Sirqul IoT Platform

Table of Contents

1. Overview . . . 1
   1.1. Version information . . . 1
   1.2. Contact information . . . 1
   1.3. License information . . . 1
   1.4. URI scheme . . . 1
   1.5. Tags . . . 1
2. Introduction to Sirqul . . . 5
3. Getting Started Guide . . . 6
4. Mesh Network using Playfield . . . 7
5. WSSL . . . 8
6. Resources . . . 9
   6.1. AMQP . . . 9
      6.1.1. Create Consumer . . . 9
      6.1.2. Update Consumer . . . 11
      6.1.3. Create Queue . . . 12
      6.1.4. Delete Queue . . . 13
      6.1.5. Get Queue . . . 14
      6.1.6. Publish Queue . . . 14
      6.1.7. Search Queue . . . 15
      6.1.8. Update Queue . . . 16
   6.2. Account . . . 17
      6.2.1. Activate/deactivate account . . . 18
      6.2.2. Block Account . . . 19
      6.2.3. Create Account . . . 20
      6.2.4. Create Account (Encrypted Username) . . . 25
      6.2.5. Login Account (Username) . . . 30
      6.2.6. Update Location . . . 31
      6.2.7. Login Account . . . 32
      6.2.8. Login as Account via Admin . . . 33
      6.2.9. Login Account (Encrypted Username) . . . 34
      6.2.10. Logout Account . . . 35
      6.2.11. Update Password . . . 36
      6.2.12. Reset Password . . . 37
      6.2.13. Get Profile Assets . . . 38
      6.2.14. Get Account . . . 39
      6.2.15. Search Accounts . . . 40

6.2.16. Update Account . . . 42
6.2.17. Get Account Referral List . . . 48
6.2.18. Send Validation Request . . . 49
6.2.19. Request Password Reset . . . 50
6.2.20. Search Accounts by Location . . . 51
6.2.21. Get Account Settings . . . 53
6.2.22. Update Account Settings . . . 54
6.2.23. Update Account Username . . . 56
6.2.24. Save Validation Status . . . 56
6.2.25. Validate Password Reset Token . . . 57
6.2.26. Save Match Token . . . 57
6.2.27. Validate Subscription . . . 58
6.3. Achievement . . . 59
6.3.1. Create Achievement . . . 60
6.3.2. Delete Achievement . . . 62
6.3.3. Get Achievement . . . 62
6.3.4. List Achievements . . . 63
6.3.5. Get Achievement Progress . . . 65
6.3.6. Override Achievement Progress . . . 66
6.3.7. Update Achievement Progress . . . 67
6.3.8. Search Achievements . . . 68
6.3.9. List Achievement Tags . . . 70
6.3.10. Create Achievement Tier . . . 70
6.3.11. Delete Achievement Tier . . . 72
6.3.12. Get Achievement Tier . . . 73
6.3.13. Search Achievement Tiers . . . 73
6.3.14. Update Achievement Tier . . . 75
6.3.15. Update Achievement . . . 76
6.4. Analytics . . . 78
6.4.1. Get Aggregated Filtered Usage . . . 79
6.4.2. Get Filtered Usage . . . 86
6.4.3. Create Usage Record . . . 93
6.4.4. Create Multiple Usage Records . . . 96
6.4.5. Get User Activity . . . 97
6.5. AppData . . . 98
6.5.1. Get App Data . . . 98
6.5.2. Create App Data . . . 101
6.5.3. Regenerate App Data . . . 103

6.6. Application . . . 104
6.6.1. Change Appliation Status . . . 105
6.6.2. Assign the owner for the application . . . 105
6.6.3. Create Application Certificate . . . 106
6.6.4. Copy an application and its content . . . 107
6.6.5. Create Application . . . 108
6.6.6. Delete Application . . . 113
6.6.7. Get Application . . . 113
6.6.8. List Applications . . . 114
6.6.9. Create Ad Placement . . . 116
6.6.10. Delete Ad Placement . . . 118
6.6.11. Get Ad Placement . . . 119
6.6.12. Search for Ad Placements . . . 120
6.6.13. Update Ad Placement . . . 120
6.6.14. Search Applications . . . 122
6.6.15. Search for Application Settings . . . 124
6.6.16. Update Application . . . 125
6.6.17. Search Application Users . . . 130
6.6.18. Get API versions . . . 130
6.7. ApplicationConfig . . . 131
6.7.1. Create AppConfig . . . 132
6.7.2. Delete AppConfig . . . 133
6.7.3. Get AppConfig . . . 133
6.7.4. Get AppConfig by Version . . . 134
6.7.5. Search AppConfigs . . . 135
6.7.6. Update AppConfig . . . 137
6.8. Asset . . . 138
6.8.1. Create Asset . . . 138
6.8.2. Delete Asset . . . 141
6.8.3. Download Asset . . . 141
6.8.4. Get Asset . . . 142
6.8.5. Convert Offer to Creative . . . 143
6.8.6. Remove Asset from Collection . . . 144
6.8.7. Search Assets . . . 145
6.8.8. Update Asset . . . 146
6.8.9. Generation . . . 149
6.8.10. Create Asset Finished . . . 149
6.9. Assignment . . . 150

6.9.1. Search Assignment Assignees . . . 151
6.9.2. Create Assignment . . . 152
6.9.3. Delete Assignment . . . 153
6.9.4. Get Assignment . . . 153
6.9.5. Search Assignments . . . 154
6.9.6. Create Assignment Status . . . 155
6.9.7. Deletes Assignment Status . . . 157
6.9.8. Get Assignment Status . . . 157
6.9.9. Search Assignment Statuses . . . 158
6.9.10. Update Assignment Status . . . 160
6.9.11. Update Assignment . . . 161
6.10. Audience . . . 162
6.10.1. Get Age Groups . . . 163
6.10.2. Create Audience . . . 163
6.10.3. Delete Audience . . . 166
6.10.4. Get Devices . . . 166
6.10.5. Get Experiences . . . 167
6.10.6. Get Audience . . . 168
6.10.7. Get GroupedAudiences . . . 169
6.10.8. Search Audiences . . . 169
6.10.9. List Sent Suggestions . . . 171
6.10.10. List Suggestions by Audience . . . 172
6.10.11. List Offers by Audience . . . 172
6.10.12. Send Suggestions . . . 173
6.10.13. Update Audience . . . 174
6.11. Bid . . . 176
6.11.1. Create Bid . . . 177
6.11.2. Delete Bid . . . 178
6.11.3. Get Bid . . . 179
6.11.4. Update Bid . . . 180
6.12. BillableEntity . . . 181
6.12.1. Create Billable . . . 182
6.12.2. Delete Billable . . . 183
6.12.3. Get Billable . . . 183
6.12.4. Update Billable . . . 184
6.13. BillingInfo . . . 186
6.13.1. Create Payment Method . . . 187
6.13.2. Get Payment Method . . . 188

6.13.3. Search Payment Methods . . . 189
6.13.4. Update Payment Method . . . 190
6.14. CVSImport . . . 192
6.14.1. Search Status . . . 193
6.14.2. Batch Status . . . 193
6.14.3. Detail Status . . . 194
6.14.4. Upload CSV . . . 195
6.15. CargoType . . . 195
6.15.1. Create Cargo Type . . . 196
6.15.2. Search Cargo Type . . . 197
6.15.3. Get Cargo Type . . . 198
6.15.4. Update Cargo Type . . . 198
6.15.5. Delete Cargo Type . . . 199
6.16. Carrier . . . 199
6.16.1. Search Carriers . . . 200
6.17. Category . . . 201
6.17.1. Create Category . . . 202
6.17.2. Delete Category . . . 203
6.17.3. Duplicate Category . . . 204
6.17.4. Get Category . . . 204
6.17.5. Search Categories . . . 205
6.17.6. Update Category . . . 207
6.18. Collection . . . 208
6.18.1. Approve Collection . . . 209
6.18.2. Create Collection . . . 210
6.18.3. Delete Collection . . . 213
6.18.4. Get Collection . . . 214
6.18.5. Search Collections . . . 215
6.18.6. Update Collection . . . 219
6.18.7. Add Collection Users . . . 222
6.18.8. Remove Collection Users . . . 223
6.18.9. Leave Collection . . . 224
6.19. Completable Action . . . 225
6.19.1. POST /api/{version}/completable/action/create . . . 225
6.19.2. POST /api/{version}/completable/action/get . . . 225
6.19.3. POST /api/{version}/completable/action/search . . . 226
6.19.4. POST /api/{version}/completable/action/update . . . 227
6.20. Connection . . . 228

6.20.1. Add Connections . . . 228
6.20.2. Remove Connections . . . 229
6.20.3. Search Connection Groups . . . 230
6.20.4. Search Possible Connections . . . 232
6.20.5. Create or Update Connection . . . 233
6.20.6. Search Connections . . . 234
6.20.7. Get Sent Friend Requests . . . 236
6.20.8. Create or Update Connection Group . . . 237
6.20.9. Add Connection . . . 238
6.20.10. Add Connection Groups . . . 239
6.20.11. Get Connection Group . . . 240
6.20.12. Delete Connection Group . . . 241
6.20.13. Delete Connection . . . 242
6.20.14. Remove Connection Groups . . . 243
6.20.15. Accept Friend . . . 243
6.20.16. Decline Friend . . . 244
6.20.17. Delete Friend . . . 245
6.20.18. Request Friend . . . 246
6.21. Contest . . . 247
6.21.1. Create or Update Contest . . . 248
6.21.2. Approve Contest . . . 250
6.21.3. Create Contest . . . 251
6.21.4. Get Contest . . . 253
6.21.5. Delete Contest . . . 254
6.21.6. Search Contests . . . 254
6.21.7. Update Contest . . . 256
6.21.8. Vote on Contest . . . 258
6.22. Creative . . . 259
6.22.1. Find Missions . . . 260
6.22.2. Add Preview . . . 262
6.22.3. Create Creative . . . 262
6.22.4. Delete Creative . . . 264
6.22.5. Get Creative . . . 264
6.22.6. Remove Preview . . . 265
6.22.7. Search Creatives . . . 266
6.22.8. Update Creative . . . 266
6.23. Dependent . . . 268
6.23.1. Create Dependent . . . 268

6.23.2. Get Dependents . . . 268
6.23.3. Delete Dependent . . . 269
6.24. Disbursement . . . 270
6.24.1. Check Disbursements . . . 270
6.24.2. Create Disbursement . . . 271
6.24.3. Get Disbursement . . . 272
6.24.4. Process Disbursements . . . 273
6.24.5. Search Disbursements . . . 274
6.24.6. Update Disbursement . . . 275
6.25. Employee . . . 276
6.25.1. Assign Employee . . . 277
6.25.2. Assign Employee to Location . . . 278
6.25.3. Create Employee . . . 278
6.25.4. Delete Employee . . . 281
6.25.5. Get Employee . . . 281
6.25.6. Search Employees . . . 282
6.25.7. Unassign Employee . . . 284
6.25.8. Update Employee . . . 284
6.26. Event . . . 287
6.26.1. Attend Event . . . 287
6.26.2. Search Event Attendance . . . 289
6.26.3. Create Event . . . 291
6.26.4. Delete Event . . . 292
6.26.5. Get Event . . . 293
6.26.6. Search Events . . . 293
6.26.7. Update Event . . . 296
6.27. Facebook . . . 297
6.27.1. Get Facebook Token . . . 298
6.27.2. Post to Facebook . . . 298
6.28. Favorite . . . 300
6.28.1. Create Favorite . . . 300
6.28.2. Delete Favorite . . . 301
6.28.3. Get Favorite . . . 302
6.28.4. Search Favorites . . . 303
6.28.5. Who has Favorited . . . 304
6.29. Filter . . . 305
6.29.1. Create Filter . . . 306
6.29.2. Delete Filter . . . 307

6.29.3. Get Filter . . . 308
6.29.4. Update Filter . . . 308
6.30. Flag . . . 309
6.30.1. Create Flag . . . 310
6.30.2. Delete Flag . . . 311
6.30.3. Get Flag . . . 312
6.30.4. POST /api/{version}/flag/override . . . 313
6.30.5. Get Flag Threshold . . . 313
6.30.6. Update Flag Threshold . . . 314
6.31. Game . . . 315
6.31.1. Create Game . . . 316
6.31.2. Delete Game . . . 316
6.31.3. Get Game . . . 317
6.31.4. Search Games . . . 318
6.31.5. Update Game . . . 319
6.32. GameLevel . . . 320
6.32.1. Create Game Level . . . 320
6.32.2. Delete Game Level . . . 323
6.32.3. Get Game Level . . . 323
6.32.4. Get Level Questions . . . 324
6.32.5. Update Level Questions . . . 325
6.32.6. Search Game Levels . . . 325
6.32.7. Search Game Level by Billable Entity . . . 326
6.32.8. Update Game Level . . . 328
6.32.9. Get Level Words . . . 331
6.32.10. Update Level Words . . . 331
6.33. Invite . . . 332
6.33.1. Accept Invite . . . 333
6.33.2. Invite to Collection . . . 334
6.33.3. Invite to Contest . . . 335
6.33.4. Invite to Event . . . 336
6.33.5. Invite to Game Level . . . 337
6.33.6. Get Invite . . . 338
6.33.7. Invite to Mission . . . 339
6.33.8. Invite to Offer . . . 340
6.33.9. Invite to Offer Location . . . 341
6.33.10. Invite to Retailer Location . . . 341
6.34. Leaderboard . . . 342

6.34.1. Create a Leaderboard . . . 343
6.34.2. Delete a Leaderboard . . . 344
6.34.3. Get a Leaderboard . . . 345
6.34.4. Search a Leaderboard . . . 346
6.34.5. Update a Leaderboard . . . 347
6.35. Like . . . 348
6.35.1. Create Like . . . 349
6.35.2. Delete Like . . . 350
6.35.3. Search Likes . . . 351
6.36. Listing . . . 352
6.36.1. Create Listing . . . 353
6.36.2. Delete Listing . . . 354
6.36.3. Get Listing . . . 355
6.36.4. Search Listings . . . 356
6.36.5. Summary Listing . . . 357
6.36.6. Update Listing . . . 358
6.37. Location . . . 359
6.37.1. Get Location by Trilateration . . . 360
6.37.2. Get Location by IP . . . 361
6.37.3. Search Regions or Postal Codes . . . 361
6.37.4. Create Trilateration Data . . . 363
6.37.5. Create Trilateration Data . . . 364
6.38. Media . . . 364
6.38.1. Create Media . . . 365
6.38.2. Delete Media . . . 370
6.38.3. Media Get . . . 371
6.38.4. Search Media . . . 371
6.38.5. Update Media . . . 374
6.39. Mission . . . 378
6.39.1. Create Mission . . . 379
6.39.2. Delete Mission . . . 381
6.39.3. Find Missions . . . 382
6.39.4. Search Mission Formats . . . 384
6.39.5. Get Mission . . . 385
6.39.6. Import Mission . . . 386
6.39.7. Search Missions . . . 387
6.39.8. Search Missions by Billable Entity . . . 388
6.39.9. Update Mission . . . 389

6.40. MissionInvite . . . 391
6.40.1. Create Mission Invite . . . 392
6.40.2. Delete Mission Invite . . . 392
6.40.3. Get Mission Invite . . . 393
6.40.4. Search Mission Invites . . . 394
6.40.5. Update Mission Invite . . . 395
6.41. Note . . . 397
6.41.1. Note batch operations . . . 397
6.41.2. Create Note . . . 398
6.41.3. Delete Note . . . 402
6.41.4. Get Note . . . 403
6.41.5. Search Notes . . . 403
6.41.6. Update Note . . . 405
6.42. Notification . . . 409
6.42.1. Send Batch Notifications . . . 409
6.42.2. Send Custom Notifications . . . 410
6.42.3. Search for Recipients . . . 412
6.42.4. Search for Recipients . . . 414
6.42.5. Get Notifications . . . 415
6.42.6. Create Notification Template . . . 417
6.42.7. Delete Notification Template . . . 418
6.42.8. Get Notification Template . . . 419
6.42.9. Search Notification Templates . . . 420
6.42.10. Update Notification Template . . . 421
6.42.11. Register Notification Token . . . 422
6.42.12. Check Notification Tokens . . . 423
6.43. ObjectStore . . . 424
6.43.1. Create Object . . . 425
6.43.2. Create Data . . . 426
6.43.3. Search Data . . . 428
6.43.4. Get Data . . . 430
6.43.5. Update Data . . . 431
6.43.6. Delete Data . . . 433
6.43.7. Delete Object . . . 434
6.43.8. Create Field . . . 434
6.43.9. Delete Field . . . 436
6.43.10. Get Object . . . 437
6.43.11. Search Objects . . . 437

6.44. Offer . . . 438
6.44.1. Get Offer . . . 439
6.44.2. Search Offers . . . 440
6.44.3. Get Offers (Counts) . . . 443
6.44.4. Get Offer Location . . . 444
6.44.5. Get Offers (Top) . . . 445
6.44.6. Create Offer . . . 445
6.44.7. Delete Offer . . . 454
6.44.8. Get Offer . . . 455
6.44.9. Update Offer Locations . . . 455
6.44.10. Delete Offer Location . . . 456
6.44.11. Search Offer Locations . . . 457
6.44.12. Search Offers . . . 459
6.44.13. Activate Offer . . . 464
6.44.14. Search Offer Transactions . . . 465
6.44.15. Update Offer Transaction . . . 468
6.44.16. Update Offer . . . 469
6.45. OfferStatus . . . 478
6.45.1. Create Offer Status . . . 479
6.45.2. Delete Offer Status . . . 480
6.45.3. Get Offer Status . . . 481
6.45.4. Search Offer Status . . . 481
6.45.5. Update Offer Status . . . 483
6.46. Optimize . . . 484
6.46.1. Request Optimization . . . 485
6.46.2. Get Optimization Result . . . 485
6.47. Pack . . . 486
6.47.1. Create Pack . . . 487
6.47.2. Delete Pack . . . 489
6.47.3. Get Pack . . . 490
6.47.4. Search Packs . . . 490
6.47.5. Update Pack . . . 492
6.48. Pathing . . . 494
6.48.1. Calculate Path . . . 495
6.49. PostalCode . . . 495
6.49.1. Create Postal Code . . . 496
6.49.2. Delete Postal Code . . . 497
6.49.3. Get Postal Code . . . 498

6.49.4. Search Postal Codes . . . 498
6.49.5. Update Postal Code . . . 499
6.50. PreviewPersona . . . 500
6.50.1. Create Persona . . . 501
6.50.2. Delete Persona . . . 502
6.50.3. Get Persona . . . 503
6.50.4. Search Personas . . . 504
6.50.5. Update Persona . . . 504
6.51. Program . . . 505
6.51.1. Create Program . . . 506
6.51.2. Search Programs . . . 507
6.51.3. Update Program . . . 508
6.51.4. Get Program . . . 508
6.51.5. Update Program . . . 509
6.51.6. Delete Program . . . 509
6.52. PurchaseItem . . . 510
6.52.1. Create Purchase . . . 511
6.52.2. Delete Purchase . . . 513
6.52.3. Get Purchase . . . 513
6.52.4. Search Purchases . . . 514
6.52.5. Update Purchase . . . 516
6.53. PurchaseOrder . . . 518
Charges Description . . . 519
Refunds Description . . . 520
6.53.3. Create Order . . . 521
6.53.4. Delete Order . . . 523
6.53.5. Get Order . . . 523
6.53.6. Search Orders . . . 524
6.53.7. Update Order . . . 525
6.54. Question . . . 527
6.54.1. Create Question . . . 528
6.54.2. Delete Question . . . 529
6.54.3. Get Question . . . 529
6.54.4. Search Questions . . . 530
6.54.5. Update Question . . . 531
6.55. Ranking . . . 532
6.55.1. Override User Rank . . . 533
6.55.2. POST /api/{version}/ranking/personal/ranks . . . 535

6.55.3. Search Rankings . . . 536
6.55.4. Update Ranking . . . 537
6.56. Rating . . . 538
6.56.1. Search Location Rating Indexes . . . 539
6.56.2. Create Rating . . . 541
6.56.3. Delete Rating . . . 542
6.56.4. Search Rating Indexes . . . 543
6.56.5. Search Ratings . . . 546
6.56.6. Update Rating . . . 548
6.57. Region . . . 549
6.57.1. Create Region . . . 550
6.57.2. Delete Region . . . 551
6.57.3. Get Region . . . 552
6.57.4. Search Regions . . . 553
6.57.5. Update Region . . . 555
6.58. Reporting . . . 557
6.58.1. Create Offline Report . . . 557
6.58.2. Delete Offline Report . . . 559
6.58.3. Get Offline Report . . . 559
6.58.4. Search Offline Reports . . . 560
6.58.5. Run Report . . . 561
6.59. Reservation . . . 562
6.59.1. Search Availability . . . 563
6.59.2. Update Availability . . . 565
6.59.3. Search Schedule . . . 566
6.59.4. Create Reservation . . . 569
6.59.5. Delete Reservation . . . 570
6.59.6. Search Reservations . . . 570
6.60. Retailer . . . 573
6.60.1. Create Retailer . . . 574
6.60.2. Delete Retailer . . . 576
6.60.3. Get Retailer . . . 577
6.60.4. Login Retailer . . . 577
6.60.5. Search Retailers . . . 578
6.60.6. Update Retailer . . . 580
6.61. RetailerLocation . . . 583
6.61.1. Create Retailer Location (Consumer) . . . 584
6.61.2. POST /api/{version}/location/details/get . . . 586

6.61.3. Get Retailer Location (Consumer) . . . 586
6.61.4. Create Retailer Location . . . 587
6.61.5. Delete Retailer Location . . . 590
6.61.6. Get Retailer Location . . . 591
6.61.7. Distance Search Retailer Locations (Indexed) . . . 592
6.61.8. Keyword Search Retailer Locations (Indexed) . . . 593
6.61.9. Search Retailer Locations (Owned) . . . 594
6.61.10. Update Retailer Location . . . 596
6.62. RetailerV2 . . . 599
6.62.1. Search Retailers . . . 600
6.63. Route . . . 601
6.63.1. Create Route . . . 602
6.63.2. Search Routes . . . 602
6.63.3. Set Driver . . . 604
6.63.4. Get Route . . . 605
6.63.5. Update Route . . . 605
6.63.6. Delete Route . . . 606
6.63.7. Approve Route . . . 606
6.63.8. Copy Route . . . 607
6.63.9. Get Route Directions . . . 607
6.63.10. Update Route Directions . . . 608
6.63.11. Disapprove Route . . . 609
6.63.12. Optimize Route . . . 609
6.63.13. Create Route Polyline . . . 610
6.63.14. Get Route Shipments . . . 610
6.63.15. Get Route Stop . . . 611
6.63.16. Update Route Stop . . . 611
6.63.17. Delete Stop . . . 612
6.63.18. Get Shipments At Stop . . . 613
6.63.19. Get Route Stops . . . 613
6.63.20. Reorder Route Stops . . . 614
6.63.21. Reorder Route Stops . . . 615
6.64. RouteSetting . . . 615
6.64.1. Create Route Setting . . . 616
6.64.2. Search Route Settings . . . 616
6.64.3. Get Route Setting . . . 617
6.64.4. Update Route Setting . . . 618
6.64.5. Delete Route Setting . . . 618

6.65. Routing . . . 619
6.65.1. Compute Route . . . 620
6.66. ScheduledNotification . . . 620
6.66.1. Create Scheduled Notification . . . 621
6.66.2. Delete Scheduled Notification . . . 625
6.66.3. Generate Schedule Notifications . . . 626
6.66.4. Get Scheduled Notification . . . 627
6.66.5. Search Scheduled Notifications . . . 627
6.66.6. Update Scheduled Notification . . . 629
6.67. Score . . . 632
6.67.1. Create Score . . . 633
6.67.2. Get Score . . . 634
6.67.3. Search Score . . . 635
6.68. SecureApp . . . 636
6.68.1. Create Secure Application . . . 637
6.68.2. Delete Secure Application . . . 639
6.68.3. Rest Secure Application . . . 639
6.68.4. Update Secure Application . . . 640
6.68.5. Login Clear . . . 641
6.68.6. Purchase Clear . . . 642
6.69. ServiceHub . . . 643
6.69.1. Create Service Hub . . . 644
6.69.2. Search Service Hubs . . . 644
6.69.3. Update Service Hub . . . 645
6.69.4. Get Service Hub . . . 646
6.69.5. Update Service Hub . . . 646
6.69.6. Delete Service Hub . . . 647
6.70. Shipment . . . 648
6.70.1. Create Shipment . . . 648
6.70.2. Search Shipments . . . 649
6.70.3. Get Shipment . . . 650
6.70.4. Update Shipment . . . 650
6.70.5. Delete Shipment . . . 651
6.70.6. Cancel Shipment . . . 652
6.70.7. Uupdate Shipment Status . . . 652
6.71. ShipmentBatch . . . 653
6.71.1. Create Shipment Batch . . . 653
6.71.2. Search Shipment Batch . . . 654

6.71.3. Get Shipment Batch . . . 655
6.71.4. Delete Shipment Batch . . . 655
6.71.5. Get Shipment Batch Status . . . 656
6.72. Simulation . . . 657
6.72.1. Routing Simulation . . . 658
6.73. Stop . . . 659
6.73.1. Get Stop . . . 659
6.73.2. Update Stop . . . 660
6.74. StopLocation . . . 660
6.74.1. Create Location . . . 661
6.74.2. Update Location . . . 662
6.75. Subscription . . . 662
6.75.1. Create Subscription . . . 663
6.75.2. Delete Subscription . . . 664
6.75.3. Get Subscription . . . 664
6.75.4. Get Subscription Plan . . . 665
6.75.5. List Subscription Plans . . . 665
6.75.6. Update Subscription . . . 666
6.75.7. Get Subscription Usage . . . 667
6.76. Task . . . 668
6.76.1. Create Task . . . 669
6.76.2. Delete Task . . . 670
6.76.3. Get Task . . . 671
6.76.4. Search Tasks . . . 671
6.76.5. Update Task . . . 673
6.77. Territory . . . 674
6.77.1. Create Territory . . . 675
6.77.2. Delete Territory . . . 676
6.77.3. Get Territory . . . 676
6.77.4. Search Territories . . . 677
6.77.5. Update Territory . . . 678
6.78. ThemeDescriptor . . . 679
6.78.1. Create/Update Theme . . . 680
6.78.2. Get Theme . . . 682
6.78.3. Delete Theme . . . 683
6.78.4. Search Themes . . . 684
6.79. ThirdPartyCredentials . . . 685
6.79.1. Create Credential . . . 686

6.79.2. Delete Credential . . . 688
6.79.3. Get Credential . . . 688
6.79.4. POST /api/{version}/thirdparty/credential/mfa/send . . . 690
6.79.5. Search Credentials . . . 690
6.79.6. Update Credential . . . 691
6.79.7. Create Network . . . 692
6.79.8. Delete Network . . . 693
6.79.9. Get Network . . . 694
6.79.10. Search Networks . . . 695
6.79.11. Update Network . . . 696
6.79.12. Search User Credentials by Network . . . 697
6.80. Ticket . . . 698
6.80.1. Gift Tickets . . . 699
6.80.2. POST /api/{version}/purchase/list . . . 700
6.80.3. Get Ticket Count . . . 700
6.80.4. Get Ticket List . . . 701
6.80.5. Save Ticket . . . 702
6.80.6. Save Ticket with Reciept . . . 704
6.80.7. Get Ticket Offers . . . 705
6.81. Tournament . . . 706
6.81.1. Create Tournament . . . 707
6.81.2. Delete Tournament . . . 710
6.81.3. Get Tournament . . . 710
6.81.4. POST /api/{version}/tournament/object/search . . . 711
6.81.5. Search Tournament Rounds . . . 712
6.81.6. Submit Tournament Score . . . 713
6.81.7. Search Tournaments . . . 714
6.81.8. Substitute Tournament Player . . . 715
6.81.9. Update Tournament . . . 715
6.81.10. Submit Tournament Score . . . 718
6.82. Tracking . . . 719
6.82.1. Create Batch Tracking . . . 720
6.82.2. Create Tracking Leg . . . 721
6.82.3. List Tracking . . . 722
6.82.4. Get Tracking Path . . . 723
6.82.5. Get Predicted Locations . . . 724
6.82.6. Search Preferred Locations . . . 725
6.82.7. Search Tracking . . . 726

6.82.8. Create Tracking Step . . . 727
6.83. Trigger . . . 728
6.83.1. Create Trigger . . . 729
6.83.2. Delete Trigger . . . 731
6.83.3. Get Trigger . . . 731
6.83.4. Search Triggers . . . 732
6.83.5. Update Trigger . . . 734
6.84. Trip . . . 735
6.84.1. Create Trip . . . 736
6.84.2. Search Trips . . . 736
6.84.3. Search Trips . . . 738
6.84.4. Process Trip Matches . . . 739
6.84.5. Trip Notifications . . . 739
6.84.6. Get Trip . . . 740
6.84.7. Update Trip . . . 740
6.84.8. Delete Trip . . . 741
6.84.9. Cancel Trip Preference . . . 742
6.84.10. Set Trip Preference Driver . . . 742
6.84.11. Set Trip Preference Flexible . . . 743
6.84.12. Update Trip Locations . . . 744
6.84.13. Update Recurrence Locations . . . 744
6.84.14. Get Trip Matches . . . 745
6.84.15. Set Trip Preference Rider . . . 745
6.84.16. Update Trip Shipments . . . 746
6.84.17. Update Recurrence Shipments . . . 747
6.85. Twilio . . . 747
6.85.1. Buy Offer by SMS . . . 748
6.86. Twitter . . . 749
6.86.1. Authorize Twitter . . . 749
6.86.2. Login Twitter . . . 750
6.87. UserPermissions . . . 751
6.87.1. Add User . . . 752
6.87.2. Leave . . . 753
6.87.3. Remove User . . . 754
6.87.4. Approve Permissionable . . . 755
6.87.5. Search Permissionables Following Distance . . . 756
6.87.6. Search Permissionables . . . 757
6.88. Vehicle . . . 759

6.88.1. Create Vehicle . . . 759
6.88.2. Search Vehicle . . . 760
6.88.3. Get Vehicle . . . 761
6.88.4. Update Vehicle . . . 761
6.88.5. Delete Vehicle . . . 762
6.89. VehicleType . . . 762
6.89.1. Create Vehicle Type . . . 763
6.89.2. Search Vehicle Type . . . 764
6.89.3. Get Vehicle Type . . . 765
6.89.4. Update Vehicle Type . . . 765
6.89.5. Delete Vehicle Type . . . 766
6.90. Wallet . . . 766
6.90.1. Create Wallet Offers . . . 767
6.90.2. Delete Wallet Offer . . . 768
6.90.3. Get Wallet Offer . . . 769
6.90.4. Search Wallet Offers . . . 770
6.90.5. Update Wallet Offer . . . 773
6.91. Weather . . . 774
6.91.1. Search Weather . . . 775
6.92. Word . . . 776
6.92.1. Create Word . . . 776
6.92.2. Delete Word . . . 777
6.92.3. Get Word . . . 778
6.92.4. Search Words . . . 779
6.92.5. Update Word . . . 780
6.93. Workflow . . . 781
6.93.1. Run Workflow . . . 781
7. Definitions . . . 783
7.1. Account . . . 783
7.2. AccountAppSettings . . . 790
7.3. AccountListResponse . . . 791
7.4. AccountLoginResponse . . . 792
7.5. AccountMiniResponse . . . 793
7.6. AccountProfileInfo . . . 793
7.7. AccountResponse . . . 795
7.8. AccountShortResponse . . . 796
7.9. AchievementProgressResponse . . . 797
7.10. AchievementResponse . . . 799

7.11. AchievementShortResponse . . . 800
7.12. AchievementTierResponse . . . 802
7.13. Address . . . 803
7.14. AddressResponse . . . 803
7.15. AgeGroupResponse . . . 804
7.16. Album . . . 804
7.17. AlbumContestListResponse . . . 813
7.18. AlbumContestResponse . . . 814
7.19. AlbumFullResponse . . . 816
7.20. AlbumResponse . . . 821
7.21. AnswerResponse . . . 824
7.22. AppInfoResponse . . . 824
7.23. AppResponse . . . 825
7.24. AppVersion . . . 826
7.25. Application . . . 826
7.26. ApplicationConfigResponse . . . 831
7.27. ApplicationMiniResponse . . . 832
7.28. ApplicationResponse . . . 833
7.29. ApplicationSettingsResponse . . . 837
7.30. ApplicationShortResponse . . . 838
7.31. ApplicationUsageResponse . . . 840
7.32. Asset . . . 840
7.33. AssetFullResponse . . . 846
7.34. AssetListResponse . . . 848
7.35. AssetResponse . . . 849
7.36. AssetShortResponse . . . 850
7.37. Assignment . . . 852
7.38. AssignmentResponse . . . 856
7.39. AssignmentStatus . . . 857
7.40. AssignmentStatusResponse . . . 858
7.41. Audience . . . 859
7.42. AudienceDevice . . . 861
7.43. AudienceDeviceResponse . . . 861
7.44. AudienceDeviceVersionRange . . . 862
7.45. AudienceLocation . . . 862
7.46. AudienceResponse . . . 863
7.47. AudienceTargetType . . . 865
7.48. AvailabilityResponse . . . 865

7.49. BaseOfferResponse . . . 866
7.50. BidResponse . . . 868
7.51. BillableEntity . . . 869
7.52. BillableEntityResponse . . . 870
7.53. BillableEntityShortResponse . . . 871
7.54. BiometricImage . . . 871
7.55. BiometricRequest . . . 871
7.56. BlobFile . . . 871
7.57. Building . . . 872
7.58. CargoType . . . 873
7.59. Category . . . 874
7.60. CategoryResponse . . . 875
7.61. CategoryTreeResponse . . . 876
7.62. CellCarrier . . . 877
7.63. CellCarrierResponse . . . 878
7.64. ChartData . . . 879
7.65. Chronology . . . 880
7.66. CohortRegion . . . 881
7.67. CompletableActionResponse . . . 881
7.68. ConfigRoutingRequest . . . 882
7.69. ConnectedFriend . . . 883
7.70. Connection . . . 883
7.71. ConnectionGroup . . . 885
7.72. ConnectionGroupResponse . . . 886
7.73. ConnectionGroupShortResponse . . . 888
7.74. ConnectionInfoResponse . . . 888
7.75. ConnectionListResponse . . . 889
7.76. ConnectionResponse . . . 890
7.77. ConsumerInviteResponse . . . 891
7.78. Contact . . . 892
7.79. ContactInfo . . . 894
7.80. ContactInfoResponse . . . 894
7.81. ContactResponse . . . 895
7.82. CoordsResponse . . . 896
7.83. CountResponse . . . 897
7.84. CreativeResponse . . . 897
7.85. CsvImportResponse . . . 898
7.86. DateTimeField . . . 899

7.87. DateTimeFieldType . . . 900
7.88. DateTimeRange . . . 900
7.89. DateTimeZone . . . 900
7.90. Device . . . 900
7.91. DeviceResponse . . . 901
7.92. Dimensions . . . 902
7.93. Direction . . . 902
7.94. DirectionResponse . . . 903
7.95. DisbursementResponse . . . 904
7.96. DriverResponse . . . 905
7.97. DurationField . . . 906
7.98. DurationFieldType . . . 906
7.99. EmployeeResponse . . . 906
7.100. Event . . . 908
7.101. EventAttendanceResponse . . . 908
7.102. EventResponse . . . 908
7.103. Filter . . . 908
7.104. FilterResponse . . . 909
7.105. FilterTreeResponse . . . 910
7.106. FingerprintBiometricImage . . . 911
7.107. Flag . . . 911
7.108. FlagResponse . . . 912
7.109. FlagThreshold . . . 912
7.110. Game . . . 913
7.111. GameLevel . . . 914
7.112. GameLevelListResponse . . . 924
7.113. GameLevelResponse . . . 924
7.114. GameListResponse . . . 927
7.115. GameObjectListResponse . . . 927
7.116. GameObjectResponse . . . 928
7.117. GameResponse . . . 929
7.118. GeoBox . . . 930
7.119. GeoPointResponse . . . 930
7.120. GeoResponse . . . 931
7.121. GeocodeLatLngResponse . . . 931
7.122. GroupPermissions . . . 932
7.123. ImportStatuses . . . 932
7.124. Interval . . . 932

7.125. InviteResponse . . . 932
7.126. Invoice . . . 934
7.127. ItineraryResponse . . . 934
7.128. JsonArray . . . 935
7.129. JsonElement . . . 936
7.130. JsonNull . . . 937
7.131. JsonObject . . . 938
7.132. JsonPrimitive . . . 940
7.133. LeaderboardResponse . . . 941
7.134. Leg . . . 942
7.135. LegResponse . . . 944
7.136. LikableResponse . . . 945
7.137. LikeResponse . . . 946
7.138. LineItem . . . 946
7.139. ListCountResponse . . . 947
7.140. ListResponse . . . 947
7.141. Listing . . . 948
7.142. ListingFullResponse . . . 949
7.143. ListingGroupResponse . . . 950
7.144. ListingResponse . . . 950
7.145. LoadResponse . . . 951
7.146. LoadShortResponse . . . 952
7.147. LocalTime . . . 952
7.148. Location . . . 953
7.149. LocationResponse . . . 955
7.150. LocationSearchResponse . . . 956
7.151. Media . . . 957
7.152. MediaOfferResponse . . . 958
7.153. MediaResponse . . . 960
7.154. MessageListResponse . . . 961
7.155. MessageResponse . . . 961
7.156. Mission . . . 961
7.157. MissionFormatResponse . . . 966
7.158. MissionInviteResponse . . . 966
7.159. MissionListResponse . . . 967
7.160. MissionResponse . . . 967
7.161. MissionShortResponse . . . 970
7.162. MissionTask . . . 972

7.163. NameStringValueResponse . . . 973
7.164. Node . . . 973
7.165. NodeRequest . . . 973
7.166. Note . . . 974
7.167. NoteFullResponse . . . 980
7.168. NoteResponse . . . 981
7.169. NotificationMessageListResponse . . . 983
7.170. NotificationMessageResponse . . . 983
7.171. NotificationRecipientResponse . . . 984
7.172. NotificationSettingsResponse . . . 985
7.173. NotificationTemplateResponse . . . 988
7.174. Number . . . 988
7.175. ObjectStoreResponse . . . 989
7.176. Offer . . . 989
7.177. OfferListResponse . . . 998
7.178. OfferLocation . . . 999
7.179. OfferResponse . . . 1001
7.180. OfferShortResponse . . . 1007
7.181. OfferTransactionResponse . . . 1012
7.182. OfferTransactionStatusResponse . . . 1018
7.183. OrderItemRequest . . . 1019
7.184. OrderPackage . . . 1019
7.185. OrderResponse . . . 1020
7.186. Orders . . . 1021
7.187. Pack . . . 1021
7.188. PackListResponse . . . 1023
7.189. PackResponse . . . 1024
7.190. Participant . . . 1026
7.191. ParticipantResponse . . . 1027
7.192. PathingResponse . . . 1027
7.193. PaymentRequest . . . 1028
7.194. PaymentTransactionResponse . . . 1029
7.195. PaymentTypesResponse . . . 1030
7.196. PermissionResponse . . . 1032
7.197. Permissions . . . 1033
7.198. PersonalProfileResponse . . . 1033
7.199. Placement . . . 1035
7.200. PlacementResponse . . . 1037

7.201. Platform . . . 1037
7.202. PlatformResponse . . . 1038
7.203. PostalCode . . . 1039
7.204. PostalCodeResponse . . . 1040
7.205. PredictedLocationResponse . . . 1040
7.206. PreferredLocationResponse . . . 1041
7.207. PreviewPersonaResponse . . . 1042
7.208. Product . . . 1042
7.209. ProductResponse . . . 1044
7.210. ProfileInfoResponse . . . 1045
7.211. ProfileResponse . . . 1047
7.212. ProfileShortResponse . . . 1049
7.213. Program . . . 1050
7.214. PromoCode . . . 1050
7.215. PurchaseItemFullResponse . . . 1051
7.216. PurchaseItemListResponse . . . 1052
7.217. PurchaseItemResponse . . . 1053
7.218. PurchaseItemShortResponse . . . 1054
7.219. PurchaseOrderItemResponse . . . 1055
7.220. QuestionResponse . . . 1056
7.221. QueueResponse . . . 1057
7.222. RankFullResponse . . . 1058
7.223. RankListResponse . . . 1059
7.224. RankResponse . . . 1060
7.225. RatingIndexResponse . . . 1061
7.226. RatingResponse . . . 1061
7.227. Recurrence . . . 1062
7.228. Region . . . 1062
7.229. RegionResponse . . . 1064
7.230. ReportBatchResponse . . . 1066
7.231. ReportDefinition . . . 1067
7.232. ReportResponse . . . 1068
7.233. ReportTypeResponse . . . 1069
7.234. ReservationResponse . . . 1069
7.235. Retailer . . . 1070
7.236. RetailerCountResponse . . . 1073
7.237. RetailerFullResponse . . . 1073
7.238. RetailerLocation . . . 1074

7.239. RetailerLocationResponse . . . 1077
7.240. RetailerLocationShortResponse . . . 1079
7.241. RetailerOfferResponse . . . 1080
7.242. RetailerProfile . . . 1086
7.243. RetailerResponse . . . 1087
7.244. RetailerShortResponse . . . 1089
7.245. RewardResponse . . . 1089
7.246. Route . . . 1090
7.247. RouteSettings . . . 1092
7.248. RoutingListResponse . . . 1093
7.249. RoutingResponse . . . 1094
7.250. RuleListResponse . . . 1095
7.251. RuleResponse . . . 1096
7.252. ScheduledNotification . . . 1096
7.253. ScheduledNotificationFullResponse . . . 1103
7.254. ScheduledNotificationShortResponse . . . 1106
7.255. ScoreListResponse . . . 1106
7.256. ScoreResponse . . . 1107
7.257. SearchResponse . . . 1108
7.258. ServiceHub . . . 1108
7.259. ServiceType . . . 1109
7.260. Shipment . . . 1110
7.261. ShipmentBatch . . . 1112
7.262. ShipmentImportStatus . . . 1114
7.263. ShipmentOrder . . . 1115
7.264. SirqulResponse . . . 1118
7.265. SizeGroup . . . 1118
7.266. SizeGroupResponse . . . 1119
7.267. Step . . . 1119
7.268. StepResponse . . . 1120
7.269. Stop . . . 1121
7.270. StopResponse . . . 1123
7.271. Subscription . . . 1124
7.272. SubscriptionOption . . . 1127
7.273. SubscriptionOptionResponse . . . 1128
7.274. SubscriptionPlan . . . 1128
7.275. SubscriptionPlanResponse . . . 1129
7.276. SubscriptionResponse . . . 1130

7.277. SummaryRankResponse . . . 1130
7.278. TaskResponse . . . 1131
7.279. Territory . . . 1134
7.280. TerritoryResponse . . . 1134
7.281. ThemeDescriptorResponse . . . 1135
7.282. ThirdPartyCredentialResponse . . . 1135
7.283. ThirdPartyCredentials . . . 1136
7.284. ThirdPartyNetwork . . . 1137
7.285. ThirdPartyNetworkResponse . . . 1139
7.286. ThirdPartyNetworkShortResponse . . . 1139
7.287. TicketCountResponse . . . 1140
7.288. TicketListResponse . . . 1140
7.289. TicketOfferResponse . . . 1141
7.290. TicketResponse . . . 1142
7.291. TimeSlotResponse . . . 1143
7.292. TokenResponse . . . 1143
7.293. TournamentResponse . . . 1144
7.294. TriggerResponse . . . 1148
7.295. TrilatAppSettings . . . 1150
7.296. TrilatCacheRequest . . . 1151
7.297. TrilatCacheSample . . . 1151
7.298. Trip . . . 1152
7.299. Tutorial . . . 1153
7.300. TutorialResponse . . . 1154
7.301. TwiMLResponse . . . 1155
7.302. UserActivityResponse . . . 1155
7.303. UserLocationListResponse . . . 1157
7.304. UserLocationResponse . . . 1158
7.305. UserLocationSearchResponse . . . 1160
7.306. UserPermissions . . . 1160
7.307. UserPermissionsListResponse . . . 1161
7.308. UserPermissionsResponse . . . 1162
7.309. UserSettingsResponse . . . 1163
7.310. Vehicle . . . 1164
7.311. VehicleCargoSetting . . . 1165
7.312. VehicleResponse . . . 1165
7.313. VehicleType . . . 1166
7.314. Verb . . . 1167

7.315. WeatherForecastResponse 1167
7.316. WeatherResponse 1168
7.317. WordzWordResponse 1169
7.318. WrappedResponse 1170
7.319. YayOrNay 1171

Chapter 1. Overview

Sirqul provides an Engagement-as-a-Service (EaaS) IoT Platform with Smart Mesh network technology to drive engagement, operational efficiency, rapid innovation and new revenue streams. Please visit https://dev.sirqul.com/ for more documents, examples, and sample applications.

1.1. Version information

Version : 3.17

1.2. Contact information

Contact : Sirqul Info
Contact Email : info@sirqul.com

1.3. License information

Terms of service : https://dev.sirqul.com/app/webroot/dev/tos.php

1.4. URI scheme

Host : dev.sirqul.com
BasePath : /api/3.17
Schemes : HTTP, HTTPS

1.5. Tags

- AMQP
- Account
- Achievement
- Analytics
- AppData
- Application
- ApplicationConfig
- Asset
- Assignment
- Audience

- Bid
- BillableEntity
- BillingInfo
- CVSImport
- CargoType
- Carrier
- Category
- Collection
- Completable Action
- Connection
- Contest
- Creative
- Dependent
- Disbursement
- Employee
- Event
- Facebook
- Favorite
- Filter
- Flag
- Game
- GameLevel
- Invite
- Leaderboard
- Like
- Listing
- Location
- Media
- Mission
- MissionInvite
- Note
- Notification

- ObjectStore
- Offer
- OfferStatus
- Optimize
- Pack
- Pathing
- PostalCode
- PreviewPersona
- Program
- PurchaseItem
- PurchaseOrder
- Question
- Ranking
- Rating
- Region
- Reporting
- Reservation
- Retailer
- RetailerLocation
- RetailerV2
- Route
- RouteSetting
- Routing
- ScheduledNotification
- Score
- SecureApp
- ServiceHub
- Shipment
- ShipmentBatch
- Simulation
- Stop
- StopLocation

- Subscription
- Task
- Territory
- ThemeDescriptor
- ThirdPartyCredentials
- Ticket
- Tournament
- Tracking
- Trigger
- Trip
- Twilio
- Twitter
- UserPermissions
- Vehicle
- VehicleType
- Wallet
- Weather
- Word
- Workflow

Chapter 2. Introduction to Sirqul

Chapter 3. Getting Started Guide

Follow these simple steps to get started:

- Create an account by going to https://dev.sirqul.com/users/add
- Create an application by going to https://dev.sirqul.com/applications/add
- Select the newly created application from the list and find the Application Keys section to retrieve your public and rest keys used to make api calls

You are now ready to start using the various servcies

Chapter 4. Mesh Network using Playfield

The Playfield SDKs enable communications between devices without requiring an Internet connection. Apps that leverage the Playfield SDK can "daisy-chain", creating local adhoc networks that work across multiple protocols (Bluetooth, Wifi, Internet, cellular, etc.) that is not dependent on a separate hosted cloud server or solution.

Playfield turns any device into a "Server" (Host) that is capable of interacting with a plethora of connected devices enabling it to work in concert to sync information across devices, meaning "relay" & "broadcast" is supported. Playfield empowers devices to communicate with each other directly regardless of their connection protocol.

For example:

- Device A (Host) and Device B are connected to each other over Wifi.
- Device C is connected to Device B over bluetooth but does not have wifi or a connection to the Internet.
- Device A can communicate with Device C by passing information back and forth through Device B. This concept is called daisy-chaining.

In addition to this, Playfield supports host migration/network healing as well:

- If Device A (Host) loses a connection or fails for some reason, it will then migrate the host capabilities to the next best device. In this case it is Device B.
- Device B is still connected to Device C via bluetooth.
- If Device A comes back online, it will automatically connect back into the mesh to Device B.

Playfield can be used for many different scenarios. For example:

- Retail: Can be used to detect users in a physical store to see what kind of shopping habits that they exhibit.
- Automotive: Have your smartphone receive notifications and maintenance reminders for oil changes and other necessary appointments.
- Smart Home: Use your smartphone to communicate with other smart devices from within the home locally, as well as over the Internet when you are out of the house.
- Healthcare: Use indoor positioning with beacons to determine where equipment is located from within the hospital building to reduce equipment loss.

Chapter 5. WSSL

Developers can define WSSL services, aggregators (combination of services), and stylesheets (used to modify the HTTP responses) to generate their own custom response for their application. For example, there could be a news application that aggregates multiple news sources and returns the results in the way that you define, creating a single cohesive response that is simpler to use in your application.

A WSSL service is in essence, a proxy for making HTTP requests. It can be used to make HTTP requests to Sirqul services as well as any open REST API. The response is parsed and can be manipulated via a graphical user interface (GUI). Note: The WSSL GUI works with XML, but json responses are automatically converted into XML format. WSSL Services are composed of a HTTP URL, parameters (which are used in making HTTP requests), and output variables. Values in the HTTP response can be referenced and saved as “output variables”, which can then be used in WSSL Aggregators for referencing response values.

A WSSL aggregator allows the ability to combine multiple services (as well other aggregators) together to form one cohesive response. Asynchronous service requests are made when applicable, but it is also possible to conditionally run services based on the response values from another. For example, there could be a WSSL service that sends in an error analytic only if another service failed to return a valid response.

WSSL stylesheets can then be used to customize WSSL Aggregator responses even further by completely changing the structure or even the response format. For example, the XML responses generated from the WSSL Aggregators can be converted to JSON, or to simple HTML pages that can be viewed by consumers.

Chapter 6. Resources

At the foundation of Sirqul's platform are 70 APIs, smart mesh edge computing and seven industry focused, customizable white-label solution templates for rapidly deploying IoT instances. Sirqul is — device, protocol and cloud agnostic — fostering an interoperable system for building future-proof IoT solutions.

Benefits

- Increase efficiencies, productivity, engagement and ROI
- Create new revenue streams: "product-as-a-service", "pay-per-use", and "monetization of data"
- Respond to environment and market dynamics faster
- Scale dynamically to meet business requirements
- Create custom IoT solutions with 70 APIs using Magellan — a drag, drop and orchestration tool

6.1. AMQP

The Advanced Message Queuing Protocol (AMQP) is an open standard for transfering data between systems. Queues can be setup for applications so that analytic data that is sent in or generated can be published to 3rd parties. Queues can also be configured to consume data from 3rd party queues and then make a service call to take action on the payload.

Below are some example scenarios as to how these services can be used:

- Retail: publish customer location information as they are moving around the store
- Smart Home: receive door entry logs from a 3rd party IoT system

This service is important because it your system to not have to poll another system to check if there are updates such as analytic data that needs to be pulled in. This avoids unnecessary service calls to be made to perform these kinds of checks. Instead, it allows the other system to push data into your system whenever there are new updates to be made.

The Scheduled notification service works well with the AMQP service. For example, a system can have an AMQP stack deployed, take a payload format, parse, and say to perform a scheduled notification. So if there was a Smart Door system and the doorbell alarm puts a message on the queue, your system can consume the message and send an alert to the home owner that there's someone at the door.

6.1.1. Create Consumer

```
POST /api/{version}/queue/consumer/create
```

Description

Create a connection to an existing amqp queue and register as a consumer.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version to use when constructing the response	number(double)	
Query	**accountId** *required*	The logged in user ID	integer(int64)	
Query	**appKey** *required*	The application key to use when creating an analytic or service request. The account needs to have permissions to the applicaton or it will be denied.	string	
Query	**dataMapping** *required*	The data mapping information in the format of AMQPRequest	string	
Query	**deviceId** *optional*	The client deviceID	string	
Query	**exchanger** *optional*	The exchanger of the queue to connect to	string	
Query	**exchangerType** *optional*	The exchanger type of the queue to connect to	string	
Query	**hostname** *required*	The hostname of the server the queue is hosted on	string	
Query	**name** *required*	The name of the queue to connect to	string	
Query	**password** *required*	The password to access the queue to connect to	string	
Query	**port** *optional*	The port of the server the queue is hosted on	integer(int32)	`"5672"`
Query	**useSSL** *optional*	Use SSL	boolean	
Query	**username** *required*	The username to access the server the queue is hosted on	string	
Query	**virtualHost** *optional*	The virtual host defined on the server the queue is associated on	string	

Type	Name	Description	Schema	Default
Query	**workers** *optional*	The number of workers to generate	integer(int32)	"1"

Responses

HTTP Code	Description	Schema
200	successful operation	QueueResponse

6.1.2. Update Consumer

```
POST /api/{version}/queue/consumer/update
```

Description

Update an existing amqp queue's data mapping.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use when constructing the response	number(double)
Query	**accountId** *required*	The logged in user ID	integer(int64)
Query	**appKey** *required*	The application key to use when creating an analytic or service request. The account needs to have permissions to the applicaton or it will be denied.	string
Query	**dataMapping** *required*	The data mapping information in the format of AMQPRequest	string
Query	**deviceId** *optional*	The client deviceID	string
Query	**queueId** *required*	The queue to update	integer(int64)
Query	**useSSL** *optional*	Use SSL	boolean

Responses

HTTP Code	Description	Schema
200	successful operation	QueueResponse

6.1.3. Create Queue

```
POST /api/{version}/queue/create
```

Description

Create a basic AMQP queue. If the username and password and virtual host is not sepcified, the queue will be created on the virtual host assigned to the application.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version to use when constructing the response	number(double)	
Query	**accountId** *required*	The logged in user ID	integer(int64)	
Query	**analyticTags** *optional*	If provided the analytic processing will publsih to this queue instead of the default one for the provided list of tags	string	
Query	**appKey** *required*	The application key unique to each application.	string	
Query	**deviceId** *optional*	The client deviceID	string	
Query	**hostname** *optional*	Hostname	string	
Query	**name** *required*	The name of the queue to create	string	
Query	**password** *optional*	Password	string	
Query	**port** *optional*	Port	integer(int32)	
Query	**useSSL** *optional*	Use SSL	boolean	

Type	Name	Description	Schema	Default
Query	**username** *optional*	Username	string	
Query	**virtualHost** *optional*	Virtual Host	string	
Query	**workers** *optional*	The number of workers to generate	integer(int32)	"1"

Responses

HTTP Code	Description	Schema
200	successful operation	QueueResponse

6.1.4. Delete Queue

```
POST /api/{version}/queue/delete
```

Description

Delete the stored queue record and close any active connections to the AMQP servers.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use when constructing the response	number(double)
Query	**accountId** *required*	Account Id	integer(int64)
Query	**deviceId** *optional*	Device Id	string
Query	**queueId** *required*	The id of the queue to find	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.1.5. Get Queue

```
GET /api/{version}/queue/get
```

Description

Get the stored queue record. Must supply the queueId, or the name and hostname and virtualHost, or the name and appKey to find the record.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use when constructing the response	number(double)
Query	**accountId** *required*	Account Id	integer(int64)
Query	**appKey** *optional*	The application key the queue was assigned to	string
Query	**deviceId** *optional*	Device Id	string
Query	**hostname** *optional*	The hostname of the queue to find	string
Query	**name** *optional*	The name of the queue to find	string
Query	**queueId** *optional*	The id of the queue to find	integer(int64)
Query	**virtualHost** *optional*	The virtual host of the queue to find	string

Responses

HTTP Code	Description	Schema
200	successful operation	QueueResponse

6.1.6. Publish Queue

```
POST /api/{version}/queue/publish
```

Description

Publish a message to a stored queue. Must supply the queueId, or the name and hostname and virtualHost, or the name and appKey to find the record.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use when constructing the response	number(double)
Query	**appKey** *optional*	The application key the queue was assigned to	string
Query	**hostname** *optional*	Hostname	string
Query	**message** *required*	The payload to send to the queue	string
Query	**name** *optional*	The name of the queue to publish to or the analytic tag to handle if the analytic param is true	string
Query	**queueId** *optional*	The id of the queue to publish to	integer(int64)
Query	**virtualHost** *optional*	Virtual Host	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.1.7. Search Queue

```
GET /api/{version}/queue/search
```

Description

Get the queues setup for the BillableEntity's applications.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version to use when constructing the response	number(double)	
Query	**accountId** *required*	Account Id	integer(int64)	
Query	**deviceId** *optional*	Device Id	string	
Query	**limit** *optional*	Limit	integer(int32)	`"10"`
Query	**name** *optional*	The name of the queue to find	string	
Query	**queueId** *optional*	The id of the queue to find	integer(int64)	
Query	**start** *optional*	Start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	QueueResponse

6.1.8. Update Queue

```
POST /api/{version}/queue/update
```

Description

Update the basic AMQP queue.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use when constructing the response	number(double)
Query	**accountId** *required*	The logged in user ID	integer(int64)

Type	Name	Description	Schema
Query	**analyticTags** *optional*	If provided the analytic processing will publsih to this queue instead of the default one for the provided list of tags	string
Query	**appKey** *optional*	The application key unique to each application.	string
Query	**deviceId** *optional*	The client deviceID	string
Query	**hostname** *optional*	Hostname	string
Query	**password** *optional*	Password	string
Query	**port** *optional*	Port	integer(int32)
Query	**queueId** *required*	The id of the queue to update	integer(int64)
Query	**useSSL** *optional*	Use SSL	boolean
Query	**username** *optional*	Username	string
Query	**virtualHost** *optional*	Virtual Host	string
Query	**workers** *optional*	The number of workers to generate	integer(int32)

Responses

HTTP Code	Description	Schema
200	successful operation	QueueResponse

6.2. Account

The Accounts service enables your users to establish an identity in your application, in addition to allowing your users to interact & engage with other users in a community. The identify of your user is critical in understanding who they are, why they're using your application, and how you can further personalize and enhance their experience. Think of any service that you have used either currently or in the past (e.g. shopping app, personal banking web portal, home security system, etc.). Each of these services more likely than not will require you to establish an account or identify in order to continue

using their service.

Below are some example scenarios as to how these services can be used:

- Retail: Customers inside of a retail store who have the shopping app installed should be able to receive personalized notifications about sales that they would be interested in.
- Medical: Patients, doctors, and nurses should each be a different account type so that they have their own specified roles & permissions.
- Real Estate: The garage door of a smart home should automatically open for the owner of the home as they approach the driveway.
- Automotive: The driver's side door should only unlock when the registered owner approaches the vehicle.
- Robotics: The drone should deliver a package only to the customer account that requested and paid for the product.

The Accounts service is important so that personalized data & content can only be accessed by a particular individual. In addition, identifying & understanding your user accounts behavior will help you decide in making the right feature updates for your product or service. User accounts are also important in the aspect of collecting data in order to make good business decisions. For instance, learning more about the frequency of logins, average usage time per day, user behavior, and much more. All of these are critical pieces for a successful product or service.

Below are great example services that could be combined with the Accounts service:

- Wallets Service: User accounts that have purchased offers that are saved
- Notifications Service: Personalized push notifications that are targeted to user accounts that fit in a particular audience
- Connections Service: User accounts can become friends with other user accounts
- User Permissions Service: User account can access their friend's photo album, but cannot make any edits or upload new photos

6.2.1. Activate/deactivate account

```
POST /api/{version}/account/active/update
```

Description

Update account's own active flag. logged in account must satisfy these conditions to activate/deactivate an account:
1) Logged in account need to have higher access level than the deactivating account
2) Deactivating account need to belong to the selected application
3) Logged in account is the current selected application's owner OR is an executive

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API Version	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**active** *required*	true will activate the user and false will deactivate	boolean
Query	**appKey** *optional*	The application key that the user belong too	string
Query	**connectionAccountId** *required*	The account id of the user you want to modify (if this is not set, then the "accountId" parameter will be used instead)	integer(int64)
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.2.2. Block Account

```
POST /api/{version}/account/block
```

Description

Moves or removes an account into the user's blocked group.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version	number(double)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	

Type	Name	Description	Schema	Default
Query	**accountIdBeingBlocked** *required*	The id of the account to be blocked/unblocked	integer(int64)	
Query	**blockFlagValue** *optional*	Determines whether the account is blocked or unblocked	boolean	`"true"`
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string	
Query	**latitude** *optional*	The current latitude of the user	number(double)	
Query	**longitude** *optional*	The current longitude of the user	number(double)	
Query	**removeFromGroupsIfBlocked** *optional*	Determines whether the account is removed from all other groups if blocked	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.2.3. Create Account

```
POST /api/{version}/account/create
```

Description

Create a new account by role.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API Version	number(double)	
Query	**aboutUs** *optional*	Information about the user	string	

Type	Name	Description	Schema	Default
Query	**acceptedTerms** *optional*	Says whether the user has accepted the app's terms of service	boolean	`"true"`
Query	**appKey** *optional*	The application key	string	
Query	**appVersion** *optional*	The application version	string	
Query	**assetId** *optional*	The asset id to set the user's profile image	integer(int64)	
Query	**audienceIdsToAdd** *optional*	Comma separated list of audience ids to assign to the user	string	
Query	**birthday** *optional*	The birthday date of the user in milliseconds	integer(int64)	
Query	**bodyType** *optional*	The user's body type	string	
Query	**businessPhone** *optional*	The business phone number	string	
Query	**categoryIds** *optional*	A list of category ids that represent interests and associations	string	
Query	**cellPhone** *optional*	The cell phone number	string	
Query	**cellPhoneCarrier** *optional*	The cell phone carrier	string	
Query	**children** *optional*	The user's children status	string	
Query	**companionship** *optional*	The user's companionship status	string	
Query	**companionshipIndex** *optional*	The user's companionship index	integer(int32)	
Query	**deviceId** *optional*	The ID of the device	string	

Type	Name	Description	Schema	Default
Query	**deviceIdType** *optional*	The ID of the device type	string	
Query	**drink** *optional*	The user’s drinking status	string	
Query	**education** *optional*	The user’s education	string	
Query	**educationIndex** *optional*	The user’s education in a numerical value that can be used for ordering/searching	integer(int32)	
Query	**emailAddress** *optional*	The user’s contact email address (NOT the username) which is also used for email validation	string	
Query	**ethnicity** *optional*	The user’s ethnicity	string	
Query	**firstName** *optional*	If the parameter 'name' is empty or not present, this field will be used to set the user’s first name	string	
Query	**gameExperience** *optional*	Game experience level of the user {ANY, NEW, BEGINNER, INTERMEDIATE, EXPERT}	string	
Query	**gameType** *optional*	Deprecated: use appKey	string	
Query	**gender** *optional*	The gender of the user AudienceGender	string	
Query	**height** *optional*	The user’s height	string	
Query	**heightIndex** *optional*	The user’s height in a numerical value that can be used for ordering/searching	integer(int32)	
Query	**homePhone** *optional*	The home phone number	string	
Query	**hometown** *optional*	The user’s hometown	string	
Query	**inviteToken** *optional*	The inviteToken that the referrer use for this account to sign up	string	
Query	**lastName** *optional*	If the parameter 'name' is empty or not present, this field will be used to set the user’s last name	string	

Type	Name	Description	Schema	Default
Query	**latitude** *optional*	The current latitude of the user	number(double)	
Query	**longitude** *optional*	The current longitude of the user	number(double)	
Query	**maritalStatus** *optional*	The user's maritial status	string	
Query	**middleName** *optional*	If the parameter 'name' is empty or not present, this field will be used to set the user's middle name	string	
Query	**name** *optional*	The full name of the user. If this parameter is not empty, the following parameters will be ignored. prefixName, firstName, middleName, lastName, and suffixName	string	
Query	**password** *required*	The password of the user	string	
Query	**platforms** *optional*	Comma separated list of development platforms {MAC, WINDOWS, IOS, ANDROID, WINDOWSPHONE, KINDLE, UNITY3D, COCOS2D, HTML5, FACEBOOK}	string	
Query	**preferredBodyType** *optional*	The preferred body type in the account location search	string	
Query	**preferredEducation** *optional*	The preferred education in the account location search	string	
Query	**preferredEducationIndex** *optional*	The preferred education in a numerical value that can be used for ordering/searching	integer(int32)	
Query	**preferredEthnicity** *optional*	The preferred ethnicity in the account location search	string	
Query	**preferredGender** *optional*	The preferred gender in the account location search	string	
Query	**preferredLocation** *optional*	The preferred education in the account location search	string	

Type	Name	Description	Schema	Default
Query	**preferredLocationRange** *optional*	The preferred location range in the account location search	number(double)	
Query	**preferredMaxAge** *optional*	The preferred maximum age in the account location search	integer(int32)	
Query	**preferredMaxHeight** *optional*	The preferred maximum height in the account location search	integer(int32)	
Query	**preferredMinAge** *optional*	The preferred minimum age in the account location search	integer(int32)	
Query	**preferredMinHeight** *optional*	The preferred minimum height in the account location search	integer(int32)	
Query	**prefixName** *optional*	If the parameter 'name' is empty or not present, this field will be used to set the prefix of the user's name	string	
Query	**religion** *optional*	The user's reglion	string	
Query	**responseType** *optional*	Returns an AccountLoginResponse if AccountLoginResponse is passed in	string	
Query	**role** *optional*	The type of account being created {RETAILER, MEMBER, DEVELOPER, GUEST}	string	`"MEMBER"`
Query	**sendValidation** *optional*	Determines whether to send the user validation emails	boolean	
Query	**smoke** *optional*	The user's smoking status	string	
Query	**streetAddress** *optional*	The street address of the user's contact location	string	
Query	**suffixName** *optional*	If the parameter 'name' is empty or not present, this field will be used to set the suffix of the user's name	string	
Query	**tags** *optional*	Search tags	string	

Type	Name	Description	Schema	Default
Query	**title** *optional*	This field will be used to set the user's job title	string	
Query	**username** *required*	The user's username that will be used when they login	string	
Query	**zipcode** *optional*	The street zipcode of the user's contact location	string	

Responses

HTTP Code	Description	Schema
200	successful operation	AccountLoginResponse

6.2.4. Create Account (Encrypted Username)

```
POST /api/{version}/account/create/validate
```

Description

Create a new account by role (with encrypted user-name and password)

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The version	number(double)	
Query	**aboutUs** *optional*	About Us	string	
Query	**acceptedTerms** *optional*	Says whether the user has accepted the app's terms of service	boolean	`"true"`
Query	**address** *optional*	Address	string	
Query	**appKey** *optional*	The application key	string	
Query	**appVersion** *optional*	The application version	string	

Type	Name	Description	Schema	Default
Query	**assetId** *optional*	The asset id to set the user's profile image	integer(int64)	
Query	**audienceIdsToAdd** *optional*	Comma separated list of audience ids to assign to the user	string	
Query	**birthday** *optional*	The birthday date of the user in milliseconds	integer(int64)	
Query	**bodyType** *optional*	The user's body type	string	
Query	**businessPhone** *optional*	Business Phone	string	
Query	**categoryIds** *optional*	A list of category ids that represent interests and associations	string	
Query	**cellPhone** *optional*	Cell Phone	string	
Query	**cellPhoneCarrier** *optional*	Cell Phone Carrier	string	
Query	**charsetName** *optional*	Charset Name	string	`"UTF-8"`
Query	**children** *optional*	The user's children status	string	
Query	**companionship** *optional*	The user's companionship status	string	
Query	**companionshipIndex** *optional*	The user's companionship index	integer(int32)	
Query	**deviceId** *required*	The device id	string	
Query	**deviceIdType** *optional*	Device Id Type	string	
Query	**drink** *optional*	The user's drink status	string	
Query	**education** *optional*	The user's education	string	

Type	Name	Description	Schema	Default
Query	**educationIndex** *optional*	The user's education in a numerical value that can be used for ordering/searching	integer(int32)	
Query	**emailAddress** *optional*	The user's contact email address (NOT the username) which is also used for email validation	string	
Query	**ethnicity** *optional*	The user's ethnicity	string	
Query	**firstName** *optional*	If the parameter 'name' is empty or not present, this field will be used to set the user's first name	string	
Query	**gameExperience** *optional*	Game experience level of the user {ANY, NEW, BEGINNER, INTERMEDIATE, EXPERT	string	
Query	**gameType** *optional*	Deprecated: use appKey	string	
Query	**gender** *optional*	The gender of the user AudienceGender	string	
Query	**height** *optional*	The user's height	string	
Query	**heightIndex** *optional*	The user's height in a numerical value that can be used for ordering/searching	integer(int32)	
Query	**homePhone** *optional*	Home Phone	string	
Query	**hometown** *optional*	The user's hometown	string	
Query	**inviteToken** *optional*	The inviteToken that the referrer use for this account to sign up	string	
Query	**lastName** *optional*	If the parameter 'name' is empty or not present, this field will be used to set the user's last name	string	
Query	**latitude** *optional*	The current latitude of the user	number(double)	
Query	**longitude** *optional*	The current longitude of the user	number(double)	

Type	Name	Description	Schema	Default
Query	**maritalStatus** *optional*	The user's maritial status	string	
Query	**middleName** *optional*	If the parameter 'name' is empty or not present, this field will be used to set the user's middle name	string	
Query	**name** *optional*	The full name of the user. If this parameter is not empty, the following parameters will be ignored: prefixName, firstName, middleName, lastName, and suffixName	string	
Query	**password** *required*	The encrypted password of the user	string	
Query	**platforms** *optional*	Comma separated list of development platforms {MAC, WINDOWS, IOS, ANDROID, WINDOWSPHONE, KINDLE, UNITY3D, COCOS2D, HTML5, FACEBOOK	string	
Query	**preferredBodyType** *optional*	The preferred body type in the account location search	string	
Query	**preferredEducation** *optional*	The preferred education in the account location search	string	
Query	**preferredEducationIndex** *optional*	The preferred education in a numerical value that can be used for ordering/searching	integer(int32)	
Query	**preferredEthnicity** *optional*	The preferred ethnicity in the account location search	string	
Query	**preferredGender** *optional*	The preferred gender in the account location search	string	
Query	**preferredLocation** *optional*	The preferred education in the account location search	string	
Query	**preferredLocationRange** *optional*	The preferred location range in the account location search	number(double)	

Type	Name	Description	Schema	Default
Query	**preferredMaxAge** *optional*	The preferred maximum age in the account location search	integer(int32)	
Query	**preferredMaxHeight** *optional*	The preferred maximum height in the account location search	integer(int32)	
Query	**preferredMinAge** *optional*	The preferred minimum age in the account location search	integer(int32)	
Query	**preferredMinHeight** *optional*	The preferred minimum height in the account location search	integer(int32)	
Query	**prefixName** *optional*	If the parameter 'name' is empty or not present, this field will be used to set the prefix of the user's name	string	
Query	**religion** *optional*	The user's religion	string	
Query	**responseType** *optional*	Returns an AccountLoginResponse if AccountLoginResponse is passed in	string	
Query	**role** *optional*	The type of account being created {RETAILER, MEMBER, DEVELOPER, GUEST	string	`"MEMBER"`
Query	**smoke** *optional*	The user's smoke status	string	
Query	**suffixName** *optional*	If the parameter 'name' is empty or not present, this field will be used to set the suffix of the user's name	string	
Query	**tags** *optional*	Search tags	string	
Query	**title** *optional*	Title	string	
Query	**username** *required*	The encrypted email of the user, this is what will be used when they login	string	
Query	**zipcode** *optional*	The street zipcode of the user's contact location	string	

Responses

HTTP Code	Description	Schema
200	successful operation	ProfileInfoResponse

6.2.5. Login Account (Username)

```
POST /api/{version}/account/get
```

Description

Login to system with an account

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version	number(double)	
Query	**app** *optional*	Deprecated: use appKey	string	
Query	**appKey** *optional*	The application key	string	
Query	**deviceId** *optional*	The device id	string	
Query	**gameType** *optional*	Deprecated: use appKey	string	
Query	**latitude** *optional*	The current latitude of the user	number(double)	
Query	**longitude** *optional*	The current latitude of the user	number(double)	
Query	**password** *required*	The user's password	string	
Query	**responseFilters** *optional*	A comma separated list of values for filtering the returned user's profile response data	string	`"PROFILE"`
Query	**returnProfile** *optional*	Determines whether to return the user's profile response.	boolean	`"false"`
Query	**username** *required*	The user's email address they used to sign-up	string	

Responses

HTTP Code	Description	Schema
200	successful operation	ProfileResponse

6.2.6. Update Location

```
POST /api/{version}/account/location/update
```

Description

Update the account location

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**appKey** *optional*	The application key	string
Query	**clientTime** *optional*	The user's current time	integer(int64)
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string
Query	**latitude** *optional*	The current latitude of the user	number(double)
Query	**locationDescription** *optional*	The user's location description	string
Query	**longitude** *optional*	The current longitude of the user	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.2.7. Login Account

```
POST /api/{version}/account/login
```

Description

General login service that supports various authentication methods. Currently supports Facebook, Twitter, Sirqul Username, and Sirqul Phone by default. Can also support custom networks created using the {@link ThirdPartyApi}

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version to use to make the request	number(double)	
Query	**accessToken** *required*	The access token to authenticate with (ex: username or fb token)	string	
Query	**accessTokenSecret** *optional*	The secret to authenticate with (ex: password)	string	
Query	**ageRestriction** *optional*	Checks user's birthday to see if they meet an age requirement. If the user is under age, an error message is returned.	integer(int32)	`"0"`
Query	**appKey** *required*	The application key	string	
Query	**chosenAccountId** *optional*	Pass in the 2nd request to choose an account from multiple accounts matching the email, use one of the account id from the previous request	integer(int64)	
Query	**deviceId** *optional*	The unique id of the device making the request	string	
Query	**deviceIdType** *optional*	The type of device id (this is defined by the client), ie. MAC_ADDRESS, APPLE_UDID, etc	string	
Query	**emailMatch** *required*	When no accounts match a username, this determines whether passed in username as email and search the accounts matching that email, if multiple accounts matching the email is found, return list of account and a ThirdPartyCredential object	boolean	

Type	Name	Description	Schema	Default
Query	**latitude** *optional*	Used to update the user's current location	number(double)	
Query	**longitude** *optional*	Used to update the user's current location	number(double)	
Query	**networkUID** *required*	The access provider to authenticate against. This can be custom networks created using the ThirdPartyApi as well. Supported values by default include: FACEBOOK, TWITTER, USERNAME, PHONE	string	`"USERNAME"`
Query	**responseFilters** *optional*	This determines how much of the profile should be returned, see ProfileFilters	string	`"PROFILE"`
Query	**thirdPartyCredentialId** *optional*	Pass in the 2nd request to choose an account from multiple accounts matching the email, use the id from the previous ThirdPartyCredential object call	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	ProfileResponse

6.2.8. Login as Account via Admin

```
POST /api/{version}/account/login/delegate
```

Description

A login service that supports logging in as someone else (accounts that the user manages). Intended for internal use for now.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version	number(double)	
Query	**accessToken** *required*	The access token to authenticate with (ex: username or fb token)	string	

Type	Name	Description	Schema	Default
Query	**accessTokenSecret** *optional*	The secret to authenticate with (ex: password)	string	
Query	**ageRestriction** *required*	Checks user's birthday to see if they meet an age requirement. If the user is under age, an error message is returned.	integer(int32)	`"0"`
Query	**appKey** *required*	The application key	string	
Query	**delegatedAccountId** *optional*	The account id to login with	integer(int64)	
Query	**delegatedUsername** *optional*	The account to login with	string	
Query	**deviceId** *optional*	The unique id of the device making the request	string	
Query	**latitude** *optional*	Used to update the user's current location	number(double)	
Query	**longitude** *optional*	Used to update the user's current location	number(double)	
Query	**networkUID** *required*	The network UID	string	`"USERNAME"`
Query	**responseFilters** *required*	A comma separated list of values for filtering the returned response data.	string	`"PROFILE"`

Responses

HTTP Code	Description	Schema
200	successful operation	ProfileResponse

6.2.9. Login Account (Encrypted Username)

```
POST /api/{version}/account/login/validate
```

Description

ogin with encrypted user-name and password.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version	number(double)	
Query	**charsetName** *optional*	Charset Name	string	`"UTF-8"`
Query	**deviceId** *optional*	The device id	string	
Query	**gameType** *required*	The application key	string	
Query	**latitude** *optional*	The current latitude of the user	number(double)	
Query	**longitude** *optional*	The current longitude of the user	number(double)	
Query	**password** *required*	The encrypted password	string	
Query	**responseFilters** *optional*	A comma separated list of ProfileFilters for filtering the returned response data	string	`"PROFILE"`
Query	**returnProfile** *optional*	Return Profile	boolean	`"false"`
Query	**username** *required*	The user's encrypted email address they used to sign-up	string	

Responses

HTTP Code	Description	Schema
200	successful operation	ProfileResponse

6.2.10. Logout Account

```
POST /api/{version}/account/logout
```

Description

Cleans up the users data for logging out.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version	number(double)
Query	**accountId** *optional*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**deviceId** *required*	The device id (deviceId or accountId required)	string
Query	**deviceIdType** *optional*	Device Id Type	string
Query	**latitude** *optional*	The current latitude of the user	number(double)
Query	**longitude** *optional*	The current longitude of the user	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.2.11. Update Password

```
POST /api/{version}/account/passwordchange
```

Description

Update the account password.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use	number(double)

Type	Name	Description	Schema
Query	**accountId** *required*	The account to update	integer(int64)
Query	**confirmPassword** *required*	The new password to confirm, must match newPassword	string
Query	**newPassword** *required*	The new password to set, cannot be empty	string
Query	**oldPassword** *required*	The current password, used to validate access	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.2.12. Reset Password

```
POST /api/{version}/account/passwordreset
```

Description

Reset the account password. The token must be valid and not expired. Use the RequestPasswordReset end point to request a token.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use	number(double)
Query	**confirm** *required*	The new password to confirm, must match newPassword	string
Query	**password** *required*	The new password to set, cannot be empty	string
Query	**token** *required*	The token associated with the account to update, good for 24 hours	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.2.13. Get Profile Assets

```
GET /api/{version}/account/profile/assets
```

Description

Get a list of assets a person has ever uploaded. Filters the list based on parameters.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	Version	number(double)	
Query	**_i** *optional*	_i	integer(int32)	
Query	**_l** *optional*	_l	integer(int32)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**descending** *optional*	Determines whether to return the resulting list in descending or ascending order	boolean	`"true"`
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string	
Query	**latitude** *optional*	Latitude used to update the user's current location	number(double)	
Query	**limit** *optional*	Limit of the pagination	integer(int32)	`"0"`
Query	**longitude** *optional*	Longitude used to update the user's current location	number(double)	
Query	**mediaTypes** *optional*	Comma separated list of MediaType	string	
Query	**mimeTypes** *optional*	Comma separated list of mime types	string	

Type	Name	Description	Schema	Default
Query	**ownerId** *optional*	The account id of the person the user wants to view	integer(int64)	
Query	**returnNulls** *optional*	Determines whether to return null fields in the response	boolean	`"1"`
Query	**sortField** *optional*	Determines what the returning list will be sorted by (see AssetApiMap)	string	`"CREATED"`
Query	**start** *optional*	Start of the pagination	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	AssetListResponse

6.2.14. Get Account

```
GET /api/{version}/account/profile/get
```

Description

Gets a user's account profile. Application settings and account settings will also be returned for the owner of the account.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version	number(double)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**appKey** *optional*	The application key	string	
Query	**connectionAccountEmail** *optional*	Connection Account Email	string	
Query	**connectionAccountId** *optional*	The account id used to view another person's account	integer(int64)	`"0"`

Type	Name	Description	Schema	Default
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string	
Query	**gameType** *optional*	Game Type	string	
Query	**latitude** *optional*	Latitude used to update the user's current location	number(double)	
Query	**longitude** *optional*	Longitude used to update the user's current location	number(double)	
Query	**purchaseType** *optional*	Purchase Type	string	`"SIRQUL"`
Query	**responseFilters** *optional*	A comma separated list of ProfileFilters for filtering the returned response data	string	`"PROFILE"`
Query	**returnNulls** *optional*	Return Nulls	boolean	`"1"`
Query	**updateViewedDate** *optional*	Determines whether to track if a person has viewed someone's profile	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	ProfileResponse

6.2.15. Search Accounts

```
GET /api/{version}/account/profile/search
```

Description

Search for account profiles.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The Sirqul API version	number(double)	

Type	Name	Description	Schema	Default
Query	**accountId** *required*	The id of the account requesting	integer(int64)	
Query	**activeOnly** *optional*	Determines whether to return only active results. Default is false.	boolean	`"false"`
Query	**age** *optional*	Age	integer(int32)	
Query	**appKey** *required*	The application key	string	
Query	**categoryIds** *optional*	Category Ids	string	
Query	**descending** *optional*	The order to return the results. Default is false, which will return the results in ascending order.	boolean	`"false"`
Query	**emailAddress** *optional*	For querying the accounts by their email address	string	
Query	**flagCountMinimum** *optional*	Return any results that have a minimum of the specified flag count (even ones that have met the flag threshold)	integer(int64)	`"false"`
Query	**gameExperience** *optional*	Game Experience	string	
Query	**gender** *optional*	Gender	enum (MALE, FEMALE, ANY)	
Query	**keyword** *optional*	The keyword for querying the accounts by their display name	string	
Query	**latitude** *optional*	Latitude	number(double)	
Query	**limit** *optional*	The total number of record to return.	integer(int32)	`"20"`
Query	**longitude** *optional*	Longitude	number(double)	
Query	**purchaseType** *optional*	A comma separated list of PurchaseType	string	`"SIRQUL"`
Query	**radius** *optional*	Radius	number(double)	`"5.0"`

Type	Name	Description	Schema	Default
Query	**responseFilters** *optional*	A comma separated list of ProfileFilters for filtering the returned response data	string	`"PROFILE"`
Query	**returnNulls** *optional*	Return Nulls	boolean	`"true"`
Query	**roles** *optional*	Roles	string	
Query	**sortField** *optional*	The field to sort by	string	`"id"`
Query	**start** *optional*	The index into the record set to start with.	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< ProfileResponse > array

6.2.16. Update Account

```
POST /api/{version}/account/profile/update
```

Description

Edit the user's profile information

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version	number(double)	
Query	**aboutUs** *optional*	About Us	string	
Query	**acceptedTerms** *optional*	Accepted Terms	boolean	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	

Type	Name	Description	Schema	Default
Query	**age** *optional*	This is deperecated, use the birthday parameter	integer(int32)	
Query	**appBlob** *optional*	App Blob	string	
Query	**appEnableEmail** *optional*	App Enable Email	boolean	
Query	**appEnablePush** *optional*	App Enable Push	boolean	
Query	**appEnableSMS** *optional*	App Enable SMS	boolean	
Query	**appKey** *optional*	The application key	string	
Query	**assetId** *optional*	The asset id to set the user's profile image	integer(int64)	
Query	**audienceIdsToAdd** *optional*	Audience Ids to add	string	
Query	**audienceIdsToRemove** *optional*	Audience Ids to remove	string	
Query	**birthday** *optional*	The birthday date of the user in UTC milliseconds	integer(int64)	
Query	**bodyType** *optional*	The user's body type	string	
Query	**businessName** *optional*	Business Name	string	
Query	**businessPhone** *optional*	The business phone number	string	
Query	**categories** *optional*	Deprecated use categoryIds	string	
Query	**categoryIds** *optional*	A list of category ids that represent interests and associations	string	

Type	Name	Description	Schema	Default
Query	**cellPhone** *optional*	The cellular phone number	string	
Query	**cellPhoneCarrier** *optional*	The cellular service provider	string	
Query	**children** *optional*	The user's children status	string	
Query	**city** *optional*	The city of the user's contact location	string	
Query	**companionship** *optional*	The user's companionship status	string	
Query	**companionshipIndex** *optional*	The user's companionship index	integer(int32)	
Query	**connectionAccountId** *optional*	The account id used to edit another person's account	integer(int64)	
Query	**country** *optional*	The country of the user's contact location	string	
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string	
Query	**drink** *optional*	The user's drink status	string	
Query	**education** *optional*	The user's education	string	
Query	**educationIndex** *optional*	The user's education in a numerical value that can be used for ordering/searching	integer(int32)	
Query	**emailAddress** *optional*	The user's contact email address (NOT the username)	string	
Query	**ethnicity** *optional*	The user's ethnicity	string	
Query	**firstName** *optional*	If the parameter 'name' is empty or not present, this field will be used to set the user's first name	string	

Type	Name	Description	Schema	Default
Query	**gameExperience** *optional*	Game Experience	string	
Query	**gameType** *optional*	Game Type	string	
Query	**gender** *optional*	The gender of the user AudienceGender	string	
Query	**height** *optional*	The user's height	string	
Query	**heightIndex** *optional*	The user's height in a numerical value that can be used for ordering/searching	integer(int32)	
Query	**homeLatitude** *optional*	The home latitude of the user	number(double)	
Query	**homeLongitude** *optional*	The home longitude of the user	number(double)	
Query	**homePhone** *optional*	The home phone number	string	
Query	**hometown** *optional*	The user's hometown	string	
Query	**lastName** *optional*	If the parameter 'name' is empty or not present, this field will be used to set the user's last name	string	
Query	**latitude** *optional*	The current latitude of the user	number(double)	
Query	**locationVisibility** *optional*	Location Visibility	string	
Query	**longitude** *optional*	The current longitude of the user	number(double)	
Query	**makeFriendsInfoPublic** *optional*	Allow anyone to view the user's friends list	boolean	
Query	**makeGameInfoPublic** *optional*	Allow anyone to view the user's game/app info	boolean	

Type	Name	Description	Schema	Default
Query	**makeProfileInfoPublic** *optional*	Allow anyone to view the user's personal profile	boolean	
Query	**maritalStatus** *optional*	The user's marital status	string	
Query	**matchToken** *optional*	Match Token	string	
Query	**middleName** *optional*	If the parameter 'name' is empty or not present, this field will be used to set the user's middle name	string	
Query	**name** *optional*	The full name of the user. If this parameter is NOT empty, the following parameters will be ignored: prefixName, firstName, middleName, lastName, and suffixName	string	
Query	**platforms** *optional*	Platforms	string	
Query	**preferredBodyType** *optional*	The preferred body type in the account location search	string	
Query	**preferredEducation** *optional*	The preferred education in the account location search	string	
Query	**preferredEducationIndex** *optional*	The preferred education in a numerical value that can be used for ordering/searching	integer(int32)	
Query	**preferredEthnicity** *optional*	The preferred ethnicity in the account location search	string	
Query	**preferredGender** *optional*	The preferred gender in the account location search	string	
Query	**preferredLocation** *optional*	The preferred education in the account location search	string	
Query	**preferredLocationRange** *optional*	The preferred location range in the account location search	number(double)	

Type	Name	Description	Schema	Default
Query	**preferredMaxAge** *optional*	The preferred maximum age in the account location search	integer(int32)	
Query	**preferredMaxHeight** *optional*	The preferred maximum height in the account location search	integer(int32)	
Query	**preferredMinAge** *optional*	The preferred minimum age in the account location search	integer(int32)	
Query	**preferredMinHeight** *optional*	The preferred minimum height in the account location search	integer(int32)	
Query	**prefixName** *optional*	If the parameter 'name' is empty or not present, this field will be used to set the prefix of the user's name	string	
Query	**religion** *optional*	The user's religion	string	
Query	**responseFilters** *optional*	A comma separated list of ProfileFilters for filtering the returned response data	string	`"PROFILE,PROFILE_CONTACT"`
Query	**returnProfile** *optional*	Determine whether to return a ProfileInfoResponse	boolean	
Query	**role** *optional*	The account role to change to	string	
Query	**showAsZipcode** *optional*	The user's preference if they want to be shown by zipcode on a map	boolean	
Query	**showExactLocation** *optional*	The user's preference if they want to be shown by their exact location on a map	boolean	
Query	**showOthersExactLocation** *optional*	The user's preference if they want to see others exact location on a map	boolean	
Query	**smoke** *optional*	The user's smoke status	string	
Query	**state** *optional*	The state of the user's contact location	string	

Type	Name	Description	Schema	Default
Query	**streetAddress** *optional*	The street address of the user's contact location	string	
Query	**streetAddress2** *optional*	Additional address information (such as a suite number, floor number, building name, or PO Box)	string	
Query	**suffixName** *optional*	If the parameter 'name' is empty or not present, this field will be used to set the suffix of the user's name	string	
Query	**tags** *optional*	Tags	string	
Query	**title** *optional*	This field will be used to set the user's job title	string	
Query	**zipcode** *optional*	The zipcode of the user's contact location	string	

Responses

HTTP Code	Description	Schema
200	successful operation	ProfileInfoResponse

6.2.17. Get Account Referral List

```
GET /api/{version}/account/referral/list
```

Description

Gets a user's account profile and their referral List

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version	number(double)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	

Type	Name	Description	Schema	Default
Query	**ancestorLevelLimit** *optional*	Ancestor Level Limit	integer(int32)	
Query	**ancestorListLimit** *optional*	Ancestor list limit	integer(int32)	
Query	**ancestorListStart** *optional*	Ancestor list start	integer(int32)	
Query	**appKey** *optional*	The application key	string	
Query	**childrenChildren** *optional*	Children's children	boolean	
Query	**childrenLevelLimit** *optional*	Children Level Limit	integer(int32)	
Query	**childrenListLimit** *optional*	Children list limit	integer(int32)	
Query	**childrenListStart** *optional*	Children list start	integer(int32)	
Query	**levelLimit** *required*	Level Limit	integer(int32)	
Query	**retrieveType** *optional*	GET_CHILDREN/GET_ANCESTOR/GET_ALL	string	`"GET_ALL"`

Responses

HTTP Code	Description	Schema
200	successful operation	ProfileShortResponse

6.2.18. Send Validation Request

```
POST /api/{version}/account/requestValidateAccount
```

Description

Send an email to validate a user's account.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version	number(double)
Query	**accountId** *required*	The account id of the user	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.2.19. Request Password Reset

```
POST /api/{version}/account/requestpasswordreset
```

Description

Request that an account password be reset. The account is looked up by email address and then a link is sent via email to that account with a reset token. The token is valid for 24 hours.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	Version	number(double)	
Query	**domain** *optional*	Domain	string	
Query	**email** *required*	The email/username of the account	string	
Query	**from** *optional*	From	string	`"Sirqul"`

Type	Name	Description	Schema	Default
Query	**referer** *optional*	Referer	string	`"http://dev.sirqul.com/resetpassword"`
Query	**subUrl** *optional*	Sub Url	string	

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.2.20. Search Accounts by Location

```
GET /api/{version}/account/search
```

Description

Search accounts by their location. This only searches on users that have location data. Use ConnectionApi to perform a regular search on accounts.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API Version	number(double)	
Query	**_i** *optional*	Deprecated: use start	integer(int32)	
Query	**_l** *optional*	Deprecated: use limit	integer(int32)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**appKey** *optional*	The application key	string	
Query	**audienceIds** *optional*	The audience ids to filter on (comma separated)	string	

Type	Name	Description	Schema	Default
Query	**audienceOperator** *required*	Determines the operator on the audienceIds list	enum (AND, OR)	`"AND"`
Query	**categoryIds** *optional*	The category ids to filter on (comma separated)	string	
Query	**companionshipIndex** *optional*	The preferred companionship index	integer(int32)	
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string	
Query	**experience** *optional*	The experience to filter on	string	
Query	**filter** *optional*	Comma separated list of ConnectionApiMap, to filter results	string	
Query	**flagCountMinimum** *optional*	Filters accounts that have a minimum number of flags	integer(int64)	
Query	**gender** *optional*	The preferred gender	string	
Query	**keyword** *optional*	An optional keyword to search on, will be ignore if empty	string	
Query	**latitude** *optional*	The latitude of the user	number(double)	
Query	**limit** *optional*	The limit of the pagination	integer(int32)	`"100"`
Query	**locationLastUpdated** *optional*	Searches for user's that has updated their location since this date	integer(int64)	
Query	**longitude** *optional*	The longitude of the user	number(double)	
Query	**maxAge** *optional*	The preferred max age	integer(int32)	
Query	**minAge** *optional*	The preferred min age	integer(int32)	
Query	**postalCode** *optional*	The postal code to search on, either postalCode or the user's exact location is required	string	

Type	Name	Description	Schema	Default
Query	**q** *optional*	Deprecated: use keyword	string	
Query	**range** *optional*	The range to search on	number(double)	`"10.0"`
Query	**roles** *optional*	Filter results by account roles	string	
Query	**searchMode** *optional*	Determines whether to use CLOUDINDEX or RDS to search	enum (RDS, LUCENE, CLOUDINDEX)	
Query	**showConnectionToSearcher** *optional*	showConnectionToSearcher	boolean	`"false"`
Query	**showExactLocations** *optional*	Determines whether to return the user's exact location or not	boolean	`"false"`
Query	**start** *optional*	The starting index of the pagination	integer(int32)	`"0"`
Query	**tags** *optional*	Filter results account search tags	string	
Query	**updateCurrentLocation** *optional*	Determines whether to update the user's current location	boolean	`"false"`
Query	**updatePreferredSettings** *optional*	Determines whether to update the user's perferred settings	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	UserLocationSearchResponse

6.2.21. Get Account Settings

```
GET /api/{version}/account/settings/get
```

Description

Get the account settings for a user

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string
Query	**latitude** *optional*	The current latitude of the user	number(double)
Query	**longitude** *optional*	The current longitude of the user	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	UserSettingsResponse

6.2.22. Update Account Settings

```
POST /api/{version}/account/settings/update
```

Description

Update the account settings for a user

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)

Type	Name	Description	Schema
Query	**blockedNotifications** *optional*	The notifications to be blocked	string
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string
Query	**favoriteVisibility** *optional*	Favorite Visibility	string
Query	**latitude** *optional*	The current latitude of the user	number(double)
Query	**longitude** *optional*	The current longitude of the user	number(double)
Query	**showAsZipcode** *optional*	Show As Zipcode	boolean
Query	**showExactLocation** *optional*	Show Exact Location	boolean
Query	**showOthersExactLocation** *optional*	Show Others Exact Location	boolean
Query	**suggestionCount** *optional*	How many suggestions to receive per time frame	integer(int32)
Query	**suggestionMethod** *optional*	How suggestions are to be sent (APNS, MOBILE_NOTIFICATION, SMS)	string
Query	**suggestionTimeFrame** *optional*	The time frame in seconds, 3600 would be a 1 hour time frame	integer(int32)

Responses

HTTP Code	Description	Schema
200	successful operation	UserSettingsResponse

6.2.23. Update Account Username

```
POST /api/{version}/account/username/update
```

Description

Update account's own username and/or emailAddress

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**deviceId** *optional*	The device id	string
Query	**emailAddress** *optional*	The user's contact email address (NOT the username) which is also used for email validation	string
Query	**username** *optional*	The encrypted email of the user, this is what will be used when they login	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.2.24. Save Validation Status

```
POST /api/{version}/account/validateAccountSignup
```

Description

Validate the account's email address. The token must be valid and not expired. Use the RequestValidateAccount end point to request a new token.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use	number(double)
Query	**token** *required*	The token associated with the account to update, good for 24 hours	string

Responses

HTTP Code	Description	Schema
200	successful operation	AccountLoginResponse

6.2.25. Validate Password Reset Token

```
POST /api/{version}/account/validatepasswordreset
```

Description

Validate the password reset token. The token must be valid and not expired. Use the RequestPasswordReset end point to request a token. The user receives and email with the reset page, therefore it should be validated before bwing used to reset the password.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use	number(double)
Query	**token** *required*	The token associated with the account to update, good for 24 hours	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.2.26. Save Match Token

```
POST /api/{version}/consumer/profile/matchToken
```

Description

Save user's match token to be used for profile match making, (can also use the profile/edit endpoint)

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	Version	number(double)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**appKey** *optional*	The application key	string	
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string	
Query	**gameType** *optional*	Game Type	string	`"BOOPY"`
Query	**latitude** *optional*	The current latitude of the user	number(double)	
Query	**longitude** *optional*	The current longitude of the user	number(double)	
Query	**matchToken** *optional*	A string of numbers	string	

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.2.27. Validate Subscription

```
POST /api/{version}/subscription/validate
```

Description

Checks to see if a user still has premium access.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version	number(double)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**appKey** *optional*	The application key	string	
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string	
Query	**gameType** *optional*	Game Type	string	
Query	**latitude** *optional*	The current latitude of the user	number(double)	
Query	**longitude** *optional*	The current longitude of the user	number(double)	
Query	**responseFilters** *optional*	A comma separated list of ProfileFilters for filtering the returned response data	string	`"PROFILE"`

Responses

HTTP Code	Description	Schema
200	successful operation	ProfileResponse

6.3. Achievement

An achievement can be any thing or object that is highly compelling to users and can be rewarded to them after completing something. There are users who spend hours of time throughout the day on a single application just to earn these achievements. Going even further, users who go after these achievements want to earn them so that they can show off to their friends or community.

Below are some example scenarios as to how these services can be used:

- Retail: Customer purchases clothes at a retail store. Retailer rewards the customer with an achievement that touts them as ‘Customer of the month!’ for purchasing the most clothes during the month.
- Medical: Patient goes to their dentist for teeth to be cleaned. If patient returns next time with healthier teeth, they can be rewarded with an achievement and earn a discount on their next check-up.

- Real Estate: A sports fan is rewarded with the ‘Friendliest fan of the game!’ achievement for connecting with the most fans during the game.
- Automotive: Driver receives ‘Safest driver of the year!’ achievement from his insurance company for not getting into any accidents or disputes during the year.

Achievements are critical to keep your users engaged & interacting with your product offering. It’s important to understand what drives your users and what motivates them when using your product. Achievements can be used to analytically track the usage & time spent on your product and their progress on attempting to earn an achievement. By being able to quantify the progress and behavior of your users, it helps you to push out the content that really matters and what users find interesting. With doing so, substantially increasing the amount of time users spend on your application.

Below are services that could be combined with this service:

- Assets Service: Achievements can be in the form of a compelling graphical asset, such as a ‘badge’. Users can collect these badges based upon their actions in an application.
- Offer Service: Users who are rewarded with an achievement may also be granted with an offer as part of the accomplishment. Such as a coupon or voucher.
- Tickets Service: When an achievement is unlocked, the user may be granted with ‘tickets’, which is digital currency that can be used to purchase products in your application such as gift cards, in-app purchases, and even redeemable for cash.

6.3.1. Create Achievement

```
POST /api/{version}/achievement/create
```

Description

Updates an achievement (for developer/retailer use). User must have permissions to the application the achievement is created for.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**achievementType** *optional*	achievementType	string

Type	Name	Description	Schema
Query	**active** *optional*	active	boolean
Query	**analyticsTag** *optional*	analyticsTag	string
Query	**appKey** *required*	appKey	string
Query	**description** *optional*	description	string
Query	**deviceId** *optional*	deviceId	string
Query	**incrementRule** *optional*	incrementRule	string
Query	**limitPerDay** *optional*	limitPerDay	integer(int32)
Query	**maxIncrement** *optional*	maxIncrement	integer(int32)
Query	**metaData** *optional*	metaData	string
Query	**minIncrement** *optional*	minIncrement	integer(int32)
Query	**rankIncrement** *optional*	rankIncrement	integer(int32)
Query	**rankType** *optional*	rankType	string
Query	**title** *required*	title	string
Query	**triggerDefinition** *optional*	triggerDefinition	string
Query	**validate** *optional*	validate	boolean

Responses

HTTP Code	Description	Schema
200	successful operation	AchievementResponse

6.3.2. Delete Achievement

```
POST /api/{version}/achievement/delete
```

Description

Deletes an achievement (for developer/retailer use). User must have permissions to the application the achievement was created for.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**achievementId** *required*	achievementId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.3.3. Get Achievement

```
GET /api/{version}/achievement/get
```

Description

Get an achievement (for developer/retailer use). User must have permissions to the application the achievement is created for.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**achievementId** *required*	achievementId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**includeInactiveTiers** *required*	includeInactiveTiers	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	AchievementTierResponse

6.3.4. List Achievements

```
GET /api/{version}/achievement/list
```

Description

List achievements by billable.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	

Type	Name	Description	Schema	Default
Query	**achievementType** *optional*	achievementType	string	
Query	**activeOnly** *required*	activeOnly	boolean	`"true"`
Query	**appKey** *optional*	appKey	string	
Query	**descending** *required*	descending	boolean	`"false"`
Query	**deviceId** *optional*	deviceId	string	
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**rankType** *optional*	rankType	string	
Query	**sortField** *required*	sortField	enum (CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, TITLE, DESCRIPTION, ACHIEVEMENT_TYPE, RANK_TYPE, APPLICATION_ID, APPLICATION_NAME, BILLABLE_ID)	`"TITLE"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< AchievementShortResponse > array

6.3.5. Get Achievement Progress

```
GET /api/{version}/achievement/progress/get
```

Description

Gets a list of user achievements.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**achievementType** *optional*	achievementType	string	
Query	**appKey** *required*	appKey	string	
Query	**connectionAccountEmail** *optional*	connectionAccountEmail	string	
Query	**connectionAccountId** *optional*	connectionAccountId	integer(int64)	`"0"`
Query	**deviceId** *optional*	deviceId	string	
Query	**includeUndiscovered** *required*	includeUndiscovered	boolean	`"true"`
Query	**latitude** *optional*	latitude	number(double)	
Query	**longitude** *optional*	longitude	number(double)	
Query	**rankType** *optional*	rankType	string	
Query	**returnNulls** *required*	returnNulls	boolean	`"1"`

Responses

HTTP Code	Description	Schema
200	successful operation	< AchievementProgressResponse > array

6.3.6. Override Achievement Progress

```
POST /api/{version}/achievement/progress/override
```

Description

Allows an admin of an application to override a user's achievement progress.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**accumulativeCount** *optional*	accumulativeCount	integer(int64)
Query	**achievementId** *required*	achievementId	integer(int64)
Query	**bestMonthlyStreakCount** *optional*	bestMonthlyStreakCount	integer(int64)
Query	**bestStreakCount** *optional*	bestStreakCount	integer(int64)
Query	**bestWeeklyStreakCount** *optional*	bestWeeklyStreakCount	integer(int64)

Type	Name	Description	Schema
Query	**currentMonthlyStreakCount** *optional*	currentMonthlyStreakCount	integer(int64)
Query	**currentStreakCount** *optional*	currentStreakCount	integer(int64)
Query	**currentWeeklyStreakCount** *optional*	currentWeeklyStreakCount	integer(int64)
Query	**endDate** *optional*	endDate	integer(int64)
Query	**ownerAccountId** *required*	ownerAccountId	integer(int64)
Query	**startDate** *optional*	startDate	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.3.7. Update Achievement Progress

```
POST /api/{version}/achievement/progress/update
```

Description

Update user achievement progress.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)

Type	Name	Description	Schema
Query	**achievementId** *optional*	achievementId	integer(int64)
Query	**appKey** *required*	appKey	string
Query	**customId** *optional*	customId	integer(int64)
Query	**endDate** *optional*	endDate	integer(int64)
Query	**increment** *optional*	increment	integer(int64)
Query	**returnProgress** *optional*	returnProgress	boolean
Query	**startDate** *optional*	startDate	integer(int64)
Query	**tag** *optional*	tag	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.3.8. Search Achievements

```
GET /api/{version}/achievement/search
```

Description

Searches achievements by application for consumers.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	

Type	Name	Description	Schema	Default
Query	**accountId** *required*	accountId	integer(int64)	
Query	**achievementType** *optional*	achievementType	string	
Query	**appKey** *required*	appKey	string	
Query	**descending** *required*	descending	boolean	`"false"`
Query	**deviceId** *optional*	deviceId	string	
Query	**includeInactiveTiers** *required*	includeInactiveTiers	boolean	`"false"`
Query	**includeTiers** *required*	includeTiers	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"100"`
Query	**rankType** *optional*	rankType	string	
Query	**sortField** *required*	sortField	enum (CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, TITLE, DESCRIPTION, ACHIEVEMENT_TYPE, RANK_TYPE, APPLICATION_ID, APPLICATION_NAME, BILLABLE_ID)	`"TITLE"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< AchievementShortResponse > array

6.3.9. List Achievement Tags

```
GET /api/{version}/achievement/tag/list
```

Description

List achievement tags by application

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**appKey** *optional*	appKey	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.3.10. Create Achievement Tier

```
POST /api/{version}/achievement/tier/create
```

Description

Create a tier of an achievement (for developer/retailer use). User must have permissions to the application the achievement is created for.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**achievementId** *required*	achievementId	integer(int64)	
Query	**description** *optional*	description	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**gameId** *optional*	gameId	integer(int64)	
Query	**gameLevelId** *optional*	gameLevelId	integer(int64)	
Query	**gameObjectId** *optional*	gameObjectId	integer(int64)	
Query	**goalCount** *optional*	goalCount	integer(int64)	
Query	**groupAlbumId** *optional*	groupAlbumId	integer(int64)	
Query	**icon** *optional*	icon	ref	
Query	**iconAssetId** *optional*	iconAssetId	integer(int64)	
Query	**metaData** *optional*	metaData	string	
Query	**missionId** *optional*	missionId	integer(int64)	
Query	**packId** *optional*	packId	integer(int64)	
Query	**scoreAllInstances** *required*	scoreAllInstances	boolean	`"true"`

Type	Name	Description	Schema	Default
Query	**tierType** *required*	tierType	enum (ACCUMULATIVE, CONSECUTIVE, CONSECUTIVE_WEEKLY, CONSECUTIVE_MONTHLY)	`"ACCUMULATIVE"`
Query	**title** *optional*	title	string	

Responses

HTTP Code	Description	Schema
200	successful operation	AchievementTierResponse

6.3.11. Delete Achievement Tier

```
POST /api/{version}/achievement/tier/delete
```

Description

Deletes an achievement tier (for developer/retailer use). User must have permissions to the application the achievement was created for.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**achievementTierId** *required*	achievementTierId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.3.12. Get Achievement Tier

```
GET /api/{version}/achievement/tier/get
```

Description

Gets an achievement tier (for developer/retailer use). User must have permissions to the application the achievement was created for.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**achievementTierId** *required*	achievementTierId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.3.13. Search Achievement Tiers

```
GET /api/{version}/achievement/tier/search
```

Description

Searches achievement tiers by application.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**achievementType** *optional*	achievementType	string	
Query	**activeOnly** *optional*	activeOnly	boolean	
Query	**appKey** *required*	appKey	string	
Query	**descending** *required*	descending	boolean	`"false"`
Query	**descendingGoal** *required*	descendingGoal	boolean	`"false"`
Query	**deviceId** *optional*	deviceId	string	
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"100"`
Query	**rankType** *optional*	rankType	string	
Query	**sortField** *required*	sortField	enum (CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, TITLE, DESCRIPTION, ACHIEVEMENT_TYPE, RANK_TYPE, APPLICATION_ID, APPLICATION_NAME, BILLABLE_ID)	`"TITLE"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< AchievementTierResponse > array

6.3.14. Update Achievement Tier

```
POST /api/{version}/achievement/tier/update
```

Description

Updates a tier of an achievement (for developer/retailer use). User must have permissions to the application the achievement is created for.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**achievementTierId** *required*	achievementTierId	integer(int64)
Query	**active** *optional*	active	boolean
Query	**description** *optional*	description	string
Query	**deviceId** *optional*	deviceId	string
Query	**gameId** *optional*	gameId	integer(int64)
Query	**gameLevelId** *optional*	gameLevelId	integer(int64)
Query	**gameObjectId** *optional*	gameObjectId	integer(int64)

Type	Name	Description	Schema
Query	**goalCount** *optional*	goalCount	integer(int64)
Query	**groupAlbumId** *optional*	groupAlbumId	integer(int64)
Query	**icon** *optional*	icon	ref
Query	**iconAssetId** *optional*	iconAssetId	integer(int64)
Query	**metaData** *optional*	metaData	string
Query	**missionId** *optional*	missionId	integer(int64)
Query	**packId** *optional*	packId	integer(int64)
Query	**scoreAllInstances** *optional*	scoreAllInstances	boolean
Query	**tierType** *optional*	tierType	enum (ACCUMULATIVE, CONSECUTIVE, CONSECUTIVE_WEEKLY, CONSECUTIVE_MONTHLY)
Query	**title** *optional*	title	string

Responses

HTTP Code	Description	Schema
200	successful operation	AchievementTierResponse

6.3.15. Update Achievement

```
POST /api/{version}/achievement/update
```

Description

Updates an achievement (for developer/retailer use). User must have permissions to the application the achievement is created for.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**achievementId** *optional*	achievementId	integer(int64)
Query	**achievementType** *optional*	achievementType	string
Query	**active** *optional*	active	boolean
Query	**analyticsTag** *optional*	analyticsTag	string
Query	**description** *optional*	description	string
Query	**deviceId** *optional*	deviceId	string
Query	**incrementRule** *optional*	incrementRule	string
Query	**limitPerDay** *optional*	limitPerDay	integer(int32)
Query	**maxIncrement** *optional*	maxIncrement	integer(int32)
Query	**metaData** *optional*	metaData	string
Query	**minIncrement** *optional*	minIncrement	integer(int32)

Type	Name	Description	Schema
Query	**nullLimitPerDay** *optional*	nullLimitPerDay	boolean
Query	**nullMaxIncrement** *optional*	nullMaxIncrement	boolean
Query	**nullMinIncrement** *optional*	nullMinIncrement	boolean
Query	**rankIncrement** *optional*	rankIncrement	integer(int32)
Query	**rankType** *optional*	rankType	string
Query	**title** *optional*	title	string
Query	**triggerDefinition** *optional*	triggerDefinition	string
Query	**validate** *optional*	validate	boolean

Responses

HTTP Code	Description	Schema
200	successful operation	AchievementResponse

6.4. Analytics

Analytics is a tool to help you understand how people interact with your application, what they do when they are using your application, and why they abandon the application. All of these elements are crucial to having a successful understanding of your users, customers, and target demographics. Analytics present a simple view of your application's performance without having to study lines and lines of log data.

Let's take the basics such as usage and bounce rate. Just because you built an application does not mean that people are downloading and installing it. And just because they run the application, does not mean they use it. Recording and understanding Visits and the behaviors of unique visitors can let

you know if people are getting the benefits of your services, which features are popular, and inform you as to whether you are capturing new audiences. The bounce rate will tell you if users launch your application and quickly leave it, identifying areas that you could improve on such as content or navigation.

Going even deeper, analytics can tell you who the people are that are interacting with your application. Storing behavior tied to a Social Media account is a vital component to understanding your users because these accounts usually have valuable meta-data associated with them that can later be combined with your own analytics to provide mineable, predictive visualizations that are attractive to partners, clients, and drive adoption and engagement by making sure you are focusing on the most important services vs. what you "think" will be popular. If you are running an ad campaign or what we call "missions" and "challenges", you and your client partners are going to want to know if it is working? Having a customizable way for application owners to embed analytics that are important to their own business and understanding of the customer is important and Sirqul provides the tools necessary for centrally distributed, searchable, minable, cross device, cross application analysis to occur within a simple-to-use web dashboard.

Companies "know" that they need mobile and desktop/laptop applications, and so they build them, but how many actually know how effective those apps are? How many thousands of dollars are spent on flashy graphics, layouts that display countless products or features, and enumerable services without knowing if they are influencing the customer in a positive way? Which features are actually bringing people to the application? Do your users share the app with friends? If so, which method is better: SMS, FB, Twitter, or URL sharing? Who are these users and how can you see aggregate behaviors? Is your application keeping the users engaged long enough and often enough to make a purchase? Sirqul's analytics engine can help answer all those questions and more.

Developers use the following endpoints as a means of "instrumenting" their code. In context of computer programming, instrumentation refers to an ability to monitor or measure the level of a product's performance, to diagnose errors and to write trace information. This endpoint is used to send in custom analytic "Tags" that can be used for this purpose. For example, when an application user leaves your application and it gets put into the "background", a developer can add a call to the Sirqul servers that record that event. Please review the following code snippet, paying attention to the following parameter: "tag=EnteredBackground". Once received by the Sirqul server, the custom analytic tag "EnteredBackground" that can be queried in the database by device, in aggregate, by application, by time, by operating system, by device type, by account, by location, by version, etc.

6.4.1. Get Aggregated Filtered Usage

```
POST /api/{version}/analytics/aggregatedFilteredUsage
```

Description

Query analytics to get data used for nested graphs and charts

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**_l** *optional*	_l	integer(int32)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**ageGroup** *optional*	ageGroup	string	
Query	**appKey** *optional*	appKey	string	
Query	**applicationId** *optional*	applicationId	integer(int64)	
Query	**city** *optional*	city	string	
Query	**country** *optional*	country	string	
Query	**descending** *optional*	descending	boolean	`"false"`
Query	**device** *optional*	device	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**deviceOS** *optional*	deviceOS	string	
Query	**deviceType** *optional*	deviceType	string	

Type	Name	Description	Schema	Default
Query	**distinctCount** *optional*	distinctCount	enum (TAG_COUNT, TAG, TOKEN, MODEL, DEVICE_TYPE, DEVICE, DEVICE_OS, DEVICE_ID, IP_ADDRESS, STATE, CITY, ZIP, COUNTRY, CREATED, UPDATED, LAST_UPDATED, CLIENT_TIME, ACTIVE, CUSTOM_ID, CUSTOM_TYPE, CUSTOM_VALUE, CUSTOM_VALUE2, CUSTOM_LONG, CUSTOM_LONG2, CUSTOM_MESSAGE, CUSTOM_MESSAGE2, ACCOUNT_ID, ACCOUNT_USERNAME, ACCOUNT_DISPLAY, ACCOUNT_CREATED, ACCOUNT_GENDER, ACCOUNT_AGE_GROUP, APPLICATION_ID, APPLICATION_KEY, APPLICATION_NAME)	
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**gender** *optional*	gender	string	

Type	Name	Description	Schema	Default
Query	**groupBy** *optional*	groupBy	enum (TAG_COUNT, TAG, TOKEN, MODEL, DEVICE_TYPE, DEVICE, DEVICE_OS, DEVICE_ID, IP_ADDRESS, STATE, CITY, ZIP, COUNTRY, CREATED, UPDATED, LAST_UPDATED, CLIENT_TIME, ACTIVE, CUSTOM_ID, CUSTOM_TYPE, CUSTOM_VALUE, CUSTOM_VALUE2, CUSTOM_LONG, CUSTOM_LONG2, CUSTOM_MESSAGE, CUSTOM_MESSAGE2, ACCOUNT_ID, ACCOUNT_USERNAME, ACCOUNT_DISPLAY, ACCOUNT_CREATED, ACCOUNT_GENDER, ACCOUNT_AGE_GROUP, APPLICATION_ID, APPLICATION_KEY, APPLICATION_NAME)	`"TAG"`

Type	Name	Description	Schema	Default
Query	**groupByRoot** *optional*	groupByRoot	enum (TAG_COUNT, TAG, TOKEN, MODEL, DEVICE_TYPE, DEVICE, DEVICE_OS, DEVICE_ID, IP_ADDRESS, STATE, CITY, ZIP, COUNTRY, CREATED, UPDATED, LAST_UPDATED, CLIENT_TIME, ACTIVE, CUSTOM_ID, CUSTOM_TYPE, CUSTOM_VALUE, CUSTOM_VALUE2, CUSTOM_LONG, CUSTOM_LONG2, CUSTOM_MESSAGE, CUSTOM_MESSAGE2, ACCOUNT_ID, ACCOUNT_USERNAME, ACCOUNT_DISPLAY, ACCOUNT_CREATED, ACCOUNT_GENDER, ACCOUNT_AGE_GROUP, APPLICATION_ID, APPLICATION_KEY, APPLICATION_NAME)	`"CREATED"`
Query	**hideUnknown** *optional*	hideUnknown	boolean	`"true"`
Query	**latitude** *optional*	latitude	number(double)	

Type	Name	Description	Schema	Default
Query	**limit** *optional*	limit	integer(int32)	`"0"`
Query	**longitude** *optional*	longitude	number(double)	
Query	**model** *optional*	model	string	
Query	**responseFormat** *optional*	responseFormat	enum (HTML, XML, JSON, CSV)	`"XML"`

Type	Name	Description	Schema	Default
Query	**sortField** *optional*	sortField	enum (TAG_COUNT, TAG, TOKEN, MODEL, DEVICE_TYPE, DEVICE, DEVICE_OS, DEVICE_ID, IP_ADDRESS, STATE, CITY, ZIP, COUNTRY, CREATED, UPDATED, LAST_UPDATED, CLIENT_TIME, ACTIVE, CUSTOM_ID, CUSTOM_TYPE, CUSTOM_VALUE, CUSTOM_VALUE2, CUSTOM_LONG, CUSTOM_LONG2, CUSTOM_MESSAGE, CUSTOM_MESSAGE2, ACCOUNT_ID, ACCOUNT_USERNAME, ACCOUNT_DISPLAY, ACCOUNT_CREATED, ACCOUNT_GENDER, ACCOUNT_AGE_GROUP, APPLICATION_ID, APPLICATION_KEY, APPLICATION_NAME)	`"CREATED"`
Query	**startDate** *optional*	startDate	integer(int64)	
Query	**state** *optional*	state	string	
Query	**tag** *optional*	tag	string	

Type	Name	Description	Schema	Default
Query	**userAccountDisplay** *optional*	userAccountDisplay	string	
Query	**userAccountId** *optional*	userAccountId	integer(int64)	
Query	**userAccountUsername** *optional*	userAccountUsername	string	
Query	**zip** *optional*	zip	string	

Responses

HTTP Code	Description	Schema
200	successful operation	ChartData

6.4.2. Get Filtered Usage

```
POST /api/{version}/analytics/filteredUsage
```

Description

Query analytics to get data used for graphs and charts

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**_l** *optional*	_l	integer(int32)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**ageGroup** *optional*	ageGroup	string	
Query	**appKey** *optional*	appKey	string	

Type	Name	Description	Schema	Default
Query	**applicationId** *optional*	applicationId	integer(int64)	
Query	**city** *optional*	city	string	
Query	**country** *optional*	country	string	
Query	**customId** *optional*	customId	integer(int64)	
Query	**customLong** *optional*	customLong	integer(int64)	
Query	**customLong2** *optional*	customLong2	integer(int64)	
Query	**customMessage** *optional*	customMessage	string	
Query	**customMessage2** *optional*	customMessage2	string	
Query	**customType** *optional*	customType	string	
Query	**customValue** *optional*	customValue	number(double)	
Query	**customValue2** *optional*	customValue2	number(double)	
Query	**descending** *optional*	descending	boolean	`"false"`
Query	**device** *optional*	device	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**deviceOS** *optional*	deviceOS	string	
Query	**deviceType** *optional*	deviceType	string	

Type	Name	Description	Schema	Default
Query	**distinctCount** *optional*	distinctCount	enum (TAG_COUNT, TAG, TOKEN, MODEL, DEVICE_TYPE, DEVICE, DEVICE_OS, DEVICE_ID, IP_ADDRESS, STATE, CITY, ZIP, COUNTRY, CREATED, UPDATED, LAST_UPDATED, CLIENT_TIME, ACTIVE, CUSTOM_ID, CUSTOM_TYPE, CUSTOM_VALUE, CUSTOM_VALUE2, CUSTOM_LONG, CUSTOM_LONG2, CUSTOM_MESSAGE, CUSTOM_MESSAGE2, ACCOUNT_ID, ACCOUNT_USERNAME, ACCOUNT_DISPLAY, ACCOUNT_CREATED, ACCOUNT_GENDER, ACCOUNT_AGE_GROUP, APPLICATION_ID, APPLICATION_KEY, APPLICATION_NAME)	
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**gender** *optional*	gender	string	

Type	Name	Description	Schema	Default
Query	**groupBy** *optional*	groupBy	enum (TAG_COUNT, TAG, TOKEN, MODEL, DEVICE_TYPE, DEVICE, DEVICE_OS, DEVICE_ID, IP_ADDRESS, STATE, CITY, ZIP, COUNTRY, CREATED, UPDATED, LAST_UPDATED, CLIENT_TIME, ACTIVE, CUSTOM_ID, CUSTOM_TYPE, CUSTOM_VALUE, CUSTOM_VALUE2, CUSTOM_LONG, CUSTOM_LONG2, CUSTOM_MESSAGE, CUSTOM_MESSAGE2, ACCOUNT_ID, ACCOUNT_USERNAME, ACCOUNT_DISPLAY, ACCOUNT_CREATED, ACCOUNT_GENDER, ACCOUNT_AGE_GROUP, APPLICATION_ID, APPLICATION_KEY, APPLICATION_NAME)	`"TAG"`
Query	**hideUnknown** *optional*	hideUnknown	boolean	`"true"`
Query	**latitude** *optional*	latitude	number(double)	

Type	Name	Description	Schema	Default
Query	**limit** *optional*	limit	integer(int32)	`"0"`
Query	**longitude** *optional*	longitude	number(double)	
Query	**model** *optional*	model	string	
Query	**responseFormat** *optional*	responseFormat	enum (HTML, XML, JSON, CSV)	`"XML"`

Type	Name	Description	Schema	Default
Query	**sortField** *optional*	sortField	enum (TAG_COUNT, TAG, TOKEN, MODEL, DEVICE_TYPE, DEVICE, DEVICE_OS, DEVICE_ID, IP_ADDRESS, STATE, CITY, ZIP, COUNTRY, CREATED, UPDATED, LAST_UPDATED, CLIENT_TIME, ACTIVE, CUSTOM_ID, CUSTOM_TYPE, CUSTOM_VALUE, CUSTOM_VALUE2, CUSTOM_LONG, CUSTOM_LONG2, CUSTOM_MESSAGE, CUSTOM_MESSAGE2, ACCOUNT_ID, ACCOUNT_USERNAME, ACCOUNT_DISPLAY, ACCOUNT_CREATED, ACCOUNT_GENDER, ACCOUNT_AGE_GROUP, APPLICATION_ID, APPLICATION_KEY, APPLICATION_NAME)	`"CREATED"`
Query	**startDate** *optional*	startDate	integer(int64)	
Query	**state** *optional*	state	string	

Type	Name	Description	Schema	Default
Query	**sumColumn** *optional*	sumColumn	enum (TAG_COUNT, TAG, TOKEN, MODEL, DEVICE_TYPE, DEVICE, DEVICE_OS, DEVICE_ID, IP_ADDRESS, STATE, CITY, ZIP, COUNTRY, CREATED, UPDATED, LAST_UPDATED, CLIENT_TIME, ACTIVE, CUSTOM_ID, CUSTOM_TYPE, CUSTOM_VALUE, CUSTOM_VALUE2, CUSTOM_LONG, CUSTOM_LONG2, CUSTOM_MESSAGE, CUSTOM_MESSAGE2, ACCOUNT_ID, ACCOUNT_USERNAME, ACCOUNT_DISPLAY, ACCOUNT_CREATED, ACCOUNT_GENDER, ACCOUNT_AGE_GROUP, APPLICATION_ID, APPLICATION_KEY, APPLICATION_NAME)	
Query	**tag** *optional*	tag	string	
Query	**userAccountDisplay** *optional*	userAccountDisplay	string	

Type	Name	Description	Schema	Default
Query	**userAccountId** *optional*	userAccountId	integer(int64)	
Query	**userAccountUsername** *optional*	userAccountUsername	string	
Query	**zip** *optional*	zip	string	

Responses

HTTP Code	Description	Schema
200	successful operation	ChartData

6.4.3. Create Usage Record

```
POST /api/{version}/analytics/usage
```

Description

Record an analytic record for a known state within the application.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**achievementIncrement** *optional*	achievementIncrement	integer(int64)
Query	**appKey** *optional*	appKey	string
Query	**appVersion** *optional*	appVersion	string
Query	**applicationId** *optional*	applicationId	integer(int64)

Type	Name	Description	Schema
Query	**backgroundEvent** *optional*	backgroundEvent	boolean
Query	**city** *optional*	city	string
Query	**clientTime** *optional*	clientTime	integer(int64)
Query	**country** *optional*	country	string
Query	**customId** *optional*	customId	integer(int64)
Query	**customLong** *optional*	customLong	integer(int64)
Query	**customLong2** *optional*	customLong2	integer(int64)
Query	**customMessage** *optional*	customMessage	string
Query	**customMessage2** *optional*	customMessage2	string
Query	**customType** *optional*	customType	string
Query	**customValue** *optional*	customValue	number(double)
Query	**customValue2** *optional*	customValue2	number(double)
Query	**device** *optional*	device	string
Query	**deviceId** *optional*	deviceId	string
Query	**deviceOS** *optional*	deviceOS	string
Query	**deviceType** *optional*	deviceType	string
Query	**errorMessage** *optional*	errorMessage	string

Type	Name	Description	Schema
Query	**ip** *optional*	ip	string
Query	**latitude** *optional*	latitude	number(double)
Query	**locationDescription** *optional*	locationDescription	string
Query	**longitude** *optional*	longitude	number(double)
Query	**model** *optional*	model	string
Query	**returnSummaryResponse** *optional*	returnSummaryResponse	boolean
Query	**state** *optional*	state	string
Query	**tag** *required*	tag	string
Query	**updateRanking** *optional*	updateRanking	boolean
Query	**userAgent** *optional*	userAgent	string
Query	**zip** *optional*	zip	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

Example

```
$ curl -X POST \
 https://www.sirqul.com/api/3.16/analytics/usage \
 -H 'Application-Key: 5c9956f22fdf5a7d225dcb154f4d3912' \
 -H 'Application-Rest-Key: 3804c22449e7301165bf89342bcbcaee' \
 -H 'Content-Type: application/x-www-form-urlencoded' \
 -d 'tag=RUSSELL_TEST&deviceType=HANDHELD&deviceOS=IOS'
```

6.4.4. Create Multiple Usage Records

```
POST /api/{version}/analytics/usage/batch
```

Description

Sends multiple analytics. Can be used to send in the user's stored usage when they did not have internet access. Should not include more than 100 items per batch.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *required*	appKey	string
Query	**appVersion** *optional*	appVersion	string
Query	**data** *required*	data	string
Query	**device** *required*	device	string
Query	**deviceId** *optional*	deviceId	string
Query	**deviceOS** *optional*	deviceOS	string
Query	**deviceType** *optional*	deviceType	string
Query	**model** *optional*	model	string

Type	Name	Description	Schema
Query	**returnSummaryResponse** *optional*	returnSummaryResponse	boolean

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.4.5. Get User Activity

```
GET /api/{version}/analytics/useractivity
```

Description

Get an activity feed by user.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**limit** *required*	limit	integer(int32)
Query	**start** *required*	start	integer(int32)

Responses

HTTP Code	Description	Schema
200	successful operation	< UserActivityResponse > array

6.5. AppData

The AppData service allows you to build out the JSON of large amounts of data and put it into cache, so that when a user needs to load content on an app or website, the data is already pre-loaded. This service should be used when you have a lot of data that needs to be loaded and you want to minimize the amount of time the user has to wait before being able to access content. This works great for game applications, where it may contain hundreds to thousands of game levels that need to appear on one single view.

Below are some example scenarios as to how these services can be used:

- You have a game application where game levels are split into different groups called 'Packs'. A pack could have hundreds of game levels. In order to allow the user to open the pack without having to wait minutes to be able to select a game level, cache the game levels via the AppData service so that the next time the user loads the pack again, they will not need to wait at all
- You have a photo sharing application where an 'album' can contain hundreds of your photos. In order to allow the user to open the album without having to wait minutes to be able to view a photo or video, cache the items in the album via the AppData service so that the next time the user loads the album again, they will not need to wait at all

The AppData service is important because it prevents frustrating experiences for the user to occur. When a user has to wait an unreasonable amount of time before viewing their desired content inside of an app, there's a high possibly that the user may never return to the app again. The AppData service will reduce and possibly eliminate wait time for users so that they have a seamless experience inside of an application.

Below are example services that could be combined with the AppData service:

- The Albums, Assets, and Notes service works well together with the AppData because when combining assets and notes into an Album, the data could get quite large. The AppData service should be used to cache data from assets and notes so that the user does not have to load this information every single time they open up an album inside of a photo sharing application.
- The Packs, Game Levels, Missions, and Offers services work well together because the data in combination of these services could also get quite large. The AppData service should be used to cache data from offers, game levels, and packs, so that when the user loads their mission, they will not have to wait long for the content to appear

6.5.1. Get App Data

```
POST /api/{version}/app/get
```

Description

Get the application data structure. The basic structure is a node tree, with the root node being a

AppResponse. The response contains the user's profile, messages from the system, and a list of MissionResponse. A mission can have any number of GameResponses but typically is a single game type. A game then has any number of PackResponses which help group the game levels. Packs are then composed of any number of GameLevelResponses.

Using the various parameters can return the applications default mission (built-in packs to play), the list of community levels published, the user's saved levels, or explicity levels desired. You can choose to include the profile or not, or just return parts of the profile. You can also filter out game levels that have been published with a higher version of the application.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**_i** *optional*	_i	integer(int32)	
Query	**_l** *optional*	_l	integer(int32)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appVersion** *optional*	appVersion	string	
Query	**dateCreated** *optional*	dateCreated	integer(int64)	
Query	**descending** *optional*	descending	boolean	`"true"`
Query	**deviceId** *optional*	deviceId	string	
Query	**filter** *optional*	filter	string	`"PUBLIC"`
Query	**gameIds** *optional*	gameIds	string	
Query	**gameLevelIds** *optional*	gameLevelIds	string	
Query	**gameObjectCount** *optional*	gameObjectCount	boolean	`"true"`

Type	Name	Description	Schema	Default
Query	**gameType** *optional*	gameType	string	
Query	**includeGameData** *optional*	includeGameData	boolean	`"true"`
Query	**includeHigherVersionLevels** *optional*	includeHigherVersionLevels	boolean	`"false"`
Query	**includeHigherVersionPacks** *optional*	includeHigherVersionPacks	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"0"`
Query	**missionIds** *optional*	missionIds	string	
Query	**ownerId** *optional*	ownerId	integer(int64)	
Query	**packIds** *optional*	packIds	string	
Query	**purchaseType** *optional*	purchaseType	string	`"SIRQUL"`
Query	**q** *optional*	q	string	
Query	**responseGroups** *optional*		string	
Query	**sortField** *optional*	sortField	string	`"LEVEL_CREATED"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	AppResponse

6.5.2. Create App Data

```
POST /api/{version}/app/post
```

Description

Publish the application data structure. Can be used to save levels and scores. It then returns the application data structure.

The basic structure is a node tree, with the root node being a AppResponse. The response contains the user's profile, messages from the system, and a list of MissionResponse. A mission can have any number of GameResponses but typically is a single game type. A game then has any number of PackResponses which help group the game levels. Packs are then composed of any number of GameLevelResponses.

Using the various parameters can return the applications default mission (built-in packs to play), the list of community levels published, the user's saved levels, or explicity levels desired. You can choose to include the profile or not, or just return parts of the profile. You can also filter out game levels that have been published with a higher version of the application

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**_i** *optional*	_i	integer(int32)	
Query	**_l** *optional*	_l	integer(int32)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appVersion** *optional*	appVersion	string	
Query	**data** *required*	data	string	
Query	**dateCreated** *optional*	dateCreated	integer(int64)	

Type	Name	Description	Schema	Default
Query	**descending** *optional*	descending	boolean	`"true"`
Query	**deviceId** *optional*	deviceId	string	
Query	**filter** *optional*	filter	string	`"PUBLIC"`
Query	**gameIds** *optional*	gameIds	string	
Query	**gameLevelIds** *optional*	gameLevelIds	string	
Query	**gameObjectCount** *optional*	gameObjectCount	boolean	`"true"`
Query	**gameType** *required*	gameType	string	
Query	**includeGameData** *optional*	includeGameData	boolean	`"true"`
Query	**includeHigherVersionLevels** *optional*	includeHigherVersionLevels	boolean	`"false"`
Query	**includeHigherVersionPacks** *optional*	includeHigherVersionPacks	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"0"`
Query	**missionIds** *optional*	missionIds	string	
Query	**ownerId** *optional*	ownerId	integer(int64)	
Query	**packIds** *optional*	packIds	string	

Type	Name	Description	Schema	Default
Query	**purchaseType** *optional*	purchaseType	string	`"SIRQUL"`
Query	**q** *optional*	q	string	
Query	**responseGroups** *optional*		string	
Query	**sortField** *optional*	sortField	string	`"LEVEL_UPDATED"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	AppResponse

6.5.3. Regenerate App Data

```
POST /api/{version}/app/regen
```

Description

Regenerate the app data cache for apps

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**apiVersion** *optional*	apiVersion	string
Query	**appKey** *optional*	appKey	string

Type	Name	Description	Schema
Query	**buildVersion** *optional*	buildVersion	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.6. Application

In order to start using the Sirqul suite of APIs and web interfaces, developers must first create an application. You can create, publish, and manage your application(s) within the Sirqul ecosystem. Our APIs and web interfaces provide the flexibility you need to make real-time changes and be up-and-running in a matter of minutes.

Below are some example scenarios as to how these services can be used:

- Retail: Retail store releases a shopping application for customers to install so that they can view all of the latest news and sales updates.
- Healthcare: Patient downloads a hospital application to setup doctor appointments and to also communicate with the hospital staff members remotely via video chat.
- Smart Home/Communities: Home owner installs an application to manage all of his smart home devices through his phone device
- Smart Vehicle: Car owner installs an application on his phone to manage the climate control and start the engine remotely.
- Robotics: User installs an application on his device to be able to control his drone's movements and actions

If you want to utilize any of Sirqul's 60+ services, you must have an application created and configured properly. Once an application is created, developers can start to utilize the various APIs and functionality that enable you to manage your applications. For instance, enable push notifications for your application to send personalized messages to your users. In addition, create ads through our advertising platform and push content that is relevant to your users interests. Once you have an application established in our ecosystem, you can really start creating compelling and differentiating features and functionality by utilizing the various services that we have available. Each of our services are like building blocks that can be taken and integrated inside of your application.

Below are great example services that could be combined with this service:

- Application Configuration Service: Create compelling content or ads and deploy them onto your application on the fly, without a server or client update.

- Notifications Service: Enable your application to support push notifications. Push notifications can then be sent to your users with custom and tailored messaging.
- Ads Service: Create and deploy ads that make sense to your users that are compelling and relevant to their interests.
- Analytics Service: Track the type of usage your users have in your application. Use this data to drive your business decisions to enhance the overall experience of your application.

6.6.1. Change Appliation Status

```
POST /api/{version}/application/active
```

Description

Set the application's active flag to true/false. This effectively activates or deactivates the application.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**accountId** *required*	The account used to perform the delete, must have rights to edit the application.	integer(int64)
Query	**active** *required*	If true then set to active, false otherwise	boolean
Query	**appKey** *required*	The key of the application to be deleted	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.6.2. Assign the owner for the application

```
POST /api/{version}/application/assignowner
```

Description

Assign the owner for the application

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**accountId** *required*	The account id of the logged user (deviceId or accountId required)	integer(int64)
Query	**appKey** *optional*	The key of the application	string
Query	**appOwnerId** *optional*	The id of the account being set to be the owner of the app	integer(int64)
Query	**applicationId** *optional*	Application Id	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	ApplicationResponse

6.6.3. Create Application Certificate

```
POST /api/{version}/application/certificate/create
```

Description

Uploads a certificate for an application that the user has access to.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**accountId** *required*	The account used to perform the delete, must have rights to edit the application.	integer(int64)
Query	**appKey** *required*	The key of the application	string
Query	**certificate** *optional*	Certificate	ref

Type	Name	Description	Schema
Query	**deviceId** *optional*	Device Id	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.6.4. Copy an application and its content

```
POST /api/{version}/application/copy
```

Description

Copy an existig application and its content.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**accountId** *required*	The account id of the user	integer(int64)
Query	**appKey** *required*	The app key of the application to copy	string
Query	**newAppNameSlug** *required*	The new normalized application name for updating with for analytic tags and leaderboards	string
Query	**oldAppNameSlug** *required*	The normalized application name that should be replaced	string

Responses

HTTP Code	Description	Schema
200	successful operation	ApplicationResponse

6.6.5. Create Application

```
POST /api/{version}/application/create
```

Description

Create an application record and one placement record for that application. You can create more placements for this application by using {@link createApplicationPlacement}.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version being used for the request	number(double)	
Query	**about** *optional*	The description of the application	string	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**activityDescription** *optional*	The description of the application in the activity feed	string	
Query	**apiVersion** *optional*	The current API version the application uses	number(float)	
Query	**appIconAssetId** *optional*	The application icon asset id	integer(int64)	
Query	**appLogoAssetId** *optional*	The application logo asset id	integer(int64)	
Query	**appName** *required*	The name of the application	string	
Query	**authorizedCount** *optional*	How many servers the license will support	integer(int32)	`"1"`
Query	**authorizedServers** *optional*	The list of ip addresses of servers the license will support, leave null for any server	string	
Query	**buildVersion** *optional*	The current build version of the application	number(float)	`"1.0"`

Type	Name	Description	Schema	Default
Query	**bundleId** *optional*	The application bundle identifier (format - com.company.appName)	string	
Query	**categoryIds** *optional*	List of categories to apply	string	
Query	**createObjectStore** *optional*	Generate a custom object store to use	boolean	`"true"`
Query	**deviceId** *optional*	The unique id of the device making the request (deviceId or accountId required)	string	
Query	**downloadUrls** *optional*	Json formatted downloadUrls. <pre> <code> {'ios': 'the-url-to-app-store', 'android': 'the-url-to-google-play' </code> </pre>	string	
Query	**enableAPNSBadge** *optional*	Enables setting the APNS badge value in the payload	boolean	
Query	**enableOutsideAdjustments** *optional*	Determines whether to enable logic to adjust users outside during trilateration	boolean	
Query	**eulaVersion** *optional*	The EULA version	string	
Query	**facebookAppId** *optional*	The Facebook application id	string	
Query	**facebookAppSecret** *optional*	The Facebook application secret	string	
Query	**globalTickets** *optional*	Determines whether earned tickets are applied across all applications	boolean	`"false"`
Query	**googleApiKey** *optional*	This is the either the 'server key' or 'browser key' generated from google to enable Google Cloud Messaging	string	
Query	**hasGameData** *optional*	Determines whether the application uses services to save custom game objects	boolean	`"true"`
Query	**hintCost** *optional*	The cost of hints	integer(int32)	`"11"`

Type	Name	Description	Schema	Default
Query	**includeInReport** *optional*	Enables using the application in session reports	boolean	
Query	**invitePageUrl** *optional*	The url to the application invite page	string	
Query	**inviteWelcomeText** *optional*	The text to display on the invite page	string	
Query	**ipsFloor** *optional*	Create floor tables for Ips	boolean	
Query	**landingPageUrl** *optional*	The landing page URL	string	
Query	**localAdsEnabled** *optional*	Local Ads Enabled	boolean	`"false"`
Query	**maxSampleSize** *optional*	Determines what the maximum sample size during trilateration	integer(int32)	
Query	**maxScore** *optional*	The maximum score that will be possible	integer(int32)	`"125"`
Query	**metaData** *optional*	The application meta data. Defined by the client	string	
Query	**minRSSI** *optional*	Determines what the minimum acceptable RSSI value	number(double)	
Query	**minimumSessionLength** *optional*	Minimum Session Length	integer(int32)	
Query	**modules** *optional*	List modules allowed to be served by the server, possible values include: COMMON, MEDIA, OFFER, GAME, SOCIAL, CONSUMER_WEBSITE, ALL	string	`"ALL"`
Query	**placementDescription** *optional*	The description of the placement	string	
Query	**placementHeight** *optional*	The height of a custom ad size	integer(int32)	

Type	Name	Description	Schema	Default
Query	**placementMetaData** *optional*	The ad placement meta data. Defined by the client	string	
Query	**placementName** *optional*	The name of the placement	string	
Query	**placementRefreshInterval** *optional*	The refresh interval in seconds	integer(int32)	
Query	**placementSize** *optional*	The ad placement size {BANNER, LEADERBOARD, SKYSCRAPER, INTERSTITIAL, CUSTOM (this is required if a placements is to be created)	enum (CONFIG, BANNER, LEADERBOARD, SKYSCRAPER, VIDEO, ZIP, INTERSTITIAL, CUSTOM1, CUSTOM2, CUSTOM3, CUSTOM4, CUSTOM5, CUSTOM6, CUSTOM7, CUSTOM8, CUSTOM9, CUSTOM10)	
Query	**placementWidth** *optional*	The width of a custom ad size	integer(int32)	
Query	**platforms** *optional*	A json object with a list of supported platforms. <pre> <code> {platforms:[{ 'deviceId':1, 'minimum':230, 'maximum':421, 'downloadUrl':'http://app.store.com/download', 'description':'description and version info'] </code> </pre> the minumum and maximum version values the version numbers with the dots removed and right padded with 0 up to a 3 digit long value.	string	
Query	**productionMode** *optional*	Determines whether the application uses production or sandbox services	boolean	`"false"`

Type	Name	Description	Schema	Default
Query	**publicContentApproval** *optional*	Determine whether or not public content requires admin approval before it becomes public	boolean	`"false"`
Query	**publicNotifications** *optional*	Public Notifications	boolean	
Query	**scoringType** *optional*	The type of scoring this application will use {GAME_LEVEL, GAME_OBJECT	enum (GAME_LEVEL, GAME_OBJECT)	`"GAME_LEVEL"`
Query	**sessionGapLength** *optional*	Session Gap Length	integer(int32)	
Query	**showInActivities** *optional*	Determines whether to show the application in the activity feed	boolean	
Query	**sqootApiKey** *optional*	Sqoot Api Key	string	
Query	**ticketsPerPoint** *optional*	The point-to-ticket conversion ratio	number(float)	`"0.037"`
Query	**trilatProcessingType** *optional*	Determines how to process trilateration data. Possible values include: DEFAULT, FINGERPRINT	enum (DEFAULT, FINGERPRINT, FINGERPRINT_V2)	`"FINGERPRINT"`
Query	**updateEULADate** *optional*	Determines whether to update the EULA date	boolean	
Query	**urlScheme** *optional*	The protocal the app uses to load the app via a browser	string	
Query	**useMatchingAlgorithm** *optional*	Use Matching Algorithm	boolean	

Responses

HTTP Code	Description	Schema
200	successful operation	ApplicationResponse

6.6.6. Delete Application

```
POST /api/{version}/application/delete
```

Description

Set the deleted timestamp to current time. This effectively deletes the application since all queries should ignore any records with a deleted timestamp

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**accountId** *required*	The account used to perform the delete, must have rights to edit the application.	integer(int64)
Query	**appKey** *optional*	The key of the application to be deleted	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.6.7. Get Application

```
POST /api/{version}/application/get
```

Description

Get a specific application by appKey

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**appKey** *optional*	The key of the application	string

Type	Name	Description	Schema
Query	**applicationId** *optional*	Application Id	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	ApplicationResponse

6.6.8. List Applications

```
POST /api/{version}/application/list
```

Description

List active applications matching the criteria (as a consumer)

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version being used for the request	number(double)	
Query	**_i** *optional*	_i	integer(int32)	
Query	**_l** *optional*	_l	integer(int32)	
Query	**accountId** *required*	The account id of the application owner/manager	integer(int64)	
Query	**activeOnly** *optional*	Return only active results	boolean	`"true"`
Query	**applicationIds** *optional*	The list of application ids, comma separated. If provided will ignore all other params.	string	
Query	**categoryIds** *optional*	The list of category ids to filter the list by	string	
Query	**descending** *optional*	The order to return the search results	boolean	`"true"`

Type	Name	Description	Schema	Default
Query	**deviceIds** *optional*	The list of targeted device ids, comma separated; possible values are retreived via /api/{version/audience/devices com.sirqul.geoapps.common.service.AudienceApi#getDevices getDevices (pass in deviceId values)	string	
Query	**deviceVersions** *optional*	The list of targeted device version ranges that are aligned with the provided devices list (see devices param for possible values), comma separated; examples are: 2.3-X, 0-5.1.4, 4.3.1-6.1.4 where X is no maximum and 0 is no minimum.	string	
Query	**filterBillable** *optional*	Determines whether to only return applications that the user has access to	boolean	
Query	**hasAds** *optional*	Filter results on whether the application supports ads or not. Ignore this parameter to return all results.	boolean	
Query	**hasObjectStore** *optional*	Only include applications with a object store (default is false)	boolean	`"false"`
Query	**keyword** *optional*	The keyword used to search for title, about, and description fields	string	
Query	**limit** *optional*	The limit of the pagination	integer(int32)	`"20"`
Query	**platforms** *optional*	Deprecated, use deviceIds and deviceVersions	string	
Query	**publicNotifications** *optional*	Filter results on whether the application is available for public trigger notifications	boolean	
Query	**q** *optional*	Q	string	

Type	Name	Description	Schema	Default
Query	**sortField** *optional*	The column to sort the search on, possible values include: UPDATED (default), CREATED, TITLE	enum (CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, HAS_ADVERTISEMENTS, PUBLIC_NOTIFICATIONS, PLACEMENTS, BILLABLE_ENTITY_ID, BILLABLE_ENTITY_NAME, RESPONSIBLE_DISPLAY, TITLE, NAME, ABOUT, DESCRIPTION, APPLICATION_ID, APP_TYPE, GAME_TYPE, BUNDLE_ID, SCORING_TYPE, LANDING_PAGE_URL, EULA_VERSION, BUILD_VERSION, API_VERSION)	`"UPDATED"`
Query	**start** *optional*	The start of the pagination	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< ApplicationShortResponse > array

6.6.9. Create Ad Placement

```
POST /api/{version}/application/placement/create
```

Description

Creates a new ad placement for an application.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**active** *optional*	Sets the placement to be active/inactive	boolean
Query	**appKey** *required*	The appKey of the application the ad placement is for	string
Query	**defaultImageId** *optional*	Default Image Id	integer(int64)
Query	**description** *optional*	The description of the placement	string
Query	**deviceId** *optional*	The unique id of the device making the request (deviceId or accountId required)	string
Query	**height** *optional*	The height of a custom ad size	integer(int32)
Query	**metaData** *optional*	The ad placement meta data. Defined by the client	string
Query	**name** *optional*	The name of the placement	string
Query	**refreshInterval** *optional*	The refresh interval in seconds	integer(int32)

Type	Name	Description	Schema
Query	**size** *required*	The ad placement size {BANNER, LEADERBOARD, SKYSCRAPER, INTERSTITIAL, CUSTOM	enum (CONFIG, BANNER, LEADERBOARD, SKYSCRAPER, VIDEO, ZIP, INTERSTITIAL, CUSTOM1, CUSTOM2, CUSTOM3, CUSTOM4, CUSTOM5, CUSTOM6, CUSTOM7, CUSTOM8, CUSTOM9, CUSTOM10)
Query	**width** *optional*	The width of a custom ad size	integer(int32)

Responses

HTTP Code	Description	Schema
200	successful operation	PlacementRespons e

6.6.10. Delete Ad Placement

```
POST /api/{version}/application/placement/delete
```

Description

Deletes an ad placement for an application.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**deviceId** *optional*	The unique id of the device making the request (deviceId or accountId required)	string

Type	Name	Description	Schema
Query	**placementId** *required*	The id of the placement to delete, the user must have rights to the application the ad placement is for	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	PlacementRespons e

6.6.11. Get Ad Placement

```
POST /api/{version}/application/placement/get
```

Description

Get details of an ad placement

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**deviceId** *optional*	The unique id of the device making the request (deviceId or accountId required)	string
Query	**placementId** *required*	The id of the placement	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	PlacementRespons e

6.6.12. Search for Ad Placements

```
POST /api/{version}/application/placement/search
```

Description

Searches placements for an application.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version to use to make the request	number(double)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**appKey** *required*	The key of the application	string	
Query	**deviceId** *optional*	The unique id of the device making the request (deviceId or accountId required)	string	
Query	**limit** *optional*	The limit of the pagination	integer(int32)	`"100"`
Query	**start** *optional*	The start of the pagination	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< PlacementResponse > array

6.6.13. Update Ad Placement

```
POST /api/{version}/application/placement/update
```

Description

Updates an ad placement for an application.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**active** *optional*	Sets the placement to be active/inactive	boolean
Query	**defaultImageId** *optional*	Default Image Id	integer(int64)
Query	**description** *optional*	The description of the placement	string
Query	**deviceId** *optional*	The unique id of the device making the request (deviceId or accountId required)	string
Query	**height** *optional*	The height of a custom ad size	integer(int32)
Query	**metaData** *optional*	The ad placement meta data. Defined by the client	string
Query	**name** *optional*	The name of the placement	string
Query	**placementId** *required*	The id of the placement to update, the user must have rights to the application the ad placement is for	integer(int64)
Query	**refreshInterval** *optional*	The refresh interval in seconds	integer(int32)
Query	**size** *optional*	The ad placement size {BANNER, LEADERBOARD, SKYSCRAPER, INTERSTITIAL, CUSTOM	enum (CONFIG, BANNER, LEADERBOARD, SKYSCRAPER, VIDEO, ZIP, INTERSTITIAL, CUSTOM1, CUSTOM2, CUSTOM3, CUSTOM4, CUSTOM5, CUSTOM6, CUSTOM7, CUSTOM8, CUSTOM9, CUSTOM10)

Type	Name	Description	Schema
Query	**width** *optional*	The width of a custom ad size	integer(int32)

Responses

HTTP Code	Description	Schema
200	successful operation	PlacementRespons e

6.6.14. Search Applications

```
POST /api/{version}/application/search
```

Description

Search for applications matching the criteria that the logged in user has access to

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version being used for the request	number(double)	
Query	**_i** *optional*	_i	integer(int32)	
Query	**_l** *optional*	_l	integer(int32)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**activeOnly** *optional*	Return only active results	boolean	`"false"`
Query	**descending** *optional*	The order to return the search results	boolean	`"false"`
Query	**deviceId** *optional*	The unique id of the device making the request (deviceId or accountId required)	string	
Query	**hasAds** *optional*	Filter results on whether the application supports ads or not. Ignore this parameter to return all results.	boolean	

Type	Name	Description	Schema	Default
Query	**keyword** *optional*	The keyword used to search	string	
Query	**latitude** *optional*	The location of the device	number(double)	
Query	**limit** *optional*	The limit of the pagination	integer(int32)	`"100"`
Query	**longitude** *optional*	The location of the device	number(double)	
Query	**publicNotifications** *optional*	Filter results on whether the application is available for public trigger notifications	boolean	
Query	**q** *optional*	Q	string	
Query	**qSearchFields** *optional*	The columns to applied the keyword search to	string	`"NAME,ABOUT,BILLABLE_ENTITY_NAME"`

Type	Name	Description	Schema	Default
Query	**sortField** *optional*	The column to sort the search on	enum (CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, HAS_ADVERTISEMENTS, PUBLIC_NOTIFICATIONS, PLACEMENTS, BILLABLE_ENTITY_ID, BILLABLE_ENTITY_NAME, RESPONSIBLE_DISPLAY, TITLE, NAME, ABOUT, DESCRIPTION, APPLICATION_ID, APP_TYPE, GAME_TYPE, BUNDLE_ID, SCORING_TYPE, LANDING_PAGE_URL, EULA_VERSION, BUILD_VERSION, API_VERSION)	`"APPLICATION_ID"`
Query	**start** *optional*	The start of the pagination	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< ApplicationResponse > array

6.6.15. Search for Application Settings

```
POST /api/{version}/application/settings/search
```

Description

Returns a list of applications that the user has logged into before, and returns specific settings for that application and user

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version	number(double)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**connectionAccountId** *optional*	The account id used to view another person's account	integer(int64)	
Query	**descending** *optional*	The order to return the search results	boolean	`"false"`
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string	
Query	**keyword** *optional*	The string to search applications on	string	
Query	**limit** *optional*	The limit per result set for pagination	integer(int32)	`"20"`
Query	**sortField** *optional*	The column to sort the search on. Possible values include: CREATED, UPDATED, APPLICATION_TITLE	string	`"APPLICATION_TITLE"`
Query	**start** *optional*	The start index for pagination	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	ApplicationSettingsResponse

6.6.16. Update Application

```
POST /api/{version}/application/update
```

Description

Update an application record

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version being used for the request	number(double)	
Query	**about** *optional*	The description of the application	string	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**active** *optional*	Sets whether the retailer is active or inactive (hidden from consumers)	boolean	
Query	**activityDescription** *optional*	The description of the application in the activity feed	string	
Query	**apiVersion** *optional*	The current API version the application uses	number(float)	
Query	**appIconAssetId** *optional*	The application icon asset id	integer(int64)	
Query	**appKey** *required*	The application key for updating an existing application	string	
Query	**appLogoAssetId** *optional*	The application logo asset id	integer(int64)	
Query	**appName** *optional*	The name of the application	string	
Query	**authorizedCount** *optional*	How many servers the license will support	integer(int32)	"1"
Query	**authorizedServers** *optional*	The list of ip addresses of servers the license will support, leave null for any server	string	
Query	**buildVersion** *optional*	The current build version of the application	number(float)	
Query	**bundleId** *optional*	The application bundle identifier (format - com.company.appName)	string	

Type	Name	Description	Schema	Default
Query	**categoryIds** *optional*	List of categories to apply	string	
Query	**createObjectStore** *optional*	Generate a custom object store to use	boolean	
Query	**deviceId** *optional*	The unique id of the device making the request (deviceId or accountId required)	string	
Query	**downloadUrls** *optional*	Json formatted downloadUrls. <pre> <code> {'ios': 'the-url-to-app-store', 'android': 'the-url-to-google-play' </code> </pre>	string	
Query	**enableAPNSBadge** *optional*	Enables setting the APNS badge value in the payload	boolean	
Query	**enableOutsideAdjustments** *optional*	Determines whether to enable logic to adjust users outside during trilateration	boolean	
Query	**eulaVersion** *optional*	The EULA version	string	
Query	**facebookAppId** *optional*	The Facebook application id	string	
Query	**facebookAppSecret** *optional*	The Facebook application secret	string	
Query	**globalTickets** *optional*	Determines whether earned tickets are applied across all applications	boolean	
Query	**googleApiKey** *optional*	This is the either the 'server key' or 'browser key' generated from google to enable Google Cloud Messaging	string	
Query	**hasGameData** *optional*	Determines whether the application uses services to save custom game objects	boolean	
Query	**hintCost** *optional*	The cost of hints	integer(int32)	
Query	**includeInReport** *optional*	Enables using the application in session reports	boolean	

Type	Name	Description	Schema	Default
Query	**invitePageUrl** *optional*	The url to the application invite page	string	
Query	**inviteWelcomeText** *optional*	The text to display on the invite page	string	
Query	**landingPageUrl** *optional*	The landing page URL	string	
Query	**localAdsEnabled** *optional*	Local Ads Enabled	boolean	
Query	**maxSampleSize** *optional*	Determines what the maximum sample size during trilateration	integer(int32)	
Query	**maxScore** *optional*	The maximum score that will be possible	integer(int32)	
Query	**metaData** *optional*	The application meta data. Defined by the client.	string	
Query	**minRSSI** *optional*	Determines what the minimum acceptable RSSI value	number(double)	
Query	**minimumSessionLength** *optional*	Minimum Session Length	integer(int32)	
Query	**modules** *optional*	List modules allowed to be served by the server, possible values include: COMMON, MEDIA, OFFER, GAME, SOCIAL, CONSUMER_WEBSITE, ALL	string	`"ALL"`
Query	**platforms** *optional*	A json object with a list of platforms supported. {platforms:[{ 'deviceId':1, 'minimum':230, 'maximum':421, 'downloadUrl':'http://app.store.com/download', 'description':'description and version info'] the minumum and maximum version values the version numbers with the dots removed and right padded with 0 up to a 3 digit long value.	string	
Query	**productionMode** *optional*	Determines whether the application uses production or sandbox services	boolean	

Type	Name	Description	Schema	Default
Query	**publicContentApproval** *optional*	Determine whether or not public content requires admin approval before it becomes public	boolean	
Query	**publicNotifications** *optional*	Public Notifications	boolean	
Query	**scoringType** *optional*	The type of scoring this application will use {GAME_LEVEL, GAME_OBJECT	enum (GAME_LEVEL, GAME_OBJECT)	
Query	**sessionGapLength** *optional*	Session Gap Length	integer(int32)	
Query	**showInActivities** *optional*	Determines whether to show the application in the activity feed	boolean	
Query	**sqootApiKey** *optional*	Sqoot Api Key	string	
Query	**ticketsPerPoint** *optional*	The point-to-ticket conversion ratio	number(float)	
Query	**trilatProcessingType** *optional*	Determines how to process trilateration data. Possible values include: DEFAULT, FINGERPRINT	enum (DEFAULT, FINGERPRINT, FINGERPRINT_V2)	
Query	**updateEULADate** *optional*	Determines whether to update the EULA date	boolean	
Query	**urlScheme** *optional*	The protocal the app uses to load the app via a browser	string	
Query	**useMatchingAlgorithm** *optional*	Use Matching Algorithm	boolean	

Responses

HTTP Code	Description	Schema
200	successful operation	ApplicationResponse

6.6.17. Search Application Users

```
POST /api/{version}/application/users
```

Description

Get a list of users per application

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version to use to make the request	number(double)	
Query	**_i** *optional*	_i	integer(int32)	
Query	**_l** *optional*	_l	integer(int32)	
Query	**appKey** *required*	The application key	string	
Query	**keyword** *optional*	The keyword used to search	string	
Query	**limit** *optional*	The limit of the pagination	integer(int32)	`"20"`
Query	**q** *optional*	Q	string	
Query	**since** *optional*	Return accounts that have been active after this date (UNIX time-stamp in milliseconds)	integer(int64)	
Query	**start** *optional*	The start of the pagination	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	AccountListResponse

6.6.18. Get API versions

```
POST /api/{version}/application/versions
```

Description

Will return a comma separated list of numbers, newest first. For example: 3.0, 2.2, 2.1, 1.8

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.7. ApplicationConfig

The application configuration service allows you to dynamically update files, themes, and the look & feel inside of your application through the server and not by client-side updates. This allows you to customize your applications without having to require any client-side changes. This is extremely important when you need to make updates on-the-fly in order to meet certain deadlines or seamlessly push out seasonal promotions or content updates.

Below are some example scenarios as to how these services can be used:

- Retail: Push out promotional content or ads inside of the customer's shopping app without a client update
- Medical: Patient sees an ad on their Dentist application about a summer promotional discount on teeth cleaning
- Real Estate: Home owner sees a holiday theme update on their smart TV menu UI
- Automotive: Car owner sees an ad pop up on their phone for a free car wash at the nearby gas station.

It's absolutely critical that you have the ability to generate and push out content on-the-fly without any sort of client side update. Otherwise, you will be forced to submit application updates to the various app stores, and this will cost time and resources. Instead, utilize our application configuration service to apply all of these updates dynamically, saving you all the hassle of submitting your application.

One compelling example that can make a huge impact to your application is by being able to update

the theme and look & feel of your application. You can easily customize the menu interface to make it appear as the theme fits with the user's personal interests. This is important to get your users on-board and talking about it with their friends and community. To go even further, we can support the ability to schedule these dynamic content deployments based upon the date & time that you need them to apply in your applications. Simply schedule the time and day you need the content to be pushed out. The application will then receive the update once the day & time is triggered.

Below are great example services that could be combined with this service:

- Applications Service: Enable your application to dynamically push out content on-the-fly without client updates
- Assets Service: Combine your ad message with visually compelling assets and schedule the deployment to the application
- Creative Service: Utilize the banner ad type and schedule the deployment to the application

6.7.1. Create AppConfig

```
POST /api/{version}/appconfig/create
```

Description

Creates a new application configuration. If the configVersion provided already exists for the given app, an invalid response is returned and the application configuration won't be created.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version	number(double)
Query	**accountId** *required*	The account ID of the user	integer(int64)
Query	**appKey** *required*	The application key that the newly created applicationConfig will be associated to	string
Query	**assetId** *required*	The json assetId that stores the configuration detail.	integer(int64)
Query	**configVersion** *required*	The application configuration, has to be unique within the application	string
Query	**retailerId** *optional*	The retailer id for retailer specific configurations	integer(int64)

Type	Name	Description	Schema
Query	**retailerLocationId** *optional*	The retailer location id for retailer location specific configurations	integer(int64)
Query	**udid** *optional*	The device udid for device specific configurations	string

Responses

HTTP Code	Description	Schema
200	successful operation	ApplicationConfigResponse

6.7.2. Delete AppConfig

```
POST /api/{version}/appconfig/delete
```

Description

Mark the application configuration for deletion.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version	number(double)
Query	**accountId** *required*	The account ID of the user	integer(int64)
Query	**configId** *required*	The config ID of the application configuration to delete	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.7.3. Get AppConfig

```
POST /api/{version}/appconfig/get
```

Description

Gets the appConfig data by the given configId. If appConfig cannot be found, it returns an invalid response.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version	number(double)
Query	**accountId** *required*	The account ID of the user	integer(int64)
Query	**configId** *required*	The config ID of the application configuration	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	ApplicationConfigResponse

6.7.4. Get AppConfig by Version

```
POST /api/{version}/appconfig/getbyversion
```

Description

Gets the appConfig data by the given appKey and app configVersion number.If the appKey is is invalid or appConfig is not found, it returns an invalid response.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version	number(double)	

Type	Name	Description	Schema	Default
Query	**allowOlderVersions** *optional*	Determines whether to return older config versions if the exact version is not found. If this happens, will try to return the latest version.	boolean	`"false"`
Query	**appKey** *required*	The application key	string	
Query	**configVersion** *required*	The version of the application configuration	string	
Query	**retailerId** *optional*	Only returns the config that matches the given retailer	integer(int64)	
Query	**retailerLocationId** *optional*	Only returns the config that matches the given retailer location	integer(int64)	
Query	**udid** *optional*	Only returns only returns the config that matches the given device udid	string	

Responses

HTTP Code	Description	Schema
200	successful operation	ApplicationConfigResponse

6.7.5. Search AppConfigs

```
POST /api/{version}/appconfig/search
```

Description

Gets all versions of application configurations in a particular app by the given appKey.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version	number(double)	
Query	**accountId** *required*	The account ID of the user	integer(int64)	

Type	Name	Description	Schema	Default
Query	**appKey** *optional*	The application key to filter results by application Leaving this empty will return all application configurations for all applications (executive user only)	string	
Query	**configVersion** *optional*	Config Version	string	
Query	**descending** *optional*	Determines whether the results are in descending or ascending order	boolean	`"true"`
Query	**limit** *optional*	The limit for pagination (There is a hard limit of 100)	integer(int32)	`"20"`
Query	**retailerId** *optional*	Only returns the configs that matches the given retailer	integer(int64)	
Query	**retailerLocationId** *optional*	Only returns the configs that matches the given retailer location	integer(int64)	
Query	**sortField** *optional*	Orders the results by: CREATED, UPDATED, ACTIVE, APPLICATION_ID, APPLICATION_KEY, APPLICATION_NAME, RETAILER_NAME, RETAILER_ID, RETAILER_LOCATION_NAME, RETAILER_LOCATION_ID, OFFER_TITLE, OFFER_ID, OFFER_LOCATION_ID, CONFIG_VERSION_MINOR, CONFIG_VERSION_MAJOR, CONFIG_VERSION_INDEX	string	`"CONFIG_VERSION_INDEX"`
Query	**start** *optional*	The start index for pagination	integer(int32)	`"0"`
Query	**udid** *optional*	Only returns only returns the configs that matches the given device udid	string	

Responses

HTTP Code	Description	Schema
200	successful operation	< ApplicationConfigResponse > array

6.7.6. Update AppConfig

```
POST /api/{version}/appconfig/update
```

Description

pdates an existing application configuration. If the configVersion provided already exists for the given app the application configuration won't be updated.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version	number(double)
Query	**accountId** *required*	The account ID of the user	integer(int64)
Query	**appKey** *optional*	The application key that the updated applicationConfig will be associated to	string
Query	**assetId** *optional*	The json assetId that stores the configuration detail.	integer(int64)
Query	**configId** *required*	The config ID of the application configuration to update	integer(int64)
Query	**configVersion** *optional*	The application configuration, has to be unique within the application	string
Query	**retailerId** *optional*	The retailer id for retailer specific configurations	integer(int64)
Query	**retailerLocationId** *optional*	The retailer location id for retailer location specific configurations	integer(int64)
Query	**udid** *optional*	The device udid for device specific configurations	string

Responses

HTTP Code	Description	Schema
200	successful operation	ApplicationConfigResponse

6.8. Asset

Assets are a list that can be made up of photos, videos, audio, text, and many other various object types. Assets can be used for scenarios such as an image used for an ad banner, a large HD image used for the application menu background, visually appealing iconography injected inside of your application, and so much more. If you want to create a visually compelling experience for your users, then utilizing the Assets for your product or application is a must.

Below are some example scenarios as to how these services can be used:

- Retail: When a customer opens up their shopping app, they can see images associated with each of the clothes in the catalog. These images can be considered assets.
- Medical: Patient sees a list of medicine that they have been prescribed to take each day. Each of these medicines has an image associated with it so that they can visually associate the name of the medicine to the capsule itself.
- Real Estate: Sports fan sees a smart booth at the stadium. The booth shows an image on the display about $5 beer at the beer garden. The graphic used is be a high quality image asset.

The Assets service allows you to create a visually compelling experience for your users. Use the Assets service if your application requires the use of HD media content, and you need to able to display a list of these items to your users. For example, if your application contains a catalog of retail products, and each product requires a high quality photo or media content to appeal to your users, utilize the Assets service to upload the content for each of these items in the list. If the application contains a list of patients that require a photograph of each patient for verification purposes, the Assets service allows you to upload the photographs in the appropriate resolution. In addition, iconography should consist of utilizing appropriate assets so that your users understand what they mean. The Assets service will help you build a visually compelling application in order to appropriately communicate to your users.

Below are great example services that could be combined with the Accounts service:

- Accounts Service: Each user account requires a profile image in order to verify their identification. Use the Assets Service to allow users to upload an image.
- Albums Service: Your application may have a list of albums. Each album can contain assets. Use the albums service to allow users to upload assets to each of these albums.
- Creative Service: You can create an ad banner to be pushed out to your application. The ad banner is associated with an image.

6.8.1. Create Asset

```
POST /api/{version}/asset/create
```

Description

Uploads an asset to server and returns an asset id which can be used to assign to various objects.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**addToDefaultAlbum** *optional*	addToDefaultAlbum	string	
Query	**addToMediaLibrary** *optional*	addToMediaLibrary	boolean	
Query	**albumId** *optional*	albumId	integer(int64)	
Query	**app** *optional*	app	string	
Query	**appKey** *optional*	appKey	string	
Query	**attachedMedia** *optional*	attachedMedia	ref	
Query	**attachedMediaString** *optional*	attachedMediaString	string	
Query	**attachedMediaStringContentType** *optional*	attachedMediaStringContentType	string	
Query	**attachedMediaStringFileName** *optional*	attachedMediaStringFileName	string	
Query	**attachedMediaUrl** *optional*	attachedMediaUrl	string	

Type	Name	Description	Schema	Default
Query	**caption** *optional*	caption	string	
Query	**collectionId** *optional*	collectionId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**locationDescription** *optional*	locationDescription	string	
Query	**longitude** *optional*	longitude	number(double)	
Query	**media** *optional*	media	ref	
Query	**mediaString** *optional*	mediaString	string	
Query	**mediaStringContentType** *optional*	mediaStringContentType	string	
Query	**mediaStringFileName** *optional*	mediaStringFileName	string	
Query	**mediaUrl** *optional*	mediaUrl	string	
Query	**metaData** *optional*	metaData	string	
Query	**returnNulls** *required*	returnNulls	boolean	"1"
Query	**searchTags** *optional*	searchTags	string	
Query	**versionCode** *optional*	versionCode	integer(int32)	
Query	**versionName** *optional*	versionName	string	

Responses

HTTP Code	Description	Schema
200	successful operation	AssetResponse

6.8.2. Delete Asset

```
POST /api/{version}/asset/delete
```

Description

Delete an asset.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**assetId** *required*	assetId	string
Query	**deviceId** *optional*	deviceId	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.8.3. Download Asset

```
POST /api/{version}/asset/download/{filename}
```

Description

Downloads an asset from the server for assets that have been uploaded to the server.

Parameters

Type	Name	Description	Schema
Path	**filename** *required*	filename	string
Path	**version** *required*	version	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.8.4. Get Asset

```
POST /api/{version}/asset/get
```

Description

Gets the full asset response including attached likes and notes.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**assetId** *required*	assetId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**noteDescending** *required*	noteDescending	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	AssetFullResponse

6.8.5. Convert Offer to Creative

```
POST /api/{version}/asset/morph
```

Description

Converts an offer image + text into a creative image.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**adSize** *required*	adSize	enum (CONFIG, BANNER, LEADERBOARD, SKYSCRAPER, VIDEO, ZIP, INTERSTITIAL, CUSTOM1, CUSTOM2, CUSTOM3, CUSTOM4, CUSTOM5, CUSTOM6, CUSTOM7, CUSTOM8, CUSTOM9, CUSTOM10)
Query	**backgroundSize** *optional*	backgroundSize	string
Query	**creativeId** *optional*	creativeId	integer(int64)
Query	**height** *optional*	height	integer(int32)
Query	**offerId** *required*		integer(int64)
Query	**template** *optional*	template	string

Type	Name	Description	Schema
Query	**width** *optional*	width	integer(int32)

Responses

HTTP Code	Description	Schema
200	successful operation	AssetShortRespons e

6.8.6. Remove Asset from Collection

```
POST /api/{version}/asset/remove
```

Description

Remove assets from collections

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**albumId** *optional*	albumId	integer(int64)	
Query	**assetId** *required*	assetId	string	
Query	**collectionId** *optional*	collectionId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**longitude** *optional*	longitude	number(double)	

Type	Name	Description	Schema	Default
Query	**removeFromDefaultAlbums** *required*	removeFromDefaultAlbums	boolean	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.8.7. Search Assets

```
POST /api/{version}/asset/search
```

Description

Searches for assets

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activeOnly** *required*	activeOnly	boolean	`"true"`
Query	**albumIds** *optional*	albumIds	string	
Query	**appKey** *optional*	appKey	string	
Query	**assetIds** *optional*	assetIds	string	
Query	**descending** *required*	descending	boolean	`"true"`
Query	**deviceId** *optional*	deviceId	string	

Type	Name	Description	Schema	Default
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**mediaType** *optional*	mediaType	string	
Query	**mimeType** *optional*	mimeType	string	
Query	**returnApp** *optional*	returnApp	boolean	
Query	**searchMediaLibrary** *optional*	searchMediaLibrary	boolean	
Query	**sortField** *required*	sortField	string	`"UPDATED"`
Query	**start** *required*	start	integer(int32)	`"0"`
Query	**updatedBefore** *optional*	updatedBefore	integer(int64)	
Query	**updatedSince** *optional*	updatedSince	integer(int64)	
Query	**versionCode** *optional*	versionCode	integer(int32)	
Query	**versionName** *optional*	versionName	string	

Responses

HTTP Code	Description	Schema
200	successful operation	< AssetResponse > array

6.8.8. Update Asset

```
POST /api/{version}/asset/update
```

Description

Updates an asset's meta data. If an album reference is passed in, the participants with write permissions are allowed to edit the asset. Otherwise, only the asset up-loader has permission to edit the data.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**albumId** *optional*	albumId	integer(int64)
Query	**appKey** *optional*	appKey	string
Query	**assetId** *required*	assetId	integer(int64)
Query	**attachedAssetId** *optional*	attachedAssetId	integer(int64)
Query	**attachedMedia** *optional*	attachedMedia	ref
Query	**attachedMediaString** *optional*	attachedMediaString	string
Query	**attachedMediaStringContentType** *optional*	attachedMediaStringContentType	string
Query	**attachedMediaStringFileName** *optional*	attachedMediaStringFileName	string
Query	**attachedMediaUrl** *optional*	attachedMediaUrl	string
Query	**caption** *optional*	caption	string

Type	Name	Description	Schema
Query	**deviceId** *optional*	deviceId	string
Query	**latitude** *optional*	latitude	number(double)
Query	**locationDescription** *optional*	locationDescription	string
Query	**longitude** *optional*	longitude	number(double)
Query	**media** *optional*	media	ref
Query	**mediaString** *optional*	mediaString	string
Query	**mediaStringContentType** *optional*	mediaStringContentType	string
Query	**mediaStringFileName** *optional*	mediaStringFileName	string
Query	**mediaUrl** *optional*	mediaUrl	string
Query	**metaData** *optional*	metaData	string
Query	**searchTags** *optional*	searchTags	string
Query	**versionCode** *optional*	versionCode	integer(int32)
Query	**versionName** *optional*	versionName	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.8.9. Generation

```
POST /api/{version}/charts/generate
```

Description

Downloads charts into one png file

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *optional*	appKey	string
Query	**mediaUrl** *required*	mediaUrl	string
Query	**startElement** *optional*	startElement	string
Query	**timeout** *optional*	timeout	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.8.10. Create Asset Finished

```
POST /api/{version}/consumer/media/add/finished
```

Description

Triggers a notification to be sent to all album participants when a user add assets to an album.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**albumId** *required*	albumId	integer(int64)
Query	**assetIds** *required*		string
Query	**deviceId** *optional*	deviceId	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.9. Assignment

Assignments are like tasks that can be given to your internal employees or management team. For example, your retail store employees may have a sales quota that they must meet by the end of the month. You can use assignments to keep track of the status on your customer leads. Whether or not they have been contacted, if they seem interested, if you need to schedule another meeting, and so forth. For example, keep track of the status if a potential patient has been contacted for a follow-up visit. The Assignments service is essential to enable your business to meet its goals.

Below are some example scenarios as to how these services can be used:

- Retail: Sales person has products that they need to sell, and has a quota that they need to meet by the end of the month. Use the Assignments service to track the performance of this employee.
- Medical: Hospital staff has reached out to patients that need a follow-up appointment. Keep track if whether or not the patients have responded.
- Real Estate: Keep track if the food vendor at a sports stadium has sold enough beer to meet the quota for the day.

Assignments are important to make sure that your business is meeting its goals. Whether it is the number of calls that they have to make to hospital patients in need of a primary doctor, or if it's the homes around particular neighborhoods in a region that would be interested in a smart home solution. Think of any scenario where you need to reach out to your customers and be able to manage the communication and potential leads. To grow your user base of your business or service, you need to be

able to measure your performance and have the ability to make adjustments based upon the data that is collected. The assignments service is capable in helping you make these decisions.

Below are services that could be combined with this service:

- Analytics Service: Use the Analytics service to keep track of your sales quotas, phone calls, communications, etc. Set it up on a web dashboard so that you can review performance in real-time.
- Employee Service: Employees of your business or service should always have goals such as increasing sales numbers, getting more customers in the door, more revenue, etc. Utilize the Assignments service to help create these goals.
- Reporting Service: Generate performance reports so that your team can review if whether or not the employees are meeting their goals for the day, week, month, year, etc.

6.9.1. Search Assignment Assignees

```
POST /api/{version}/assignment/assignee/search
```

Description

Search for avaiable users for creating or updating assignment.

Parameters

Type	Name	Schema
Path	**version** *required*	number(double)
Query	**accountId** *required*	integer(int64)
Query	**keyword** *optional*	string

Responses

HTTP Code	Description	Schema
200	successful operation	< AccountMiniResponse > array

6.9.2. Create Assignment

```
POST /api/{version}/assignment/create
```

Description

Create an assignment.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*		integer(int64)
Query	**active** *optional*		boolean
Query	**assigneeAccountId** *required*		integer(int64)
Query	**description** *optional*		string
Query	**metaData** *optional*		string
Query	**name** *required*		string
Query	**retailerLocationId** *optional*		integer(int64)
Query	**tags** *optional*		string

Responses

HTTP Code	Description	Schema
200	successful operation	AssignmentResponse

6.9.3. Delete Assignment

```
POST /api/{version}/assignment/delete
```

Description

Delete an assignment.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**assignmentId** *required*	assignmentId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.9.4. Get Assignment

```
POST /api/{version}/assignment/get
```

Description

Get the details of an assignment.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)

Type	Name	Description	Schema
Query	**assignmentId** *required*	assignmentId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	AssignmentResponse

6.9.5. Search Assignments

```
POST /api/{version}/assignment/search
```

Description

Search for assignments by the given parameters.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activeOnly** *required*	activeOnly	boolean	`"false"`
Query	**assigneeAccountIds** *optional*	assigneeAccountIds	string	
Query	**creatorAccountId** *optional*	creatorAccountId	integer(int64)	
Query	**currentStatusType** *optional*	currentStatusType	enum (NEW, IN_PROGRESS, SUBSCRIBED, ARCHIVED)	
Query	**descending** *required*	descending	boolean	`"true"`

Type	Name	Description	Schema	Default
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**retailerLocationIds** *optional*	retailerLocationIds	string	
Query	**sortField** *required*	sortField	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, ASSIGNEE_ID, CREATOR_ID, LOCATION_ID, LOCATION_NAME, CURRENT_STATUS, CURRENT_STATUS_TYPE, STATUSES)	`"UPDATED"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< AssignmentResponse > array

6.9.6. Create Assignment Status

```
POST /api/{version}/assignment/status/create
```

Description

Create an assignment status.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**active** *optional*	active	boolean
Query	**assignmentId** *required*	assignmentId	integer(int64)
Query	**closure** *optional*	closure	enum (PHONE, SITE_VISIT, PHONE_SITE, WEB)
Query	**connection** *optional*	connection	enum (INITIAL, FOLLOW_UP, DECLINED)
Query	**followUp** *optional*	followUp	integer(int64)
Query	**message** *optional*	message	string
Query	**method** *optional*	method	enum (PHONE, SITE_VISIT, EMAIL)
Query	**scheduledNotificationId** *optional*	scheduledNotificationId	integer(int64)
Query	**status** *optional*	status	enum (ARCHIVED, SUBSCRIBED, UNSUBSCRIBED, CONTACTED, DECLINED, NOT_CONTACTED)
Query	**toDo** *optional*	toDo	enum (SITE_VISIT, PHONE, RECONTACT, RENEWAL, CREDIT)

Responses

HTTP Code	Description	Schema
200	successful operation	AssignmentStatusResponse

6.9.7. Deletes Assignment Status

```
POST /api/{version}/assignment/status/delete
```

Description

Deletes an assignment status.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**assignmentStatusId** *required*	assignmentStatusId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.9.8. Get Assignment Status

```
POST /api/{version}/assignment/status/get
```

Description

Get an assignment status.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)

Type	Name	Description	Schema
Query	**assignmentStatusId** *required*	assignmentStatusId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	AssignmentStatusResponse

6.9.9. Search Assignment Statuses

```
POST /api/{version}/assignment/status/search
```

Description

Search on assignment statuses.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activeOnly** *required*	activeOnly	boolean	`"false"`
Query	**assigneeAccountId** *optional*	assigneeAccountId	integer(int64)	
Query	**assignmentId** *optional*	assignmentId	integer(int64)	
Query	**creatorAccountId** *optional*	creatorAccountId	integer(int64)	
Query	**descending** *required*	descending	boolean	`"true"`

Type	Name	Description	Schema	Default
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**retailerLocationId** *optional*	retailerLocationId	integer(int64)	
Query	**sortField** *required*	sortField	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, ASSIGNEE_ID, CREATOR_ID, LOCATION_ID, LOCATION_NAME, CURRENT_STATUS, CURRENT_STATUS_TYPE, TODO, CONNECTION, METHOD, STATUS, CLOSURE, MESSAGE, FOLLOW_UP)	`"CREATED"`
Query	**start** *required*	start	integer(int32)	`"0"`
Query	**statusType** *optional*	statusType	enum (ARCHIVED, SUBSCRIBED, UNSUBSCRIBED, CONTACTED, DECLINED, NOT_CONTACTED)	

Responses

HTTP Code	Description	Schema
200	successful operation	< AssignmentStatusResponse > array

6.9.10. Update Assignment Status

```
POST /api/{version}/assignment/status/update
```

Description

Updates an assignment status.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**active** *optional*	active	boolean
Query	**assignmentStatusId** *required*	assignmentStatusId	integer(int64)
Query	**closure** *optional*	closure	enum (PHONE, SITE_VISIT, PHONE_SITE, WEB)
Query	**connection** *optional*	connection	enum (INITIAL, FOLLOW_UP, DECLINED)
Query	**followUp** *optional*	followUp	integer(int64)
Query	**message** *optional*	message	string
Query	**method** *optional*	method	enum (PHONE, SITE_VISIT, EMAIL)
Query	**scheduledNotificationId** *optional*	scheduledNotificationId	integer(int64)

Type	Name	Description	Schema
Query	**status** *optional*	status	enum (ARCHIVED, SUBSCRIBED, UNSUBSCRIBED, CONTACTED, DECLINED, NOT_CONTACTED)
Query	**toDo** *optional*	toDo	enum (SITE_VISIT, PHONE, RECONTACT, RENEWAL, CREDIT)

Responses

HTTP Code	Description	Schema
200	successful operation	AssignmentStatusResponse

6.9.11. Update Assignment

```
POST /api/{version}/assignment/update
```

Description

Updates an assignment.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*		integer(int64)
Query	**active** *optional*		boolean
Query	**assigneeAccountId** *optional*		integer(int64)
Query	**assignmentId** *required*		integer(int64)

Type	Name	Description	Schema
Query	**description** *optional*		string
Query	**metaData** *optional*		string
Query	**name** *optional*		string
Query	**retailerLocationId** *optional*		integer(int64)
Query	**tags** *optional*		string

Responses

HTTP Code	Description	Schema
200	successful operation	AssignmentResponse

6.10. Audience

The Audience service allows you to create, define, and manage the target users for your product or service. Being able to define your audience is essential when trying to target items to users that have their own personal interests that may be different than others. In general, you want to maximize your investment on marketing and promoting content to your users, and only guide the right messaging to customers most likely to respond.

Below are some example scenarios as to how these services can be used:

- Retail: Customers who use the retail app should be segmented into specific target groups based upon their browsing and purchasing patterns.
- Medical: The hospital may have a list of patients that they want to promote special discounts for various types of check-ups. The hospital should be able to segment these patients based upon their past/current diagnosis, family history, etc.
- Real Estate: The grocery store may want to promote homes that have grocery deliveries. Segment these homeowners based upon their grocery purchase patterns and what they might be interested in purchasing.

Being able to segment your user into audiences is important so that you can send the appropriate messaging and content. You can segment your audience by age, gender, operating system, device,

application, interests, location, pathing, and more. With these various categories, you have the freedom to configure how these messages and content should be delivered to your users. Customize these messaging so that they are tailored and personalized so that your users do not lose interest and increasing the likelihood of a response.

Below are great example services that could be combined with the Accounts service:

- Creative Service: Ads should be tailored with the appropriate graphics and targeted to a specific audience most likely to respond
- Offer Service: Offers should be pushed out to specific audiences that would be interested based upon their buying habits.
- Preview Persona Service: Be able to send an ad or content to an application, and be able to preview what it would look like from the user's perspective. This is a great way to validate your campaigns and promotion efforts.
- Pathing Service: Based upon your user's pathing trends (ex. Driving patterns, walking patterns, etc.) push out content based upon where you predict they will be and also where they are going to be at a particular time in the day.

6.10.1. Get Age Groups

```
POST /api/{version}/audience/ageGroups
```

Description

Gets the list of available age groups that can be selected by consumers and retailers targeting offers.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	< AgeGroupResponse > array

6.10.2. Create Audience

```
POST /api/{version}/audience/create
```

Description

Create a user defined audience.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**ageGroups** *optional*	ageGroups	string	
Query	**appKey** *optional*	appKey	string	
Query	**applicationIds** *optional*	applicationIds	string	
Query	**associateDescription** *optional*	associateDescription	string	
Query	**associateId** *optional*	associateId	integer(int64)	
Query	**associateType** *optional*	associateType	string	
Query	**audienceType** *optional*	audienceType	string	
Query	**categoryIds** *optional*	categoryIds	string	
Query	**cohortRegionsData** *optional*	cohortRegionsData	string	
Query	**description** *optional*	description	string	

Type	Name	Description	Schema	Default
Query	**deviceIds** *optional*	deviceIds	string	
Query	**deviceVersions** *optional*	deviceVersions	string	
Query	**devices** *optional*	devices	string	
Query	**endTimeOffset** *optional*	endTimeOffset	integer(int32)	
Query	**gameExperienceLevel** *optional*	gameExperienceLevel	string	
Query	**gender** *optional*	gender	string	
Query	**groupingId** *optional*	groupingId	string	
Query	**locations** *optional*	locations	string	
Query	**metaData** *optional*	metaData	string	
Query	**name** *required*		string	
Query	**radius** *optional*	radius	string	
Query	**searchTags** *optional*	searchTags	string	
Query	**sendSuggestion** *required*	sendSuggestion	boolean	`"true"`
Query	**startTimeOffset** *optional*	startTimeOffset	integer(int32)	
Query	**trilaterationTypes** *optional*	trilaterationTypes	string	

Type	Name	Description	Schema	Default
Query	**useOrder** *optional*	useOrder	boolean	
Query	**visibility** *optional*	visibility	string	

Responses

HTTP Code	Description	Schema
200	successful operation	AudienceResponse

6.10.3. Delete Audience

```
POST /api/{version}/audience/delete
```

Description

Delete an audience. The audience and account must be valid and have the appropirate permissions to view the content.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**audienceId** *required*	audienceId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.10.4. Get Devices

```
POST /api/{version}/audience/devices
```

Description

Gets the list of available devices that can be selected by consumers and retailers.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**includeInactive** *required*	includeInactive	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	< AudienceDeviceResponse > array

6.10.5. Get Experiences

```
POST /api/{version}/audience/experiences
```

Description

Gets the list of available experiences that can be selected by consumers and retailers.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.10.6. Get Audience

```
POST /api/{version}/audience/get
```

Description

Get an audience. The audience and account must be valid and have the appropriate permissions to view the content.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**albumTypesForCount** *optional*	albumTypesForCount	string	
Query	**appKey** *optional*	appKey	string	
Query	**audienceId** *required*	audienceId	integer(int64)	
Query	**returnAccountCount** *required*	returnAccountCount	boolean	`"false"`
Query	**returnAlbumCount** *required*	returnAlbumCount	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	AudienceResponse

6.10.7. Get GroupedAudiences

```
POST /api/{version}/audience/grouped/get
```

Description

Get a group of audiences. The audience and account must be valid and have the appropriate permissions to view the content.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**audienceGroupingId** *required*	audienceGroupingId	string

Responses

HTTP Code	Description	Schema
200	successful operation	AudienceResponse

6.10.8. Search Audiences

```
POST /api/{version}/audience/search
```

Description

Get the list audiences owned by the account

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	

Type	Name	Description	Schema	Default
Query	**activeOnly** *required*	activeOnly	boolean	`"false"`
Query	**albumIds** *optional*	albumIds	string	
Query	**albumTypesForCount** *optional*	albumTypesForCount	string	
Query	**appKey** *optional*	appKey	string	
Query	**audienceType** *optional*	audienceType	string	
Query	**descending** *required*	descending	boolean	`"false"`
Query	**groupByGroupingId** *required*	groupByGroupingId	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**returnAccountCount** *required*	returnAccountCount	boolean	`"false"`
Query	**returnAlbumCount** *required*	returnAlbumCount	boolean	`"false"`
Query	**returnGlobal** *optional*	returnGlobal	boolean	
Query	**sendSuggestion** *optional*	sendSuggestion	boolean	

Type	Name	Description	Schema	Default
Query	**sortField** *required*	sortField	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, NAME, DESCRIPTION, SEND_SUGGESTION, OWNER_ID, OWNER_DISPLAY, GENDER)	`"NAME"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< AudienceResponse > array

6.10.9. List Sent Suggestions

```
POST /api/{version}/audience/suggestion/latest
```

Description

Return list of recent trigger suggestions that have been sent to the user.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**suggestionType** *required*	suggestionType	string	`"OfferLocation"`

Type	Name	Description	Schema	Default
Query	**timeframe** *required*	timeframe	integer(int32)	`"3600"`

Responses

HTTP Code	Description	Schema
200	successful operation	OfferListResponse

6.10.10. List Suggestions by Audience

```
POST /api/{version}/audience/suggestion/list
```

Description

List either Missions or Offers that the user matches the assigned audience.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**limit** *required*	limit	integer(int32)	`"5"`
Query	**suggestionType** *required*	suggestionType	string	`"OfferLocation"`

Responses

HTTP Code	Description	Schema
200	successful operation	OfferListResponse

6.10.11. List Offers by Audience

```
POST /api/{version}/audience/suggestion/offersByAudience
```

Description

Get a list of offer locations based on audience information provided.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**age** *optional*	age	integer(int32)	
Query	**categoryIds** *optional*	categoryIds	string	
Query	**gender** *optional*	gender	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**limit** *required*	limit	integer(int32)	`"5"`
Query	**longitude** *optional*	longitude	number(double)	

Responses

HTTP Code	Description	Schema
200	successful operation	OfferListResponse

6.10.12. Send Suggestions

```
POST /api/{version}/audience/suggestion/send
```

Description

Use the accountId to determine the associated BillableEntity. From there get a list of all triggers associated with the BillableEntity.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**latitude** *required*	latitude	number(double)
Query	**longitude** *required*	longitude	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.10.13. Update Audience

```
POST /api/{version}/audience/update
```

Description

Update a user defined audience.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**active** *optional*	active	boolean
Query	**ageGroups** *optional*	ageGroups	string
Query	**appKey** *optional*	appKey	string

Type	Name	Description	Schema
Query	**applicationIds** *optional*	applicationIds	string
Query	**associateDescription** *optional*	associateDescription	string
Query	**associateId** *optional*	associateId	integer(int64)
Query	**associateType** *optional*	associateType	string
Query	**audienceId** *required*	audienceId	integer(int64)
Query	**audienceType** *optional*	audienceType	string
Query	**categoryIds** *optional*	categoryIds	string
Query	**cohortRegionsData** *optional*	cohortRegionsData	string
Query	**description** *optional*	description	string
Query	**deviceIds** *optional*	deviceIds	string
Query	**deviceVersions** *optional*	deviceVersions	string
Query	**devices** *optional*	devices	string
Query	**endTimeOffset** *optional*	endTimeOffset	integer(int32)
Query	**gameExperienceLevel** *optional*	gameExperienceLevel	string
Query	**gender** *optional*	gender	string
Query	**groupingId** *optional*	groupingId	string

Type	Name	Description	Schema
Query	**locations** *optional*	locations	string
Query	**metaData** *optional*	metaData	string
Query	**name** *optional*	name	string
Query	**radius** *optional*	radius	string
Query	**searchTags** *optional*	searchTags	string
Query	**sendSuggestion** *optional*	sendSuggestion	boolean
Query	**startTimeOffset** *optional*	startTimeOffset	integer(int32)
Query	**trilaterationTypes** *optional*	trilaterationTypes	string
Query	**useOrder** *optional*	useOrder	boolean
Query	**visibility** *optional*	visibility	string

Responses

HTTP Code	Description	Schema
200	successful operation	AudienceResponse

6.11. Bid

The real time bidding system allows content providers to bid against other content providers for prioritizing delivering their content to customers. Content providers will allocate a maximum bid and total spend for a period. Whenever the content provider's bid is the highest the content is delivered to the customer and the provider is charged the bid amount. Currently Sirqul supports bidding on advertisements but this could easily be applied to Offer (Coupons, Products, Events, etc), Location, Album, and many other content searches.

Below are some example scenarios as to how these services can be used:

- Retail: a retailer or business can create ads for products and specials that are then available to applications and websites to display to their customers. The asset morph service can create ads from offers as well.
- Medical: a medical practice can assign a value to an open slot and then for each doctor add up the total. When searching for a new doctor the total for each doctor sets the bid price so that the one with the least patients is listed first. When a doctor is selected the bid price will then go down.
- Real Estate: seller agents can get their listings first vs. others if they are willing to a higher price per listing. Clicking the ads will take the user to the seller's listing.
- Automotive: the car senses it needs new tires and then gets a list of tire dealers ordered by the bid price each dealership is willing to pay.
- Robotics: using each robot's battery percentage a a bid price is created where the higher the charge the greater the bid. When requesting which robot to use the one with the biggest charge would be listed first.

Applications that have large data sets submitted by competing providers would benefit from this advanced way of deciding which content to provide. Once the full set of content is determined based on targeting this system presents a way to prioritize the content while also monetizing the solution.

Below are great example services that could be combined with this service:

- Offers Service: using the asset morphing service offer and product listings can be converted to ads to increase the campaign's reach.
- Album Service: by building an ad for each album brands can then compete for listing their product line albums.
- Object Store Service: whatever custom solution is designed within the object store can be referenced in the custom data field of the ads and then bid on normally. The custom data field can also specify what to do when a customer clicks the ad.

6.11.1. Create Bid

```
POST /api/{version}/bid/create
```

Description

Creates a bid on a biddable object

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The api version	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**amountPerAction** *required*	The bid amount for actions. For ads, this is the amount that will be taken for each click.	number(double)
Query	**amountPerView** *required*	The bid amount for views. For ads, this is the amount that will be taken for each impression.	number(double)
Query	**biddableId** *required*	The id of the biddable object	integer(int64)
Query	**biddableType** *required*	A biddable object type. Possible values include: CREATIVE (ads).	string
Query	**budgetAmount** *required*	The allocated budget amount that will be used	number(double)
Query	**budgetSchedule** *required*	The schedule for when the allocated budget amount is reset	string
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string

Responses

HTTP Code	Description	Schema
200	successful operation	BidResponse

6.11.2. Delete Bid

```
POST /api/{version}/bid/delete
```

Description

Deleted a bid on a biddable object

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The api version	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**bidId** *required*	The bid id	integer(int64)
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.11.3. Get Bid

```
POST /api/{version}/bid/get
```

Description

Get the bid details of a biddable object

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The api version	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**bidId** *required*	The bid id	integer(int64)
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string

Responses

HTTP Code	Description	Schema
200	successful operation	BidResponse

6.11.4. Update Bid

```
POST /api/{version}/bid/update
```

Description

Updates a bid on a biddable object

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The api version	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**amountPerAction** *optional*	The bid amount for actions. For ads, this is the amount that will be taken for each click.	number(double)
Query	**amountPerView** *optional*	The bid amount for views. For ads, this is the amount that will be taken for each impression.	number(double)
Query	**bidId** *required*	The bid id	integer(int64)
Query	**budgetAmount** *optional*	The allocated budget amount that will be used	number(double)
Query	**budgetSchedule** *optional*	The schedule for when the allocated budget amount is reset	string
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string

Responses

HTTP Code	Description	Schema
200	successful operation	BidResponse

6.12. BillableEntity

The Billable Entity of an account is assumed responsible to receive all billing inquires that require a payment to be made. An example scenario to utilize the billing entity service is when a customer owes your business or entity a payment under a monthly billing model. If you are invoicing the customer on a monthly billing cycle, the inquiry must be sent to the billable entity of the account that you are charging. As it is critical to ensure safe and secure transactions to be made, only one billable entity is allowed per account.

Below are some example scenarios as to how these services can be used:

- Retail: Retailer sends invoice to a customer for an online purchase that they made. The customer has a billable entity established and sends payment that is owed to the retailer.
- Medical: Patient is billed for their medication subscription. The hospital sends the invoice to the billable entity of the patient's account.
  Real Estate: Homeowner has a monthly subscription for delivery of groceries made to his apartment. The customer is invoiced on a monthly basis based upon all items that were purchased during each billing cycle.
- Automotive: Car owner passes a toll bridge every weekday morning each month in order to get to work. As the car owner is the billable entity of his account, he is invoiced at the end of the month.
- Robotics: Smart vending machine uses a monthly billing model to invoice employees in an office. The employees are invoiced monthly based upon all goods that were purchased during the month.

Is is important that the billable entity of the customer is established so that the invoices are sent to the appropriate destination. For example, if your product or service allows customers to make purchases, and if you bill them based upon a monthly cycle, each transaction must be recorded in an invoice and then sent to the correct individual. As there can be multiple individuals under a single account, the billable entity is important to get right to ensure payments come in on time.

Below are great example services that could be combined with the Billable Entity service:

- Accounts Service: There can only be a single billable entity that is associated with the account that owes a payment.
- Subscriptions Service: The billable entity can be invoiced based upon a monthly subscription model.
- Billing Service: The billing service is the method of transaction used, and will be recorded as so in the invoice that is sent to the assigned billable entity on the account.

6.12.1. Create Billable

```
POST /api/{version}/billable/create
```

Description

reate a billable entity for an account. The creator is assumed to be the responsible account. An account can only have one billable entity

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**accountId** *required*	The unique accountId that made the request (either deviceId or accountId must be used)	integer(int64)
Query	**authorizeNetApiKey** *optional*	Authorize Net Api Key	string
Query	**authorizeNetTransactionKey** *optional*	Authorize Net Transaction Key	string
Query	**businessPhone** *optional*	The business phone	string
Query	**businessPhoneExt** *optional*	The business phone extension	string
Query	**city** *optional*	The city	string
Query	**deviceId** *optional*	The unique device identifier that made the request (either deviceId or accountId must be used)	string
Query	**name** *optional*	The name of the entity responsible for billing	string
Query	**postalCode** *optional*	The postal code	string
Query	**state** *optional*	The state	string

Type	Name	Description	Schema
Query	**streetAddress** *optional*	The street address	string
Query	**streetAddress 2** *optional*	Additional address information (such as a suite number, floor number, building name, or PO Box)	string

Responses

HTTP Code	Description	Schema
200	successful operation	BillableEntityResponse

6.12.2. Delete Billable

```
POST /api/{version}/billable/delete
```

Description

Mark the billable as deleted

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**accountId** *required*	The account used to perform the delete, must have rights to edit the billable entity.	integer(int64)
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.12.3. Get Billable

```
POST /api/{version}/billable/get
```

Description

Used to determine the associated BillableEntity of an account

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version to use to make the request	number(double)	
Query	**accountId** *required*	The unique accountId that made the request (either deviceId or accountId must be used)	integer(int64)	
Query	**deviceId** *optional*	The unique device identifier that made the request (either deviceId or accountId must be used)	string	
Query	**includeCounts** *optional*	Determines whether to include the retailer dash board counts into the response	boolean	`"false"`
Query	**includePayments** *optional*	Include Payments	boolean	`"true"`

Responses

HTTP Code	Description	Schema
200	successful operation	BillableEntityResponse

6.12.4. Update Billable

```
POST /api/{version}/billable/update
```

Description

Updates the billable record for an account

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**accountId** *required*	The unique accountId that made the request (either deviceId or accountId must be used). The account must have rights to edit the billable entity.	integer(int64)
Query	**authorizeNetApiKey** *optional*	Authorize Net Api Key	string
Query	**authorizeNetTransactionKey** *optional*	Authorize Net Transaction Key	string
Query	**businessPhone** *optional*	The business phone	string
Query	**businessPhoneExt** *optional*	The business phone extension	string
Query	**city** *optional*	The city	string
Query	**deviceId** *optional*	The unique device identifier that made the request (either deviceId or accountId must be used)	string
Query	**name** *optional*	The name of the entity responsible for billing	string
Query	**postalCode** *optional*	The postal code	string
Query	**state** *optional*	The state	string
Query	**streetAddress** *optional*	The street address	string
Query	**streetAddress2** *optional*	Additional address information (such as a suite number, floor number, building name, or PO Box)	string

Responses

HTTP Code	Description	Schema
200	successful operation	BillableEntityResponse

6.13. BillingInfo

The billing services allows customer accounts to save and manage a credit card on file. The customer account can then use that credit card to purchase virtual currency or valid offers in the system. The virtual currency is just like a gift card that can then be used at a later time to purchase valid offer types, those offer types include products, coupons, vouchers, event entry, media downloads, and tickets. Any solution that would like to generate revenue using Sirqul Offers needs to implement these services. These services work in conjuction with the Authorize.net payment gateway for secure online playments. All funds are transacted using the payment gateway in real time and then disbursements are made to the offer owner at a later date determined by the solution you define.

Below are some example scenarios as to how these services can be used:

- Retail: Customers are presented a list of products to purchase. After reviewing the product the customer decides to purchase the product which is added to the customer's order list and then shipped.
- Medical: A patient is granted a $30 meal credit per day during inpatient hospital stay. Using that credit the patient can order from the cafeteria and have it deducted from their current balance.
- Real Estate: Customers are presented with in home apps to enhance the smart home experience such as light dimmers based on time of day. The customer can purchase the app and then download it for use.
- Automotive: Customer is warned of an imminent service requirement, say an oil change. Using the in car display they can schedule and pay for the service appointment.

Online security is critical when dealing with customer credit cards and payment transactions. Sirqul takes the worry out of the equation with secure services and a gateway partner with extensive experience protecting your assets. The billing service provide a simple, easy yet secure way to store credit card and virtual credits. Once stored it then provides one-click purchasing of offers within the system, making the customer experience the best possible.

Below are services that could be combined with this service:

- Account: a customer account is required.
- Wallet: list of purchased items and means of tracking the status of each transaction.
- Offers: list of offers available; offer types include: products, vouchers, coupons, events, media, and tickets.
- Reports: view the purchase history of customers

6.13.1. Create Payment Method

```
POST /api/{version}/billing/create
```

Description

Add a new method of payment.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version to use to make the request	number(double)	
Query	**accountId** *required*	The account used to perform the the request	integer(int64)	
Query	**accountName** *optional*	Account Name	string	
Query	**accountNumber** *optional*	The bank account number	string	
Query	**address** *optional*	The billing address of the credit card	string	
Query	**authToken** *optional*	An authorization token for providers that provide this (like Amazon Payments)	string	
Query	**bankName** *optional*	The bank name	string	
Query	**ccv** *optional*	The 3 digit confirmation code	string	
Query	**city** *optional*	The billing city of the credit card	string	
Query	**country** *optional*	Country	string	
Query	**creditCardNumber** *optional*	The full credit card number to store on file	string	
Query	**defaultPaymentMethod** *optional*	Whether this should be the default payment method	boolean	`"true"`

Type	Name	Description	Schema	Default
Query	**expirationDate** *optional*	The credit card expiration date YYYY-MM	string	
Query	**firstName** *optional*	The first name on the credit card	string	
Query	**lastName** *optional*	The last name on the credit card	string	
Query	**paymentMethodNickname** *optional*	The nickname to give the payment method	string	
Query	**phone** *optional*	The billing phone of the credit card	string	
Query	**postalCode** *optional*	The billing zip code of the credit card	string	
Query	**provider** *optional*	The payment provider (see PaymentMethodProvider	string	`"AUTHORIZE_NET"`
Query	**routingNumber** *optional*	The bank routing number	string	
Query	**state** *optional*	The billing state of the credit card	string	
Query	**taxId** *optional*	Tax Id	string	

Responses

HTTP Code	Description	Schema
200	successful operation	PaymentTypesResponse

6.13.2. Get Payment Method

```
POST /api/{version}/billing/get
```

Description

Get the details of the user's payment method or their current default method of payment

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**accountId** *required*	The account used to perform the the request	integer(int64)
Query	**paymentMethodId** *optional*	The payment method to return details on. If this is not set, then the user's default payment method will be returned.	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	PaymentTypesResponse

6.13.3. Search Payment Methods

```
POST /api/{version}/billing/search
```

Description

Search the payment methods of an account

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	Version	number(double)	
Query	**accountId** *required*	Account Id	integer(int64)	
Query	**descending** *optional*	Descending	boolean	`"true"`
Query	**keyword** *optional*	Keyword	string	

Type	Name	Description	Schema	Default
Query	**limit** *optional*	Limit	integer(int32)	`"5"`
Query	**provider** *optional*	Provider	string	`"AUTHORIZE_NET"`
Query	**sortField** *optional*	Sort Field	string	`"UPDATED"`
Query	**start** *optional*	Start	integer(int32)	`"0"`
Query	**type** *optional*	Type	string	

Responses

HTTP Code	Description	Schema
200	successful operation	PaymentTypesResponse

6.13.4. Update Payment Method

```
POST /api/{version}/billing/update
```

Description

Update a method of payment. If the paymentMethodId is not passed in then will update their default payment method.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**accountId** *required*	The account used to perform the the request	integer(int64)
Query	**accountName** *optional*	Account Name	string
Query	**accountNumber** *optional*	The bank account number	string

Type	Name	Description	Schema
Query	**address** *optional*	Address	string
Query	**bankName** *optional*	The bank name	string
Query	**ccv** *optional*	The 3 digit confirmation code	string
Query	**city** *optional*	City	string
Query	**country** *optional*	Country	string
Query	**creditCardNumber** *optional*	The full credit card number to store on file	string
Query	**defaultPaymentMethod** *optional*	Default Payment Method	boolean
Query	**expirationDate** *optional*	The credit card expiration date YYYY-MM	string
Query	**firstName** *optional*	First Name	string
Query	**lastName** *optional*	Last Name	string
Query	**paymentMethodId** *optional*	Payment Method Id	integer(int64)
Query	**paymentMethodNickname** *optional*	Payment Method Nickname	string
Query	**phone** *optional*	Phone	string
Query	**postalCode** *optional*	Postal Code	string
Query	**routingNumber** *optional*	Routing Number	string

Type	Name	Description	Schema
Query	**state** *optional*	State	string
Query	**taxId** *optional*	Tax Id	string

Responses

HTTP Code	Description	Schema
200	successful operation	PaymentTypesResponse

6.14. CVSImport

The CSVImport service allows you to import specific data into tables inside of a database. This is useful when you need to seed large amounts of data such as businesses and products. For example, if you have an application that contains all of the businesses in the United States, and you have a CSV file that contains all of the businesses that includes their name, contact information, store hours, images, etc. you can use this service to import all of this information into your application by a click of a button.

Below are some example scenarios as to how these services can be used:

- You run a business and want to upload all of your products into a digital POS so that customers can find these items inside of an application. Up until now, everything had been a manual process and your products were never exposed on any website or application. Use the CSVImport service by having a CSV of all your cataloged products, vouchers, and coupons, and import this data so that your store's products can now be advertised onto applications for customers to view, add items to their wallet, and even make purchases before picking up the item at your store.

The CSVImport service is important because it saves you a lot of time and effort when trying to seed or onboarding businesses and products inside of your database or application. Also, not only can you import new businesses and products with the CSVImport service, you can also update existing businesses and products to keep this information as up-to-date as possible. Businesses are constantly changing and products change even faster. Use this service to update this information periodically

Below are example services that could be combined with the CSVImport service:

- Scheduled Tasks service works great with the CSVImport service, because you could dynamically feed new businesses or products into your application using a nightly update. This allows your application or service to always up-to-date with new data as well as updates to existing data
- Categories Service is also recommended to use with the CSVImport, so that you can categorize the types of businesses that you need to import such as Restaurants, Entertainment, Grocery, etc.

6.14.1. Search Status

```
POST /api/{version}/csvimport/batch/list
```

Description

Retrieves batches for a user.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	CsvImportResponse

6.14.2. Batch Status

```
POST /api/{version}/csvimport/batch/status
```

Description

Checks status of batch upload.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)

Type	Name	Description	Schema
Query	**accountId** *required*	accountId	integer(int64)
Query	**batchId** *required*	batchId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	CsvImportResponse

6.14.3. Detail Status

```
POST /api/{version}/csvimport/batch/status/details
```

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**batchId** *required*	batchId	integer(int64)	
Query	**limit** *required*	limit	integer(int64)	`"20"`
Query	**responseGroup** *required*	responseGroup	enum (SUMMARY, DETAILS, ERRORS, ALL)	`"ALL"`
Query	**start** *required*	start	integer(int64)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.14.4. Upload CSV

```
POST /api/{version}/csvimport/upload
```

Description

Uploads a CSV import file.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**fileFormat** *required*	fileFormat	enum (EXCEL, MYSQL, RFC4180, TDF)	`"RFC4180"`
Query	**importFile** *required*	importFile	ref	
Query	**uploadType** *required*	uploadType	enum (OFFERS, RETAILERS, RETAILERLOCATIONS, CATEGORIES, FILTERS)	

Responses

HTTP Code	Description	Schema
200	successful operation	CsvImportResponse

6.15. CargoType

The Cargo Type service allows you to add a new cargo type. You can also update, delete, or get details of a specific cargo type. A cargo type is associated to a specific service hub. A cargo type specifies properties of a shipment, including weight, dimensions, and palletizable features. The properties affect how the shipments are organized into routes via route optimization.

The Cargo Type service is primarily for shippers, drivers, and dispatchers who will be transporting various types of cargo.

Below are some example scenarios as to how these services can be used:

- Retail: Macy's is tracking shipments to their various stores, they use cargo types to determine what type of shipment is being used. Examples include: pallet, box, etc.
- Medical: A hospital can have different processes to handle shipments based on the cargo type. Examples include: biohazard, electronic equipment, paperwork, etc.
- Real Estate: Condo managers can use cargo type to differenciate shipments for the condo (building supplies, lobby items, etc) vs the residents (personal shipments, etc).
- Automotive: Using cargo type, drivers can choose the right vehicle for the job.
- Robotics: Using cargo type, robots know what type of cargo they are handling and do the safest, most efficient work they can.

This service is used to determine the type of cargo that is being shipped, handled, or otherwise dealt with.

Below are great example services that could be combined with this service:

- Billing service: Charge a customer based on the cargo type they select. For example, a pallet may cost more to ship than a box.
- Analytics service: Determine which types of cargo are shipped the most, or determine what cargo types specific demographics tend to ship.
- Routing service: Optimize routes based on the type of cargo being shipped.
- Notification service: Notify recipients when receiving a package with a cargo type that may require immediate assistance from the recipient.

6.15.1. Create Cargo Type

```
POST /api/{version}/cargo/type
```

Description

Create new cargo type

Parameters

Type	Name	Schema
Path	**version** *required*	string

Type	Name	Schema
Body	**body** *optional*	CargoType

Responses

HTTP Code	Description	Schema
200	successful operation	CargoType

6.15.2. Search Cargo Type

```
GET /api/{version}/cargo/type
```

Description

Search for types of cargo

Parameters

Type	Name	Schema	Default
Path	**version** *required*	number(double)	
Query	**activeOnly** *required*	boolean	`"true"`
Query	**descending** *required*	boolean	`"false"`
Query	**hubId** *optional*	integer(int64)	
Query	**limit** *required*	integer(int32)	`"20"`
Query	**retailerId** *optional*	integer(int64)	
Query	**sortField** *required*	string	`"id"`
Query	**start** *required*	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< CargoType > array

6.15.3. Get Cargo Type

```
GET /api/{version}/cargo/type/{cargoTypeId}
```

Description

Get an existing cargo type

Parameters

Type	Name	Schema
Path	**cargoTypeId** *required*	integer(int64)
Path	**version** *required*	string

Responses

HTTP Code	Description	Schema
200	successful operation	CargoType

6.15.4. Update Cargo Type

```
PUT /api/{version}/cargo/type/{cargoTypeId}
```

Description

Update an existing cargo type

Parameters

Type	Name	Schema
Path	**cargoTypeId** *required*	integer(int64)
Path	**version** *required*	string
Body	**body** *optional*	CargoType

Responses

HTTP Code	Description	Schema
200	successful operation	CargoType

6.15.5. Delete Cargo Type

```
DELETE /api/{version}/cargo/type/{cargoTypeId}
```

Description

Delete a type of cargo

Parameters

Type	Name	Schema
Path	**cargoTypeId** *required*	integer(int64)
Path	**version** *required*	string

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

6.16. Carrier

The Carrier service returns a list of emails that the particular phone carrier supports. For example, if the user's phone carrier is AT&T, then the format would be "1231231234@txt.att.net". This is a way so

that if you want to send SMS messages to your users, but don't want to use a service like Twilio that costs money, you could use email-to-text method. As long as the user of the account chooses which phone carrier they are assigned to in addition to their mobile number, this will be possible.

Below are some example scenarios as to how these services can be used:

- If user wants to receive alerts whenever there are changes to their upcoming trip itinerary, they must enter their phone number and phone carrier (e.g. AT&T, T-Mobile, Verizon, etc.). They should then receive SMS alerts via email

The Carrier service is important because some businesses that want to run a pilot don't want to incur costs of using services like Twilio, as it could get expensive depending on how many SMS messages are sent each day. This is an alternative service/method that allows you to send scheduled or manual SMS alerts to your users without having to incur these costs.

Below are example services that could be combined with the Carrier service:

- Scheduled Notifications, Triggers, and Notes work well together with the Carrier service because users will be able to receive scheduled SMS alerts whenever there are updates being made to their service, receive an SMS alert via Trigger when they enter a geofence, and also Notes that are added into the SMS message

6.16.1. Search Carriers

```
POST /api/{version}/carrier/search
```

Description

Search on supported mobile telephone carriers that can be used to send SMS notifications via email.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**activeOnly** *required*	activeOnly	boolean	`"true"`
Query	**descending** *required*	descending	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`

Type	Name	Description	Schema	Default
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< CellCarrierResponse > array

6.17. Category

A category is a classification or grouping that can be applied to many of the Sirqul services. Categories are defined as a tree structure with parent-child relationships, each item can only have 1 parent. Using the category services application/solution owners can manage the available categories unique to that solution and how they are related to each other. If authorized categories can be globally available to all solutions.

Categorization is important whenever searching a large database of items, otherwise you have no way to delineate the individual items and to find what you are looking for. The classification allows the customer to better self select the kind of information they would like returned. This becomes even more critical for personalization and targeting information to the appropriate audience of customers.

Below are some example scenarios as to how these services can be used:

- Retail: A restaurant application may define a set of categories by cuisine with sub-genres of those under each cuisine.
- Medical: Doctors and nurses can be categorized by specialty.
- Real Estate: Rooms might be designated by use so that lighting by default is dimmer for entertainment places versus brighter for work spaces or kitchens.
- Robotics: Parts could be categorized to indicate the sorting, fragility, or include/disclude.

The are several services in the system that utilize the categories to facilitate searching and matching, they are: Applications, Accounts, Audiences, Rating, Listings, Offers, Retailer Locations, and Retailers. It is recommended if you are generating data for these services then you should think about using categories to better classify the items for your customers. The more data present the more critical this meta-data information becomes.

Below are services that may be combined with this service:

- Accounts Service: each person can be assigned any number of categories which can then be used to

match audiences, offers, and other items to them.

- Audience Service: categories defined for an audience will match to the account categories so you can match interests.
- Rating: rate categories so that you can rate on any number of factors.
- Offers, Retailer Locations, and Retailers Services: classify offers, business, and locations for searching.

6.17.1. Create Category

```
POST /api/{version}/category/create
```

Description

Create a new category.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**active** *optional*	active	boolean
Query	**appKey** *optional*	appKey	string
Query	**assetId** *optional*	assetId	integer(int64)
Query	**description** *optional*	description	string
Query	**externalCategorySlug** *optional*	externalCategorySlug	string
Query	**externalId** *optional*	externalId	string
Query	**externalType** *optional*	externalType	string
Query	**metaData** *optional*	metaData	string

Type	Name	Description	Schema
Query	**name** *required*	name	string
Query	**parentCategoryId** *optional*	parentCategoryId	integer(int64)
Query	**sqootSlug** *optional*	sqootSlug	string
Query	**type** *optional*	type	string

Responses

HTTP Code	Description	Schema
200	successful operation	CategoryTreeResponse

6.17.2. Delete Category

```
POST /api/{version}/category/delete
```

Description

Delete a category.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**categoryId** *required*		integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.17.3. Duplicate Category

```
POST /api/{version}/category/duplicate
```

Description

Duplicate a category, including all its children.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *optional*	appKey	string
Query	**categoryId** *required*		integer(int64)
Query	**parentCategoryId** *optional*	parentCategoryId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	CategoryTreeResponse

6.17.4. Get Category

```
POST /api/{version}/category/get
```

Description

Get the details of a specific category. Recursively include all child categories and their children.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**categoryId** *required*	categoryId	integer(int64)	
Query	**returnExternal** *required*	returnExternal	boolean	`"true"`

Responses

HTTP Code	Description	Schema
200	successful operation	CategoryTreeResponse

6.17.5. Search Categories

```
POST /api/{version}/category/search
```

Description

Search for categories.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activeOnly** *required*	activeOnly	boolean	`"true"`
Query	**appKey** *optional*	appKey	string	

Type	Name	Description	Schema	Default
Query	**categoryId** *optional*	categoryId	string	
Query	**descending** *required*	descending	boolean	`"false"`
Query	**exactMatch** *required*	exactMatch	boolean	`"false"`
Query	**excludeExternalType** *optional*	excludeExternalType	boolean	
Query	**externalType** *optional*	externalType	string	
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**responseGroup** *optional*	responseGroup	enum (ALL, GLOBAL, MINE)	
Query	**returnExternal** *required*	returnExternal	boolean	`"true"`
Query	**rootOnly** *optional*	rootOnly	boolean	
Query	**sortField** *required*	sortField	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, NAME, DISPLAY)	`"DISPLAY"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< CategoryResponse > array

6.17.6. Update Category

```
POST /api/{version}/category/update
```

Description

Update a category.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**active** *optional*	active	boolean
Query	**assetId** *optional*	assetId	integer(int64)
Query	**categoryId** *required*		integer(int64)
Query	**description** *optional*	description	string
Query	**externalCategorySlug** *optional*	externalCategorySlug	string
Query	**externalId** *optional*	externalId	string
Query	**externalType** *optional*	externalType	string
Query	**metaData** *optional*	metaData	string

Type	Name	Description	Schema
Query	**name** *optional*	name	string
Query	**parentCategoryId** *optional*	parentCategoryId	integer(int64)
Query	**sqootSlug** *optional*	sqootSlug	string
Query	**type** *optional*	type	string

Responses

HTTP Code	Description	Schema
200	successful operation	CategoryTreeResponse

6.18. Collection

An album is simply a collection of items including words, photos, videos, audio, tasks, and other various objects that are presented to the user of an application. For each of these files that are uploaded into an album, users have the ability to comment, like, and dislike each of these assets. Albums can be shared with friends & other people that you have assigned in a group. Users can upload as many photos, videos, and audio files as they choose, assign each with its own name & description.

Below are some example scenarios as to how these services can be used:

- Retail: Customer opens up their shopping app and is invited to view a collection of clothes with images that are on sale for the weekend.
- Medical: Patient documents can be stored as a collection for the hospital staff members to pull up when they are needed.
- Real Estate: Home owner has a collection of automated tasks that should be triggered during the day, including automating the air conditioning, coffee being brewed, the TV turning on at a specific time of day, etc.
- Automotive: Car owner has a collection of automated tasks that should be triggered, such as the car unlocking as he approaches, air conditioning turning on if it's above a certain temperature, etc.
- Robotics: Assembly factory robots have a plan of automated tasks that they must complete.

Albums are essential in any application that involves having to keep a collection of items for the user. For example, imagine a powerful tool that enables users to share their photos and audio securely, with

the people they care about the most. Now, give users the ability to create groups, leave text and voice comments, tag their location, narrate an image, and chat in real-time. These are all compelling features that motivate users to stay active and involved inside of your application.

Below are great example services that could be combined with the Accounts service:

- Assets Service: User has the ability to store all of their images on a trip they took inside of a collection.
- Like Service: Users can 'like' an image or file that a user has uploaded inside of a collection shared to the community.
- Note Service: Users can leave notes to an image or file that a user has uploaded inside of a collection shared to the community.
- User Permissions Service: Owner of a collection of assets can share the collection so that other users can access it and upload files themselves.

6.18.1. Approve Collection

```
POST /api/{version}/collection/approve
```

Description

Sets the approval status of a collection.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**albumId** *required*	albumId	integer(int64)	
Query	**approvalStatus** *required*	approvalStatus	enum (PENDING, REJECTED, APPROVED, FEATURED)	`"APPROVED"`
Query	**deviceId** *optional*	deviceId	string	

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.18.2. Create Collection

```
POST /api/{version}/collection/create
```

Description

Create an collection.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**albumType** *optional*	albumType	string	
Query	**albumTypeId** *required*	albumTypeId	integer(int64)	`"0"`
Query	**anonymous** *required*	anonymous	boolean	`"0"`
Query	**appKey** *optional*	appKey	string	
Query	**approvalStatus** *optional*	approvalStatus	enum (PENDING, REJECTED, APPROVED, FEATURED)	
Query	**assetId** *optional*	assetId	integer(int64)	
Query	**assetsToAdd** *optional*	assetsToAdd	string	
Query	**attachedMedia** *optional*	attachedMedia	ref	

Type	Name	Description	Schema	Default
Query	**attachedMediaURL** *optional*	attachedMediaURL	string	
Query	**audienceIds** *optional*	audienceIds	string	
Query	**audienceOperator** *optional*	audienceOperator	enum (AND, OR)	
Query	**categoryFilterIds** *optional*	categoryFilterIds	string	
Query	**categoryIds** *optional*	categoryIds	string	
Query	**cellPhone** *optional*	cellPhone	string	
Query	**city** *optional*	city	string	
Query	**coverAssetNullable** *required*	coverAssetNullable	boolean	"1"
Query	**description** *optional*	description	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**fullAddress** *optional*	fullAddress	string	
Query	**gameType** *optional*	gameType	string	
Query	**includeAllAppUsersAsMembers** *optional*	includeAllAppUsersAsMembers	boolean	
Query	**includeAudiencesAsMembers** *optional*	includeAudiencesAsMembers	boolean	

Type	Name	Description	Schema	Default
Query	**includeCoverInAssetList** *required*	includeCoverInAssetList	boolean	"0"
Query	**latitude** *optional*	latitude	number(double)	
Query	**locationDescription** *optional*	locationDescription	string	
Query	**longitude** *optional*	longitude	number(double)	
Query	**media** *optional*	media	ref	
Query	**mediaURL** *optional*	mediaURL	string	
Query	**metaData** *optional*	metaData	string	
Query	**postalCode** *optional*	postalCode	string	
Query	**publicAdd** *required*	publicAdd	boolean	"0"
Query	**publicDelete** *required*	publicDelete	boolean	"0"
Query	**publicRead** *required*	publicRead	boolean	"0"
Query	**publicWrite** *required*	publicWrite	boolean	"0"
Query	**startDate** *optional*	startDate	integer(int64)	
Query	**state** *optional*	state	string	
Query	**streetAddress** *optional*	streetAddress	string	
Query	**streetAddress2** *optional*	streetAddress2	string	

Type	Name	Description	Schema	Default
Query	**subType** *optional*	subType	string	
Query	**tags** *optional*	tags	string	
Query	**title** *required*	title	string	
Query	**visibility** *optional*	visibility	enum (PUBLIC, PRIVATE, FRIENDS)	`"PRIVATE"`

Responses

HTTP Code	Description	Schema
200	successful operation	AlbumResponse

6.18.3. Delete Collection

```
POST /api/{version}/collection/delete
```

Description

Deletes a collection

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**albumId** *required*	albumId	integer(int64)
Query	**deviceId** *optional*	deviceId	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.18.4. Get Collection

```
GET /api/{version}/collection/get
```

Description

Get a collection.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**albumId** *required*	albumId	integer(int64)	
Query	**assetPreviewSize** *required*	assetPreviewSize	integer(int32)	`"50"`
Query	**audiencePreviewSize** *required*	audiencePreviewSize	integer(int32)	`"20"`
Query	**connectionPreviewSize** *required*	connectionPreviewSize	integer(int32)	`"20"`
Query	**deviceId** *optional*	deviceId	string	
Query	**likePreviewSize** *required*	likePreviewSize	integer(int32)	`"20"`
Query	**notePreviewSize** *required*	notePreviewSize	integer(int32)	`"20"`

Type	Name	Description	Schema	Default
Query	**returnNulls** *required*	returnNulls	boolean	`"true"`

Responses

HTTP Code	Description	Schema
200	successful operation	AlbumResponse

6.18.5. Search Collections

```
GET /api/{version}/collection/search
```

Description

Searches on collections.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**albumIds** *optional*	albumIds	string	
Query	**albumType** *optional*	albumType	string	
Query	**albumTypeId** *optional*	albumTypeId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**assetPreviewSize** *required*	assetPreviewSize	integer(int32)	`"0"`
Query	**audienceIds** *optional*	audienceIds	string	

Type	Name	Description	Schema	Default
Query	**audiencePreviewSize** *required*	audiencePreviewSize	integer(int32)	`"0"`
Query	**categoryFilterIds** *optional*	categoryFilterIds	string	
Query	**categoryIds** *optional*	categoryIds	string	
Query	**connectionAccountId** *optional*	connectionAccountId	integer(int64)	
Query	**connectionPreviewSize** *required*	connectionPreviewSize	integer(int32)	`"0"`
Query	**createdBefore** *optional*	createdBefore	integer(int64)	
Query	**createdSince** *optional*	createdSince	integer(int64)	
Query	**dateCreated** *optional*	dateCreated	integer(int64)	
Query	**descending** *required*	descending	boolean	`"true"`
Query	**deviceId** *optional*	deviceId	string	
Query	**endedBefore** *optional*	endedBefore	integer(int64)	
Query	**endedSince** *optional*	endedSince	integer(int64)	
Query	**filter** *required*	filter	string	`"PUBLIC"`
Query	**flagCountMinimum** *optional*	flagCountMinimum	integer(int64)	
Query	**includeCompletable** *required*	includeCompletable	boolean	`"false"`

Type	Name	Description	Schema	Default
Query	**includeFavorited** *required*	includeFavorited	boolean	`"false"`
Query	**includeInactive** *optional*	includeInactive	boolean	`"false"`
Query	**includeLiked** *required*	includeLiked	boolean	`"false"`
Query	**includePermissions** *required*	includePermissions	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**likePreviewSize** *required*	likePreviewSize	integer(int32)	`"0"`
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**limitPerAlbumType** *optional*	limitPerAlbumType	integer(int32)	
Query	**longitude** *optional*	longitude	number(double)	
Query	**mediaId** *optional*	mediaId	integer(int64)	
Query	**minStackPerPage** *required*	minStackPerPage	integer(int32)	`"20"`
Query	**notePreviewSize** *required*	notePreviewSize	integer(int32)	`"0"`
Query	**ownerId** *optional*	ownerId	integer(int64)	
Query	**range** *required*	range	number(double)	`"25.0"`

Type	Name	Description	Schema	Default
Query	**removeFlaggedContent** *required*	removeFlaggedContent	boolean	`"true"`
Query	**searchMode** *optional*	searchMode	enum (RDS, LUCENE, CLOUDINDEX)	
Query	**sortField** *required*	sortField	string	`"ALBUM_CREATED"`
Query	**stackDetails** *required*	stackDetails	boolean	`"false"`
Query	**stackPaginationIdentifier** *optional*	stackPaginationIdentifier	string	
Query	**stackSearch** *required*	stackSearch	boolean	`"false"`
Query	**stackWindowSize** *optional*	stackWindowSize	integer(int32)	
Query	**start** *required*	start	integer(int32)	`"0"`
Query	**startedBefore** *optional*	startedBefore	integer(int64)	
Query	**startedSince** *optional*	startedSince	integer(int64)	
Query	**subType** *optional*	subType	string	
Query	**updatedBefore** *optional*	updatedBefore	integer(int64)	
Query	**updatedSince** *optional*	updatedSince	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	< AlbumResponse > array

6.18.6. Update Collection

```
POST /api/{version}/collection/update
```

Description

Update a collection.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**active** *optional*	active	boolean
Query	**albumId** *required*	albumId	integer(int64)
Query	**albumType** *optional*	albumType	string
Query	**albumTypeId** *optional*	albumTypeId	integer(int64)
Query	**anonymous** *optional*	anonymous	boolean
Query	**assetId** *optional*	assetId	integer(int64)
Query	**assetsToAdd** *optional*	assetsToAdd	string
Query	**assetsToRemove** *optional*	assetsToRemove	string
Query	**audienceIds** *optional*	audienceIds	string

Type	Name	Description	Schema
Query	**audienceIdsToAdd** *optional*	audienceIdsToAdd	string
Query	**audienceIdsToRemove** *optional*	audienceIdsToRemove	string
Query	**audienceOperator** *optional*	audienceOperator	enum (AND, OR)
Query	**categoryFilterIds** *optional*	categoryFilterIds	string
Query	**categoryIds** *optional*	categoryIds	string
Query	**cellPhone** *optional*	cellPhone	string
Query	**city** *optional*	city	string
Query	**description** *optional*	description	string
Query	**deviceId** *optional*	deviceId	string
Query	**endDate** *optional*	endDate	integer(int64)
Query	**fullAddress** *optional*	fullAddress	string
Query	**includeAllAppUsersAsMembers** *optional*	includeAllAppUsersAsMembers	boolean
Query	**includeAudiencesAsMembers** *optional*	includeAudiencesAsMembers	boolean
Query	**latitude** *optional*	latitude	number(double)

Type	Name	Description	Schema
Query	**locationDescription** *optional*	locationDescription	string
Query	**longitude** *optional*	longitude	number(double)
Query	**media** *optional*	media	ref
Query	**mediaURL** *optional*	mediaURL	string
Query	**metaData** *optional*	metaData	string
Query	**postalCode** *optional*	postalCode	string
Query	**publicAdd** *optional*	publicAdd	boolean
Query	**publicDelete** *optional*	publicDelete	boolean
Query	**publicRead** *optional*	publicRead	boolean
Query	**publicWrite** *optional*	publicWrite	boolean
Query	**startDate** *optional*	startDate	integer(int64)
Query	**state** *optional*	state	string
Query	**streetAddress** *optional*	streetAddress	string
Query	**streetAddress2** *optional*	streetAddress2	string
Query	**subType** *optional*	subType	string
Query	**tags** *optional*	tags	string
Query	**title** *optional*	title	string

Type	Name	Description	Schema
Query	**visibility** *optional*	visibility	enum (PUBLIC, PRIVATE, FRIENDS)

Responses

HTTP Code	Description	Schema
200	successful operation	AlbumResponse

6.18.7. Add Collection Users

```
POST /api/{version}/collection/user/add
```

Description

Add users to an album as participants.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**add** *optional*	add	boolean	`"0"`
Query	**albumId** *required*	albumId	integer(int64)	
Query	**connectionGroups** *optional*	connectionGroups	string	
Query	**connections** *optional*	connections	string	
Query	**delete** *optional*	delete	boolean	`"0"`
Query	**deviceId** *optional*	deviceId	string	

Type	Name	Description	Schema	Default
Query	**includeFriendGroup** *required*	includeFriendGroup	boolean	`"0"`
Query	**read** *optional*	read	boolean	`"1"`
Query	**write** *optional*	write	boolean	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.18.8. Remove Collection Users

```
POST /api/{version}/collection/user/delete
```

Description

Remove participants of an album.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**albumId** *required*	albumId	integer(int64)	
Query	**connectionGroups** *optional*	connectionGroups	string	
Query	**connections** *optional*	connections	string	
Query	**deviceId** *optional*	deviceId	string	

Type	Name	Description	Schema	Default
Query	**removeFriendGroup** *required*	removeFriendGroup	boolean	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.18.9. Leave Collection

```
POST /api/{version}/collection/user/leave
```

Description

Allows a user to leave an album (they are no longer considered a participant). The album creator cannot leave their own albums.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**albumId** *required*	albumId	integer(int64)
Query	**deviceId** *optional*	deviceId	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.19. Completable Action

6.19.1. POST /api/{version}/completable/action/create

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**action** *optional*	action	enum (NONE, SEEN, ATTEMPTED, COMPLETED)
Query	**completableId** *required*	completableId	integer(int64)
Query	**completableType** *required*	completableType	string
Query	**deviceId** *optional*	deviceId	string
Query	**metaData** *optional*	metaData	string
Query	**notificationCount** *optional*	notificationCount	integer(int32)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.19.2. POST /api/{version}/completable/action/get

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)

Type	Name	Description	Schema
Query	**accountId** *required*	accountId	integer(int64)
Query	**completableActionId** *required*	completableActionId	integer(int64)
Query	**deviceId** *optional*	deviceId	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.19.3. POST /api/{version}/completable/action/search

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**completableIds** *optional*	completableIds	string	
Query	**completableType** *optional*	completableType	string	
Query	**descending** *required*	descending	boolean	`"false"`
Query	**deviceId** *optional*	deviceId	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**ownerAccountId** *optional*	ownerAccountId	integer(int64)	

Type	Name	Description	Schema	Default
Query	**sortField** *required*	sortField	enum (ID, CREATED, UPDATED)	`"ID"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.19.4. POST /api/{version}/completable/action/update

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**action** *optional*	action	enum (NONE, SEEN, ATTEMPTED, COMPLETED)
Query	**completableActionId** *required*	completableActionId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**metaData** *optional*	metaData	string
Query	**notificationCount** *optional*	notificationCount	integer(int32)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.20. Connection

The connection services allows for the creation of social graphs by relating accounts to one another, forming custom groups, and facilitating friend/follow requests. Using the standard accept friend request model users can build up a list of friends within the Sirqul system; then they can form groups, share content, and many other social features critical to engaging solutions. Using a list of friends your solutions could show the recent content submissions such as comments, tasks, or event accepts to name just a few.

Below are some example scenarios as to how these services can be used:

- Retail: A loyalty group can be contacted with early-bird specials and other VIP announcements,
- Medical: Create support groups for segments of patients to interact.
- Real Estate: Connect with friends and family so the home system can greet guests as they enter the home.
- Automotive: By following the service center manager the owner can get updates and changes to the service center hours and provided services.
- Robotics: A cohort of automatons can notify the group of tasks being completed.

Use this service to empower users to create relationships and grow your community. Creating engagement and social communities will take your solutions to the next level.

Below are great example services that could be combined with this service:

- Account Service: get the profiles of follows and see their latest actions or comments; a doctor can see what actions the nurses have been taking with a patient.
- Ranking Service: see how family members rank against each other for chores completed this week.
- Favorites Service: retrieve a list of favorite offers (products, events, media, etc), locations, or album.
- Album Service: invite all members of a group to an album to discuss a book, movie, or upcoming medical procedure.

6.20.1. Add Connections

```
POST /api/{version}/connection/group/addConnections
```

Description

Adds a list of connections to a group.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**connectionAccountIds** *optional*	connectionAccountIds	string
Query	**connectionGroupId** *required*	connectionGroupId	integer(int64)
Query	**connectionIds** *optional*	connectionIds	string
Query	**deviceId** *optional*	deviceId	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.20.2. Remove Connections

```
POST /api/{version}/connection/group/removeConnections
```

Description

Remove a list of connections from a group.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**connectionAccountIds** *optional*	connectionAccountIds	string
Query	**connectionGroupId** *required*	connectionGroupId	integer(int64)
Query	**connectionIds** *optional*	connectionIds	string
Query	**deviceId** *optional*	deviceId	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.20.3. Search Connection Groups

```
POST /api/{version}/connection/group/search
```

Description

Gets a user's private groups and default groups.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activeOnly** *required*	activeOnly	boolean	`"false"`
Query	**appKey** *optional*	appKey	string	
Query	**descending** *required*	descending	boolean	`"true"`
Query	**deviceId** *optional*	deviceId	string	
Query	**keyword** *optional*	keyword	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**longitude** *optional*	longitude	number(double)	
Query	**sortField** *required*	sortField	enum (CREATED, UPDATED, ACTIVE, NAME, DESCRIPTION, CAN_VIEW_PROFILE_INFO, CAN_VIEW_GAME_INFO, CAN_VIEW_FRIEND_INFO, INVITE_CODE, OWNER_DISPLAY, OWNER_USERNAME)	`"CREATED"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< ConnectionInfoResponse > array

6.20.4. Search Possible Connections

```
POST /api/{version}/connection/search
```

Description

Search for accounts that the user may not have a connection with.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**_i** *optional*	_i	integer(int32)	
Query	**_l** *optional*	_l	integer(int32)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**descending** *required*	descending	boolean	`"false"`
Query	**deviceId** *optional*	deviceId	string	
Query	**gameType** *optional*	gameType	string	
Query	**hasLocation** *optional*	hasLocation	boolean	
Query	**keyword** *optional*	keyword	string	
Query	**latitude** *optional*	latitude	number(double)	

Type	Name	Description	Schema	Default
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**longitude** *optional*	longitude	number(double)	
Query	**q** *optional*	q	string	
Query	**returnNulls** *required*	returnNulls	boolean	`"true"`
Query	**sortField** *optional*	sortField	string	
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	ConnectionListResponse

6.20.5. Create or Update Connection

```
POST /api/{version}/consumer/connection/add
```

Description

Creates or updates the connection of the user and another account. Allows a user to follow, block, mark as trusted, and/or add someone to a group.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	

Type	Name	Description	Schema	Default
Query	**connectionAccountId** *optional*	connectionAccountId	integer(int64)	"0"
Query	**connectionId** *optional*	connectionId	integer(int64)	"0"
Query	**connectionResponse** *optional*	connectionResponse	boolean	
Query	**deviceId** *optional*	deviceId	string	
Query	**gameType** *optional*	gameType	string	
Query	**groupId** *optional*	groupId	integer(int64)	"0"
Query	**ignoreFriendRequest** *optional*	ignoreFriendRequest	boolean	
Query	**isBlocked** *optional*	isBlocked	boolean	
Query	**isContact** *optional*	isContact	boolean	
Query	**isFollowing** *optional*	isFollowing	boolean	
Query	**isTrusted** *optional*	isTrusted	boolean	
Query	**pendingId** *optional*	pendingId	integer(int64)	"0"

Responses

HTTP Code	Description	Schema
200	successful operation	ConnectionResponse

6.20.6. Search Connections

```
POST /api/{version}/consumer/connection/get
```

Description

Gets the connections.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**_i** *optional*	_i	integer(int32)	
Query	**_l** *optional*	_l	integer(int32)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**connectionAccountId** *optional*	connectionAccountId	integer(int64)	
Query	**descending** *required*	descending	boolean	`"false"`
Query	**deviceId** *optional*	deviceId	string	
Query	**filter** *required*	filter	string	`"FRIENDS,EXCLUDE_THIRD_PARTY"`
Query	**keyword** *optional*	keyword	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**limit** *required*	limit	integer(int32)	`"0"`
Query	**longitude** *optional*	longitude	number(double)	
Query	**q** *optional*	q	string	

Type	Name	Description	Schema	Default
Query	**returnNulls** *required*	returnNulls	boolean	`"true"`
Query	**sortField** *required*	sortField	string	`"DISPLAY"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	ConnectionListResponse

6.20.7. Get Sent Friend Requests

```
POST /api/{version}/consumer/connection/getRequested
```

Description

Gets the connection sent friend requests.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**deviceId** *optional*	deviceId	string

Responses

HTTP Code	Description	Schema
200	successful operation	ConnectionListResponse

6.20.8. Create or Update Connection Group

```
POST /api/{version}/consumer/connection/group
```

Description

Creates a new private group.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	
Query	**assetId** *optional*	assetId	integer(int64)	
Query	**canViewFriendInfo** *optional*	canViewFriendInfo	boolean	"1"
Query	**canViewGameInfo** *optional*	canViewGameInfo	boolean	"1"
Query	**canViewProfileInfo** *optional*	canViewProfileInfo	boolean	"1"
Query	**connections** *optional*	connections	string	
Query	**description** *optional*	description	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**groupId** *optional*	groupId	integer(int64)	
Query	**latitude** *optional*	latitude	number(double)	

Type	Name	Description	Schema	Default
Query	**longitude** *optional*	longitude	number(double)	
Query	**metaData** *optional*	metaData	string	
Query	**name** *optional*	name	string	
Query	**returnNulls** *required*	returnNulls	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.20.9. Add Connection

```
POST /api/{version}/consumer/connection/group/addConnection
```

Description

Adds a connection to a group.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**connectionAccountId** *optional*	connectionAccountId	integer(int64)	`"0"`
Query	**connectionId** *optional*	connectionId	integer(int64)	`"0"`
Query	**deviceId** *optional*	deviceId	string	

Type	Name	Description	Schema	Default
Query	**groupId** *required*	groupId	integer(int64)	
Query	**latitude** *optional*	latitude	number(double)	
Query	**longitude** *optional*	longitude	number(double)	
Query	**pendingId** *optional*	pendingId	integer(int64)	`"0"`
Query	**returnNulls** *required*	returnNulls	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.20.10. Add Connection Groups

```
POST /api/{version}/consumer/connection/group/addSubGroup
```

Description

Add sub groups to a group.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**groupId** *required*	groupId	integer(int64)	
Query	**latitude** *optional*	latitude	number(double)	

Type	Name	Description	Schema	Default
Query	**longitude** *optional*	longitude	number(double)	
Query	**returnNulls** *required*	returnNulls	boolean	`"false"`
Query	**subGroupIds** *required*	subGroupIds	string	

Responses

HTTP Code	Description	Schema
200	successful operation	ConnectionGroupResponse

6.20.11. Get Connection Group

```
POST /api/{version}/consumer/connection/group/details/get
```

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**combineConnections** *required*	combineConnections	boolean	`"true"`
Query	**deviceId** *optional*	deviceId	string	
Query	**groupId** *optional*	groupId	integer(int64)	
Query	**latitude** *optional*	latitude	number(double)	
Query	**longitude** *optional*	longitude	number(double)	

Responses

HTTP Code	Description	Schema
200	successful operation	ConnectionGroupResponse

6.20.12. Delete Connection Group

```
POST /api/{version}/consumer/connection/group/remove
```

Description

Remove a user's group.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**groupId** *required*	groupId	integer(int64)	
Query	**latitude** *optional*	latitude	number(double)	
Query	**longitude** *optional*	longitude	number(double)	
Query	**returnNulls** *required*	returnNulls	boolean	`"true"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.20.13. Delete Connection

```
POST /api/{version}/consumer/connection/group/removeConnection
```

Description

Removes the connection from group.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**connectionAccountId** *optional*	connectionAccountId	integer(int64)	`"0"`
Query	**connectionId** *optional*	connectionId	integer(int64)	`"0"`
Query	**deviceId** *optional*	deviceId	string	
Query	**groupId** *required*	groupId	integer(int64)	
Query	**latitude** *optional*	latitude	number(double)	
Query	**longitude** *optional*	longitude	number(double)	
Query	**pendingId** *optional*	pendingId	integer(int64)	`"0"`
Query	**returnNulls** *required*	returnNulls	boolean	`"true"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.20.14. Remove Connection Groups

```
POST /api/{version}/consumer/connection/group/removeSubGroup
```

Description

Remove sub groups from a group

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**groupId** *required*	groupId	integer(int64)	
Query	**latitude** *optional*	latitude	number(double)	
Query	**longitude** *optional*	longitude	number(double)	
Query	**returnNulls** *required*	returnNulls	boolean	`"false"`
Query	**subGroupIds** *required*	subGroupIds	string	

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.20.15. Accept Friend

```
POST /api/{version}/consumer/friend/accept
```

Description

Accept a friend request and optionally sends a notification.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**friendAccountId** *required*	friendAccountId	integer(int64)	
Query	**gameType** *optional*	gameType	string	
Query	**notificationMessage** *optional*	notificationMessage	string	
Query	**notifyFriend** *required*	notifyFriend	boolean	`"true"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.20.16. Decline Friend

```
POST /api/{version}/consumer/friend/reject
```

Description

Request a friend request and optionally sends a notification.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**friendAccountId** *required*	friendAccountId	integer(int64)	
Query	**gameType** *optional*	gameType	string	
Query	**notificationMessage** *optional*	notificationMessage	string	
Query	**notifyFriend** *optional*	notifyFriend	boolean	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.20.17. Delete Friend

```
POST /api/{version}/consumer/friend/remove
```

Description

Removes a friend from the user's friends list.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**friendAccountId** *required*	friendAccountId	integer(int64)	
Query	**notifyFriend** *optional*	notifyFriend	boolean	`"0"`
Query	**removeFromGroups** *optional*	removeFromGroups	boolean	`"1"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.20.18. Request Friend

```
POST /api/{version}/consumer/friend/request
```

Description

Sends a friend request notification to another user.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	

Type	Name	Description	Schema	Default
Query	**approvalNeeded** *required*	approvalNeeded	boolean	`"true"`
Query	**deviceId** *optional*	deviceId	string	
Query	**friendAccountId** *required*	friendAccountId	integer(int64)	
Query	**gameType** *optional*	gameType	string	
Query	**notificationMessage** *optional*	notificationMessage	string	

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.21. Contest

The contests service allows you to push out content and gain feedback from your users via comments, ratings, likes, and much more. A contest itself can be in the form of a question that allows your users to choose between options or choices. The options or choices themselves can be in the form of text or media assets for users to then select. As an example, a contest can be a poll to allow your users to choose between the types of deals or products they would like to see when they enter your retail store location. Based upon the selections, you can make better business decisions to help improve customer satisfaction and expectations.

Below are some example scenarios as to how these services can be used:

- Retail: Retailer sends a poll to their customers about the product line they may want to see for the next season.
- Medical: Patients can be given two time choices for appointments that are available for next week. The patient can respond by choosing which time they prefer.
- Real Estate: Grocery delivery service asks the homeowner if they want the delivery of their goods to be done over the weekend or during a weekday. The homeowner can choose their preference.
- Automotive: The smart car may ask the user which gas station suggestions they prefer when running low on fuel. After the user chooses their preference, the smart car will only provide

options to the gas stations that were selected.

Learning more about your users and their preferences is important for being able to improve the experience for each individual. The content that is shown to each user should be personalized and appealing. This will increase the likelihood that they spend more time on your product.

Below are great example services that could be combined with the Accounts service:

- Assets Service: Contests can utilize art assets so that the messaging is compelling for users to participate and interact with.
- Likes Service: The options that a user can choose from in a Contest can be 'liked', which can also be delegated as the user's selection.
- Analytics Service: Based upon the user's selections for your contest campaign, you should be able to dynamically review the activity and results that are coming in.
- Reports Service: Based upon a collection of information for all contests that have been completed for the campaign, generate reports that will help you make the right business decisions.

6.21.1. Create or Update Contest

```
POST /api/{version}/consumer/album/contest
```

Description

Creates or updates a contest.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**albumContestId** *optional*	albumContestId	integer(int64)	
Query	**albumId1** *optional*	albumId1	integer(int64)	
Query	**albumId2** *optional*	albumId2	integer(int64)	
Query	**appKey** *optional*	appKey	string	

Type	Name	Description	Schema	Default
Query	**connectionGroupIdsToAdd** *optional*	connectionGroupIdsToAdd	string	
Query	**connectionIdsToAdd** *optional*	connectionIdsToAdd	string	
Query	**contestType** *optional*	contestType	string	
Query	**description** *optional*	description	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**gameType** *optional*	gameType	string	
Query	**includeFriendGroup** *required*	includeFriendGroup	boolean	`"0"`
Query	**latitude** *optional*	latitude	number(double)	
Query	**locationDescription** *optional*	locationDescription	string	
Query	**longitude** *optional*	longitude	number(double)	
Query	**metaData** *optional*	metaData	string	
Query	**publicAdd** *required*	publicAdd	boolean	`"0"`
Query	**publicDelete** *required*	publicDelete	boolean	`"0"`
Query	**publicRead** *required*	publicRead	boolean	`"1"`
Query	**publicWrite** *required*	publicWrite	boolean	`"0"`

Type	Name	Description	Schema	Default
Query	**removeAlbum1** *optional*	removeAlbum1	boolean	
Query	**removeAlbum2** *optional*	removeAlbum2	boolean	
Query	**startDate** *optional*	startDate	integer(int64)	
Query	**title** *optional*	title	string	
Query	**visibility** *required*	visibility	enum (PUBLIC, PRIVATE, FRIENDS)	`"PUBLIC"`

Responses

HTTP Code	Description	Schema
200	successful operation	AlbumContestResponse

6.21.2. Approve Contest

```
POST /api/{version}/consumer/album/contest/approve
```

Description

Sets the approval status of a contest.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**albumContestId** *required*	albumContestId	integer(int64)	

Type	Name	Description	Schema	Default
Query	**approvalStatus** *required*	approvalStatus	enum (PENDING, REJECTED, APPROVED, FEATURED)	`"APPROVED"`
Query	**deviceId** *optional*	deviceId	string	

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.21.3. Create Contest

```
POST /api/{version}/consumer/album/contest/create
```

Description

Creates a contest.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**albumContestId** *optional*	albumContestId	integer(int64)	
Query	**albumId1** *optional*	albumId1	integer(int64)	
Query	**albumId2** *optional*	albumId2	integer(int64)	
Query	**albumIds** *optional*	albumIds	string	
Query	**appKey** *optional*	appKey	string	

Type	Name	Description	Schema	Default
Query	**connectionGroupIdsToAdd** *optional*	connectionGroupIdsToAdd	string	
Query	**connectionIdsToAdd** *optional*	connectionIdsToAdd	string	
Query	**contestType** *optional*	contestType	string	
Query	**description** *optional*	description	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**gameType** *optional*	gameType	string	
Query	**includeFriendGroup** *required*	includeFriendGroup	boolean	`"0"`
Query	**latitude** *optional*	latitude	number(double)	
Query	**locationDescription** *optional*	locationDescription	string	
Query	**longitude** *optional*	longitude	number(double)	
Query	**metaData** *optional*	metaData	string	
Query	**publicAdd** *required*	publicAdd	boolean	`"0"`
Query	**publicDelete** *required*	publicDelete	boolean	`"0"`
Query	**publicRead** *required*	publicRead	boolean	`"1"`
Query	**publicWrite** *required*	publicWrite	boolean	`"0"`

Type	Name	Description	Schema	Default
Query	**startDate** *optional*	startDate	integer(int64)	
Query	**title** *optional*	title	string	
Query	**visibility** *required*	visibility	enum (PUBLIC, PRIVATE, FRIENDS)	`"PUBLIC"`

Responses

HTTP Code	Description	Schema
200	successful operation	AlbumContestResponse

6.21.4. Get Contest

```
GET /api/{version}/consumer/album/contest/get
```

Description

Gets the contest object including the likes and notes

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**albumContestId** *required*	albumContestId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	AlbumContestResponse

6.21.5. Delete Contest

```
POST /api/{version}/consumer/album/contest/remove
```

Description

Deletes a contest.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**albumContestId** *required*	albumContestId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.21.6. Search Contests

```
GET /api/{version}/consumer/album/contest/search
```

Description

Searches on contests.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**_i** *optional*	_i	integer(int32)	
Query	**_l** *optional*	_l	integer(int32)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**appType** *optional*	appType	string	
Query	**contestType** *optional*	contestType	string	
Query	**dateCreated** *optional*	dateCreated	integer(int64)	
Query	**descending** *required*	descending	boolean	`"true"`
Query	**deviceId** *optional*	deviceId	string	
Query	**filter** *required*	filter	string	`"MINE"`
Query	**gameType** *optional*	gameType	string	
Query	**keyword** *optional*	keyword	string	
Query	**latitude** *optional*	latitude	number(double)	

Type	Name	Description	Schema	Default
Query	**limit** *required*	limit	integer(int32)	`"0"`
Query	**longitude** *optional*	longitude	number(double)	
Query	**ownerId** *optional*	ownerId	integer(int64)	
Query	**q** *optional*	q	string	
Query	**sortField** *required*	sortField	string	`"CREATED"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	AlbumContestListResponse

6.21.7. Update Contest

```
POST /api/{version}/consumer/album/contest/update
```

Description

Updates a contest.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**albumContestId** *required*	albumContestId	integer(int64)	

Type	Name	Description	Schema	Default
Query	**albumId1** *optional*	albumId1	integer(int64)	
Query	**albumId2** *optional*	albumId2	integer(int64)	
Query	**albumIds** *optional*	albumIds	string	
Query	**appKey** *optional*	appKey	string	
Query	**connectionGroupIdsToAdd** *optional*	connectionGroupIdsToAdd	string	
Query	**connectionIdsToAdd** *optional*	connectionIdsToAdd	string	
Query	**contestType** *optional*	contestType	string	
Query	**description** *optional*	description	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**gameType** *optional*	gameType	string	
Query	**includeFriendGroup** *required*	includeFriendGroup	boolean	`"0"`
Query	**latitude** *optional*	latitude	number(double)	
Query	**locationDescription** *optional*	locationDescription	string	
Query	**longitude** *optional*	longitude	number(double)	
Query	**metaData** *optional*	metaData	string	

Type	Name	Description	Schema	Default
Query	**publicAdd** *required*	publicAdd	boolean	`"0"`
Query	**publicDelete** *required*	publicDelete	boolean	`"0"`
Query	**publicRead** *required*	publicRead	boolean	`"1"`
Query	**publicWrite** *required*	publicWrite	boolean	`"0"`
Query	**removeAlbum1** *optional*	removeAlbum1	boolean	
Query	**removeAlbum2** *optional*	removeAlbum2	boolean	
Query	**startDate** *optional*	startDate	integer(int64)	
Query	**title** *optional*	title	string	
Query	**visibility** *required*	visibility	enum (PUBLIC, PRIVATE, FRIENDS)	`"PUBLIC"`

Responses

HTTP Code	Description	Schema
200	successful operation	AlbumContestResponse

6.21.8. Vote on Contest

```
POST /api/{version}/consumer/album/contest/vote
```

Description

Vote on an album in a contest.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**albumContestId** *required*	albumContestId	integer(int64)
Query	**albumId** *required*	albumId	integer(int64)
Query	**contestType** *optional*	contestType	string
Query	**deviceId** *optional*	deviceId	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	AlbumContestResponse

6.22. Creative

The creative services enable you to create and manage advertisements or editorials that appear in various solutions. The creative services support a set of standard advertisement sizes as well as custom sizes to fit specific solution needs. Unlike standard advertisement services creatives support custom actions for clicks; allowing deep linking to other parts of applications, replacing the application style, or standard web link loading. The custom sizing and actions enables a variety of solutions such as how-to information (for learning the application), advertising of in-app purchases, showing the top ranking account, or a branded application overlay. All of the content for these solutions can be easily managed from the advertisement dashboard without a new application release.

Below are some example scenarios as to how these services can be used:

- Retail: use Sirqul's asset morphing service to convert offers into advertisements for greater reach into applications that only support advertisements, clicking the advert can direct the customer to

the purchasing flow either in-application or online.

- Medical: remind employees of best practices and procedures they should be following during long loading screens or in just after logging in.
- Real Estate: Build a chore list that can be clicked to mark it as complete.
- Automotive: using the car's GPS get geolocated advertisements targeted to the owner of the vehicle allowing the manufacture to provide satellite radio as a free service.

The creative services need to be used in conjunction with the account, mission and audience services to be a fully featured solution. The mission services allows you to build a campaign that sets a time frame, audience targeting, and limit the total spend. The audience service allows you to specify the location, device types, and customer demographics. The account service associated demographics with customers to allow for targets matching.

Below are examples of how the services work together:

- Account Service: the customer may provide their gender, age, interests (categories), and other demographics.
- Audience Service: the advertisement owner will specify the type of customers they wish to target, the kinds of devices (IOS or Android for example), and the location + radius of the targeting.
- Mission Service: Joins the audience, creatives, and timeframe together to produce a targeted, time bound campaign.

6.22.1. Find Missions

```
POST /api/{version}/ads/find
```

Description

Get a set of ad filtered by the parameters provided.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**allocatesTickets** *optional*	allocatesTickets	boolean	

Type	Name	Description	Schema	Default
Query	**appKey** *required*	appKey	string	
Query	**appVersion** *optional*	appVersion	string	
Query	**device** *optional*	device	string	
Query	**deviceIdentifier** *optional*	deviceIdentifier	integer(int64)	
Query	**deviceVersion** *optional*	deviceVersion	string	
Query	**includeAudiences** *optional*	includeAudiences	boolean	`"false"`
Query	**latitude** *optional*	latitude	number(double)	
Query	**limit** *optional*	limit	integer(int32)	`"5"`
Query	**longitude** *optional*	longitude	number(double)	
Query	**missionIds** *optional*	missionIds	string	
Query	**randomize** *required*	randomize	boolean	`"true"`
Query	**start** *optional*	start	integer(int32)	`"0"`
Query	**targetedAdsOnly** *required*	targetedAdsOnly	boolean	`"false"`
Query	**type** *optional*	type	string	

Responses

HTTP Code	Description	Schema
200	successful operation	< MissionResponse > array

6.22.2. Add Preview

```
POST /api/{version}/creative/addpreview
```

Description

Enable this ad for preview for this account.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**creativeId** *required*	creativeId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.22.3. Create Creative

```
POST /api/{version}/creative/create
```

Description

Create a creative

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**action** *optional*		string	
Query	**active** *required*	active	boolean	`"false"`
Query	**appVersion** *optional*	appVersion	string	
Query	**assetImageId** *optional*	assetImageId	integer(int64)	
Query	**balance** *optional*	balance	number(double)	
Query	**data** *optional*	data	string	
Query	**description** *optional*	description	string	
Query	**metaData** *optional*	metaData	string	
Query	**missionId** *optional*	missionId	integer(int64)	
Query	**name** *required*	name	string	
Query	**offerId** *optional*	offerId	integer(int64)	
Query	**referenceId** *optional*	referenceId	integer(int64)	
Query	**suffix** *optional*		string	
Query	**type** *optional*	type	string	
Query	**waitForAsset** *required*	waitForAsset	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	CreativeResponse

6.22.4. Delete Creative

```
POST /api/{version}/creative/delete
```

Description

Delete a creative

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**creativeId** *required*	creativeId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.22.5. Get Creative

```
POST /api/{version}/creative/get
```

Description

Get a creative

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**creativeId** *required*	creativeId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	CreativeResponse

6.22.6. Remove Preview

```
POST /api/{version}/creative/removepreview
```

Description

Remove this ad for preview for this account.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**creativeId** *required*	creativeId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.22.7. Search Creatives

```
POST /api/{version}/creative/search
```

Description

Get a list of levels for an application, just those the account has permissions to view.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *required*	appKey	string	
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**missionId** *optional*	missionId	integer(int64)	
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< CreativeResponse > array

6.22.8. Update Creative

```
POST /api/{version}/creative/update
```

Description

Update a creative

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**action** *optional*		string
Query	**active** *optional*	active	boolean
Query	**appVersion** *optional*	appVersion	string
Query	**assetImageId** *optional*	assetImageId	integer(int64)
Query	**balance** *optional*	balance	number(double)
Query	**creativeId** *required*	creativeId	integer(int64)
Query	**data** *optional*	data	string
Query	**description** *optional*	description	string
Query	**metaData** *optional*	metaData	string
Query	**missionId** *optional*	missionId	integer(int64)
Query	**name** *optional*	name	string
Query	**referenceId** *optional*	referenceId	integer(int64)
Query	**suffix** *optional*		string
Query	**type** *optional*	type	string

Responses

HTTP Code	Description	Schema
200	successful operation	CreativeResponse

6.23. Dependent

6.23.1. Create Dependent

```
POST /api/{version}/dependent/{accountId}
```

Description

Create a dependent

Parameters

Type	Name	Schema
Path	**accountId** *required*	integer(int64)
Path	**version** *required*	number(double)
Body	**body** *optional*	Account

Responses

HTTP Code	Description	Schema
200	successful operation	Connection

6.23.2. Get Dependents

```
GET /api/{version}/dependent/{accountId}
```

Description

Get your list of dependents

Parameters

Type	Name	Schema
Path	**accountId** *required*	integer(int64)
Path	**version** *required*	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	< Connection > array

6.23.3. Delete Dependent

```
DELETE /api/{version}/dependent/{accountId}/{dependentId}
```

Description

Delete a dependent

Parameters

Type	Name	Schema
Path	**accountId** *required*	integer(int64)
Path	**dependentId** *required*	integer(int64)
Path	**version** *required*	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	Connection

6.24. Disbursement

The Disbursements service allows you to pay third-party users or vendors electronically and scheduled as opposed to by doing manual hand-written checks. For example, you have a marketplace business where there are vendors signed up that offer delivery services for customers that need shipments delivered. You will accept payments from the shippers, take a % processing fee, and need to schedule payments to your vendors what is owed to them with the remaining amount. The disbursements API will connect to an existing payments processing system like Bill.com or Authorize.net, which will then distribute money from your bank account to each of the vendor's bank accounts like a direct deposit. This can be scheduled on a weekly, monthly, or even on a per transaction basis.

Below are some example scenarios as to how these services can be used:

- Retail: Merchant disburses money to their employees through a direct deposit into their bank accounts
- Stadiums: The retail shops in a stadium will disbursement money to their employees through a direct deposit into their bank accounts
- Buildings & Spaces: Leverage the disbursements service to deposit money owed to each of the commercial owners of a building

The Disbursements service is important so that businesses can deposit money owed automatically to employees and vendors in a safe and secure manner. When there are hundreds or thousands of entities that need to be paid, having it scheduled and automated is the only way your business can scale. In addition, the disbursements service reduces the chances of errors or mistakes of incorrect funds being distributed. Since disbursements deals with real money, it's important that it is done in a secure way, meaning, if it is done automatically and scheduled, there is less of a chance for someone to tamper or making errors since they will not be handing and writing checks manually.

Below are example services that could be combined with the Accounts service:

- Wallets Service: Wallets can store the amount of money that is owed to them
- Notifications Service: Vendors can be alerted with notifications once the money from the disbursements service is been deposited
- Billable Entity Service: The information for the billable entity is what the disbursements service will use to deposit money to
- Employees: Disbursements can distribute money directly to the bank accounts of the employees of a retailer location

6.24.1. Check Disbursements

```
POST /api/{version}/disbursement/check
```

Description

Check Disbursements

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	Version	number(double)	
Query	**disbursementId** *optional*	Disbursement Id	integer(int64)	
Query	**limit** *optional*	Limit	integer(int32)	`"20"`

Responses

HTTP Code	Description	Schema
200	successful operation	DisbursementResponse

6.24.2. Create Disbursement

```
POST /api/{version}/disbursement/create
```

Description

Create Disbursement

Parameters

Type	Name	Description	Schema
Path	**version** *required*	Version	number(double)
Query	**accountId** *required*	Account Id	integer(int64)
Query	**amount** *required*	Amount	number
Query	**comment** *optional*	Comment	string

Type	Name	Description	Schema
Query	**externalId** *optional*	External Id	string
Query	**introspectionParams** *optional*	Introspection Params	string
Query	**originalSenderAccountId** *optional*	Original Sender Account Id	integer(int64)
Query	**provider** *required*	Provider	enum (AUTHORIZE_NET, AMAZON_FPS, BILL_COM, CLEAR, CLEAR_MERCHANT)
Query	**receiverAccountId** *required*	Receiver Account Id	integer(int64)
Query	**scheduledDate** *optional*	Scheduled Date	integer(int64)
Query	**title** *optional*	Title	string
Query	**type** *optional*	Type	string

Responses

HTTP Code	Description	Schema
200	successful operation	DisbursementResponse

6.24.3. Get Disbursement

```
POST /api/{version}/disbursement/get
```

Description

Get Disbursement

Parameters

Type	Name	Description	Schema
Path	**version** *required*	Version	number(double)
Query	**accountId** *required*	Account Id	integer(int64)
Query	**disbursementId** *required*	Disbursement Id	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	DisbursementResponse

6.24.4. Process Disbursements

```
POST /api/{version}/disbursement/process
```

Description

Process Disbursements

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	Version	number(double)	
Query	**disbursementId** *optional*	Disbursement Id	integer(int64)	
Query	**limit** *optional*	Limit	integer(int32)	`"20"`

Responses

HTTP Code	Description	Schema
200	successful operation	DisbursementResponse

6.24.5. Search Disbursements

```
POST /api/{version}/disbursement/search
```

Description

Search Disbursements

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	Version	number(double)	
Query	**accountId** *required*	Account Id	integer(int64)	
Query	**activeOnly** *optional*	Active Only	boolean	`"false"`
Query	**afterDate** *optional*	After Date	integer(int64)	
Query	**beforeDate** *optional*	Before Date	integer(int64)	
Query	**externalId** *optional*	External Id	string	
Query	**limit** *optional*	Limit	integer(int32)	`"20"`
Query	**providers** *optional*	Providers	string	
Query	**receiverAccountId** *optional*	Receiver Account Id	integer(int64)	
Query	**senderAccountId** *optional*	Sender Account Id	integer(int64)	

Type	Name	Description	Schema	Default
Query	**start** *optional*	Start	integer(int32)	`"0"`
Query	**statuses** *optional*	Statuses	string	
Query	**type** *optional*	Type	string	

Responses

HTTP Code	Description	Schema
200	successful operation	< DisbursementResponse > array

6.24.6. Update Disbursement

```
POST /api/{version}/disbursement/update
```

Description

Update Disbursement

Parameters

Type	Name	Description	Schema
Path	**version** *required*	Version	number(double)
Query	**accountId** *required*	Account Id	integer(int64)
Query	**amount** *optional*	Amount	number
Query	**comment** *optional*	Comment	string
Query	**disbursementId** *required*	Disbursement Id	integer(int64)
Query	**externalId** *optional*	External Id	string

Type	Name	Description	Schema
Query	**introspectionParams** *optional*	Introspection Params	string
Query	**provider** *optional*	Provider	enum (AUTHORIZE_NET, AMAZON_FPS, BILL_COM, CLEAR, CLEAR_MERCHANT)
Query	**retry** *optional*	Retry	boolean
Query	**scheduledDate** *optional*	Scheduled Date	integer(int64)
Query	**title** *optional*	Title	string

Responses

HTTP Code	Description	Schema
200	successful operation	DisbursementResponse

6.25. Employee

Manage a hierarchical account structure for employees, organizational structures, and other taxonomies. Employee records can be assigned a roll and to a location they are associated with. A manager can be assigned and if no manager is provided this is the root of that tree (many roots all allowed). Retail businesses, volunteer organizations, paramilitary groups, amongst other can all benefit from services that organize their membership roles. A command and control system may use this to determine how to appropriately traverse the command structure for an answer to a question.

Below are some example scenarios as to how these services can be used:

- Retail: a retailer with several locations and employee roles could register each employee and restrict access to parts of the system based on the assigned role.
- Medical: when a nurse needs a prescription authorized their doctor (manager) can be contacted and if they are busy their supervisor can be contacted.
- Real Estate: each family member could have a role and a content approver, when the teenager wants to watch a rated R show a push notification is sent to the dad (approver) so that he can accept or decline the request.

Organizing resources and authorization structures is useful across a spectrum of professional and recreational fields. The use of roles a simple way of blocking access to whole features of applications. Getting authorization from a superior is a good options when you want to limit but not block access entirely to certain features or content.

Below are great example services that could be combined with this service:

- Notification Service: contact a superior for temporary access to content.
- Offer Service: limit which statuses an account can set for offers, require a manager for refunds for example.
- Dashboards: show or hide features based on the current account's role.

6.25.1. Assign Employee

```
POST /api/{version}/employee/assign
```

Description

Assign An existing account to be an employee

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**employeeAccountId** *required*	employeeAccountId	integer(int64)
Query	**managerAccountId** *required*	managerAccountId	integer(int64)
Query	**role** *optional*	role	string

Responses

HTTP Code	Description	Schema
200	successful operation	EmployeeResponse

6.25.2. Assign Employee to Location

```
POST /api/{version}/employee/assignToLocation
```

Description

Assign or unassign the account to a retailer location.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**assign** *required*	assign	boolean	`"true"`
Query	**employeeAccountId** *optional*	employeeAccountId	integer(int64)	
Query	**retailerLocationId** *required*	retailerLocationId	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.25.3. Create Employee

```
POST /api/{version}/employee/create
```

Description

Create a new account record with the provided information.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appBlob** *optional*	appBlob	string
Query	**assetId** *optional*	assetId	integer(int64)
Query	**assignedDeviceId** *optional*	assignedDeviceId	string
Query	**businessPhone** *optional*	businessPhone	string
Query	**cellPhone** *optional*	cellPhone	string
Query	**cellPhoneCarrier** *optional*	cellPhoneCarrier	string
Query	**city** *optional*	city	string
Query	**country** *optional*	country	string
Query	**emailAddress** *optional*	emailAddress	string
Query	**firstName** *optional*	firstName	string
Query	**gender** *optional*	gender	enum (MALE, FEMALE, ANY)
Query	**homePhone** *optional*	homePhone	string
Query	**lastName** *optional*	lastName	string
Query	**managerAccountId** *required*	managerAccountId	integer(int64)

Type	Name	Description	Schema
Query	**middleName** *optional*	middleName	string
Query	**name** *optional*	name	string
Query	**password** *required*	password	string
Query	**prefixName** *optional*	prefixName	string
Query	**retailerLocationIds** *optional*	retailerLocationIds	string
Query	**role** *optional*	role	string
Query	**settingsAppKey** *optional*	settingsAppKey	string
Query	**state** *optional*	state	string
Query	**streetAddress** *optional*	streetAddress	string
Query	**streetAddress2** *optional*	streetAddress2	string
Query	**suffixName** *optional*	suffixName	string
Query	**title** *optional*	title	string
Query	**username** *required*	username	string
Query	**zipcode** *optional*	zipcode	string

Responses

HTTP Code	Description	Schema
200	successful operation	EmployeeResponse

6.25.4. Delete Employee

```
POST /api/{version}/employee/delete
```

Description

Set the deleted date field which marks the record as deleted.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**employeeAccountId** *required*	employeeAccountId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.25.5. Get Employee

```
POST /api/{version}/employee/get
```

Description

Get the account record for the account id provided.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)

Type	Name	Description	Schema
Query	**employeeAccountId** *required*	employeeAccountId	integer(int64)
Query	**settingsAppKey** *optional*	settingsAppKey	string

Responses

HTTP Code	Description	Schema
200	successful operation	EmployeeResponse

6.25.6. Search Employees

```
POST /api/{version}/employee/search
```

Description

Use the accountId to determine the associated BillableEntity. From there get a list of all accounts associated as managers/employees.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**_i** *optional*	_i	integer(int32)	
Query	**_l** *optional*	_l	integer(int32)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activeOnly** *required*	activeOnly	boolean	`"true"`
Query	**categoryIds** *optional*	categoryIds	string	
Query	**descending** *required*	descending	boolean	`"false"`

Type	Name	Description	Schema	Default
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**managedOnly** *optional*	managedOnly	boolean	
Query	**q** *optional*	q	string	
Query	**query** *optional*	query	string	
Query	**retailerId** *optional*	retailerId	integer(int64)	
Query	**retailerLocationId** *optional*	retailerLocationId	integer(int64)	
Query	**role** *optional*	role	string	
Query	**settingsAppKey** *optional*	settingsAppKey	string	
Query	**sortField** *optional*	sortField	enum (ID, DISPLAY, CREATED, UPDATED, ACTIVE, DELETED, LAST_LOGGED_IN, CONTACT_EMAIL, ACCOUNT_TYPE, RETAILER_LOCATION_NAME, RETAILER_NAME)	
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< EmployeeResponse > array

6.25.7. Unassign Employee

```
POST /api/{version}/employee/unassign
```

Description

Unassign An existing account to be an employee

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**employeeAccountId** *required*	employeeAccountId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	EmployeeResponse

6.25.8. Update Employee

```
POST /api/{version}/employee/update
```

Description

Update the account record with the provided information.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**active** *optional*	active	boolean
Query	**appBlob** *optional*	appBlob	string
Query	**assetId** *optional*	assetId	integer(int64)
Query	**assignedDeviceId** *optional*	assignedDeviceId	string
Query	**businessPhone** *optional*	businessPhone	string
Query	**cellPhone** *optional*	cellPhone	string
Query	**cellPhoneCarrier** *optional*	cellPhoneCarrier	string
Query	**city** *optional*	city	string
Query	**country** *optional*	country	string
Query	**emailAddress** *optional*	emailAddress	string
Query	**employeeAccountId** *required*	employeeAccountId	integer(int64)
Query	**firstName** *optional*	firstName	string
Query	**gender** *optional*	gender	enum (MALE, FEMALE, ANY)
Query	**homePhone** *optional*	homePhone	string

Type	Name	Description	Schema
Query	**lastName** *optional*	lastName	string
Query	**managerAccountId** *optional*	managerAccountId	integer(int64)
Query	**middleName** *optional*	middleName	string
Query	**name** *optional*	name	string
Query	**password** *optional*	password	string
Query	**prefixName** *optional*	prefixName	string
Query	**retailerLocationIds** *optional*	retailerLocationIds	string
Query	**role** *optional*	role	string
Query	**settingsAppKey** *optional*	settingsAppKey	string
Query	**state** *optional*	state	string
Query	**streetAddress** *optional*	streetAddress	string
Query	**streetAddress2** *optional*	streetAddress2	string
Query	**suffixName** *optional*	suffixName	string
Query	**title** *optional*	title	string
Query	**zipcode** *optional*	zipcode	string

Responses

HTTP Code	Description	Schema
200	successful operation	EmployeeResponse

6.26. Event

The event services supplement the offer event and listing services to make attending and searching easier and more feature rich. Once a customer finds a listing they like and a location showing the listing they can create an event that they, their friends, and anyone else can attend. A listing is a broadcast or occasion that can be viewed in many places; such as the super bowl. Once a customer finds a business, home, or other location where it is being aired they can create an event at that location to view the listing. Enhanced searching is also provided so customers can find events for their favorite teams, genres, and interests. Location owners or patrons can rate the location as a good place for team/participant so that other customers can search for locations supporting their interests. A similar search can be done to find attendees with similar interests.

Below are some example scenarios as to how these services can be used:

- Retail: a location that shows sports events can host events for teams participating in a match; customer can indicate they are going and exchange messages leading up to the event to generate buzz.
- Medical: a lecture series or employee training is sent out to many locations and employees can self-select where a majority wish to attend the training.
- Real Estate: a group of friends can collaborate on whose house to watch the local team's upcoming event and send message about what to bring.

This is an easy way to crowd source events at businesses or other locations. There are a multitude of possible showings or happenings to present and this allows owners to poll their customers for feedback on which to promote. Used with the rating service location owners can begin to understand the interests of their client base and target promotions that better satisfy their needs.

Below are the services that should be combined with this service:

- Listing Service: list upcoming soccer games for Real Madrid and use the event search to find locations that rate as supporter spots.
- Rating Service: by rating locations by category affiliations are formed to the listing participants (teams or groups).

6.26.1. Attend Event

```
POST /api/{version}/event/attend
```

Description

Specify whether the user is attending an event at a particular location. This can also be used as a "check-in" action.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**listingId** *optional*	listingId	integer(int64)	
Query	**longitude** *optional*	longitude	number(double)	
Query	**offerLocationId** *optional*	offerLocationId	integer(int64)	
Query	**retailerLocationId** *optional*	retailerLocationId	integer(int64)	
Query	**status** *required*	status	integer(int32)	`"0"`
Query	**transactionId** *optional*	transactionId	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	OfferResponse

6.26.2. Search Event Attendance

```
POST /api/{version}/event/attendance/search
```

Description

Searches on event type transactions. This can be used to see who is attending an event.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**affiliatedCategoryIds** *optional*	affiliatedCategoryIds	string	
Query	**customerAccountIds** *optional*	customerAccountIds	string	
Query	**descending** *required*	descending	boolean	`"true"`
Query	**deviceId** *optional*	deviceId	string	
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**listingId** *optional*	listingId	integer(int64)	
Query	**offerId** *optional*	offerId	integer(int64)	
Query	**offerLocationId** *optional*	offerLocationId	integer(int64)	

Type	Name	Description	Schema	Default
Query	**retailerId** *optional*	retailerId	integer(int64)	
Query	**retailerLocationId** *optional*	retailerLocationId	integer(int64)	
Query	**sortField** *required*	sortField	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, ACTIVATED, EXPIRES, TITLE, SUBTITLE, DETAILS, OFFER_TYPE, SPECIAL_OFFER_TYPE, OFFER_VISIBILITY, REDEEMABLE_START, REDEEMABLE_END, CUSTOMER_ID, CUSTOMER_DISPLAY, RETAILER_ID, RETAILER_NAME, RETAILER_LOCATION_ID, RETAILER_LOCATION_NAME, BILLABLE_ENTITY_ID, BILLABLE_ENTITY_NAME, RESPONSIBLE_DISPLAY)	`"CREATED"`
Query	**start** *required*	start	integer(int32)	`"0"`
Query	**startDate** *optional*	startDate	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	< EventAttendanceResponse > array

6.26.3. Create Event

```
POST /api/{version}/event/create
```

Description

Create a private event to share with associates.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *required*	active	boolean	`"true"`
Query	**categoryIds** *optional*	categoryIds	string	
Query	**details** *optional*	details	string	
Query	**filterIds** *optional*	filterIds	string	
Query	**imageAssetId** *optional*	imageAssetId	integer(int64)	
Query	**metaData** *optional*	metaData	string	
Query	**redeemableEnd** *optional*	redeemableEnd	integer(int64)	
Query	**redeemableStart** *optional*	redeemableStart	integer(int64)	

Type	Name	Description	Schema	Default
Query	**retailerLocationIds** *optional*	retailerLocationIds	string	
Query	**subTitle** *optional*	subTitle	string	
Query	**title** *required*	title	string	

Responses

HTTP Code	Description	Schema
200	successful operation	OfferResponse

6.26.4. Delete Event

```
POST /api/{version}/event/delete
```

Description

Delete an event that the user has permissions to.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**eventId** *required*	eventId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.26.5. Get Event

```
POST /api/{version}/event/get
```

Description

Get an event.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**eventId** *required*	eventId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	OfferResponse

6.26.6. Search Events

```
POST /api/{version}/event/search
```

Description

Searches on events that the account has access to.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	

Type	Name	Description	Schema	Default
Query	**activeOnly** *required*	activeOnly	boolean	`"false"`
Query	**categoryIds** *optional*	categoryIds	string	
Query	**descending** *required*	descending	boolean	`"false"`
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**filterIds** *optional*	filterIds	string	
Query	**keyword** *optional*	keyword	string	
Query	**limit** *optional*	limit	integer(int32)	`"20"`
Query	**offerAudienceIds** *optional*	offerAudienceIds	string	

Type	Name	Description	Schema	Default
Query	**sortField** *required*	sortField	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, ACTIVATED, EXPIRES, REDEEMABLE_START, REDEEMABLE_END, TITLE, SUBTITLE, DETAILS, OFFER_TYPE, SPECIAL_OFFER_TYPE, OFFER_VISIBILITY, ESTIMATED_VALUE, VOUCHER_PRICE, VOUCHER_DISCOUNT_PRICE, FULL_PRICE, DICOUNT_PRICE, TICKETS_REWARD, AVAILABILITY_DATE, RELEASE_DATE, RETAILER_ID, RETAILER_NAME, RETAILER_LOCATION_ID, RETAILER_LOCATION_NAME, BILLABLE_ENTITY_ID, BILLABLE_ENTITY_NAME, RESPONSIBLE_DISPLAY)	`"TITLE"`
Query	**start** *optional*	start	integer(int32)	`"0"`
Query	**stateDate** *optional*	stateDate	integer(int64)	

Type	Name	Description	Schema	Default
Query	**transactionAudienceIds** *optional*	transactionAudienceIds	string	

Responses

HTTP Code	Description	Schema
200	successful operation	< OfferShortResponse > array

6.26.7. Update Event

```
POST /api/{version}/event/update
```

Description

Update a private event to share with associates.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**active** *optional*	active	boolean
Query	**categoryIds** *optional*	categoryIds	string
Query	**details** *optional*	details	string
Query	**eventId** *required*	eventId	integer(int64)
Query	**filterIds** *optional*	filterIds	string
Query	**imageAssetId** *optional*	imageAssetId	integer(int64)

Type	Name	Description	Schema
Query	**metaData** *optional*	metaData	string
Query	**redeemableEnd** *optional*	redeemableEnd	integer(int64)
Query	**redeemableStart** *optional*	redeemableStart	integer(int64)
Query	**retailerLocationIds** *optional*	retailerLocationIds	string
Query	**subTitle** *optional*	subTitle	string
Query	**title** *optional*	title	string

Responses

HTTP Code	Description	Schema
200	successful operation	OfferResponse

6.27. Facebook

Authenticate users in Sirqul by leveraging Facebook. Sirqul makes authentication via the third party social network seamlessly. It allows you to pull information from their Facebook account such as likes, profile information, friends, and so forth, as long as the user grants permission to do so. The service is secure so that this information cannot be accessed by other users unless they are granted permission by both the user and also through the application.

Below are some example scenarios as to how these services can be used:

- Retail: Allow users to login through their Facebook account so that all of their favorites and likes data can be leveraged to match them with products in your store
- Smart Home: Allow your user to login via Facebook to a custom music application installed in your home, which can then pull information from your Facebook account to recommend music based upon their likes and interests

The Facebook service is important because it allows your app to leverage additional information about your user that may already be available through their Facebook account. This means that you don't have to do anything within your application to build features in order to obtain information such as

interests, likes, page follows, and so forth. This is an extremely easy and seamless way for your application to ask permission to the user to obtain this information.

- The Categories and Filters service works well with the Facebook service because it can provide your users with relevant content based upon the information that is pulled in from Facebook such as interests, likes, etc. which can all be managed as categories and filters.

6.27.1. Get Facebook Token

```
POST /api/{version}/facebook/getfbtoken
```

Description

Gets a user's Facebook token.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	TokenResponse

6.27.2. Post to Facebook

```
POST /api/{version}/facebook/graph
```

Description

Make Facebook posts on behalf of the user.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *optional*	appKey	string
Query	**assetId** *optional*	assetId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**event** *required*	event	string
Query	**gameType** *optional*	gameType	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)
Query	**permissionableId** *optional*	permissionableId	integer(int64)
Query	**permissionableType** *optional*	permissionableType	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.28. Favorite

Favorites are a list of items that a customer has selected as important or memorable. Currently offers, retailer locations, albums, and categories can be favorited so the customer can easily return to that item later or share it with their friends. You can also retrieve a list of who has favorited an item to see who thinks it is important or popular. Favoriting is a great way to make a wish list of things to purchase, a list of places to visit on a trip, or a list of song collections posted by others.

Below are some example scenarios as to how these services can be used:

- Retail: a shopping app shows a product the customer wants but not right away, just favorite it and think on the purchase.
- Real Estate: the family builds many playlist of music but the father selects a few as favorites; when the house "wakes up" in the evening the entertainment center turns on a selects one of the favorite playlists as ambient sounds.
- Automotive: the HuD detects that refueling has occurred and can prompt the driver to indicate if this is a favorite spot; when low on gas and hear a favorite location a notification can be sent suggesting a stop to fill up.

The favorite service is a quick and easy way to maintain a list of things of interest per account. Many applications utilize this as a core construct; pintrest, instagram, email (star or important), facebook (saved post), and many more. Information retrieval for customer is vital for applications with ever changing content.

Below are services that could be combined with this service:

- Album Services: using the album service you can create an instagram-like application where customers can then favorite posts for later viewing.
- Offer Services: any sort of purchasing solution that needs a way to save a potential purchase can use favorites to achieve this.
- Category Services: a set of categories could be a list of soccer teams, by favoriting a team then a personalized experience can be built around that team (or teams).

6.28.1. Create Favorite

```
POST /api/{version}/favorite/create
```

Description

Adds an offer, offer location, retailer location, or category to your favorites.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**favoritableId** *required*	favoritableId	integer(int64)
Query	**favoritableType** *required*	favoritableType	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	WrappedResponse

6.28.2. Delete Favorite

```
POST /api/{version}/favorite/delete
```

Description

Removes a favorited item from the user's favorites list.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)

Type	Name	Description	Schema
Query	**deviceId** *optional*	deviceId	string
Query	**favoritableId** *optional*	favoritableId	integer(int64)
Query	**favoritableType** *optional*	favoritableType	string
Query	**favoriteId** *optional*	favoriteId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.28.3. Get Favorite

```
POST /api/{version}/favorite/get
```

Description

Retrieves a single favorited item.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**favoriteId** *required*	favoriteId	integer(int64)
Query	**latitude** *optional*	latitude	number(double)

Type	Name	Description	Schema
Query	**longitude** *optional*	longitude	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	WrappedResponse

6.28.4. Search Favorites

```
POST /api/{version}/favorite/search
```

Description

Searches on the user's favorites.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activeOnly** *required*	activeOnly	boolean	`"true"`
Query	**connectionAccountId** *optional*	connectionAccountId	integer(int64)	
Query	**descending** *required*	descending	boolean	`"true"`
Query	**deviceId** *optional*	deviceId	string	
Query	**favoritableType** *required*	favoritableType	string	
Query	**keyword** *optional*	keyword	string	

Type	Name	Description	Schema	Default
Query	**latitude** *optional*	latitude	number(double)	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**longitude** *optional*	longitude	number(double)	
Query	**returnFullResponse** *required*	returnFullResponse	boolean	`"true"`
Query	**secondaryType** *optional*	secondaryType	string	
Query	**sortField** *required*	sortField	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, DISPLAY)	`"CREATED"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	SearchResponse

6.28.5. Who has Favorited

```
POST /api/{version}/favorite/whois
```

Description

Searches for everyone that has favorited an item

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**favoritableId** *required*	favoritableId	integer(int64)	
Query	**favoritableType** *required*	favoritableType	string	
Query	**keyword** *optional*	keyword	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**longitude** *optional*	longitude	number(double)	
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< AccountResponse > array

6.29. Filter

A filter is a classification or grouping that can be applied to many of the Sirqul services. Filters are defined as a tree structure with parent-child relationships, each item can only have 1 parent. Using the filter services application/solution owners can manage the available filters unique to that solution and how they are related to each other. If authorized, filters can be globally available to all solutions.

Categorization is important whenever searching a large database of items, otherwise you have no way to delineate the individual items and to find what you are looking for. The classification allows the customer to better self select the kind of information they would like returned. This becomes even

more critical for personalization and targeting information to the appropriate audience of customers.

Below are some example scenarios as to how these services can be used:

- Retail: A restaurant application may define a set of filters for price range, valet/parking options, and supported cards.
- Medical: Equipement can have filters for power requirements, size, or operation certificates.
- Real Estate: Rooms might be designated by bed type, view, and maximum occupancy.
- Robotics: Assembly Parts could be filtered to indicate the type and what facing it should have before being worked on.

The are several services in the system that utilize the filters to facilitate searching and matching, they are: Offers, Retailer Locations, and Retailers. It is recommended if you are generating data for these services then you should think about using filters to better classify the items for your customers. The more data present the more critical this meta-data information becomes.

Below are services that may be combined with this service:

- Offer Services: products (offer type) can be classified by pickup/delivery options, sizes, or colors so customers can see a variety of choices for the item.
- Retailer Locations: for a restaurant listing app, businesses can set a price range, parking options, or atmosphere from a list of filter choices when setting up their business.
- Retailers Services: filters defined for a business automatically applied those filters to the locations and offers.

6.29.1. Create Filter

```
POST /api/{version}/filter/create
```

Description

Create a filter

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**active** *optional*	active	boolean

Type	Name	Description	Schema
Query	**appKey** *optional*	appKey	string
Query	**description** *optional*	description	string
Query	**externalId** *optional*	externalId	string
Query	**externalType** *optional*	externalType	string
Query	**metaData** *optional*	metaData	string
Query	**name** *required*	name	string
Query	**parentFilterId** *optional*	parentFilterId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	FilterTreeResponse

6.29.2. Delete Filter

```
POST /api/{version}/filter/delete
```

Description

Delete a filter.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**filterId** *required*		integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.29.3. Get Filter

```
POST /api/{version}/filter/get
```

Description

Get the details of a specific filter. Recursively include all child filters and their children.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**filterId** *required*	filterId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	FilterTreeResponse

6.29.4. Update Filter

```
POST /api/{version}/filter/update
```

Description

Update a filter.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)

Type	Name	Description	Schema
Query	**accountId** *required*	accountId	integer(int64)
Query	**active** *optional*	active	boolean
Query	**description** *optional*	description	string
Query	**externalId** *optional*	externalId	string
Query	**externalType** *optional*	externalType	string
Query	**filterId** *required*		integer(int64)
Query	**metaData** *optional*	metaData	string
Query	**name** *optional*	name	string
Query	**parentFilterId** *optional*	parentFilterId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	FilterTreeResponse

6.30. Flag

The flag service allows a customer to flag an object that a customer deems inappropriate or offensive. Flagable objects include accounts, albums, album contests, assets, game levels, and theme descriptors. Once an object has been flagged it will no longer be included in content searches for the customer doing the flagging. Each flagable object has a threshold which can be set, once that threshold has been exceeded then the object will no longer be included in all content searches.
Below are some example scenarios as to how these services can be used:

- Retail: products being sold have a album associated with it so customers can post pictures of themselves with it; if an inappropriate image is posted it can be flagged and eliminated by the community.
- Medical: multiple x-rays are taken of a fracture; with a threshold of 1 the technician can easily discard poor quality images.

- Robotics: cleaning bots are tasked with an area and there are issues accessing the area, it is flagged and eventually removed from the task list as an invalid location.

Whenever community sourced content is used a level of moderation is necessary so that the community can self police the content. Moderation of content is important for community based solutions but can be used by self contained systems when you don't want to delete or inactivate the content. Empowering the customers and community creates a strong user base which can help promote and vitalize the solution.

Below are services that are combined with this service:

- Album Service: the content search of albums will restrict items based on the account making the request and the flags applied to each item.
- Account Service: once an account is flagged then all album, game level, and other content is restricted.
- Game Level Service: game level/tasks can be customer submitted and then self moderated by the community using flagging.

6.30.1. Create Flag

```
POST /api/{version}/flag/create
```

Description

Allows a user to flag an object that the user deems inappropriate or offensive. Flagable objects include accounts, albums, album contests, assets, game levels, and theme descriptors

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**flagDescription** *optional*	flagDescription	string
Query	**flagableId** *required*	flagableId	integer(int64)

Type	Name	Description	Schema
Query	**flagableType** *required*	flagableType	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.30.2. Delete Flag

```
POST /api/{version}/flag/delete
```

Description

Deletes a flag.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**flagableId** *optional*	flagableId	integer(int64)
Query	**flagableType** *optional*	flagableType	string
Query	**itemBeingFlaggedId** *optional*	itemBeingFlaggedId	integer(int64)

Type	Name	Description	Schema
Query	**itemBeingFlaggedType** *optional*	itemBeingFlaggedType	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.30.3. Get Flag

```
POST /api/{version}/flag/get
```

Description

Gets the details on whether the user has flagged a particular flagable object.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**flagableId** *required*	flagableId	integer(int64)
Query	**flagableType** *required*	flagableType	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	FlagResponse

6.30.4. POST /api/{version}/flag/override

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**flagCount** *required*	flagCount	integer(int32)
Query	**flagableId** *required*	flagableId	integer(int64)
Query	**flagableType** *required*	flagableType	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.30.5. Get Flag Threshold

```
POST /api/{version}/flag/threshold/get
```

Description

Get the flag threshold value on an object type for a particular application.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)

Type	Name	Description	Schema
Query	**appKey** *required*	appKey	string
Query	**itemBeingFlaggedType** *required*	itemBeingFlaggedType	string

Responses

HTTP Code	Description	Schema
200	successful operation	CountResponse

6.30.6. Update Flag Threshold

```
POST /api/{version}/flag/threshold/update
```

Description

Update the flag threshold on an object type for a particular application.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *required*		string
Query	**deviceId** *optional*	deviceId	string
Query	**itemBeingFlaggedType** *required*	itemBeingFlaggedType	string
Query	**threshold** *required*	threshold	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	CountResponse

6.31. Game

Manage task-based content that can be sourced internally by developers or externally via the community. The game level object accepts whatever content format the solution desires: csv, json, xml, binary, text, etc and wraps that with relevant metadata focused on task completion. Metadata includes, but is not limited to: the name, a description, the skill level suggested, tutorials on how to complete, and image assets. An easy example of tasks are levels for a game; other examples are photo upload challenges, posting invites to facebook, content voting, etc. Any solution that wants to task or challenge it's customers can use these services to build a set of these or ask the community to generate their own.

Below are some example scenarios as to how these services can be used:

- Retail: a coffee chain sets up a photo challenge to take pictures in front of as many locations a possible.
- Real Estate: a level contains the thermostat schedule for a particular day and can be altered by the homeowner as needed.
- Automotive: a student driver program could lead a parent and student through a series of lessons in preparation for a driver's test.
- Robotics: each level is a series of commands the automated robot should perform.

Because the game level accepts any format of data it can be used for literally anything as long as the solution can encode and decode the data. Combined with the pack and mission services you can begin to build up a sequential series of tasks that once completed gives rewards to the customer for their participation.

Below are great example services that could be combined with this service:

- Pack Service: group a set of level together into a pack; the set can then be sold as DLC, unlocked through level completion, and even level the progression order can be defined.
- Mission Service: by adding a set of levels to a mission the solution can then reward the customer once all levels have been complete; this could be offers or in game currency (tickets).
- Score Service: levels are completed by sending a score and a time to complete; if enabled the score can be used to automatically allocate application currency (tickets) which can be used to enhance the customer's experience.

6.31.1. Create Game

```
POST /api/{version}/game/create
```

Description

Create a game

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**description** *optional*	description	string	
Query	**includeGameData** *optional*	includeGameData	boolean	`"false"`
Query	**metaData** *optional*	metaData	string	
Query	**packIds** *optional*	packIds	string	
Query	**title** *optional*	title	string	

Responses

HTTP Code	Description	Schema
200	successful operation	GameResponse

6.31.2. Delete Game

```
POST /api/{version}/game/delete
```

Description

Delete a game.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**gameId** *optional*	gameId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	GameResponse

6.31.3. Get Game

```
POST /api/{version}/game/get
```

Description

Get a game.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**gameId** *optional*	gameId	integer(int64)	
Query	**includeGameData** *optional*	includeGameData	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	GameResponse

6.31.4. Search Games

```
POST /api/{version}/game/search
```

Description

Get the list of games available to the account.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**includeGameData** *optional*	includeGameData	boolean	`"false"`
Query	**includeInactive** *optional*	includeInactive	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**limit** *optional*	limit	integer(int32)	`"20"`
Query	**start** *optional*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< GameResponse > array

6.31.5. Update Game

```
POST /api/{version}/game/update
```

Description

Update a game

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	
Query	**description** *optional*	description	string	
Query	**gameId** *optional*	gameId	integer(int64)	
Query	**includeGameData** *optional*	includeGameData	boolean	`"false"`
Query	**metaData** *optional*	metaData	string	
Query	**packIds** *optional*	packIds	string	
Query	**title** *optional*	title	string	

Responses

HTTP Code	Description	Schema
200	successful operation	GameResponse

6.32. GameLevel

Manage task-based content that can be sourced internally by developers or externally via the community. The game level object accepts whatever content format the solution desires: csv, json, xml, binary, text, etc and wraps that with relevant metadata focused on task completion. Metadata includes, but is not limited to: the name, a description, the skill level suggested, tutorials on how to complete, and image assets. An easy example of tasks are levels for a game; other examples are photo upload challenges, posting invites to facebook, content voting, etc. Any solution that wants to task or challenge it's customers can use these services to build a set of these or ask the community to generate their own.

Below are some example scenarios as to how these services can be used:

- Retail: a coffee chain sets up a photo challenge to take pictures in front of as many locations a possible.
- Real Estate: a level contains the thermostat schedule for a particular day and can be altered by the homeowner as needed.
- Automotive: a student driver program could lead a parent and student through a series of lessons in preparation for a driver's test.
- Robotics: each level is a series of commands the automated robot should perform.

Because the game level accepts any format of data it can be used for literally anything as long as the solution can encode and decode the data. Combined with the pack and mission services you can begin to build up a sequential series of tasks that once completed gives rewards to the customer for their participation.

Below are great example services that could be combined with this service:

- Pack Service: group a set of level together into a pack; the set can then be sold as DLC, unlocked through level completion, and even level the progression order can be defined.
- Mission Service: by adding a set of levels to a mission the solution can then reward the customer once all levels have been complete; this could be offers or in game currency (tickets).
- Score Service: levels are completed by sending a score and a time to complete; if enabled the score can be used to automatically allocate application currency (tickets) which can be used to enhance the customer's experience.

6.32.1. Create Game Level

```
POST /api/{version}/level/create
```

Description

Create a game level. Currently does NOT support game objects.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *required*	active	boolean	`"false"`
Query	**allocateTickets** *required*	allocateTickets	boolean	`"false"`
Query	**appKey** *optional*	appKey	string	
Query	**appVersion** *optional*	appVersion	string	
Query	**assetIconId** *optional*	assetIconId	integer(int64)	
Query	**assetImageId** *optional*	assetImageId	integer(int64)	
Query	**balance** *optional*	balance	number(double)	
Query	**connectionGroupIds** *optional*	connectionGroupIds	string	
Query	**connectionIds** *optional*	connectionIds	string	
Query	**description** *optional*	description	string	
Query	**difficulty** *optional*	difficulty	string	

Type	Name	Description	Schema	Default
Query	**friendGroup** *optional*	friendGroup	boolean	
Query	**gameData** *required*	gameData	string	
Query	**gameDataSuffix** *required*	gameDataSuffix	string	
Query	**metaData** *optional*	metaData	string	
Query	**name** *required*	name	string	
Query	**offerId** *optional*	offerId	integer(int64)	
Query	**points** *optional*	points	integer(int64)	
Query	**ticketCount** *required*	ticketCount	integer(int64)	`"0"`
Query	**ticketType** *optional*	ticketType	string	
Query	**tutorialAlignment** *required*	tutorialAlignment	string	`"NONE"`
Query	**tutorialImageAssetId** *optional*	tutorialImageAssetId	integer(int64)	
Query	**tutorialMessage** *optional*	tutorialMessage	string	
Query	**tutorialTitle** *optional*	tutorialTitle	string	
Query	**visibility** *optional*	visibility	string	

Responses

HTTP Code	Description	Schema
200	successful operation	GameLevelResponse

6.32.2. Delete Game Level

```
POST /api/{version}/level/delete
```

Description

Delete a game level. The level and account must be valid and have the appropirate permissions to view the content.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**levelId** *required*	levelId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.32.3. Get Game Level

```
POST /api/{version}/level/get
```

Description

Get a game level. The level and account must be valid and have the appropirate permissions to view the content.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**includeGameData** *required*	includeGameData	boolean	`"false"`
Query	**levelId** *required*	levelId	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	GameLevelResponse

6.32.4. Get Level Questions

```
POST /api/{version}/level/questions/get
```

Description

Get questions within a level.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**levelId** *required*	levelId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	QuestionResponse

6.32.5. Update Level Questions

```
POST /api/{version}/level/questions/update
```

Description

Updates a level with question game objects.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**levelId** *required*	levelId	integer(int64)
Query	**questionIds** *required*	questionIds	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.32.6. Search Game Levels

```
POST /api/{version}/level/search
```

Description

Get a list of levels for an application, just those the account has permissions to view.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *required*	appKey	string	
Query	**appVersion** *optional*	appVersion	string	
Query	**descending** *required*	descending	boolean	`"true"`
Query	**filters** *optional*	filters	string	
Query	**includeGameData** *required*	includeGameData	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**sortField** *required*	sortField	string	`"LEVEL_UPDATED"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	GameLevelListResponse

6.32.7. Search Game Level by Billable Entity

```
POST /api/{version}/level/searchByBillableEntity
```

Description

Searches on game levels that the logged in user has access to. A user would have access if the creator of

the game level is managed under the same BillableEntity.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activeOnly** *required*	activeOnly	boolean	`"false"`
Query	**appKey** *optional*	appKey	string	
Query	**descending** *required*	descending	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int64)	`"20"`

Type	Name	Description	Schema	Default
Query	**sortField** *required*	sortField	enum (LEVEL_ACTIVE, LEVEL_NAME, LEVEL_DESCRIPTION, LEVEL_CREATED, LEVEL_UPDATED, LEVEL_LIKES, LEVEL_DISLIKES, LEVEL_NOTES, LEVEL_PLAYS, LEVEL_DOWNLOADS, LEVEL_QUITS, LEVEL_COMPLETES, LEVEL_VERSION, LEVEL_MISSION_TYPE, LEVEL_OWNER_DISPLAY, GAME_OWNER_DISPLAY, GAME_TITLE, GAME_DESCRIPTION, GAME_LIKES, GAME_DISLIKES, APP_NAME, APP_SCORING_TYPE)	`"LEVEL_NAME"`
Query	**start** *required*	start	integer(int64)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	GameLevelResponse

6.32.8. Update Game Level

```
POST /api/{version}/level/update
```

Description

Update a game level. Currently does NOT support game objects.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	
Query	**allocateTickets** *optional*	allocateTickets	boolean	
Query	**appKey** *optional*	appKey	string	
Query	**appVersion** *optional*	appVersion	string	
Query	**assetIconId** *optional*	assetIconId	integer(int64)	
Query	**assetImageId** *optional*	assetImageId	integer(int64)	
Query	**balance** *optional*	balance	number(double)	
Query	**connectionGroupIds** *optional*	connectionGroupIds	string	
Query	**connectionIds** *optional*	connectionIds	string	
Query	**description** *optional*	description	string	
Query	**difficulty** *optional*	difficulty	string	
Query	**friendGroup** *optional*	friendGroup	boolean	

Type	Name	Description	Schema	Default
Query	**gameData** *optional*	gameData	string	
Query	**gameDataSuffix** *optional*	gameDataSuffix	string	
Query	**levelId** *required*	levelId	integer(int64)	
Query	**metaData** *optional*	metaData	string	
Query	**name** *optional*	name	string	
Query	**offerId** *optional*	offerId	integer(int64)	
Query	**points** *optional*	points	integer(int64)	
Query	**ticketCount** *required*	ticketCount	integer(int64)	`"0"`
Query	**ticketType** *optional*	ticketType	string	
Query	**tutorialAlignment** *optional*	tutorialAlignment	string	
Query	**tutorialImageAssetId** *optional*	tutorialImageAssetId	integer(int64)	
Query	**tutorialMessage** *optional*	tutorialMessage	string	
Query	**tutorialTitle** *optional*	tutorialTitle	string	
Query	**visibility** *optional*	visibility	string	

Responses

HTTP Code	Description	Schema
200	successful operation	GameLevelResponse

6.32.9. Get Level Words

```
POST /api/{version}/level/words/get
```

Description

Get words within a level.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**levelId** *required*	levelId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	WordzWordResponse

6.32.10. Update Level Words

```
POST /api/{version}/level/words/update
```

Description

Updates a level with word game objects.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**levelId** *required*	levelId	integer(int64)
Query	**wordIds** *required*	wordIds	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.33. Invite

The invite service provides an easy way for a customer to share and invite their friends to experience events, albums, contests, game levels, missions, offers, and locations. The invitation is a web URL that can be shared out from the application by whatever means desired; email, sms, facebook post, etc. By clicking on the URL the invitee can be deep linked within the app (if installed) or directed to a website to login or register and download the application. Once the invitee's account is determined the item is shown and is associated with the account as is appropriate. Using the invite service is an easy way for customer to attract their friends to the application creating a viral buzz about it.

Below are some example scenarios as to how these services can be used:

- Retail: use the offer invite to send a coupon to all of a customer's friends; for each coupon added to a wallet reward 1 ticket. Tickets can be used to get discounts or get access to specialty coupons.
- Medical: when a set of x-rays is ready for review send a message to the doctor and once clicked it is added to their patient list.
- Real Estate: a parent can create a chore (game level) and then invite the children to complete it.
- Automotive: a web based road trip planner could send an invite to the customer that when accessed in the car save the location as a destination.

Attracting new customers to an application is critical to it's success. When friends engage in the same activities usage is re-enforced and the application can gain momentum and keep existing customers.

Below are great example services that could be combined with this service:

- Game Level Service: share tasks with friends or a cohort to either distribute the work or to

encourage peer participation.

- Album Service: let people know about an artist or fashion collect you like.
- Offers Service: find a great deal on hotels, share it so that others can cash in on the special while it lasts.

6.33.1. Accept Invite

```
POST /api/{version}/invite/accept
```

Description

Allows a user to accept an invite. The user could also become the inviter's friend.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**albumContestId** *optional*		integer(int64)	
Query	**albumId** *optional*		integer(int64)	
Query	**appKey** *optional*		string	
Query	**autoAttendEvent** *required*		boolean	`"false"`
Query	**autoFavoriteOffer** *required*		boolean	`"false"`
Query	**autoFavoriteOfferLocation** *required*		boolean	`"false"`

Type	Name	Description	Schema	Default
Query	**autoFavoriteRetailerLocation** *required*		boolean	`"false"`
Query	**autoFriend** *required*		boolean	`"true"`
Query	**missionId** *optional*		integer(int64)	
Query	**offerId** *optional*		integer(int64)	
Query	**offerLocationId** *optional*		integer(int64)	
Query	**retailerLocationId** *optional*		integer(int64)	
Query	**token** *required*	token	string	

Responses

HTTP Code	Description	Schema
200	successful operation	ConsumerInviteResponse

6.33.2. Invite to Collection

```
POST /api/{version}/invite/album
```

Description

Allows a user to invite people to gain access to a collection. This will generate an invite token, which when used, will give the invitee access to a collection (whether it is private or not). The invitee will also become the user's friend when the invitation is accepted.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**albumId** *optional*	albumId	integer(int64)
Query	**appId** *optional*	appId	integer(int64)
Query	**appKey** *optional*	appKey	string
Query	**deviceId** *optional*	deviceId	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	InviteResponse

6.33.3. Invite to Contest

```
POST /api/{version}/invite/albumContest
```

Description

Allows a user to invite people to gain access to a contest. This will generate an invite token, which when used, will give the invitee access to a contest (whether it is private or not). The invitee will also become the user's friend when the invitation is accepted.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)

Type	Name	Description	Schema
Query	**accountId** *required*	accountId	integer(int64)
Query	**albumContestId** *optional*	albumContestId	integer(int64)
Query	**appId** *optional*	appId	integer(int64)
Query	**appKey** *optional*	appKey	string
Query	**deviceId** *optional*	deviceId	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	InviteResponse

6.33.4. Invite to Event

```
POST /api/{version}/invite/event
```

Description

Allows a user to invite people to attend an event. This will generate an invite token, which when used, will allow the invitee to add the offer to their wallet.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)

Type	Name	Description	Schema
Query	**appKey** *required*	appKey	string
Query	**listingId** *required*	listingId	integer(int64)
Query	**receiverAccountIds** *optional*	receiverAccountIds	string
Query	**retailerLocationId** *optional*	retailerLocationId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	InviteResponse

6.33.5. Invite to Game Level

```
POST /api/{version}/invite/gameLevel
```

Description

Allows a user to invite people to gain access to an album. This will generate an invite token, which when used, will give the invitee access to an album (whether it is private or not). The invitee will also become the user's friend when the invitation is accepted.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appId** *optional*	appId	integer(int64)
Query	**appKey** *optional*	appKey	string

Type	Name	Description	Schema
Query	**deviceId** *optional*	deviceId	string
Query	**gameLevelId** *optional*	gameLevelId	integer(int64)
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	InviteResponse

6.33.6. Get Invite

```
POST /api/{version}/invite/get
```

Description

This is used to determine whether an invite token is valid. If the token is valid, this will also return information about who invited the user, and what they are invited to.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *optional*		integer(int64)
Query	**albumContestId** *optional*		integer(int64)
Query	**albumId** *optional*		integer(int64)
Query	**appKey** *optional*		string

Type	Name	Description	Schema
Query	**missionId** *optional*		integer(int64)
Query	**offerId** *optional*		integer(int64)
Query	**offerLocationId** *optional*		integer(int64)
Query	**retailerLocationId** *optional*		integer(int64)
Query	**token** *optional*		string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.33.7. Invite to Mission

```
POST /api/{version}/invite/mission
```

Description

Allows a user to invite people to gain access to a mission. This will generate an invite token, which when used, will give the invitee access to a mission (whether it is private or not). The invitee will also become the user's friend when the invitation is accepted.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appId** *optional*	appId	integer(int64)

Type	Name	Description	Schema
Query	**appKey** *optional*	appKey	string
Query	**deviceId** *optional*	deviceId	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)
Query	**missionId** *optional*	missionId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	InviteResponse

6.33.8. Invite to Offer

```
POST /api/{version}/invite/offer
```

Description

Allows a user to invite people to favorite an offer. This will generate an invite token, which when used, will give the invitee the offer in their favorite's list.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *required*	appKey	string
Query	**offerId** *required*	offerId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	InviteResponse

6.33.9. Invite to Offer Location

```
POST /api/{version}/invite/offerLocation
```

Description

Allows a user to invite people to favorite an offer location. This will generate an invite token, which when used, will give the invitee the offer location in their favorite's list.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *required*	appKey	string
Query	**offerLocationId** *required*	offerLocationId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	InviteResponse

6.33.10. Invite to Retailer Location

```
POST /api/{version}/invite/retailerLocation
```

Description

Allows a user to invite people to favorite a retailer location. This will generate an invite token, which

when used, will give the invitee the retailer location in their favorite's list.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**albumId** *optional*	albumId	integer(int64)
Query	**appKey** *required*	appKey	string
Query	**retailerLocationId** *required*	retailerLocationId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	InviteResponse

6.34. Leaderboard

The Ranking API allows you to setup various "Leaderboards" within your application. Some examples of Ranking can be "highest score" or "most shares" or "most games played". Ranking allows your users to compete against each other which drives more usage of your application. The Ranking(s) can be displayed across various time ranges such as "daily, weekly, or monthly."

Below are some example scenarios as to how the Rating service can be used:

- Retail: Setup Ranking so your customers can compete against each other to win various prizes or recognition such as "most store visits" on a monthly business.
- Medical: Create an internal Leaderboard to show leading employees in various categories such as "patients served".
- Smart Home: Use Ranking to get feedback on power usage peak times during the week, month or year. Use the data to adjust settings on devices to maximize energy efficiencies.
- Automotive: Use Ranking to track miles driven by drivers in a fleet to show most miles driven per week, per month, etc.
- Robotics: Use Ranking to keep track of uptime or number of products produced by various

assembly lines.

The Ranking API is important because it allows you to see and share scoring standings across time ranges.

Below are great example services that could be combined with the Ranking service:

- Missions: Use Missions in combination with Ranking to create a Leaderboard for most Missions accomplished by application users.
- Invites: Create a Leaderboard using Ranking to display a listing of users who successfully invite the most other users to download and use your application.
- Favorites: Use Ranking with Favorites to display the rank of the favorite items in your application.
- Retailers: Use Ranking in combination with Retailers to show a list of the most visited Retail for your business locations.
- Purchase Item: Use Rankings in combination with Purchase Item to create a list of most purchased items within your application.

6.34.1. Create a Leaderboard

```
POST /api/{version}/leaderboard/create
```

Description

Create a Leaderboard

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**bannerAssetId** *optional*	bannerAssetId	integer(int64)	
Query	**bannerMedia** *optional*	bannerMedia	ref	
Query	**description** *optional*	description	string	

Type	Name	Description	Schema	Default
Query	**iconAssetId** *optional*	iconAssetId	integer(int64)	
Query	**iconMedia** *optional*	iconMedia	ref	
Query	**leaderboardMode** *required*	leaderboardMode	string	`"GLOBAL"`
Query	**limitation** *optional*	limitation	integer(int32)	
Query	**metaData** *optional*	metaData	string	
Query	**rankType** *required*	rankType	string	`"POINTS"`
Query	**sortField** *required*	sortField	string	`"TOTAL"`
Query	**title** *optional*	title	string	

Responses

HTTP Code	Description	Schema
200	successful operation	LeaderboardResponse

6.34.2. Delete a Leaderboard

```
POST /api/{version}/leaderboard/delete
```

Description

Delete a Leaderboard

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)

Type	Name	Description	Schema
Query	**accountId** *required*	accountId	integer(int64)
Query	**leaderboardId** *required*	leaderboardId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.34.3. Get a Leaderboard

```
POST /api/{version}/leaderboard/get
```

Description

Get a Leaderboard

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**includeFullRankingList** *required*	includeFullRankingList	boolean	`"false"`
Query	**leaderboardId** *required*	leaderboardId	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	LeaderboardResponse

6.34.4. Search a Leaderboard

```
POST /api/{version}/leaderboard/search
```

Description

Search a Leaderboard

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**descending** *optional*	descending	boolean	`"false"`
Query	**globalOnly** *optional*	globalOnly	boolean	`"false"`
Query	**includeAppResponse** *optional*	includeAppResponse	boolean	`"false"`
Query	**includeInactive** *optional*	includeInactive	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**leaderboardIds** *optional*	leaderboardIds	string	
Query	**limit** *optional*	limit	integer(int32)	`"20"`
Query	**rankTypes** *optional*	rankTypes	string	

Type	Name	Description	Schema	Default
Query	**sortField** *required*	sortField	enum (TITLE, DESCRIPTION, CREATED, UPDATED, RANK_TYPE, RANK_MODE)	`"TITLE"`
Query	**start** *optional*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< SirqulResponse > array

6.34.5. Update a Leaderboard

```
POST /api/{version}/leaderboard/update
```

Description

Update a Leaderboard

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**active** *optional*	active	boolean
Query	**appKey** *optional*	appKey	string
Query	**bannerAssetId** *optional*	bannerAssetId	integer(int64)

Type	Name	Description	Schema
Query	**bannerMedia** *optional*	bannerMedia	ref
Query	**description** *optional*	description	string
Query	**iconAssetId** *optional*	iconAssetId	integer(int64)
Query	**iconMedia** *optional*	iconMedia	ref
Query	**leaderboardId** *required*	leaderboardId	integer(int64)
Query	**leaderboardMode** *optional*	leaderboardMode	string
Query	**limitation** *optional*	limitation	integer(int32)
Query	**metaData** *optional*	metaData	string
Query	**rankType** *optional*	rankType	string
Query	**sortField** *optional*	sortField	string
Query	**title** *optional*	title	string

Responses

HTTP Code	Description	Schema
200	successful operation	LeaderboardResponse

6.35. Like

The like service allows customers to like or dislike objects similar to a Facebook like. Customers may like or dislike accounts, albums, album contests, assets, game levels, notes, and theme descriptors. The service will only count a single like or dislike per customer and one will replace the current status. The current like/dislike status can be reset back to neutral if desired. The like/dislike can also be used for other concepts like approve/disapprove or affirmation by a community of customers.

Below are some example scenarios as to how these services can be used:

- Retail: allowing customer to like menu items or restaurants can allow users to quickly see what is trending as popular.
- Medical: a doctor can use a like to approve or disapprove a procedure or prescription.
- Real Estate: as items are depleted from the refrigerator they can be liked or disliked to build a grocery list that is automatically ordered and delivered at the end of the week.
- Automotive: a car application could have a traffic alert system where once an accident alert is added by one driver it can be confirmed by multiple drivers using a like.

One of the defining features of Facebook, Instagram, and other social applications is the ability to like items and the feature should be included any time a solution wants to drive customer engagement and social awareness.

Below are great example services that could be combined with this service:

- Album Service: a customer would like a collection of music or a physician could dislike a set of x-rays so they are retaken.
- Game Level Service: after completing the task the customer can like or dislike it so that others can be aware of its appeal.
- Note Service: just like Facebook, customers can like a comment.
- Asset Service: similar to Instagram a customer would like an asset (image) which would then send a notification to their friends.

6.35.1. Create Like

```
POST /api/{version}/like
```

Description

Allows a user to like or dislike accounts, albums, album contests, assets, game levels, notes, and theme descriptors. Multiple likes\dislikes on the same object will replace the previous one.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	

Type	Name	Description	Schema	Default
Query	**app** *optional*	app	string	
Query	**appKey** *optional*	appKey	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**gameType** *optional*	gameType	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**likableId** *required*		integer(int64)	
Query	**likableType** *required*	likableType	string	
Query	**like** *required*	like	boolean	"1"
Query	**longitude** *optional*	longitude	number(double)	
Query	**permissionableId** *optional*	permissionableId	integer(int64)	
Query	**permissionableType** *optional*	permissionableType	string	

Responses

HTTP Code	Description	Schema
200	successful operation	LikableResponse

6.35.2. Delete Like

```
POST /api/{version}/like/delete
```

Description

Removes a like. This will make the user "neutral".

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**latitude** *optional*	latitude	number(double)
Query	**likableId** *required*		integer(int64)
Query	**likableType** *required*	likableType	string
Query	**longitude** *optional*	longitude	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	LikableResponse

6.35.3. Search Likes

```
POST /api/{version}/like/search
```

Description

Search for likes on a likable object.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	

Type	Name	Description	Schema	Default
Query	**descending** *required*	descending	boolean	`"true"`
Query	**deviceId** *optional*	deviceId	string	
Query	**likableId** *required*	likableId	integer(int64)	
Query	**likableType** *required*	likableType	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**sortField** *required*	sortField	enum (ID)	`"ID"`
Query	**start** *required*	start	integer(int32)	`"0"`
Query	**updatedBefore** *optional*	updatedBefore	integer(int64)	
Query	**updatedSince** *optional*	updatedSince	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	< LikeResponse > array

6.36. Listing

A listing can be thought of as an event that is not tied to a specific location such as a sport or event being broadcast. Listings have a number of participants (actors, players, teams, etc.) that can be seen by the viewer. Think of listings as a customizable program guide. To be clear, the users are not physically at the particular listing, but they can experience the content being broadcast or streamed on their television, tablet, and even on their mobile smartphone devices. For instance, you can watch the World Cup finals in your living room while interacting with other friends and fans from all over the world.

Below are some example scenarios as to how these services can be used:

Retail: Retailer broadcasts the New York Fashion Show as a listing, which customers can view on their persona smartphone devices.

- Medical: Doctors can create a listing of a seminar being broadcast in which other other doctors and associates can view remotely.
- Real Estate: The World Cup game is being broadcast as a listing and the homeowner can view it on the television in their living room.

It is important for being able to track and manage all of the various listings in an organized manner with all of the relevant participants that are a part of the listing. In addition, listings can be used to drive more user traffic into your product or service all based upon a single point of interest. If your application allows users to broadcast these listings, this can help to create more engagement and interaction. Which may contribute toward increasing the amount of time your customers spend on your product.

Below are great example services that could be combined with the Listings service:

- Participants Service: When a listing is created, the participants must be identified. These participants are the sports teams, actors of the event, and so forth.
- Events Service: An event can be created for the listing. The event can be at a bar, broadcasting the sports game. Customers of the bar can attend the event and watch the listing that is broadcasted.

6.36.1. Create Listing

```
POST /api/{version}/listing/create
```

Description

Creates a listing.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	
Query	**description** *optional*	description	string	

-

Type	Name	Description	Schema	Default
Query	**end** *optional*	end	integer(int64)	
Query	**externalGroupId** *optional*	externalGroupId	string	
Query	**externalId** *optional*	externalId	string	
Query	**filterIds** *optional*	filterIds	string	
Query	**isPrivate** *required*	isPrivate	boolean	`"false"`
Query	**locationDescription** *optional*	locationDescription	string	
Query	**locationName** *optional*	locationName	string	
Query	**metaData** *optional*	metaData	string	
Query	**name** *required*	name	string	
Query	**start** *optional*	start	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	ListingFullResponse

6.36.2. Delete Listing

```
POST /api/{version}/listing/delete
```

Description

Delete a listing.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**listingId** *required*	listingId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.36.3. Get Listing

```
POST /api/{version}/listing/get
```

Description

Get a listing by id.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**listingId** *required*	listingId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	ListingFullRespons e

6.36.4. Search Listings

```
POST /api/{version}/listing/search
```

Description

Search for event listings from the start time to end time

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activeOnly** *required*	activeOnly	boolean	`"false"`
Query	**categoryIds** *optional*	categoryIds	string	
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**filterIds** *optional*	filterIds	string	
Query	**keyword** *optional*	keyword	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**longitude** *optional*	longitude	number(double)	
Query	**start** *required*	start	integer(int32)	`"0"`
Query	**startDate** *optional*	startDate	integer(int64)	
Query	**useListingOrderIds** *required*	useListingOrderIds	boolean	`"true"`

Responses

HTTP Code	Description	Schema
200	successful operation	< ListingResponse > array

6.36.5. Summary Listing

```
POST /api/{version}/listing/summary
```

Description

Search for a list of summary listings from the start time up to 8 days out.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**categoryIds** *optional*	categoryIds	string	
Query	**daysToInclude** *required*	daysToInclude	integer(int32)	`"15"`
Query	**startDate** *optional*	startDate	integer(int64)	
Query	**useListingOrderIds** *required*	useListingOrderIds	boolean	`"true"`

Responses

HTTP Code	Description	Schema
200	successful operation	< ListingGroupResponse > array

6.36.6. Update Listing

```
POST /api/{version}/listing/update
```

Description

Updates a listing.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**active** *optional*	active	boolean
Query	**description** *optional*	description	string
Query	**end** *optional*	end	integer(int64)
Query	**externalGroupId** *optional*	externalGroupId	string
Query	**externalId** *optional*	externalId	string
Query	**filterIds** *optional*	filterIds	string
Query	**isPrivate** *optional*	isPrivate	boolean
Query	**listingId** *required*	listingId	integer(int64)
Query	**locationDescription** *optional*	locationDescription	string
Query	**locationName** *optional*	locationName	string
Query	**metaData** *optional*	metaData	string

Type	Name	Description	Schema
Query	**name** *optional*	name	string
Query	**start** *optional*	start	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	ListingFullRespons e

6.37. Location

The Geographic Locations service allows an object or multiple objects to be located and tracked within an indoor or outdoor environment. Trilateration is a technique that the service uses to determine the location of these objects. This is created for being able to track residents or customers inside of a building that carry a smartphone device or wearables. We can identify information such as dwell time, directional traffic flow, ingress/egress of entrances and exits, and much more. We can track objects that are up to 40 feet away from an Edysen device via bluetooth with less than 1 foot accuracy of a object's position. After gathering this information, create reports that informs you about your customer behavior to help improve their experience. For example, understand where the main traffic flow is in your retail store so that you can place your products in the most optimal areas where they will be seen.

Below are some example scenarios as to how these services can be used:

- Retail: Add Edysen devices in the floorplan of the retail store. Using trilateration and passive WiFi tracking, determine the average dwell time of your customers, in addition to where your customers spend the most time/least amount of time
- Hospitals: Allow hospital administrators to see a dashboard of all patients inside of the hospital that are wearing a wearable device.
- Commercial Office Spaces: Use Trilateration to determine where devices are in your space. Track supplies, where they are moved, and identify employees that used them last
- Robotics: Use trilateration to determine the paths that your robotics/drones are taking, and determine if their paths need to be further optimized

The Geographic Locations service is important because it will provide you with information that will help increase revenue opportunities for your business. For example, use the trilateration to determine the footpaths that your customers take within your retail store. Determine if products you have placed within the store get as much attention as possible based upon the footpaths of your customers. The geographic locations service is also important because it can help track devices that may have been

misplaced. When you add a BLE tag onto a device, use trilateration to determine where the device has been placed within the space, in addition, understand where the device has been moved around in the past N hours, days, weeks, etc.

Below are example services that could be combined with the Geographic Locations service:

- Mesh Networks Service: Place Edysen devices within a floorplan, which uses Mesh Network to directly connect device-to-device together so that data can be transmitted and passed along with one another vs. all reporting directly to a single server. This alleviates the network load by leveraging each device as its own server.
- Reporting Service: Generate reports for information gather through trilateration such as dwell time, foot traffic, average distance travelled, and so forth.
- Pathing Service: Generate visual lines/paths for each user's journey using the trilateration data captured on foot traffic

6.37.1. Get Location by Trilateration

```
POST /api/{version}/account/location/trilaterate
```

Description

Send in device data and calculate a position based on signal strengths.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**data** *optional*	data	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)
Query	**responseFilters** *optional*	responseFilters	string

Responses

HTTP Code	Description	Schema
200	successful operation	GeoPointResponse

6.37.2. Get Location by IP

```
POST /api/{version}/location/ip
```

Description

Get location information based on an IP address.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**ip** *optional*	ip	string

Responses

HTTP Code	Description	Schema
200	successful operation	CoordsResponse

6.37.3. Search Regions or Postal Codes

```
POST /api/{version}/location/search
```

Description

Searches geographic locations by proximity via address or keyword.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	

Type	Name	Description	Schema	Default
Query	**_i** *optional*	_i	integer(int32)	
Query	**_l** *optional*	_l	integer(int32)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**currentLatitude** *optional*	currentLatitude	number(double)	
Query	**currentLongitude** *optional*	currentLongitude	number(double)	
Query	**currentlatitude** *optional*	currentlatitude	number(double)	
Query	**currentlongitude** *optional*	currentlongitude	number(double)	
Query	**deviceId** *optional*	deviceId	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**query** *optional*	query	string	
Query	**searchRange** *required*	searchRange	number(double)	`"5.0"`
Query	**selectedMapLatitude** *optional*	selectedMapLatitude	number(double)	
Query	**selectedMapLongitude** *optional*	selectedMapLongitude	number(double)	
Query	**selectedMaplatitude** *optional*	selectedMaplatitude	number(double)	
Query	**selectedMaplongitude** *optional*	selectedMaplongitude	number(double)	

Type	Name	Description	Schema	Default
Query	**start** *required*	start	integer(int32)	`"0"`
Query	**useGeocode** *required*	useGeocode	boolean	`"false"`
Query	**zipCode** *optional*	zipCode	string	
Query	**zipcode** *optional*	zipcode	string	

Responses

HTTP Code	Description	Schema
200	successful operation	LocationSearchResponse

6.37.4. Create Trilateration Data

```
POST /api/{version}/location/trilaterate/cache
```

Description

Creates trilateration samples for a source device (i.e. a router).

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**data** *optional*	data	string
Query	**dataFile** *optional*	dataFile	ref
Query	**minimumSampleSize** *optional*	minimumSampleSize	integer(int32)
Query	**sourceTime** *optional*	sourceTime	integer(int64)

Type	Name	Description	Schema
Query	**udid** *required*	udid	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

Consumes

- `multipart/form-data`

6.37.5. Create Trilateration Data

```
POST /api/{version}/location/trilaterate/cache/submit
```

Description

Creates trilateration samples for a source device (i.e. a router).

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Body	**body** *optional*		TrilatCacheRequest

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.38. Media

The Media service allows you to upload files like images, videos, files, and more to a media type offer for your business. These can be used to promote your business and brand identity by uploading images of your product and services on a website or application. For example, if you have an application that requires a lot of visuals like images, videos, graphics, etc. this service will be used frequently.

Below are some example scenarios as to how these services can be used:

- Retail: If you are a retailer and have an application, use the media service as a way to upload images and videos to help promote the store
- Healthcare: Use the media service to upload an image of an injury so that the doctor can see it and diagnose the issue
- Smart Home/Communities: Upload an image to other community members in your apartment if you are looking to sell some of your furniture
- Smart Vehicle: Take a photo or video of an incident and it can be uploaded and saved so that it can be submitted to your insurance company
- Robotics: Drones can take pictures and videos which can then be uploaded onto your website or application

The Media service is important because it allows owners of websites and applications to upload and display static or dynamic content. A media offer could be scheduled to display dynamically for advertisements on your website and application. This is important if your website or application has content that needs to be dynamically updated each day. Using the media service to promote items for sale using high quality images and videos for sale during the holidays is important because it can be scheduled to only appear for a specified period of time, and can also be targeted to specific audiences. In addition, a Media like a movie may require many metadata information to be associated with it such as the title, description, run time, genre, ratings, comments, and much more. The Media service offers the ability to associate many different metadata information to it.

Below are example services that could be combined with the Media service:

- Asset Service: You can upload one or many assets to a media
- Album Service: You can upload one or many albums to a media, where an album can contain multiple assets
- Retailer Location: Retailer locations can have a media associated with it that can be used for advertising

6.38.1. Create Media

```
POST /api/{version}/media/create
```

Description

Create a media offering.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activated** *optional*	activated	integer(int64)	
Query	**active** *required*	active	boolean	`"true"`
Query	**addedLimit** *required*	addedLimit	integer(int32)	`"0"`
Query	**asin** *optional*	asin	string	
Query	**author** *optional*	author	string	
Query	**availability** *optional*	availability	string	
Query	**availabilityDate** *optional*	availabilityDate	integer(int64)	
Query	**availabilitySummary** *optional*	availabilitySummary	string	
Query	**availableLimit** *required*	availableLimit	integer(int32)	`"0"`
Query	**availableLimitPerUser** *required*	availableLimitPerUser	integer(int32)	`"0"`
Query	**barcodeAssetId** *optional*	barcodeAssetId	integer(int64)	
Query	**barcodeEntry** *optional*	barcodeEntry	string	
Query	**barcodeType** *required*	barcodeType	enum (NONE, UPC, CODE_128, QR, CUSTOM_MEDIA)	`"NONE"`

Type	Name	Description	Schema	Default
Query	**catalogNumbers** *optional*	catalogNumbers	string	
Query	**categoryIds** *optional*	categoryIds	string	
Query	**collectionIds** *optional*	collectionIds	string	
Query	**conditionType** *optional*	conditionType	enum (NEW, USED, REFURBISHED, OTHER)	
Query	**details** *optional*	details	string	
Query	**discountPrice** *required*	discountPrice	number(double)	`"0.0"`
Query	**duration** *optional*	duration	integer(int32)	
Query	**expires** *optional*	expires	integer(int64)	
Query	**externalRedeemOptions** *optional*	externalRedeemOptions	string	
Query	**externalUrl** *optional*	externalUrl	string	
Query	**featured** *optional*	featured	boolean	
Query	**filterIds** *optional*	filterIds	string	
Query	**finePrint** *optional*	finePrint	string	
Query	**fullPrice** *required*	fullPrice	number(double)	`"0.0"`
Query	**imageAssetId** *optional*	imageAssetId	integer(int64)	
Query	**imageAssetId1** *optional*	imageAssetId1	integer(int64)	

Type	Name	Description	Schema	Default
Query	**imageAssetId2** *optional*	imageAssetId2	integer(int64)	
Query	**imageAssetId3** *optional*	imageAssetId3	integer(int64)	
Query	**imageAssetId4** *optional*	imageAssetId4	integer(int64)	
Query	**imageAssetId5** *optional*	imageAssetId5	integer(int64)	
Query	**isbn** *optional*	isbn	string	
Query	**maxPrints** *required*	maxPrints	integer(int32)	`"0"`
Query	**mediaType** *optional*	mediaType	enum (APK, APPLICATION, IMAGE, AUDIO, VIDEO, MULTIPART, BAR_CODE, TEXT, OTHER)	
Query	**metaData** *optional*	metaData	string	
Query	**noExpiration** *required*	noExpiration	boolean	`"false"`
Query	**offerVisibility** *required*	offerVisibility	enum (PUBLIC, LISTABLE, REWARDABLE, TRIGGERABLE, PRIVATE)	`"PUBLIC"`
Query	**parentalRating** *optional*	parentalRating	string	
Query	**publisher** *optional*	publisher	string	

Type	Name	Description	Schema	Default
Query	**redeemableEnd** *optional*	redeemableEnd	integer(int64)	
Query	**redeemableStart** *optional*	redeemableStart	integer(int64)	
Query	**releaseDate** *optional*	releaseDate	integer(int64)	
Query	**replaced** *optional*	replaced	boolean	
Query	**retailerLocationIds** *optional*	retailerLocationIds	string	
Query	**showRedeemed** *optional*	showRedeemed	boolean	
Query	**showRemaining** *optional*	showRemaining	boolean	
Query	**specialOfferType** *required*	specialOfferType	enum (ALL, RESERVABLE, REGULAR_OFFER, ACT_NOW, GET_THERE_NOW, SQOOT, TICKETS, YIPIT)	`"GET_THERE_NOW"`
Query	**subDetails** *optional*	subDetails	string	
Query	**subTitle** *optional*	subTitle	string	
Query	**ticketPrice** *required*	ticketPrice	integer(int64)	`"0"`
Query	**ticketPriceType** *optional*	ticketPriceType	string	
Query	**ticketsReward** *optional*	ticketsReward	integer(int64)	

Type	Name	Description	Schema	Default
Query	**ticketsRewardType** *optional*	ticketsRewardType	string	
Query	**title** *required*	title	string	
Query	**viewLimit** *required*	viewLimit	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	MediaOfferResponse

6.38.2. Delete Media

```
POST /api/{version}/media/delete
```

Description

Delete a media offering that the user has permissions to.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**mediaId** *required*	mediaId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.38.3. Media Get

```
POST /api/{version}/media/get
```

Description

Get a media offering.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**mediaId** *required*	mediaId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	MediaOfferResponse

6.38.4. Search Media

```
POST /api/{version}/media/search
```

Description

Searches on events that the account has access to.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	

Type	Name	Description	Schema	Default
Query	**activeOnly** *required*	activeOnly	boolean	`"false"`
Query	**categoryIds** *optional*	categoryIds	string	
Query	**descending** *required*	descending	boolean	`"false"`
Query	**filterIds** *optional*	filterIds	string	
Query	**keyword** *optional*	keyword	string	
Query	**limit** *optional*	limit	integer(int32)	`"20"`

Type	Name	Description	Schema	Default
Query	**sortField** *required*	sortField	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, ACTIVATED, EXPIRES, REDEEMABLE_START, REDEEMABLE_END, TITLE, SUBTITLE, DETAILS, OFFER_TYPE, SPECIAL_OFFER_TYPE, OFFER_VISIBILITY, ESTIMATED_VALUE, VOUCHER_PRICE, VOUCHER_DISCOUNT_PRICE, FULL_PRICE, DICOUNT_PRICE, TICKETS_REWARD, AVAILABILITY_DATE, RELEASE_DATE, RETAILER_ID, RETAILER_NAME, RETAILER_LOCATION_ID, RETAILER_LOCATION_NAME, BILLABLE_ENTITY_ID, BILLABLE_ENTITY_NAME, RESPONSIBLE_DISPLAY)	`"TITLE"`
Query	**start** *optional*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< MediaOfferResponse > array

6.38.5. Update Media

```
POST /api/{version}/media/update
```

Description

Update a media offering.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**activated** *optional*	activated	integer(int64)
Query	**active** *optional*	active	boolean
Query	**addedLimit** *optional*	addedLimit	integer(int32)
Query	**asin** *optional*	asin	string
Query	**author** *optional*	author	string
Query	**availability** *optional*	availability	string
Query	**availabilityDate** *optional*	availabilityDate	integer(int64)

Type	Name	Description	Schema
Query	**availabilitySummary** *optional*	availabilitySummary	string
Query	**availableLimit** *optional*	availableLimit	integer(int32)
Query	**availableLimitPerUser** *optional*	availableLimitPerUser	integer(int32)
Query	**barcodeAssetId** *optional*	barcodeAssetId	integer(int64)
Query	**barcodeEntry** *optional*	barcodeEntry	string
Query	**barcodeType** *optional*	barcodeType	enum (NONE, UPC, CODE_128, QR, CUSTOM_MEDIA)
Query	**catalogNumbers** *optional*	catalogNumbers	string
Query	**categoryIds** *optional*	categoryIds	string
Query	**collectionIds** *optional*	collectionIds	string
Query	**conditionType** *optional*	conditionType	enum (NEW, USED, REFURBISHED, OTHER)
Query	**details** *optional*	details	string
Query	**discountPrice** *optional*	discountPrice	number(double)
Query	**duration** *optional*	duration	integer(int32)
Query	**expires** *optional*	expires	integer(int64)
Query	**externalRedeemOptions** *optional*	externalRedeemOptions	string

Type	Name	Description	Schema
Query	**externalUrl** *optional*	externalUrl	string
Query	**featured** *optional*	featured	boolean
Query	**filterIds** *optional*	filterIds	string
Query	**finePrint** *optional*	finePrint	string
Query	**fullPrice** *optional*	fullPrice	number(double)
Query	**imageAssetId** *optional*	imageAssetId	integer(int64)
Query	**imageAssetId1** *optional*	imageAssetId1	integer(int64)
Query	**imageAssetId2** *optional*	imageAssetId2	integer(int64)
Query	**imageAssetId3** *optional*	imageAssetId3	integer(int64)
Query	**imageAssetId4** *optional*	imageAssetId4	integer(int64)
Query	**imageAssetId5** *optional*	imageAssetId5	integer(int64)
Query	**isbn** *optional*	isbn	string
Query	**maxPrints** *optional*	maxPrints	integer(int32)
Query	**mediaId** *required*	mediaId	integer(int64)
Query	**mediaType** *optional*	mediaType	enum (APK, APPLICATION, IMAGE, AUDIO, VIDEO, MULTIPART, BAR_CODE, TEXT, OTHER)
Query	**metaData** *optional*	metaData	string

Type	Name	Description	Schema
Query	**noExpiration** *optional*	noExpiration	boolean
Query	**offerLocations** *optional*	offerLocations	string
Query	**offerVisibility** *optional*	offerVisibility	enum (PUBLIC, LISTABLE, REWARDABLE, TRIGGERABLE, PRIVATE)
Query	**parentalRating** *optional*	parentalRating	string
Query	**publisher** *optional*	publisher	string
Query	**redeemableEnd** *optional*	redeemableEnd	integer(int64)
Query	**redeemableStart** *optional*	redeemableStart	integer(int64)
Query	**releaseDate** *optional*	releaseDate	integer(int64)
Query	**replaced** *optional*	replaced	boolean
Query	**retailerLocationIds** *optional*	retailerLocationIds	string
Query	**showRedeemed** *optional*	showRedeemed	boolean
Query	**showRemaining** *optional*	showRemaining	boolean

Type	Name	Description	Schema
Query	**specialOfferType** *optional*	specialOfferType	enum (ALL, RESERVABLE, REGULAR_OFFER, ACT_NOW, GET_THERE_NOW, SQOOT, TICKETS, YIPIT)
Query	**subDetails** *optional*	subDetails	string
Query	**subTitle** *optional*	subTitle	string
Query	**ticketPrice** *optional*	ticketPrice	integer(int64)
Query	**ticketPriceType** *optional*	ticketPriceType	string
Query	**ticketsReward** *optional*	ticketsReward	integer(int64)
Query	**ticketsRewardType** *optional*	ticketsRewardType	string
Query	**title** *optional*	title	string
Query	**viewLimit** *optional*	viewLimit	integer(int32)

Responses

HTTP Code	Description	Schema
200	successful operation	MediaOfferResponse

6.39. Mission

The Missions service enables you to create tasks that can be completed by users that reward users for their accomplishment. It can be used in such a way to motivate users to accomplish what they are assigned to and to be kept engaged with the service.

Managers and admins can create custom missions from a developer portal that can be generated on

the fly and shown within an application or service immediately. Missions are not always required to have a reward.

Below are some example scenarios as to how these services can be used:

- Retail: Management sets up a mission for their employees to see if they can meet their quota. If the quota is reached, the employee gets a bonus.
  Medical:
- Real Estate: Parents can create missions for their kids to see how well they can conserve energy for a reward. Once a certain threshold is met, the goal is attained and the mission is complete.
- Automotive: Missions can be created by insurance providers that empower users to drive safe in order to earn rewards and incentives to their insurance policy.
- Robotics: Once a mission has been complete, it will trigger the next set of missions that need to be completed by a particular machine and continue down this path until the required actions are finished.

The essence of Missions is to incentivize users to accomplish a set of goals and reward them for their accomplishment. While the reward is not required, it can also be used as a tool to give users a task list that can be completed at some point in time.

Below are great example services that could be combined with this service:

- Game Levels: Game Levels are contained within missions and contain many of the rules and tasks that are associated with the mission. Multiple game levels can be contained in the same mission.
- Offers: You can use offers as a rewards mechanism that can be given to users upon completing a mission.
- Notifications: You can send users notifications that new missions are available to complete.

6.39.1. Create Mission

```
POST /api/{version}/mission/create
```

Description

Create a user defined mission.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	

Type	Name	Description	Schema	Default
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	`"false"`
Query	**advancedReporting** *required*	advancedReporting	boolean	`"false"`
Query	**allocateTickets** *required*	allocateTickets	boolean	`"false"`
Query	**audienceIds** *optional*	audienceIds	string	
Query	**balance** *optional*	balance	number(double)	
Query	**creativeIds** *optional*	creativeIds	string	
Query	**description** *optional*	description	string	
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**formatType** *required*	formatType	string	`"CUSTOM"`
Query	**gameLevelIds** *optional*	gameLevelIds	string	
Query	**metaData** *optional*	metaData	string	
Query	**missionTask** *optional*	missionTask	string	
Query	**offerId** *optional*	offerId	integer(int64)	
Query	**points** *optional*	points	integer(int64)	
Query	**startDate** *optional*	startDate	integer(int64)	
Query	**subType** *optional*	subType	string	

Type	Name	Description	Schema	Default
Query	**ticketCount** *required*	ticketCount	integer(int64)	`"0"`
Query	**ticketType** *optional*	ticketType	string	
Query	**title** *required*	title	string	

Responses

HTTP Code	Description	Schema
200	successful operation	MissionResponse

6.39.2. Delete Mission

```
POST /api/{version}/mission/delete
```

Description

Delete a mission.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**missionId** *required*	missionId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

The creative services enable you to create and manage advertisements or editorials that appear in various solutions. The creative services support a set of standard advertisement sizes as well as custom

sizes to fit specific solution needs. Unlike standard advertisement services creatives support custom actions for clicks; allowing deep linking to other parts of applications, replacing the application style, or standard web link loading. The custom sizing and actions enables a variety of solutions such as how-to information (for learning the application), advertising of in-app purchases, showing the top ranking account, or a branded application overlay. All of the content for these solutions can be easily managed from the advertisement dashboard without a new application release.

Below are some example scenarios as to how these services can be used:

- Retail: use Sirqul's asset morphing service to convert offers into advertisements for greater reach into applications that only support advertisements, clicking the advert can direct the customer to the purchasing flow either in-application or online.
- Medical: remind employees of best practices and procedures they should be following during long loading screens or in just after logging in.
- Real Estate: Build a chore list that can be clicked to mark it as complete.
- Automotive: using the car's GPS get geolocated advertisements targeted to the owner of the vehicle allowing the manufacture to provide satellite radio as a free service.

The creative services need to be used in conjunction with the account, mission and audience services to be a fully featured solution. The mission services allows you to build a campaign that sets a time frame, audience targeting, and limit the total spend. The audience service allows you to specify the location, device types, and customer demographics. The account service associated demographics with customers to allow for targets matching.

Below are examples of how the services work together:

- Account Service: the customer may provide their gender, age, interests (categories), and other demographics.
- Audience Service: the advertisement owner will specify the type of customers they wish to target, the kinds of devices (IOS or Android for example), and the location + radius of the targeting.
- Mission Service: Joins the audience, creatives, and timeframe together to produce a targeted, time bound campaign.

6.39.3. Find Missions

```
POST /api/{version}/mission/find
```

Description

Get a set of ad filtered by the parameters provided.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**allocatesTickets** *optional*	allocatesTickets	boolean	
Query	**appKey** *required*	appKey	string	
Query	**appVersion** *optional*	appVersion	string	
Query	**device** *optional*	device	string	
Query	**deviceIdentifier** *optional*	deviceIdentifier	integer(int64)	
Query	**deviceVersion** *optional*	deviceVersion	string	
Query	**includeAudiences** *optional*	includeAudiences	boolean	`"false"`
Query	**includeGameData** *optional*	includeGameData	boolean	`"true"`
Query	**latitude** *optional*	latitude	number(double)	
Query	**limit** *optional*	limit	integer(int32)	`"5"`
Query	**longitude** *optional*	longitude	number(double)	
Query	**missionIds** *optional*	missionIds	string	
Query	**randomize** *required*	randomize	boolean	`"true"`

Type	Name	Description	Schema	Default
Query	**start** *optional*	start	integer(int32)	`"0"`
Query	**suffix** *optional*	suffix	string	
Query	**targetedAdsOnly** *required*	targetedAdsOnly	boolean	`"false"`
Query	**type** *optional*	type	string	

Responses

HTTP Code	Description	Schema
200	successful operation	MissionResponse

6.39.4. Search Mission Formats

```
POST /api/{version}/mission/format/search
```

Description

Searches on pre-defined mission formats

This information is used to generate the *gameData* parameter when creating a Game Level. A Mission can then be created using the generated game level ID.

Possbile Mission Formats by *formatType*:
(All format types require a points field. This is a digit that represents the number of points the consumer will receive upon completion of the mission.)

- PHOTO - {"type": "photo", "albumId": %d "points":%d} - Create an album and pass in the albumId.
- VOTE - {"type": "vote", "ids": [%s], "points": %d} - Create an album contest and pass in the albumContestId for "ids". You can pass in multiple contests by separating the IDs with a comma.
- ASK - {"type": "ask", "albumId": %d, "points": %d} - Create an album and pass in the albumId.
- ANSWER - {"type": "answer", "albumId": %d, "points": %d} - Create an album and pass in the albumId.
- AD - {"type":"click","action":"%s", points:%d} - Pass in a griply link for "action".
- GAME - This does not require Json generation. An existing game level is used when creating a

mission.

- All other types don't require anything else other than points.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**activeOnly** *required*	activeOnly	boolean	`"false"`
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< MissionFormatResponse > array

6.39.5. Get Mission

```
POST /api/{version}/mission/get
```

Description

Get a mission.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**missionId** *required*	missionId	integer(int64)	

Type	Name	Description	Schema	Default
Query	**returnCreative** *required*	returnCreative	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	MissionResponse

6.39.6. Import Mission

```
POST /api/{version}/mission/import
```

Description

Create a mission using a source item such as an offer location.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**adSize** *optional*	adSize	enum (CONFIG, BANNER, LEADERBOARD, SKYSCRAPER, VIDEO, ZIP, INTERSTITIAL, CUSTOM1, CUSTOM2, CUSTOM3, CUSTOM4, CUSTOM5, CUSTOM6, CUSTOM7, CUSTOM8, CUSTOM9, CUSTOM10)	

Type	Name	Description	Schema	Default
Query	**appKey** *required*	appKey	string	
Query	**keyword** *optional*	keyword	string	
Query	**latitude** *required*	latitude	number(double)	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**longitude** *required*	longitude	number(double)	
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.39.7. Search Missions

```
POST /api/{version}/mission/search
```

Description

Get the list missions available to the account.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**includeAudiences** *optional*	includeAudiences	boolean	`"false"`

Type	Name	Description	Schema	Default
Query	**includeGameData** *optional*	includeGameData	boolean	`"false"`
Query	**includeInactive** *optional*	includeInactive	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**limit** *optional*	limit	integer(int32)	`"20"`
Query	**start** *optional*	start	integer(int32)	`"0"`
Query	**subType** *optional*	subType	string	
Query	**suffix** *optional*	suffix	string	

Responses

HTTP Code	Description	Schema
200	successful operation	< MissionResponse > array

6.39.8. Search Missions by Billable Entity

```
POST /api/{version}/mission/searchByBillableEntity
```

Description

Use the accountId to determine the associated BillableEntity. From there get a list of all accounts associated as managers. Get the list missions owned by all associated managers.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	

Type	Name	Description	Schema	Default
Query	**accountId** *required*	accountId	integer(int64)	
Query	**includeAudiences** *optional*	includeAudiences	boolean	`"false"`
Query	**includeGameData** *optional*	includeGameData	boolean	`"false"`
Query	**includeInactive** *optional*	includeInactive	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**limit** *optional*	limit	integer(int32)	`"20"`
Query	**start** *optional*	start	integer(int32)	`"0"`
Query	**suffix** *optional*	suffix	string	

Responses

HTTP Code	Description	Schema
200	successful operation	< MissionResponse > array

6.39.9. Update Mission

```
POST /api/{version}/mission/update
```

Description

Update a mission.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	
Query	**advancedReporting** *optional*	advancedReporting	boolean	
Query	**allocateTickets** *required*	allocateTickets	boolean	`"false"`
Query	**audienceIds** *optional*	audienceIds	string	
Query	**balance** *optional*	balance	number(double)	
Query	**creativeIds** *optional*	creativeIds	string	
Query	**description** *optional*	description	string	
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**gameLevelIds** *optional*	gameLevelIds	string	
Query	**metaData** *optional*	metaData	string	
Query	**missionId** *required*	missionId	integer(int64)	
Query	**offerId** *optional*	offerId	integer(int64)	
Query	**points** *optional*	points	integer(int64)	
Query	**startDate** *optional*	startDate	integer(int64)	
Query	**subType** *optional*	subType	string	

Type	Name	Description	Schema	Default
Query	**ticketCount** *required*	ticketCount	integer(int64)	`"0"`
Query	**ticketType** *optional*	ticketType	string	
Query	**title** *optional*	title	string	

Responses

HTTP Code	Description	Schema
200	successful operation	MissionResponse

6.40. MissionInvite

The Mission Invite (Challenges) service serves as a challenges to be issued to users. Users can accept these challenges (similar to tasks) and complete them in order to earn a reward or prize in return. Mission Invites can also be thought of as a way to assign tasks that are executed by users or automated devices and machines.

Below are some example scenarios as to how these services can be used:

- Retail: Mission Invites can be available to employees to meet sales quotas. Upon reaching sales quotas, bonuses could be rewarded upon completion.
- Medical: Mission Invites can be available to employees as daily tasks that need to be completed based upon scheduled appointments. Medical equipment can also have a Mission Invite to be returned back to its original location when it is done being used.
- Real Estate: Mission Invites can be available to household members to complete chores on time.
- Automotive: Mission invites can be used to set safe driving milestones for the users.
- Robotics: Mission invites can be considered tasks that are used to be executed on a daily basis. Completion of mission invites could dictate whether or not production is on track or not.

Mission Invites are a great way of tracking progress of a particular item. You can also incentivize users to encourage them to complete their challenges/tasks with rewards.

Below are great example services that could be combined with this service:

- Offers Service: Create an offer that provides some level of incentive for users to complete their Mission Invite.
- Scheduled Tasks Service: Once a Mission Invite is complete, trigger another Mission Invite to be

worked on next. Continue this chain to create a task list that is to be completed by a user or via an automated process.

6.40.1. Create Mission Invite

```
POST /api/{version}/mission/invite/create
```

Description

Create the mission invite. An account can only be invited to a mission one time. For missions that require user submission and reviewing the permissionableType and permissionableId need to be provided.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**includeGameData** *optional*	includeGameData	boolean	`"true"`
Query	**joinCode** *optional*	joinCode	string	
Query	**missionId** *optional*	missionId	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	MissionResponse

6.40.2. Delete Mission Invite

```
POST /api/{version}/mission/invite/delete
```

Description

Update the mission invite status to quit.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**includeGameData** *optional*	includeGameData	boolean	`"true"`
Query	**missionId** *optional*	missionId	integer(int64)	
Query	**missionInviteId** *optional*	missionInviteId	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.40.3. Get Mission Invite

```
POST /api/{version}/mission/invite/get
```

Description

Get the mission invite. An account can only be invited to a mission one time.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	

Type	Name	Description	Schema	Default
Query	**accountId** *required*	accountId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**includeGameData** *optional*	includeGameData	boolean	`"true"`
Query	**includeScores** *optional*	includeScores	string	
Query	**missionId** *optional*	missionId	integer(int64)	
Query	**missionInviteId** *optional*	missionInviteId	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	MissionResponse

6.40.4. Search Mission Invites

```
POST /api/{version}/mission/invite/search
```

Description

Get a list of mission invites that the account has.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	

Type	Name	Description	Schema	Default
Query	**appVersion** *optional*	appVersion	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**filterByBillable** *optional*	filterByBillable	boolean	`"false"`
Query	**includeGameData** *optional*	includeGameData	boolean	`"true"`
Query	**keyword** *optional*	keyword	string	
Query	**lastUpdated** *optional*	lastUpdated	integer(int64)	
Query	**limit** *optional*	limit	integer(int32)	`"20"`
Query	**missionId** *optional*	missionId	integer(int64)	
Query	**missionTypes** *optional*	missionTypes	string	
Query	**start** *optional*	start	integer(int32)	`"0"`
Query	**status** *optional*	status	string	

Responses

HTTP Code	Description	Schema
200	successful operation	< MissionResponse > array

6.40.5. Update Mission Invite

```
POST /api/{version}/mission/invite/update
```

Description

Update the mission invite status. An account can only be invited to a mission one time. For missions that require user submission and reviewing the permissionableType and permissionableId need to be provided.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**gameLevelId** *optional*	gameLevelId	integer(int64)	
Query	**includeGameData** *optional*	includeGameData	boolean	`"true"`
Query	**missionId** *optional*	missionId	integer(int64)	
Query	**missionInviteId** *optional*	missionInviteId	integer(int64)	
Query	**packId** *optional*	packId	integer(int64)	
Query	**permissionableId** *optional*	permissionableId	integer(int64)	
Query	**permissionableType** *optional*	permissionableType	string	
Query	**status** *optional*	status	string	`"ACTIVE"`

Responses

HTTP Code	Description	Schema
200	successful operation	MissionResponse

6.41. Note

The Notes service enables users to leave comments and messages on content that can be viewable by other users who have the correct permissions to view it. Notes can be used in a variety of ways such as a comment on a photo, or used as a way to message users back and forth in a messaging app.

Notes can be left on a variety of objects such as albums, assets, game levels, and much more. The idea is to give users the ability to interact with the object itself as well as other users to foster engagement overall usage.

Below are some example scenarios as to how these services can be used:

- Retail: Notes can be used as a form of communication between customers and employees when it comes to product information and sales.
- Medical: Practitioners, nurses, and their patients can communicate with each other based upon x rays, diagnosis, and other medical information in a secure manner.
- Real Estate: Families who have a 'smart home' can set and schedule events and make sure that everyone is on the same page in terms of what is going on in the household.
- Automotive: Can be used to communicate with maintenance vendors and service providers for quotes and rates.
- Robotics: Can be used when a status is updated and recorded to an album or some other service to indicate what happened and when (e.g. hardware malfunction would be timestamped and noted for Robotic Arm 1, which can be reviewed later by the owner of this particular piece of hardware).

This service is simply used to inform users of what is going on by facilitating a way to communicate with one another.

Below are great example services that could be combined with this service:

- Albums Service: Use albums to contain photos of defective products that are produced from manufacturing with the comments + timestamps of when they occur.
- Game Levels Service: A list of tasks that must be complete, users can make a note on the game level in order to communicate with each other about the requirements of the game level or the their results of completing the task.

6.41.1. Note batch operations

```
POST /api/{version}/note/batch
```

Description

Perform batch operations on notes.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	The unique accountId that made the request (either deviceId or accountId must be used)	integer(int64)
Query	**batchOperation** *optional*	Special operation for notable, current supported value is DELETE_ALL_NOTES_IN_NOTABLE	string
Query	**deviceId** *optional*	The unique device identifier that made the request (either deviceId or accountId must be used)	string
Query	**notableId** *required*	The id of the notable object	integer(int64)
Query	**notableType** *required*	The notable object type {ALBUM, ALBUM_CONTEST, ASSET, GAME_LEVEL, OFFER, OFFER_LOCATION, RETAILER, RETAILER_LOCATION, THEME_DESCRIPTOR}	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.41.2. Create Note

```
POST /api/{version}/note/create
```

Description

This is used to leave a comment (note) on a notable object (i.e. albums, album contests, assets, game levels, offers, offer locations, retailers, retailer locations, and theme descriptors). Leaving a comment on a notable object will be visiable to everyone who has access to view the object.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**assetAddToDefaultAlbum** *optional*	assetAddToDefaultAlbum	string	
Query	**assetAddToMediaLibrary** *optional*	assetAddToMediaLibrary	boolean	
Query	**assetAlbumId** *optional*	assetAlbumId	integer(int64)	
Query	**assetApp** *optional*	assetApp	string	
Query	**assetAttachedMedia** *optional*	assetAttachedMedia	ref	
Query	**assetAttachedMediaString** *optional*	assetAttachedMediaString	string	
Query	**assetAttachedMediaStringContentType** *optional*	assetAttachedMediaStringContentType	string	
Query	**assetAttachedMediaStringFileName** *optional*	assetAttachedMediaStringFileName	string	
Query	**assetAttachedMediaUrl** *optional*	assetAttachedMediaUrl	string	
Query	**assetCaption** *optional*	assetCaption	string	

Type	Name	Description	Schema	Default
Query	**assetCollectionId** *optional*	assetCollectionId	integer(int64)	
Query	**assetIds** *optional*	assetIds	string	
Query	**assetLatitude** *optional*	assetLatitude	number(double)	
Query	**assetLocationDescription** *optional*	assetLocationDescription	string	
Query	**assetLongitude** *optional*	assetLongitude	number(double)	
Query	**assetMedia** *optional*	assetMedia	ref	
Query	**assetMediaString** *optional*	assetMediaString	string	
Query	**assetMediaStringContentType** *optional*	assetMediaStringContentType	string	
Query	**assetMediaStringFileName** *optional*	assetMediaStringFileName	string	
Query	**assetMediaUrl** *optional*	assetMediaUrl	string	
Query	**assetMetaData** *optional*	assetMetaData	string	
Query	**assetReturnNulls** *required*	assetReturnNulls	boolean	"1"
Query	**assetSearchTags** *optional*	assetSearchTags	string	

Type	Name	Description	Schema	Default
Query	**assetVersionCode** *optional*	assetVersionCode	integer(int32)	
Query	**assetVersionName** *optional*	assetVersionName	string	
Query	**comment** *required*	comment	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**initializeAsset** *required*	initializeAsset	boolean	`"false"`
Query	**latitude** *optional*	latitude	number(double)	
Query	**locationDescription** *optional*	locationDescription	string	
Query	**longitude** *optional*	longitude	number(double)	
Query	**metaData** *optional*	metaData	string	
Query	**notableId** *required*	notableId	integer(int64)	
Query	**notableType** *required*	notableType	string	
Query	**permissionableId** *optional*	permissionableId	integer(int64)	
Query	**permissionableType** *optional*	permissionableType	string	
Query	**returnFullResponse** *required*	returnFullResponse	boolean	`"false"`
Query	**tags** *required*	tags	string	

Responses

HTTP Code	Description	Schema
200	successful operation	NoteResponse

6.41.3. Delete Note

```
POST /api/{version}/note/delete
```

Description

Sets a comment (note) as deleted.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *optional*	appKey	string
Query	**deviceId** *optional*	deviceId	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)
Query	**noteId** *required*	noteId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.41.4. Get Note

```
POST /api/{version}/note/get
```

Description

Get for a note based on its Id.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**noteId** *required*	noteId	integer(int64)	
Query	**returnFullResponse** *required*	returnFullResponse	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	NoteResponse

6.41.5. Search Notes

```
POST /api/{version}/note/search
```

Description

Search for notes on a notable object.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**descending** *required*	descending	boolean	`"true"`
Query	**deviceId** *optional*	deviceId	string	
Query	**flagCountMinimum** *optional*	flagCountMinimum	integer(int64)	
Query	**flagsExceedThreshold** *required*	flagsExceedThreshold	boolean	`"false"`
Query	**includeInactive** *optional*	includeInactive	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**notableId** *optional*	notableId	integer(int64)	
Query	**notableType** *optional*	notableType	string	
Query	**returnFullResponse** *required*	returnFullResponse	boolean	`"false"`

Type	Name	Description	Schema	Default
Query	**sortField** *required*	sortField	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, OWNER_DISPLAY, NOTABLE_TYPE, NOTE_TAG, NOTE_COUNT, LIKES, DISLIKES)	`"ID"`
Query	**start** *required*	start	integer(int32)	`"0"`
Query	**updatedBefore** *optional*	updatedBefore	integer(int64)	
Query	**updatedSince** *optional*	updatedSince	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	< NoteResponse > array

6.41.6. Update Note

```
POST /api/{version}/note/update
```

Description

Update an existing comment (note). Only the creator of the note have permission to update.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	

Type	Name	Description	Schema	Default
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	
Query	**appKey** *optional*	appKey	string	
Query	**assetAddToDefaultAlbum** *optional*	assetAddToDefaultAlbum	string	
Query	**assetAddToMediaLibrary** *optional*	assetAddToMediaLibrary	boolean	
Query	**assetAlbumId** *optional*	assetAlbumId	integer(int64)	
Query	**assetApp** *optional*	assetApp	string	
Query	**assetAttachedMedia** *optional*	assetAttachedMedia	ref	
Query	**assetAttachedMediaString** *optional*	assetAttachedMediaString	string	
Query	**assetAttachedMediaStringContentType** *optional*	assetAttachedMediaStringContentType	string	
Query	**assetAttachedMediaStringFileName** *optional*	assetAttachedMediaStringFileName	string	
Query	**assetAttachedMediaUrl** *optional*	assetAttachedMediaUrl	string	
Query	**assetCaption** *optional*	assetCaption	string	
Query	**assetCollectionId** *optional*	assetCollectionId	integer(int64)	

Type	Name	Description	Schema	Default
Query	**assetIds** *optional*	assetIds	string	
Query	**assetLatitude** *optional*	assetLatitude	number(double)	
Query	**assetLocationDescription** *optional*	assetLocationDescription	string	
Query	**assetLongitude** *optional*	assetLongitude	number(double)	
Query	**assetMedia** *optional*	assetMedia	ref	
Query	**assetMediaString** *optional*	assetMediaString	string	
Query	**assetMediaStringContentType** *optional*	assetMediaStringContentType	string	
Query	**assetMediaStringFileName** *optional*	assetMediaStringFileName	string	
Query	**assetMediaUrl** *optional*	assetMediaUrl	string	
Query	**assetMetaData** *optional*	assetMetaData	string	
Query	**assetReturnNulls** *required*	assetReturnNulls	boolean	"1"
Query	**assetSearchTags** *optional*	assetSearchTags	string	
Query	**assetVersionCode** *optional*	assetVersionCode	integer(int32)	

Type	Name	Description	Schema	Default
Query	**assetVersionName** *optional*	assetVersionName	string	
Query	**comment** *optional*	comment	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**locationDescription** *optional*	locationDescription	string	
Query	**longitude** *optional*	longitude	number(double)	
Query	**metaData** *optional*	metaData	string	
Query	**noteId** *required*	noteId	integer(int64)	
Query	**permissionableId** *optional*	permissionableId	integer(int64)	
Query	**permissionableType** *optional*	permissionableType	string	
Query	**returnFullResponse** *required*	returnFullResponse	boolean	`"false"`
Query	**tags** *required*	tags	string	
Query	**updateAsset** *required*	updateAsset	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	NoteResponse

6.42. Notification

The Notifications service enables you to contact your users through different methods of communication: Email, SMS/MMS, Apple Push Notifications, and Google Cloud messaging. You can send all of your users custom and scheduled notifications or you can target a specific group of users.

Automated notifications that are triggered based upon actions within an app can also be customized and sent (E.g. someone liked a photo) to the appropriate users who have access to the content.

Below are some example scenarios as to how these services can be used:

- Retail: Set scheduled notifications to customers for ongoing sales and discounts and/or send notifications to team managers when sales quotas are met for employees.
- Medical: Set scheduled notifications for nurses, doctors, and other practitioners to make sure that they arrive on to their appointments on time. Get notified when medical devices need scheduled maintenance services.
- Real Estate: Schedule notifications when you detect movement or sound in your house for security purposes.
- Automotive: Get notifications for maintenance reminders, scheduling, and overall health of your vehicle.
- Robotics: Get notifications for when a device is malfunctioning, underperforming, or goes offline.

Reminders and notifications are an integral part of letting users know of what and when things are happening.

Below are great example services that could be combined with this service:

- Scheduled Tasks Service: Based upon the rules and triggers that you set up using Scheduled Tasks, you can have custom notifications be triggered based upon these actions.
- Offers Service: Push relevant content to your users based upon their demographics and preferences. Offers can be coupons, vouchers, products, people, and much more.
- Albums Service: Similar to offers, Albums and their contents can also be pushed to other users based upon their demographics and preferences.

6.42.1. Send Batch Notifications

```
POST /api/{version}/notification/batch
```

Description

Send notifications to all users of an application. Only someone with permissions to the application can do this.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *required*		string	
Query	**conduit** *required*	conduit	string	`"MOBILE_NOTIFICATION"`
Query	**contentId** *optional*	contentId	integer(int64)	
Query	**contentName** *optional*	contentName	string	
Query	**contentType** *optional*	contentType	string	
Query	**customMessage** *required*	customMessage	string	
Query	**deepLinkURI** *optional*	deepLinkURI	string	
Query	**parentId** *optional*	parentId	integer(int64)	
Query	**parentType** *optional*	parentType	string	

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.42.2. Send Custom Notifications

```
POST /api/{version}/notification/custom
```

Description

Send your own custom notification to a user. NOTE: the EventType of these notifications will be CUSTOM. Notifications sent to yourself will currently be ignored.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**actionCategory** *optional*	actionCategory	string	
Query	**appKey** *optional*	appKey	string	
Query	**conduit** *required*	conduit	string	`"MOBILE_NOTIFICATION"`
Query	**contentId** *optional*	contentId	integer(int64)	
Query	**contentName** *optional*	contentName	string	
Query	**contentType** *optional*	contentType	string	
Query	**customMessage** *optional*	customMessage	string	
Query	**deepLinkURI** *optional*	deepLinkURI	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**eventType** *required*	eventType	string	`"CUSTOM"`
Query	**friendOnlyAPNS** *required*	friendOnlyAPNS	boolean	`"1"`
Query	**gameType** *optional*	gameType	string	

Type	Name	Description	Schema	Default
Query	**includeFriendGroup** *required*	includeFriendGroup	boolean	`"0"`
Query	**latitude** *optional*	latitude	number(double)	
Query	**longitude** *optional*	longitude	number(double)	
Query	**parentId** *optional*	parentId	integer(int64)	
Query	**parentType** *optional*	parentType	string	
Query	**receiverAccountIds** *optional*	receiverAccountIds	string	
Query	**subject** *optional*	subject	string	

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.42.3. Search for Recipients

```
POST /api/{version}/notification/recipient/search
```

Description

Search for application users to send notifications.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	

Type	Name	Description	Schema	Default
Query	**appKey** *optional*	appKey	string	
Query	**audienceId** *optional*	audienceId	integer(int64)	
Query	**audienceIds** *optional*	audienceIds	string	
Query	**conduit** *optional*	conduit	string	
Query	**connectionGroupIds** *optional*	connectionGroupIds	string	
Query	**descending** *required*	descending	boolean	`"false"`
Query	**deviceId** *optional*	deviceId	string	
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"100"`
Query	**recipientAccountIds** *optional*	recipientAccountIds	string	
Query	**sortField** *required*	sortField	enum (ID, HAS_SMS, HAS_EMAIL, HAS_APNS, HAS_GCM, APPLICATION_ID, APPLICATION_NAME, ACCOUNT_ID, ACCOUNT_USERNAME, ACCOUNT_DISPLAY, ACCOUNT_TYPE)	`"ACCOUNT_DISPLAY"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< NotificationRecipientResponse > array

6.42.4. Search for Recipients

```
POST /api/{version}/notification/recipient/search/count
```

Description

Search for application users to send notifications.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**audienceId** *optional*	audienceId	integer(int64)	
Query	**audienceIds** *optional*	audienceIds	string	
Query	**conduit** *optional*	conduit	string	
Query	**connectionGroupIds** *optional*	connectionGroupIds	string	
Query	**descending** *required*	descending	boolean	`"false"`
Query	**deviceId** *optional*	deviceId	string	
Query	**keyword** *optional*	keyword	string	

Type	Name	Description	Schema	Default
Query	**limit** *required*	limit	integer(int32)	`"1"`
Query	**recipientAccountIds** *optional*	recipientAccountIds	string	
Query	**sortField** *optional*	sortField	enum (ID, HAS_SMS, HAS_EMAIL, HAS_APNS, HAS_GCM, APPLICATION_ID, APPLICATION_NAME, ACCOUNT_ID, ACCOUNT_USERNAME, ACCOUNT_DISPLAY, ACCOUNT_TYPE)	
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< NotificationRecipientResponse > array

6.42.5. Get Notifications

```
POST /api/{version}/notification/search
```

Description

Get a list of notifications for a user. If the "markAsRead" parameter is set to true, the returned notifications will be marked as "read" after the response has been sent. By default, read messages will not be returned, so to see read messages, set "returnReadMessages" to true.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**actionCategory** *optional*	actionCategory	string	
Query	**appKey** *optional*	appKey	string	
Query	**conduits** *optional*	conduits	string	
Query	**connectionAccountId** *optional*	connectionAccountId	integer(int64)	
Query	**contentIds** *optional*	contentIds	string	
Query	**contentTypes** *optional*	contentTypes	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**eventType** *optional*	eventType	string	
Query	**fromDate** *optional*	fromDate	integer(int64)	
Query	**keyword** *optional*	keyword	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**limit** *required*	limit	integer(int32)	`"100"`
Query	**longitude** *optional*	longitude	number(double)	
Query	**markAsRead** *required*	markAsRead	boolean	`"1"`
Query	**parentIds** *optional*	parentIds	string	

Type	Name	Description	Schema	Default
Query	**parentTypes** *optional*	parentTypes	string	
Query	**returnReadMessages** *required*	returnReadMessages	boolean	`"1"`
Query	**returnSent** *required*	returnSent	boolean	`"0"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	NotificationMessageListResponse

6.42.6. Create Notification Template

```
POST /api/{version}/notification/template/create
```

Description

Create a notification template. Developers will only be able to create notification templates for their own applications.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *optional*	appKey	string
Query	**body** *required*	body	string
Query	**conduit** *required*	conduit	string

Type	Name	Description	Schema
Query	**customType** *optional*	customType	string
Query	**disabled** *optional*	disabled	boolean
Query	**event** *optional*	event	string
Query	**sendToAllParticipants** *optional*	sendToAllParticipants	boolean
Query	**tags** *optional*	tags	string
Query	**title** *required*	title	string
Query	**updateNotificationCount** *optional*	updateNotificationCount	boolean

Responses

HTTP Code	Description	Schema
200	successful operation	NotificationTemplateResponse

6.42.7. Delete Notification Template

```
POST /api/{version}/notification/template/delete
```

Description

Deletes a notification template. Developers will only be able to delete notification templates for their own applications.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)

Type	Name	Description	Schema
Query	**accountId** *required*	accountId	integer(int64)
Query	**notificationTemplateId** *required*	notificationTemplateId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	NotificationTemplateResponse

6.42.8. Get Notification Template

```
POST /api/{version}/notification/template/get
```

Description

Get the details of a notification template. Developers will only be able to see notification templates for their own applications.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**notificationTemplateId** *required*	notificationTemplateId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	NotificationTemplateResponse

6.42.9. Search Notification Templates

```
POST /api/{version}/notification/template/search
```

Description

Search for notification templates on owned applications.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**conduit** *optional*	conduit	string	
Query	**customType** *optional*	customType	string	
Query	**descending** *required*	descending	boolean	`"true"`
Query	**event** *optional*	event	string	
Query	**globalOnly** *optional*	globalOnly	boolean	
Query	**includeGlobal** *required*	includeGlobal	boolean	`"true"`
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**reservedOnly** *optional*	reservedOnly	boolean	
Query	**sortField** *required*	sortField	string	`"CREATED"`

Type	Name	Description	Schema	Default
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	NotificationTemplateResponse

6.42.10. Update Notification Template

```
POST /api/{version}/notification/template/update
```

Description

Update a notification template. Developers will only be able to update notification templates for their own applications.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**body** *optional*	body	string
Query	**customType** *optional*	customType	string
Query	**disabled** *optional*	disabled	boolean
Query	**notificationTemplateId** *required*	notificationTemplateId	integer(int64)
Query	**sendToAllParticipants** *optional*	sendToAllParticipants	boolean

Type	Name	Description	Schema
Query	**tags** *optional*	tags	string
Query	**title** *optional*	title	string
Query	**updateNotificationCount** *optional*	updateNotificationCount	boolean

Responses

HTTP Code	Description	Schema
200	successful operation	NotificationTemplateResponse

6.42.11. Register Notification Token

```
POST /api/{version}/notification/token
```

Description

Register a token to send application dependent notifications like Google Cloud Messaging, or Apple Push Notifications.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	`"true"`
Query	**appKey** *optional*	appKey	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**environment** *optional*	environment	string	

Type	Name	Description	Schema	Default
Query	**gameType** *optional*	gameType	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**longitude** *optional*	longitude	number(double)	
Query	**pushType** *required*	pushType	enum (APNS, APNS_DEV, GCM, BAIDU, XGPUSH, JPUSH)	
Query	**token** *required*	token	string	

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.42.12. Check Notification Tokens

```
POST /api/{version}/notification/token/check
```

Description

Checks if push tokens are still valid.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**limit** *required*	limit	integer(int32)	`"500"`

Type	Name	Description	Schema	Default
Query	**pushType** *required*	pushType	enum (APNS, APNS_DEV, GCM, BAIDU, XGPUSH, JPUSH)	

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.43. ObjectStore

Storing data on the cloud is simple and easy with Sirqul's Object Store. Specify any key-value pairs you wish to define and add the record to the Object Store. You can then retrieve, update, or delete the record using the returned identifier. There is robust search system to find a list of records given a defined set of search criteria. The Object Store can be used to define custom objects, extend the features of existing Sirqul objects, or create relationships between Sirqul objects. Each application can define it's own Object Store records allowing for whatever custom solution you can think of.

Below are some example scenarios as to how these services can be used:

- Retail: when an item is returned an Object Store record can be created to track the reason, date, amount refunded, condition, employee, and other data currently not available with the Offer Product record.
- Medical: a mobile scanning machine is defined using an object store record, using the geopoint extension it can be given a location and indexed; this makes searching for it fast and highly efficient.
- Real Estate: create a keyless pin code object that can be assigned to various family members, the dog walker, and a house cleaner. Whenever a pin is used log the activity in another object or to the analytics service.
- Robotics: each robot has a unique configuration record entered in the object store, these can be managed online via a dashboard.

Ideally Sirqul's building blocks will provide the services and content holders necessary to create a successful solution. However, we know we cannot create everything for every situation and the Object Store allows solutions to define and create whatever items need to be stored in the cloud.

Below are great example services that could be combined with this service:

- Offer Service: each product has building instructions, warranty information, and other documents. These extended features can be associated with each Offer Product to enhance the current Sirqul

metadata.

- Account Service: as the solution begins to determine user behaviors entries can be added to the Object Store which can then can be used to set light and heating schedules in the home.
- Game Level Service: the Object Store specifies the templates available for the tasks the solution supports. When a task is generated a template is selected to help guide the author so that valid tasks are generated.

6.43.1. Create Object

```
POST /api/{version}/object/create
```

Description

Create an Object Store table. By default tables will have the columns: id, created, updated, deleted. Names og objects should be camel case with the first letter capitalized, for example: MyTableName.

Duplicate object names are not allowed.

The object name cannot be any of the following reserved words: ACCESSIBLE, ADD, ALL, ALTER, ANALYZE, AND, AS, ASC, ASENSITIVE, BEFORE, BETWEEN, BIGINT, BINARY, BLOB, BOTH, BY, CALL, CASCADE, CASE, CHANGE, CHAR, CHARACTER, CHECK, COLLATE, COLUMN, CONDITION, CONSTRAINT, CONTINUE, CONVERT, CREATE, CROSS, CURRENT_, ATE, CURRENT_TIME, CURRENT_TIMESTAMP, CURRENT_USER, CURSOR, DATABASE, DATABASES, DAY_HOUR, DAY_MICROSECOND, DAY_MINUTE, DAY_SECOND, DEC, DECIMAL, DECLARE, DEFAULT, DELAYED, DELETE, DESC, DESCRIBE, DETERMINISTIC, DISTINCT, DISTINCTROW, DIV, DOUBLE, DROP, DUAL, EACH, ELSE, ELSEIF, ENCLOSED, ESCAPED, EXISTS, EXIT, EXPLAIN, FALSE, FETCH, FLOAT, FLOAT4, FLOAT8, FOR, FORCE, FOREIGN, FROM, FULLTEXT, GRANT, GROUP, HAVING, HIGH_PRIORITY, HOUR_MICROSECOND, HOUR_MINUTE, HOUR_SECOND, IF, IGNORE, IN, INDEX, INFILE, INNER, INOUT, INSENSITIVE, INSERT, INT, INT1, INT2, INT3, INT4, INT8, INTEGER, INTERVAL, INTO, IS, ITERATE, JOIN, KEY, KEYS, KILL, LEADING, LEAVE, LEFT, LIKE, LIMIT, LINEAR, LINES, LOAD, LOCALTIME, LOCALTIMESTAMP, LOCK, LONG, LONGBLOB, LONGT, XT, LOOP, LOW_PRIORITY, MASTER_SSL_VERIFY_SERVER_CERT, MATCH, MAXVALUE, MEDIUMBLOB, MEDIUMINT, MEDIUMTEXT, MIDDLEINT, MINUTE_MICROSECOND, MINUTE_SECOND, MOD, MODIFIES, NATURAL, NOT, NO_WRITE_TO_BINLOG, NULL, NUMERIC, ON, OPTIMIZE, OPTION, OPTIONALLY, OR, ORDER, OUT, OUTER, OUTFILE, PRECISION, PRIMARY, PROCEDURE, PURGE, RANGE, READ, READS, READ_WRITE, REAL, REFERENCES, REGEXP, RELEASE, RENAME, REPEAT, REPLACE, REQUIRE, RESIGNAL, RESTRICT, RETURN, REVOKE, RIGHT, RLIKE, SCHEMA, SCHEMAS, SECOND_MICROSECOND, SELECT, SENSITIVE, SEPARATOR, SET, SHOW, SIGNAL, SMALLINT, SPATIAL, SPECIFIC, SQL, SQLEXCEPTION, SQLSTATE, SQLWARNING, SQL_BIG_RESULT, SQL_CALC_FOUND_ROWS, SQL_SMALL_RESULT, SSL, STARTING, STRAIGHT_JOIN, TABLE, TERMINATED, THEN, TINYBLOB, TINYINT, TINYTEXT, TO, TRAILING, TRIGGER, TRUE, NDO, UNION, UNIQUE, UNLOCK, UNSIGNED, UPDATE, USAGE, USE, USING, UTC_DATE, UTC_TIME, UTC_TIMESTAMP, VALUES, VARBINARY, VARCHAR, VARCHARACTER, VARYING, WHEN, WHERE, WHILE, WITH, WRITE, XOR, YEAR_MONTH, ZEROFILL, GENERAL, IGNORE_SERVER_IDS, MASTER_HEARTBEAT_PERIOD,

SLOW.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *required*	appKey	string
Query	**objectName** *required*	objectName	string

Responses

HTTP Code	Description	Schema
200	successful operation	ObjectStoreResponse

6.43.2. Create Data

```
POST /api/{version}/object/data/{objectName}
```

Description

Create a record for the specified object. If the object does not exist then a new one will be created prior to inserting the record. If any of the fields included does not exist for the object then they are added to the object.

The data type for the new fields are inferred from the JSON type, they are:

- DATE → quoted string of the format: yyyy-MM-dd hh:mm:ss
- STRING → quoted string of any format except a date
- NUMBER → unquoted number
- BOOLEAN → unquoted string of the format: true or false

The data structure is:

```
{
    "stringField":"value",
    "dateField":"2014-01-01 00:00:00",
    "numberField":23.98,
    "booleanField":true
}
```

Duplicate field names are not allowed.

The field name cannot be any of the following reserved words: ACCESSIBLE, ADD, ALL, ALTER, ANALYZE, AND, AS, ASC, ASENSITIVE, BEFORE, BETWEEN, BIGINT, BINARY, BLOB, BOTH, BY, CALL, CASCADE, CASE, CHANGE, CHAR, CHARACTER, CHECK, COLLATE, COLUMN, CONDITION, CONSTRAINT, CONTINUE, CONVERT, CREATE, CROSS, CURRENT_, ATE, CURRENT_TIME, CURRENT_TIMESTAMP, CURRENT_USER, CURSOR, DATABASE, DATABASES, DAY_HOUR, DAY_MICROSECOND, DAY_MINUTE, DAY_SECOND, DEC, DECIMAL, DECLARE, DEFAULT, DELAYED, DELETE, DESC, DESCRIBE, DETERMINISTIC, DISTINCT, DISTINCTROW, DIV, DOUBLE, DROP, DUAL, EACH, ELSE, ELSEIF, ENCLOSED, ESCAPED, EXISTS, EXIT, EXPLAIN, FALSE, FETCH, FLOAT, FLOAT4, FLOAT8, FOR, FORCE, FOREIGN, FROM, FULLTEXT, GRANT, GROUP, HAVING, HIGH_PRIORITY, HOUR_MICROSECOND, HOUR_MINUTE, HOUR_SECOND, IF, IGNORE, IN, INDEX, INFILE, INNER, INOUT, INSENSITIVE, INSERT, INT, INT1, INT2, INT3, INT4, INT8, INTEGER, INTERVAL, INTO, IS, ITERATE, JOIN, KEY, KEYS, KILL, LEADING, LEAVE, LEFT, LIKE, LIMIT, LINEAR, LINES, LOAD, LOCALTIME, LOCALTIMESTAMP, LOCK, LONG, LONGBLOB, LONGT, XT, LOOP, LOW_PRIORITY, MASTER_SSL_VERIFY_SERVER_CERT, MATCH, MAXVALUE, MEDIUMBLOB, MEDIUMINT, MEDIUMTEXT, MIDDLEINT, MINUTE_MICROSECOND, MINUTE_SECOND, MOD, MODIFIES, NATURAL, NOT, NO_WRITE_TO_BINLOG, NULL, NUMERIC, ON, OPTIMIZE, OPTION, OPTIONALLY, OR, ORDER, OUT, OUTER, OUTFILE, PRECISION, PRIMARY, PROCEDURE, PURGE, RANGE, READ, READS, READ_WRITE, REAL, REFERENCES, REGEXP, RELEASE, RENAME, REPEAT, REPLACE, REQUIRE, RESIGNAL, RESTRICT, RETURN, REVOKE, RIGHT, RLIKE, SCHEMA, SCHEMAS, SECOND_MICROSECOND, SELECT, SENSITIVE, SEPARATOR, SET, SHOW, SIGNAL, SMALLINT, SPATIAL, SPECIFIC, SQL, SQLEXCEPTION, SQLSTATE, SQLWARNING, SQL_BIG_RESULT, SQL_CALC_FOUND_ROWS, SQL_SMALL_RESULT, SSL, STARTING, STRAIGHT_JOIN, TABLE, TERMINATED, THEN, TINYBLOB, TINYINT, TINYTEXT, TO, TRAILING, TRIGGER, TRUE, NDO, UNION, UNIQUE, UNLOCK, UNSIGNED, UPDATE, USAGE, USE, USING, UTC_DATE, UTC_TIME, UTC_TIMESTAMP, VALUES, VARBINARY, VARCHAR, VARCHARACTER, VARYING, WHEN, WHERE, WHILE, WITH, WRITE, XOR, YEAR_MONTH, ZEROFILL, GENERAL, IGNORE_SERVER_IDS, MASTER_HEARTBEAT_PERIOD, SLOW

The following field names are reserved (cannot be used directly) and are automatically included during object creation: ID, OBJECTID, CREATED, UPDATED, DELETED. Additionally the field names must start with a letter or number.

Parameters

Type	Name	Description	Schema
Header	**Application-Key** *required*		string
Path	**objectName** *required*	objectName	string
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Body	**body** *optional*		string

Responses

HTTP Code	Description	Schema
200	successful operation	ObjectStoreResponse

6.43.3. Search Data

```
GET /api/{version}/object/data/{objectName}
```

Description

Search for records given the specified criteria. The criteria is a defined set of json values used to build a query

The search criteria structure is:

```
{
    "field1": {
        "operation1":"value1",
        "operation2":"value2"
    },
    "field2": {
        "operation3":["value3","value4"]
    }
}
```

The supported operations are:

- eq → equal to
- neq → not equal to
- lt → less than
- lte → less than or equal to
- gt → greater than
- gte → greater than or equal to
- in → contained in (array)
- nin → not contained in (array)
- exist→ true: some value set, false: no value set (null)

This example will find any record with a score between 500 and 5000 that was updated any time after Jan 1, 2014 and is for a player named Bob or Ben:

```
{
    "score":{"gte":500, "lte":5000},
    "updated":{"gte":"2014-01-01 00:00:00"},
    "playerName":"{"in":["Bob","Ben"]}
}
```

All operations across all fields are AND clauses.

Parameters

Type	Name	Description	Schema	Default
Header	**Application-Key** *required*		string	
Path	**objectName** *required*	objectName	string	
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**count** *required*	count	boolean	`"false"`
Query	**criteria** *optional*	criteria	string	

Type	Name	Description	Schema	Default
Query	**include** *optional*	include	string	
Query	**limit** *required*	limit	integer(int64)	`"20"`
Query	**order** *optional*	order	string	
Query	**start** *required*	start	integer(int64)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	ObjectStoreResponse

6.43.4. Get Data

```
GET /api/{version}/object/data/{objectName}/{objectId}
```

Description

Get a specific record from a specified object.

Parameters

Type	Name	Description	Schema
Header	**Application-Key** *required*		string
Path	**objectId** *required*	objectId	string
Path	**objectName** *required*	objectName	string
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)

Type	Name	Description	Schema
Query	**include** *optional*	include	string

Responses

HTTP Code	Description	Schema
200	successful operation	ObjectStoreResponse

6.43.5. Update Data

```
PUT /api/{version}/object/data/{objectName}/{objectId}
```

Description

Update a record for the specified object. If the object does not exist the request will be rejected, use the data create service for the first entry. If any of the fields included does not exist for the object then they are added to the object.

The data type for the new fields are inferred from the JSON type, they are:

- DATE → quoted string of the format: yyyy-MM-dd hh:mm:ss
- STRING → quoted string of any format except a date
- NUMBER → unquoted number
- BOOLEAN → unquoted string of the format: true or false

The data structure is:

```
{
    "stringField":"value",
    "dateField":"2014-01-01 00:00:00",
    "numberField":23.98,
    "booleanField":true
}
```

Duplicate field names are not allowed.

The field name cannot be any of the following reserved words: ACCESSIBLE, ADD, ALL, ALTER, ANALYZE, AND, AS, ASC, ASENSITIVE, BEFORE, BETWEEN, BIGINT, BINARY, BLOB, BOTH, BY, CALL, CASCADE, CASE, CHANGE, CHAR, CHARACTER, CHECK, COLLATE, COLUMN, CONDITION, CONSTRAINT,

CONTINUE, CONVERT, CREATE, CROSS, CURRENT_, ATE, CURRENT_TIME, CURRENT_TIMESTAMP, CURRENT_USER, CURSOR, DATABASE, DATABASES, DAY_HOUR, DAY_MICROSECOND, DAY_MINUTE, DAY_SECOND, DEC, DECIMAL, DECLARE, DEFAULT, DELAYED, DELETE, DESC, DESCRIBE, DETERMINISTIC, DISTINCT, DISTINCTROW, DIV, DOUBLE, DROP, DUAL, EACH, ELSE, ELSEIF, ENCLOSED, ESCAPED, EXISTS, EXIT, EXPLAIN, FALSE, FETCH, FLOAT, FLOAT4, FLOAT8, FOR, FORCE, FOREIGN, FROM, FULLTEXT, GRANT, GROUP, HAVING, HIGH_PRIORITY, HOUR_MICROSECOND, HOUR_MINUTE, HOUR_SECOND, IF, IGNORE, IN, INDEX, INFILE, INNER, INOUT, INSENSITIVE, INSERT, INT, INT1, INT2, INT3, INT4, INT8, INTEGER, INTERVAL, INTO, IS, ITERATE, JOIN, KEY, KEYS, KILL, LEADING, LEAVE, LEFT, LIKE, LIMIT, LINEAR, LINES, LOAD, LOCALTIME, LOCALTIMESTAMP, LOCK, LONG, LONGBLOB, LONGT, XT, LOOP, LOW_PRIORITY, MASTER_SSL_VERIFY_SERVER_CERT, MATCH, MAXVALUE, MEDIUMBLOB, MEDIUMINT, MEDIUMTEXT, MIDDLEINT, MINUTE_MICROSECOND, MINUTE_SECOND, MOD, MODIFIES, NATURAL, NOT, NO_WRITE_TO_BINLOG, NULL, NUMERIC, ON, OPTIMIZE, OPTION, OPTIONALLY, OR, ORDER, OUT, OUTER, OUTFILE, PRECISION, PRIMARY, PROCEDURE, PURGE, RANGE, READ, READS, READ_WRITE, REAL, REFERENCES, REGEXP, RELEASE, RENAME, REPEAT, REPLACE, REQUIRE, RESIGNAL, RESTRICT, RETURN, REVOKE, RIGHT, RLIKE, SCHEMA, SCHEMAS, SECOND_MICROSECOND, SELECT, SENSITIVE, SEPARATOR, SET, SHOW, SIGNAL, SMALLINT, SPATIAL, SPECIFIC, SQL, SQLEXCEPTION, SQLSTATE, SQLWARNING, SQL_BIG_RESULT, SQL_CALC_FOUND_ROWS, SQL_SMALL_RESULT, SSL, STARTING, STRAIGHT_JOIN, TABLE, TERMINATED, THEN, TINYBLOB, TINYINT, TINYTEXT, TO, TRAILING, TRIGGER, TRUE, NDO, UNION, UNIQUE, UNLOCK, UNSIGNED, UPDATE, USAGE, USE, USING, UTC_DATE, UTC_TIME, UTC_TIMESTAMP, VALUES, VARBINARY, VARCHAR, VARCHARACTER, VARYING, WHEN, WHERE, WHILE, WITH, WRITE, XOR, YEAR_MONTH, ZEROFILL, GENERAL, IGNORE_SERVER_IDS, MASTER_HEARTBEAT_PERIOD, SLOW

The following field names are reserved (cannot be used directly) and are automatically included during object creation: ID, OBJECTID, CREATED, UPDATED, DELETED. Additionally the field names must start with a letter or number.

Parameters

Type	Name	Description	Schema
Header	**Application-Key** *required*		string
Path	**objectId** *required*	objectId	string
Path	**objectName** *required*	objectName	string
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)

Type	Name	Description	Schema
Body	**body** *optional*		string

Responses

HTTP Code	Description	Schema
200	successful operation	ObjectStoreResponse

6.43.6. Delete Data

```
DELETE /api/{version}/object/data/{objectName}/{objectId}
```

Description

Delete a record for the specified object. Cannot be undone so use only when abolutely sure.

Parameters

Type	Name	Description	Schema
Header	**Application-Key** *required*		string
Path	**objectId** *required*	objectId	string
Path	**objectName** *required*	objectName	string
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	ObjectStoreResponse

6.43.7. Delete Object

```
POST /api/{version}/object/delete
```

Description

Delete and Object in the store. This will delete the table and clean up and foreign keys referencing it. Cannot be undone so use only when abolutely sure.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *required*	appKey	string
Query	**objectName** *required*	objectName	string

Responses

HTTP Code	Description	Schema
200	successful operation	ObjectStoreResponse

6.43.8. Create Field

```
POST /api/{version}/object/field/add
```

Description

Add a field to a specific object. The field name should be camel case with the first letter lower case, for example: myFieldName.

Duplicate field names are not allowed.

The field name cannot be any of the following reserved words: ACCESSIBLE, ADD, ALL, ALTER, ANALYZE, AND, AS, ASC, ASENSITIVE, BEFORE, BETWEEN, BIGINT, BINARY, BLOB, BOTH, BY, CALL, CASCADE, CASE, CHANGE, CHAR, CHARACTER, CHECK, COLLATE, COLUMN, CONDITION, CONSTRAINT,

CONTINUE, CONVERT, CREATE, CROSS, CURRENT_, ATE, CURRENT_TIME, CURRENT_TIMESTAMP, CURRENT_USER, CURSOR, DATABASE, DATABASES, DAY_HOUR, DAY_MICROSECOND, DAY_MINUTE, DAY_SECOND, DEC, DECIMAL, DECLARE, DEFAULT, DELAYED, DELETE, DESC, DESCRIBE, DETERMINISTIC, DISTINCT, DISTINCTROW, DIV, DOUBLE, DROP, DUAL, EACH, ELSE, ELSEIF, ENCLOSED, ESCAPED, EXISTS, EXIT, EXPLAIN, FALSE, FETCH, FLOAT, FLOAT4, FLOAT8, FOR, FORCE, FOREIGN, FROM, FULLTEXT, GRANT, GROUP, HAVING, HIGH_PRIORITY, HOUR_MICROSECOND, HOUR_MINUTE, HOUR_SECOND, IF, IGNORE, IN, INDEX, INFILE, INNER, INOUT, INSENSITIVE, INSERT, INT, INT1, INT2, INT3, INT4, INT8, INTEGER, INTERVAL, INTO, IS, ITERATE, JOIN, KEY, KEYS, KILL, LEADING, LEAVE, LEFT, LIKE, LIMIT, LINEAR, LINES, LOAD, LOCALTIME, LOCALTIMESTAMP, LOCK, LONG, LONGBLOB, LONGT, XT, LOOP, LOW_PRIORITY, MASTER_SSL_VERIFY_SERVER_CERT, MATCH, MAXVALUE, MEDIUMBLOB, MEDIUMINT, MEDIUMTEXT, MIDDLEINT, MINUTE_MICROSECOND, MINUTE_SECOND, MOD, MODIFIES, NATURAL, NOT, NO_WRITE_TO_BINLOG, NULL, NUMERIC, ON, OPTIMIZE, OPTION, OPTIONALLY, OR, ORDER, OUT, OUTER, OUTFILE, PRECISION, PRIMARY, PROCEDURE, PURGE, RANGE, READ, READS, READ_WRITE, REAL, REFERENCES, REGEXP, RELEASE, RENAME, REPEAT, REPLACE, REQUIRE, RESIGNAL, RESTRICT, RETURN, REVOKE, RIGHT, RLIKE, SCHEMA, SCHEMAS, SECOND_MICROSECOND, SELECT, SENSITIVE, SEPARATOR, SET, SHOW, SIGNAL, SMALLINT, SPATIAL, SPECIFIC, SQL, SQLEXCEPTION, SQLSTATE, SQLWARNING, SQL_BIG_RESULT, SQL_CALC_FOUND_ROWS, SQL_SMALL_RESULT, SSL, STARTING, STRAIGHT_JOIN, TABLE, TERMINATED, THEN, TINYBLOB, TINYINT, TINYTEXT, TO, TRAILING, TRIGGER, TRUE, NDO, UNION, UNIQUE, UNLOCK, UNSIGNED, UPDATE, USAGE, USE, USING, UTC_DATE, UTC_TIME, UTC_TIMESTAMP, VALUES, VARBINARY, VARCHAR, VARCHARACTER, VARYING, WHEN, WHERE, WHILE, WITH, WRITE, XOR, YEAR_MONTH, ZEROFILL, GENERAL, IGNORE_SERVER_IDS, MASTER_HEARTBEAT_PERIOD, SLOW.

The following field names are reserved (cannot be used directly) and are automatically included during object creation: ID, OBJECTID, CREATED, UPDATED, DELETED. Additionally the field names must start with a letter or number.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *required*	appKey	string
Query	**fieldName** *required*	fieldName	string
Query	**fieldType** *required*	fieldType	string

Type	Name	Description	Schema
Query	**objectName** *required*	objectName	string

Responses

HTTP Code	Description	Schema
200	successful operation	ObjectStoreResponse

6.43.9. Delete Field

```
POST /api/{version}/object/field/delete
```

Description

Delete a field from an object. This will remove the field, indexes, and foreign keys associated with the field.

The following field names are reserved and cannot be removed from the object: ID, OBJECTID, CREATED, UPDATED, DELETED

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *required*	appKey	string
Query	**fieldName** *required*	fieldName	string
Query	**objectName** *required*	objectName	string

Responses

HTTP Code	Description	Schema
200	successful operation	ObjectStoreResponse

6.43.10. Get Object

```
POST /api/{version}/object/get
```

Description

Get the definition of an Object. Returns all field names, types, and current size. The types supported are: STRING, DATE, NUMBER, BOOLEAN, IDENTITY.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *required*	appKey	string
Query	**objectName** *required*	objectName	string

Responses

HTTP Code	Description	Schema
200	successful operation	ObjectStoreResponse

6.43.11. Search Objects

```
POST /api/{version}/object/search
```

Description

Search for Objects and return the list of names found. Use this in conjunction with the object get

service to present the current data model defined.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *required*	appKey	string	
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int64)	`"20"`
Query	**start** *required*	start	integer(int64)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	ObjectStoreResponse

6.44. Offer

The Offers API allows you to create compelling offers such as coupons, vouchers or products to attract more users and gain more engagement from your existing user base. Offers are a key component in any business and can be used to initiate a free trial, drive repeat business, and increase overall sales and revenues.

Below are some example scenarios as to how the Offers service can be used:

- Retail: Create coupons to drive users to visit your physical locations and increase the chance of impulse purchases in-store.
- Medical: Reward Employees by creating vouchers to such as “$10 Starbucks Gift Cards.” Reward your employees for various activities to increase productivity, improve customer service, and more.
- Smart Home: Use Offers to drive the purchase of add-on products, special content, and more to increase revenues.

- Automotive: Use geo-targeted and timed Offers to add more customer value to your connected vehicle. For example, “Save 5% on gas in 1 mile” or “Save 10% on an Oil Change.”
- Robotics: Use Offers to get your users to try out drone delivery. For example, “Free First Delivery”

The Offers API is important because it gives you the main business tool every business uses to attract new customers, increase repeat business and compete in the market.

Below are great example services that could be combined with the Offers service:

- Audience: Test various offers within different Audiences to optimize types of offers to send in the future.
- Territories: Use Offers in different Territories to be able to grow your business in a new market.
- Game Levels: Combine Offers with Game Levels to keep users playing your game in order to earn discounts and savings on future purchases.
- Tournaments: Combine Offers with Tournaments to get users into your contest and drive more engagement within your game with an offer incentive.

6.44.1. Get Offer

```
POST /api/{version}/offer/get
```

Description

Gets offer or offer location details as a consumer. Will check if it is a favorite if the deviceId/accountId is provided. If the offerId is provided it will look up the main offer and ignore the the offerLocationId. If no offerId is provided then an offerLocationId must be specified.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**distance** *optional*	distance	number(double)	
Query	**includeOfferLocations** *required*	includeOfferLocations	boolean	`"false"`

Type	Name	Description	Schema	Default
Query	**includeRetailerLocations** *required*	includeRetailerLocations	boolean	`"false"`
Query	**latitude** *optional*	latitude	number(double)	
Query	**longitude** *optional*	longitude	number(double)	
Query	**offerId** *optional*	offerId	integer(int64)	
Query	**offerLocationId** *optional*	offerLocationId	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	OfferResponse

6.44.2. Search Offers

```
POST /api/{version}/offer/lists
```

Description

Searches for offers as a consumer.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**categories** *optional*	categories	string	

Type	Name	Description	Schema	Default
Query	**closestOfferOnly** *optional*	closestOfferOnly	boolean	`"false"`
Query	**deviceId** *optional*	deviceId	string	
Query	**distanceUnit** *required*	distanceUnit	enum (MILES, KILOMETERS)	`"MILES"`
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**endedBefore** *optional*	endedBefore	integer(int64)	
Query	**endedSince** *optional*	endedSince	integer(int64)	
Query	**filters** *optional*	filters	string	
Query	**groupBy** *optional*	groupBy	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, OFFER_ID, ACTIVATED, EXPIRES, TITLE, SUBTITLE, DETAILS, OFFER_TYPE, SPECIAL_OFFER_TYPE, OFFER_VISIBILITY, RETAILER_ID, RETAILER_LOCATION_ID, BILLABLE_ENTITY_ID, BILLABLE_ENTITY_NAME, RESPONSIBLE_DISPLAY)	
Query	**includeCategories** *optional*	includeCategories	boolean	`"false"`

Type	Name	Description	Schema	Default
Query	**includeExpired** *optional*	includeExpired	boolean	`"false"`
Query	**includeFavorite** *optional*	includeFavorite	boolean	`"false"`
Query	**includeFilters** *optional*	includeFilters	boolean	`"false"`
Query	**includeMission** *optional*	includeMission	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**latitude** *required*	latitude	number(double)	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**locationId** *required*	locationId	integer(int64)	`"0"`
Query	**longitude** *required*	longitude	number(double)	
Query	**maxRecommendations** *required*	maxRecommendations	integer(int32)	`"6"`
Query	**offerAudienceIds** *optional*	offerAudienceIds	string	
Query	**offerId** *optional*	offerId	integer(int64)	
Query	**offerTypes** *required*	offerTypes	string	`"COUPON,VOUCHER"`
Query	**recommendOfferIds** *optional*	recommendOfferIds	string	
Query	**recommendationType** *required*	recommendationType	enum (WALLET, CLICKS, BLENDED)	`"BLENDED"`

Type	Name	Description	Schema	Default
Query	**retailerLocationIds** *optional*	retailerLocationIds	string	
Query	**searchExpression** *optional*	searchExpression	string	
Query	**searchRange** *required*	searchRange	number(double)	"5.0"
Query	**sortField** *optional*	sortField	string	
Query	**start** *required*	start	integer(int32)	"0"
Query	**startedBefore** *optional*	startedBefore	integer(int64)	
Query	**startedSince** *optional*	startedSince	integer(int64)	
Query	**supportedPostalCodes** *optional*	supportedPostalCodes	string	
Query	**tags** *optional*	tags	string	
Query	**transactionAudienceIds** *optional*	transactionAudienceIds	string	
Query	**type** *optional*	type	string	

Responses

HTTP Code	Description	Schema
200	successful operation	OfferListResponse

6.44.3. Get Offers (Counts)

```
POST /api/{version}/offer/lists/count
```

Description

Gets the offer list counts.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**distanceUnit** *required*	distanceUnit	enum (MILES, KILOMETERS)	`"MILES"`
Query	**latitude** *required*	latitude	number(double)	
Query	**longitude** *required*	longitude	number(double)	
Query	**searchRange** *required*	searchRange	number(double)	`"5.0"`

Responses

HTTP Code	Description	Schema
200	successful operation	ListCountResponse

6.44.4. Get Offer Location

```
POST /api/{version}/offer/location/get
```

Description

Gets the offer location by offer location id or udid (of a device)

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**offerLocationId** *optional*	offerLocationId	integer(int64)

Type	Name	Description	Schema
Query	**udid** *optional*	udid	string

Responses

HTTP Code	Description	Schema
200	successful operation	OfferShortResponse

6.44.5. Get Offers (Top)

```
POST /api/{version}/offer/top
```

Description

Gets the top active offers.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	OfferListResponse

6.44.6. Create Offer

```
POST /api/{version}/retailer/offer/create
```

Description

Create an offer and assign it to the provided retailer locations.

Parameters

Type	Name	Description	Schema	Default
Header	**Application-Key** *optional*		string	
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activated** *optional*	activated	integer(int64)	
Query	**active** *required*	active	boolean	`"true"`
Query	**addedLimit** *required*	addedLimit	integer(int32)	`"0"`
Query	**asin** *optional*	asin	string	
Query	**author** *optional*	author	string	
Query	**availability** *optional*	availability	string	
Query	**availabilityDate** *optional*	availabilityDate	integer(int64)	
Query	**availabilitySummary** *optional*	availabilitySummary	string	
Query	**availableLimit** *required*	availableLimit	integer(int32)	`"0"`
Query	**availableLimitPerUser** *required*	availableLimitPerUser	integer(int32)	`"0"`

Type	Name	Description	Schema	Default
Query	**barcodeAssetId** *optional*	barcodeAssetId	integer(int64)	
Query	**barcodeEntry** *optional*	barcodeEntry	string	
Query	**barcodeType** *required*	barcodeType	enum (NONE, UPC, CODE_128, QR, CUSTOM_MEDIA)	`"NONE"`
Query	**brand** *optional*	brand	string	
Query	**catalogNumbers** *optional*	catalogNumbers	string	
Query	**categoryIds** *optional*	categoryIds	string	
Query	**collectionIds** *optional*	collectionIds	string	
Query	**conditionType** *optional*	conditionType	enum (NEW, USED, REFURBISHED, OTHER)	
Query	**department** *optional*	department	string	
Query	**depth** *optional*	depth	number(double)	
Query	**details** *optional*	details	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**deviceInterference** *optional*	deviceInterference	number(double)	
Query	**devicePower** *optional*	devicePower	number(double)	
Query	**deviceType** *optional*	deviceType	string	

Type	Name	Description	Schema	Default
Query	**discountPrice** *required*	discountPrice	number(double)	"0.0"
Query	**duration** *optional*	duration	integer(int32)	
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**expires** *optional*	expires	integer(int64)	
Query	**externalId** *optional*	externalId	string	
Query	**externalRedeemOptions** *optional*	externalRedeemOptions	string	
Query	**externalUrl** *optional*	externalUrl	string	
Query	**featured** *optional*	featured	boolean	
Query	**features** *optional*	features	string	
Query	**filterIds** *optional*	filterIds	string	
Query	**finePrint** *optional*	finePrint	string	
Query	**fullPrice** *required*	fullPrice	number(double)	"0.0"
Query	**height** *optional*	height	number(double)	
Query	**idleTimeoutInSecond** *optional*	idleTimeoutInSecond	integer(int32)	
Query	**imageAssetId** *optional*	imageAssetId	integer(int64)	
Query	**imageAssetId1** *optional*	imageAssetId1	integer(int64)	

Type	Name	Description	Schema	Default
Query	**imageAssetId2** *optional*	imageAssetId2	integer(int64)	
Query	**imageAssetId3** *optional*	imageAssetId3	integer(int64)	
Query	**imageAssetId4** *optional*	imageAssetId4	integer(int64)	
Query	**imageAssetId5** *optional*	imageAssetId5	integer(int64)	
Query	**includeOfferLocations** *required*	includeOfferLocations	boolean	`"false"`
Query	**includeRetailerLocations** *required*	includeRetailerLocations	boolean	`"false"`
Query	**isbn** *optional*	isbn	string	
Query	**listingId** *optional*	listingId	integer(int64)	
Query	**maxPrints** *required*	maxPrints	integer(int32)	`"0"`
Query	**mediaType** *optional*	mediaType	enum (APK, APPLICATION, IMAGE, AUDIO, VIDEO, MULTIPART, BAR_CODE, TEXT, OTHER)	
Query	**metaData** *optional*	metaData	string	
Query	**minimumPrice** *optional*	minimumPrice	number(double)	
Query	**noExpiration** *required*	noExpiration	boolean	`"false"`

Type	Name	Description	Schema	Default
Query	**offerAudienceIds** *optional*	offerAudienceIds	string	
Query	**offerAudienceIdsToAdd** *optional*	offerAudienceIdsToAdd	string	
Query	**offerAudienceIdsToRemove** *optional*	offerAudienceIdsToRemove	string	
Query	**offerLocations** *optional*	offerLocations	string	
Query	**offerType** *required*	offerType	enum (VOUCHER, COUPON, PRODUCT, MEDIA, EVENT, DEVICE)	`"COUPON"`
Query	**offerVisibility** *required*	offerVisibility	enum (PUBLIC, LISTABLE, REWARDABLE, TRIGGERABLE, PRIVATE)	`"PUBLIC"`
Query	**parentOfferId** *optional*	parentOfferId	integer(int64)	
Query	**parentalRating** *optional*	parentalRating	string	

Type	Name	Description	Schema	Default
Query	**productType** *optional*	productType	enum (APPAREL, APPLIANCES, APPS, ARTS_CRAFTS, AUTOMOTIVE, BABY, BEAUTY, BOOKS, CLASSICAL, COLLECTIBLES, DVD, ELECTRONICS, GIFT_CARDS, GOURMET_FOOD, GROCERY, HEALTH_PERSONAL_CARE, HOME_GARDEN, INDUSTRIAL, JEWELRY, KITCHEN, LAWN_GARDEN, MAGAZINES, MISCELLANEOUS, MUSICAL_INSTRUMENTS, OFFICE_PRODUCTS, OTHER, OUTDOOR_LIVING, PC_HARDWARE, PET_SUPPLIES, SHOES, SOFTWARE, SPORTING_GOODS, TOOLS, TOYS, VIDEO_GAMES, WATCHES, WIRELESS, WIRELESS_ACCESSORIES)	
Query	**publishDate** *optional*	publishDate	integer(int64)	
Query	**publisher** *optional*	publisher	string	

Type	Name	Description	Schema	Default
Query	**rebootTimeHour** *optional*	rebootTimeHour	integer(int32)	
Query	**rebootTimeMinute** *optional*	rebootTimeMinute	integer(int32)	
Query	**redeemableEnd** *optional*	redeemableEnd	integer(int64)	
Query	**redeemableStart** *optional*	redeemableStart	integer(int64)	
Query	**releaseDate** *optional*	releaseDate	integer(int64)	
Query	**replaced** *optional*	replaced	boolean	
Query	**retailerLocationIds** *optional*	retailerLocationIds	string	
Query	**serialNumber** *optional*	serialNumber	string	
Query	**showRedeemed** *optional*	showRedeemed	boolean	
Query	**showRemaining** *optional*	showRemaining	boolean	
Query	**sizeId** *optional*	sizeId	integer(int64)	
Query	**specialOfferType** *required*	specialOfferType	enum (ALL, RESERVABLE, REGULAR_OFFER, ACT_NOW, GET_THERE_NOW, SQOOT, TICKETS, YIPIT)	`"GET_THERE_NOW"`
Query	**startDate** *optional*	startDate	integer(int64)	

Type	Name	Description	Schema	Default
Query	**studio** *optional*	studio	string	
Query	**subDetails** *optional*	subDetails	string	
Query	**subTitle** *optional*	subTitle	string	
Query	**tags** *optional*	tags	string	
Query	**ticketPrice** *required*	ticketPrice	integer(int64)	`"0"`
Query	**ticketPriceType** *optional*	ticketPriceType	string	
Query	**ticketsReward** *optional*	ticketsReward	integer(int64)	
Query	**ticketsRewardType** *optional*	ticketsRewardType	string	
Query	**title** *required*	title	string	
Query	**transactionAudienceIds** *optional*	transactionAudienceIds	string	
Query	**transactionAudienceIdsToAdd** *optional*	transactionAudienceIdsToAdd	string	
Query	**transactionAudienceIdsToRemove** *optional*	transactionAudienceIdsToRemove	string	
Query	**udid** *optional*	udid	string	
Query	**unit** *optional*	unit	enum (MILIMETER, CENTIMETER, METER, INCH, FOOT, YARD)	

Type	Name	Description	Schema	Default
Query	**viewLimit** *required*	viewLimit	integer(int32)	`"0"`
Query	**weight** *optional*	weight	number(double)	
Query	**width** *optional*	width	number(double)	

Responses

HTTP Code	Description	Schema
200	successful operation	RetailerOfferResponse

6.44.7. Delete Offer

```
POST /api/{version}/retailer/offer/delete
```

Description

Set the deleted timestamp to current time. This effectively deletes the offer since all queries should ignore any records with a deleted time stamp.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**offerId** *required*	offerId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.44.8. Get Offer

```
POST /api/{version}/retailer/offer/get
```

Description

Gets the details of an offer that the user has access to.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**includeOfferLocations** *required*	includeOfferLocations	boolean	`"false"`
Query	**includeRetailerLocations** *required*	includeRetailerLocations	boolean	`"false"`
Query	**offerId** *required*	offerId	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	RetailerOfferResponse

6.44.9. Update Offer Locations

```
POST /api/{version}/retailer/offer/location/batchUpdate
```

Description

Batch update offer locations.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**data** *required*	data	string
Query	**deviceId** *optional*	deviceId	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.44.10. Delete Offer Location

```
POST /api/{version}/retailer/offer/location/delete
```

Description

Set the deleted timestamp to current time. This effectively deletes the offer location since all queries should ignore any records with a deleted time stamp.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)

Type	Name	Description	Schema
Query	**deviceId** *optional*	deviceId	string
Query	**offerLocationId** *required*	offerLocationId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.44.11. Search Offer Locations

```
POST /api/{version}/retailer/offer/location/search
```

Description

Searches on offer locations, which are records that represent an offer that has been assigned to a retailer location. If an offer does not have any locations assigned, then it will NOT be returned.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activeOnly** *required*	activeOnly	boolean	`"false"`
Query	**asin** *optional*	asin	string	
Query	**barcodeEntry** *optional*	barcodeEntry	string	
Query	**barcodeType** *optional*	barcodeType	string	
Query	**descending** *required*	descending	boolean	`"false"`

Type	Name	Description	Schema	Default
Query	**deviceId** *optional*	deviceId	string	
Query	**deviceStatus** *optional*	deviceStatus	enum (RUNNING, WARNING, DOWN)	
Query	**includeRetailerLocations** *required*	includeRetailerLocations	boolean	`"true"`
Query	**isbn** *optional*	isbn	string	
Query	**keyword** *optional*	keyword	string	
Query	**lastNotificationSent** *optional*	lastNotificationSent	integer(int64)	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**needsNotificationSent** *optional*	needsNotificationSent	boolean	
Query	**offerType** *optional*	offerType	enum (VOUCHER, COUPON, PRODUCT, MEDIA, EVENT, DEVICE)	
Query	**retailerId** *optional*	retailerId	integer(int64)	
Query	**retailerLocationId** *optional*	retailerLocationId	integer(int64)	

Type	Name	Description	Schema	Default
Query	**sortField** *required*	sortField	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, OFFER_ID, ACTIVATED, EXPIRES, TITLE, SUBTITLE, DETAILS, OFFER_TYPE, SPECIAL_OFFER_TYPE, OFFER_VISIBILITY, RETAILER_ID, RETAILER_LOCATION_ID, BILLABLE_ENTITY_ID, BILLABLE_ENTITY_NAME, RESPONSIBLE_DISPLAY)	`"TITLE"`
Query	**specialOfferType** *optional*	specialOfferType	enum (ALL, RESERVABLE, REGULAR_OFFER, ACT_NOW, GET_THERE_NOW, SQOOT, TICKETS, YIPIT)	
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< OfferShortResponse > array

6.44.12. Search Offers

```
POST /api/{version}/retailer/offer/search
```

Description

Searches on offers that the account has access to.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**_i** *optional*	_i	integer(int32)	
Query	**_l** *optional*	_l	integer(int32)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activeOnly** *required*	activeOnly	boolean	`"false"`
Query	**appKey** *optional*	appKey	string	
Query	**asin** *optional*	asin	string	
Query	**availableOnly** *required*	availableOnly	boolean	`"false"`
Query	**barcodeEntry** *optional*	barcodeEntry	string	
Query	**barcodeType** *optional*	barcodeType	string	
Query	**categoryIds** *optional*	categoryIds	string	
Query	**couponType** *optional*	couponType	enum (VOUCHER, COUPON, PRODUCT, MEDIA, EVENT, DEVICE)	
Query	**descending** *required*	descending	boolean	`"false"`

Type	Name	Description	Schema	Default
Query	**deviceId** *optional*	deviceId	string	
Query	**deviceStatus** *optional*	deviceStatus	enum (RUNNING, WARNING, DOWN)	
Query	**endedBefore** *optional*	endedBefore	integer(int64)	
Query	**endedSince** *optional*	endedSince	integer(int64)	
Query	**filterIds** *optional*	filterIds	string	
Query	**includeCategories** *required*	includeCategories	boolean	`"false"`
Query	**includeFilters** *required*	includeFilters	boolean	`"false"`
Query	**includeOfferLocations** *required*	includeOfferLocations	boolean	`"false"`
Query	**includeRetailerLocations** *required*	includeRetailerLocations	boolean	`"false"`
Query	**isbn** *optional*	isbn	string	
Query	**keyword** *optional*	keyword	string	
Query	**lastNotificationSent** *optional*	lastNotificationSent	integer(int64)	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**needsNotificationSent** *optional*	needsNotificationSent	boolean	
Query	**offerAudienceIds** *optional*	offerAudienceIds	string	

Type	Name	Description	Schema	Default
Query	**offerType** *optional*	offerType	enum (VOUCHER, COUPON, PRODUCT, MEDIA, EVENT, DEVICE)	
Query	**offerTypes** *optional*	offerTypes	string	
Query	**offerVisibility** *required*	offerVisibility	enum (PUBLIC, LISTABLE, REWARDABLE, TRIGGERABLE, PRIVATE)	`"PUBLIC"`
Query	**q** *optional*	q	string	
Query	**retailerId** *optional*	retailerId	integer(int64)	
Query	**retailerLocationId** *optional*	retailerLocationId	integer(int64)	

Type	Name	Description	Schema	Default
Query	**sortField** *required*	sortField	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, ACTIVATED, EXPIRES, REDEEMABLE_START, REDEEMABLE_END, TITLE, SUBTITLE, DETAILS, OFFER_TYPE, SPECIAL_OFFER_TYPE, OFFER_VISIBILITY, ESTIMATED_VALUE, VOUCHER_PRICE, VOUCHER_DISCOUNT_PRICE, FULL_PRICE, DICOUNT_PRICE, TICKETS_REWARD, AVAILABILITY_DATE, RELEASE_DATE, RETAILER_ID, RETAILER_NAME, RETAILER_LOCATION_ID, RETAILER_LOCATION_NAME, BILLABLE_ENTITY_ID, BILLABLE_ENTITY_NAME, RESPONSIBLE_DISPLAY)	`"TITLE"`

Type	Name	Description	Schema	Default
Query	**specialOfferType** *optional*	specialOfferType	enum (ALL, RESERVABLE, REGULAR_OFFER, ACT_NOW, GET_THERE_NOW, SQOOT, TICKETS, YIPIT)	
Query	**start** *required*	start	integer(int32)	`"0"`
Query	**startedBefore** *optional*	startedBefore	integer(int64)	
Query	**startedSince** *optional*	startedSince	integer(int64)	
Query	**transactionAudienceIds** *optional*	transactionAudienceIds	string	

Responses

HTTP Code	Description	Schema
200	successful operation	< OfferResponse > array

6.44.13. Activate Offer

```
POST /api/{version}/retailer/offer/status
```

Description

Sets the activated date on offers. This will make offers visible for consumers.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	

Type	Name	Description	Schema	Default
Query	**active** *required*	active	boolean	`"true"`
Query	**deviceId** *optional*	deviceId	string	
Query	**offerIds** *required*	offerIds	string	

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.44.14. Search Offer Transactions

```
POST /api/{version}/retailer/offer/transaction/search
```

Description

Searches on offer transactions for offers that the account has access to.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**_i** *optional*	_i	integer(int32)	
Query	**_l** *optional*	_l	integer(int32)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activeOnly** *required*	activeOnly	boolean	`"true"`
Query	**categoryIds** *optional*	categoryIds	string	

Type	Name	Description	Schema	Default
Query	**couponType** *optional*	couponType	enum (VOUCHER, COUPON, PRODUCT, MEDIA, EVENT, DEVICE)	
Query	**customerAccountIds** *optional*	customerAccountIds	string	
Query	**descending** *required*	descending	boolean	`"true"`
Query	**deviceId** *optional*	deviceId	string	
Query	**endedBefore** *optional*	endedBefore	integer(int64)	
Query	**endedSince** *optional*	endedSince	integer(int64)	
Query	**filterByParentOffer** *optional*	filterByParentOffer	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**offerId** *optional*	offerId	integer(int64)	
Query	**offerLocationId** *optional*	offerLocationId	integer(int64)	
Query	**offerType** *optional*	offerType	enum (VOUCHER, COUPON, PRODUCT, MEDIA, EVENT, DEVICE)	
Query	**q** *optional*	q	string	
Query	**redeemableEndDate** *optional*	redeemableEndDate	integer(int64)	

Type	Name	Description	Schema	Default
Query	**redeemableStartDate** *optional*	redeemableStartDate	integer(int64)	
Query	**redeemed** *optional*	redeemed	boolean	
Query	**reservationsOnly** *optional*	reservationsOnly	boolean	
Query	**retailerId** *optional*	retailerId	integer(int64)	
Query	**retailerLocationId** *optional*	retailerLocationId	integer(int64)	
Query	**sortField** *required*	sortField	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, ACTIVATED, EXPIRES, TITLE, SUBTITLE, DETAILS, OFFER_TYPE, SPECIAL_OFFER_TYPE, OFFER_VISIBILITY, REDEEMABLE_START, REDEEMABLE_END, CUSTOMER_ID, CUSTOMER_DISPLAY, RETAILER_ID, RETAILER_NAME, RETAILER_LOCATION_ID, RETAILER_LOCATION_NAME, BILLABLE_ENTITY_ID, BILLABLE_ENTITY_NAME, RESPONSIBLE_DISPLAY)	`"CREATED"`

Type	Name	Description	Schema	Default
Query	**specialOfferType** *optional*	specialOfferType	enum (ALL, RESERVABLE, REGULAR_OFFER, ACT_NOW, GET_THERE_NOW, SQOOT, TICKETS, YIPIT)	
Query	**start** *required*	start	integer(int32)	`"0"`
Query	**startedBefore** *optional*	startedBefore	integer(int64)	
Query	**startedSince** *optional*	startedSince	integer(int64)	
Query	**validateByOfferBillableEntity** *optional*	validateByOfferBillableEntity	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	< OfferTransactionResponse > array

6.44.15. Update Offer Transaction

```
POST /api/{version}/retailer/offer/transaction/update
```

Description

Redeems an offer.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	

Type	Name	Description	Schema	Default
Query	**accountId** *required*	accountId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**offerLocationId** *optional*	offerLocationId	integer(int64)	
Query	**offerTransactionId** *required*	offerTransactionId	integer(int64)	
Query	**status** *required*	status	integer(int32)	"2"

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.44.16. Update Offer

```
POST /api/{version}/retailer/offer/update
```

Description

Update an offer, must provide a current list of retailer locations or the current offer locations will be marked as deleted.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activated** *optional*	activated	integer(int64)	
Query	**active** *optional*	active	boolean	

Type	Name	Description	Schema	Default
Query	**addedLimit** *optional*	addedLimit	integer(int32)	
Query	**asin** *optional*	asin	string	
Query	**author** *optional*	author	string	
Query	**availability** *optional*	availability	string	
Query	**availabilityDate** *optional*	availabilityDate	integer(int64)	
Query	**availabilitySummary** *optional*	availabilitySummary	string	
Query	**availableLimit** *optional*	availableLimit	integer(int32)	
Query	**availableLimitPerUser** *optional*	availableLimitPerUser	integer(int32)	
Query	**barcodeAssetId** *optional*	barcodeAssetId	integer(int64)	
Query	**barcodeEntry** *optional*	barcodeEntry	string	
Query	**barcodeType** *optional*	barcodeType	enum (NONE, UPC, CODE_128, QR, CUSTOM_MEDIA)	
Query	**brand** *optional*	brand	string	
Query	**catalogNumbers** *optional*	catalogNumbers	string	
Query	**categoryIds** *optional*	categoryIds	string	
Query	**collectionIds** *optional*	collectionIds	string	

Type	Name	Description	Schema	Default
Query	**conditionType** *optional*	conditionType	enum (NEW, USED, REFURBISHED, OTHER)	
Query	**department** *optional*	department	string	
Query	**depth** *optional*	depth	number(double)	
Query	**details** *optional*	details	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**deviceInterference** *optional*	deviceInterference	number(double)	
Query	**devicePower** *optional*	devicePower	number(double)	
Query	**deviceType** *optional*	deviceType	string	
Query	**discountPrice** *optional*	discountPrice	number(double)	
Query	**duration** *optional*	duration	integer(int32)	
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**expires** *optional*	expires	integer(int64)	
Query	**externalId** *optional*	externalId	string	
Query	**externalRedeemOptions** *optional*	externalRedeemOptions	string	
Query	**externalUrl** *optional*	externalUrl	string	
Query	**featured** *optional*	featured	boolean	

Type	Name	Description	Schema	Default
Query	**features** *optional*	features	string	
Query	**filterIds** *optional*	filterIds	string	
Query	**finePrint** *optional*	finePrint	string	
Query	**fullPrice** *optional*	fullPrice	number(double)	
Query	**height** *optional*	height	number(double)	
Query	**idleTimeoutInSecond** *optional*	idleTimeoutInSecond	integer(int32)	
Query	**imageAssetId** *optional*	imageAssetId	integer(int64)	
Query	**imageAssetId1** *optional*	imageAssetId1	integer(int64)	
Query	**imageAssetId2** *optional*	imageAssetId2	integer(int64)	
Query	**imageAssetId3** *optional*	imageAssetId3	integer(int64)	
Query	**imageAssetId4** *optional*	imageAssetId4	integer(int64)	
Query	**imageAssetId5** *optional*	imageAssetId5	integer(int64)	
Query	**includeOfferLocations** *required*	includeOfferLocations	boolean	`"false"`
Query	**includeRetailerLocations** *required*	includeRetailerLocations	boolean	`"false"`
Query	**isbn** *optional*	isbn	string	

Type	Name	Description	Schema	Default
Query	**listingId** *optional*	listingId	integer(int64)	
Query	**maxPrints** *optional*	maxPrints	integer(int32)	
Query	**mediaType** *optional*	mediaType	enum (APK, APPLICATION, IMAGE, AUDIO, VIDEO, MULTIPART, BAR_CODE, TEXT, OTHER)	
Query	**metaData** *optional*	metaData	string	
Query	**minimumPrice** *optional*	minimumPrice	number(double)	
Query	**noExpiration** *optional*	noExpiration	boolean	
Query	**offerAudienceIds** *optional*	offerAudienceIds	string	
Query	**offerAudienceIdsToAdd** *optional*	offerAudienceIdsToAdd	string	
Query	**offerAudienceIdsToRemove** *optional*	offerAudienceIdsToRemove	string	
Query	**offerId** *required*	offerId	integer(int64)	
Query	**offerLocations** *optional*	offerLocations	string	
Query	**offerType** *optional*	offerType	enum (VOUCHER, COUPON, PRODUCT, MEDIA, EVENT, DEVICE)	

Type	Name	Description	Schema	Default
Query	**offerVisibility** *optional*	offerVisibility	enum (PUBLIC, LISTABLE, REWARDABLE, TRIGGERABLE, PRIVATE)	
Query	**parentOfferId** *optional*	parentOfferId	integer(int64)	
Query	**parentalRating** *optional*	parentalRating	string	

Type	Name	Description	Schema	Default
Query	**productType** *optional*	productType	enum (APPAREL, APPLIANCES, APPS, ARTS_CRAFTS, AUTOMOTIVE, BABY, BEAUTY, BOOKS, CLASSICAL, COLLECTIBLES, DVD, ELECTRONICS, GIFT_CARDS, GOURMET_FOOD, GROCERY, HEALTH_PERSONAL_CARE, HOME_GARDEN, INDUSTRIAL, JEWELRY, KITCHEN, LAWN_GARDEN, MAGAZINES, MISCELLANEOUS, MUSICAL_INSTRUMENTS, OFFICE_PRODUCTS, OTHER, OUTDOOR_LIVING, PC_HARDWARE, PET_SUPPLIES, SHOES, SOFTWARE, SPORTING_GOODS, TOOLS, TOYS, VIDEO_GAMES, WATCHES, WIRELESS, WIRELESS_ACCESSORIES)	
Query	**publishDate** *optional*	publishDate	integer(int64)	
Query	**publisher** *optional*	publisher	string	

Type	Name	Description	Schema	Default
Query	**rebootTimeHour** *optional*	rebootTimeHour	integer(int32)	
Query	**rebootTimeMinute** *optional*	rebootTimeMinute	integer(int32)	
Query	**redeemableEnd** *optional*	redeemableEnd	integer(int64)	
Query	**redeemableStart** *optional*	redeemableStart	integer(int64)	
Query	**releaseDate** *optional*	releaseDate	integer(int64)	
Query	**replaced** *optional*	replaced	boolean	
Query	**retailerLocationIds** *optional*	retailerLocationIds	string	
Query	**serialNumber** *optional*	serialNumber	string	
Query	**showRedeemed** *optional*	showRedeemed	boolean	
Query	**showRemaining** *optional*	showRemaining	boolean	
Query	**sizeId** *optional*	sizeId	integer(int64)	
Query	**specialOfferType** *optional*	specialOfferType	enum (ALL, RESERVABLE, REGULAR_OFFER, ACT_NOW, GET_THERE_NOW, SQOOT, TICKETS, YIPIT)	
Query	**startDate** *optional*	startDate	integer(int64)	

Type	Name	Description	Schema	Default
Query	**studio** *optional*	studio	string	
Query	**subDetails** *optional*	subDetails	string	
Query	**subTitle** *optional*	subTitle	string	
Query	**tags** *optional*	tags	string	
Query	**ticketPrice** *optional*	ticketPrice	integer(int64)	
Query	**ticketPriceType** *optional*	ticketPriceType	string	
Query	**ticketsReward** *optional*	ticketsReward	integer(int64)	
Query	**ticketsRewardType** *optional*	ticketsRewardType	string	
Query	**title** *optional*	title	string	
Query	**transactionAudienceIds** *optional*	transactionAudienceIds	string	
Query	**transactionAudienceIdsToAdd** *optional*	transactionAudienceIdsToAdd	string	
Query	**transactionAudienceIdsToRemove** *optional*	transactionAudienceIdsToRemove	string	
Query	**udid** *optional*	udid	string	
Query	**unit** *optional*	unit	enum (MILIMETER, CENTIMETER, METER, INCH, FOOT, YARD)	

Type	Name	Description	Schema	Default
Query	**viewLimit** *optional*	viewLimit	integer(int32)	
Query	**weight** *optional*	weight	number(double)	
Query	**width** *optional*	width	number(double)	

Responses

HTTP Code	Description	Schema
200	successful operation	RetailerOfferResponse

6.45. OfferStatus

The Offer Status API allows you to manage offer status codes to for your business to ensure that offers are correctly purchased, added to the consumer's wallet, and redeemed appropriately. There are custom status codes that represent each status and what they do: 0: offer purchased or added to the wallet. 1: consumer has indicated they want to redeem the offer. 2: merchant has accepted the offer and marked it as redeemed. 3: consumer has archived the offer. 4: consumer has been awarded the offer (can be purchased if desired). With Offer Status, you can track each and every transaction that occurs for your offers from when the user views it to when a purchase is made. Detailed reports can then be viewed on via a web dashboard.

Below are some example scenarios as to how the Offer Status service can be used:

- Retail: Create and track multiple offers in an A/B test then view reporting to know what types of offers are redeemed most.
- Medical: Create seasonal offers for your patients and track the engagement. For example, "Discount on Flu Shots" or "Save 10% on Spring Allergy Medicine" can all be tracked and measured so you can know what kinds of offers resonate with your patients.
- Smart Home: Use Offer Status to track different types of offers that can be purchased through your smart home device such as "weather based" or "holiday focused" so you can analyze results and drive more revenue.
- Automotive: Use Offer Status to compare different offers types, such as "maintenance" or "roadside attractions." Use the metrics to continually improve the types of offers that are most effective to increase user satisfaction.
- Robotics: Use Offer Status to reveal what types of promotions increase drone deliveries. For example, "10% off printer ink" might outperform "20% on grocery delivery."

The Offer Status API is important because it gives you clear insights regarding your users' engagement along the offer path from viewing to redeeming an offer.

Below are great example services that could be combined with the Offer Status service:

- Audience: Test various offers within different Audiences to optimize types of offers to send in the future.
- Events: Use Offer Status in combination with Events to know the most profitable kinds of Events to promote to your users in the future.
- Game Levels: Test and measure offers at different Game Levels for effectiveness.

6.45.1. Create Offer Status

```
POST /api/{version}/offer/status/create
```

Description

Create an offer status record

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The version of the api to response with	number(double)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**active** *optional*	The active status	boolean	`"true"`
Query	**applicationIds** *optional*	The applications to associate the status with, if null then for all.	string	
Query	**code** *required*	The status code, must be unique	integer(int32)	
Query	**description** *optional*	The description of the status	string	
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string	
Query	**latitude** *optional*	Used to update the user's current location	number(double)	

Type	Name	Description	Schema	Default
Query	**longitude** *optional*	Used to update the user's current location	number(double)	
Query	**name** *required*	The name of the status	string	
Query	**role** *optional*	The actor that uses the status	string	`"ANY"`

Responses

HTTP Code	Description	Schema
200	successful operation	OfferTransactionStatusResponse

6.45.2. Delete Offer Status

```
POST /api/{version}/offer/status/delete
```

Description

Mark an offer status record as deleted

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The version of the api to response with	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string
Query	**latitude** *optional*	Used to update the user's current location	number(double)
Query	**longitude** *optional*	Used to update the user's current location	number(double)
Query	**statusId** *required*	The id of the record to delete	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.45.3. Get Offer Status

```
POST /api/{version}/offer/status/get
```

Description

Get an offer status record

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The version of the api to response with	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string
Query	**latitude** *optional*	Used to update the user's current location	number(double)
Query	**longitude** *optional*	Used to update the user's current location	number(double)
Query	**statusId** *required*	The id of the record to get	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	OfferTransactionStatusResponse

6.45.4. Search Offer Status

```
POST /api/{version}/offer/status/search
```

Description

Search for the available offer statuses

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The version of the api to response with	number(double)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**appKey** *optional*	The application making the request	string	
Query	**descending** *optional*	Determines whether the results are in descending order	boolean	`"true"`
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string	
Query	**includeInactive** *optional*	If true include inactive items	boolean	`"false"`
Query	**keyword** *optional*	Search the name and description fields	string	
Query	**latitude** *optional*	Used to update the user's current location	number(double)	
Query	**limit** *optional*	The limit for pagination	integer(int32)	`"20"`
Query	**longitude** *optional*	Used to update the user's current location	number(double)	
Query	**role** *optional*	The actors role, possible values are: CUSTOMER, RETAILER, or ANY	string	
Query	**sortField** *optional*	The field to sort on, possible values are: ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, NAME, DESCRIPTION, CODE, ROLE	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, NAME, DESCRIPTION, CODE, ROLE)	`"CODE"`

Type	Name	Description	Schema	Default
Query	**start** *optional*	The start index for pagination	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< OfferTransactionStatusResponse > array

6.45.5. Update Offer Status

```
POST /api/{version}/offer/status/update
```

Description

Update an offer status record

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The version of the api to response with	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**active** *optional*	The active status	boolean
Query	**applicationIds** *optional*	The applications to associate the status with, if null then for all.	string
Query	**code** *optional*	The status code, must be unique	integer(int32)
Query	**description** *optional*	The description of the status	string
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string
Query	**latitude** *optional*	Used to update the user's current location	number(double)

Type	Name	Description	Schema
Query	**longitude** *optional*	Used to update the user's current location	number(double)
Query	**name** *optional*	The name of th status	string
Query	**role** *optional*	The actor that uses the status	string
Query	**statusId** *optional*	The id of the record to update	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	OfferTransactionStatusResponse

6.46. Optimize

Route optimization is a great tool that can be used to make sure a route is using the most efficient path and most effective order of stops. Routes can be optimized based on weather, road conditions, construction, and more.

Below are some example scenarios as to how these services can be used:

- Automotive: A delivery service can run route optimization to find the most efficient route to take. The company saves money and fuel by using this optimized route.
- Medical: If multiple stops need to be made for a medical emergency, or if multiple vehicles need to be dispatched, a hostpital can use route optimization to quickly find the best path and stop order to take.
- Robotics: Optimize the route that a set of robots take to perform their tasks. Use this before a simulation for the most efficient way of finding the proper route.

With route optimization, organizations can easily make a new route on the fly (or use a previously created route) and optimize it for the most efficient path and stop order.

Below are great example services that could be combined with this service:

- Notification Service: Send a notification to a dispatcher if there is a more optimized route available than waht is currently being used. Alterantively, send a notification to a driver if their route has been optimized and updated..
- Tracking Service: Track your dispatched vehicles on the road and make sure they are following the

optimized route.

- Dashboards: Easily see all of your routes in an organized list, and choose which routes need optimization.

6.46.1. Request Optimization

```
POST /api/{version}/optimize/request
```

Description

Request and upload of shipment orders and create ShipmentImportBatch for optimization.

Parameters

Type	Name	Schema
Header	**Application-Key** *required*	string
Header	**Application-Rest-Key** *required*	string
Path	**version** *required*	string
Body	**body** *optional*	Orders

Responses

HTTP Code	Description	Schema
200	successful operation	ImportStatuses

6.46.2. Get Optimization Result

```
GET /api/{version}/optimize/result/{batchID}
```

Description

Get the results of the import batch.

Parameters

Type	Name	Schema	Default
Header	**Application-Key** *required*	string	
Header	**Application-Rest-Key** *required*	string	
Path	**batchID** *required*	string	
Path	**version** *required*	string	
Query	**limit** *required*	integer(int32)	`"1500"`
Query	**start** *required*	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< string, ShipmentOrder > map

6.47. Pack

The Packs API lets you create a collection of Game Levels that are viewable within your application. The most obvious example is creating various groups of game levels that are can be built-in, downloadable, or purchasable with digital currency. Packs are great for adding more content and playable levels within your game on an ongoing basis.

Below are some example scenarios as to how the Rating service can be used:

- Retail: Create Packs based on Audiences to highlight downloadable content that is targeted to specific types of customers such as “moms” or “diy’rs” or “sports fanatics”.
- Medical: Use Packs to deliver specific educational content to specific groups of employees via your internal learning application, such as “ER staff” or “Reception Desk staff” so they have the latest quizzes related to their particular job tasks.
- Smart Home: Use Packs to create content that is delivered to specific devices within the home based on Audience. For example, “children” or “movie geeks” can receive different entertainment based on who is listening or watching.
- Automotive: Use Packs to deliver educational content to different types of fleet drivers in your

enterprise. For example, “fork lift drivers” or “delivery trucks” can each have targeted information for learning how to do a better job via your internal education application

- Robotics: Use Packs to deliver specific content to your application users based on items that will be delivered by drone, such as “how to use” videos or “great add-on accessories” videos.

The Packs API is important because it allows you to package up specific groups of content in creative ways for your application users to consume.

Below are great example services that could be combined with the Packs service:

- Coupons or Offers: Use Coupons or Offers to be awarded to your application users when they download certain Packs.
- Groups: Use Packs in combination with Groups to deliver specific content targeted at various groups.
- Retailers: Deliver region specific Packs to various groups that have visited particular physical retail locations for your business.
- Missions: Reward your application users by unlocking Packs when they complete certain Missions.

6.47.1. Create Pack

```
POST /api/{version}/pack/create
```

Description

Create a pack.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	`"false"`
Query	**allocateTickets** *required*	allocateTickets	boolean	`"false"`
Query	**appKey** *optional*	appKey	string	

Type	Name	Description	Schema	Default
Query	**authorOverride** *optional*	authorOverride	string	
Query	**backgroundId** *optional*	backgroundId	integer(int64)	
Query	**description** *optional*	description	string	
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**gameLevelIds** *optional*	gameLevelIds	string	
Query	**gameType** *optional*	gameType	string	
Query	**highest** *required*	highest	boolean	`"true"`
Query	**imageId** *optional*	imageId	integer(int64)	
Query	**inGame** *optional*	inGame	boolean	
Query	**metaData** *optional*	metaData	string	
Query	**packOrder** *required*	packOrder	integer(int64)	`"0"`
Query	**packType** *optional*	packType	enum (TUTORIAL, BUILTIN, DOWNLOAD, THRESHOLD, THEME, TOURNAMENT)	
Query	**points** *optional*	points	integer(int64)	
Query	**price** *required*	price	integer(int32)	`"0"`
Query	**priceType** *optional*	priceType	string	

Type	Name	Description	Schema	Default
Query	**searchTags** *optional*	searchTags	string	
Query	**sequenceType** *optional*	sequenceType	enum (FIRST_AVAILABLE, ALL_AVAILABLE)	
Query	**startDate** *optional*	startDate	integer(int64)	
Query	**ticketCount** *required*	ticketCount	integer(int64)	`"0"`
Query	**ticketType** *optional*	ticketType	string	
Query	**title** *required*	title	string	

Responses

HTTP Code	Description	Schema
200	successful operation	PackResponse

6.47.2. Delete Pack

```
POST /api/{version}/pack/delete
```

Description

Delete a pack.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**packId** *required*		integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.47.3. Get Pack

```
POST /api/{version}/pack/get
```

Description

Get a pack.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**includeGameData** *required*	includeGameData	boolean	`"false"`
Query	**packId** *required*		integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	PackResponse

6.47.4. Search Packs

```
POST /api/{version}/pack/search
```

Description

Search on packs.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**descending** *required*	descending	boolean	`"false"`
Query	**includeGameData** *optional*	includeGameData	boolean	`"false"`
Query	**includeInactive** *optional*	includeInactive	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**limit** *optional*	limit	integer(int32)	`"20"`
Query	**packType** *optional*	packType	enum (TUTORIAL, BUILTIN, DOWNLOAD, THRESHOLD, THEME, TOURNAMENT)	
Query	**sortField** *required*	sortField	enum (TITLE, DESCRIPTION, CREATED, UPDATED)	`"TITLE"`
Query	**start** *optional*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< PackResponse > array

6.47.5. Update Pack

```
POST /api/{version}/pack/update
```

Description

Update a pack.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	
Query	**allocateTickets** *required*	allocateTickets	boolean	`"false"`
Query	**appKey** *optional*	appKey	string	
Query	**authorOverride** *optional*	authorOverride	string	
Query	**backgroundId** *optional*	backgroundId	integer(int64)	
Query	**description** *optional*	description	string	
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**gameLevelIds** *optional*	gameLevelIds	string	
Query	**gameType** *optional*	gameType	string	
Query	**highest** *optional*	highest	boolean	

Type	Name	Description	Schema	Default
Query	**imageId** *optional*	imageId	integer(int64)	
Query	**inGame** *optional*	inGame	boolean	
Query	**metaData** *optional*	metaData	string	
Query	**packId** *required*		integer(int64)	
Query	**packOrder** *optional*	packOrder	integer(int64)	
Query	**packType** *optional*	packType	enum (TUTORIAL, BUILTIN, DOWNLOAD, THRESHOLD, THEME, TOURNAMENT)	
Query	**points** *optional*	points	integer(int64)	
Query	**price** *optional*	price	integer(int32)	
Query	**priceType** *optional*	priceType	string	
Query	**searchTags** *optional*	searchTags	string	
Query	**sequenceType** *optional*	sequenceType	enum (FIRST_AVAILABLE, ALL_AVAILABLE)	
Query	**startDate** *optional*	startDate	integer(int64)	
Query	**ticketCount** *required*	ticketCount	integer(int64)	`"0"`
Query	**ticketType** *optional*	ticketType	string	
Query	**title** *optional*	title	string	

Responses

HTTP Code	Description	Schema
200	successful operation	PackResponse

6.48. Pathing

The Pathing API used in combination with Location computes the shortest “path” from point to point on a grid. The grid can be authored to include “exclusion points” such as walls, displays, or other solid objects. These objects are automatically taken into account when calculating the shortest path. Pathing allows your users to easily get from one area of your business to another in the most efficient way.

Below are some example scenarios as to how the Pathing service can be used:

- Retail: Use Pathing so that customers can easily find certain departments or even products in your store such as “Men’s Department” or “Shoes”
- Medical: Use Pathing to help your patients navigate your facility. For example, “Waiting Area” or “Coffee Shop” or “Restrooms” or “X-ray Room.”
- Smart Home: Use Pathing with object beacons in order to find certain items in the home such as “keys” or “wallet.”
- Automotive: Use Pathing to identify and give directions to loading docks or empty parking spaces on your premises to help your fleet drivers be more time efficient.
- Robotics: Use Pathing to program robots on a warehouse floor to take the most efficient routes to increase productivity when traveling from point to point.

The Pathing API is important because it is an excellent way to bring the power of “route mapping” into an indoor space or a previously non-mapped space within large facilities.

Below are great example services that could be combined with the Pathing service:

- Audiences: Use different Pathing routes for different Audiences to create better flow within your venue. For example, ticket holders at a Sports Stadium going to different levels can be shown the best routes.
- Missions: Use Pathing in combination with Missions to guide your users from point to point as quickly as possible. This can also be used in a business environment to guide employees through your facilities.
- Offers: Combine Pathing with Offers and beacons so that users get rewarded as they go from point to point through your store or business.
- Packs: Combine Pathing with Packs so customers can unlock area-specific content as needed along the route they are taking within your store or business. For example, unlock “Gardening Tips” videos when customers reach the Gardening section of your store.

6.48.1. Calculate Path

```
POST /api/{version}/pathing/compute
```

Description

Calculates the shortest path from point to point on a grid

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**data** *required*	data	string	
Query	**directions** *required*	directions	boolean	`"true"`
Query	**exclusionsAssetId** *optional*	exclusionsAssetId	integer(int64)	
Query	**maxFeetPerMin** *optional*	maxFeetPerMin	number(double)	
Query	**reducePath** *required*	reducePath	boolean	`"true"`
Query	**units** *required*	units	enum (METRIC, IMPERIAL)	`"METRIC"`

Responses

HTTP Code	Description	Schema
200	successful operation	PathingResponse

6.49. PostalCode

The Postal Code API allows for various actions on (or to) Postal Codes within an application. These actions include create, delete, get, update, and search. Postal Codes are also the basis for creating “Regions” or “Territories” within your application. Your application users can enter or update their Postal Code for their address. Blocks of Postal Codes can be identified as active service areas for

particular businesses such as package or food delivery, or, home services. Think of typical applications that you have used such as banking apps to find an ATM, or Retailer apps to locate nearby stores.

Below are some example scenarios as to how the Postal Code service can be used:

- Retail: Customers can search for retail locations within a specific Postal Code, or, find retail locations within their current Postal Code.
- Medical: Doctors or Nurses can be assigned to make house calls to patients within certain Postal Codes. Or, various medical facilities can be recommended to patients based on Postal Codes.
- Real Estate: See groupings of properties within specific Postal Codes.
- Automotive: Display maps by Postal Code where ride-sharing services are available.
- Robotics: Drone services can be set to operate or deliver only within specific Postal Codes.

The Postal Code API is important because it operates off of a known geographic parameter. It's also important because it can be used to generate Regions or Territories within your application. Postal Codes can be used both as information to end users of your application, or, used as data to gain insights as to the general locations of your end users.

Below are great example services that could be combined with the PostalCode service:

- Notifications Service: Send push notifications only to users within a specific Postal Code(s), Region or Territory.
- Retailer Service: Display Retail locations within a specific Postal Code, Region or Territory.
- Analytics Service: See Postal Code data relative to time spent by end users within certain Postal Code, Region or Territory.
- Employee Service: Show location of Employees for assigning work or deliveries geographically closer to their current location in a specific Postal Code, Region or Territory.

6.49.1. Create Postal Code

```
POST /api/{version}/postalCode/create
```

Description

Create a Postal Code

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	

Type	Name	Description	Schema	Default
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	`"true"`
Query	**city** *optional*	city	string	
Query	**code** *required*	code	string	
Query	**latitude** *required*	latitude	number(double)	
Query	**longitude** *required*	longitude	number(double)	
Query	**stateCode** *optional*	stateCode	string	

Responses

HTTP Code	Description	Schema
200	successful operation	PostalCodeResponse

6.49.2. Delete Postal Code

```
POST /api/{version}/postalCode/delete
```

Description

Delete a Postal Code

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)

Type	Name	Description	Schema
Query	**postalCodeId** *required*	postalCodeId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.49.3. Get Postal Code

```
POST /api/{version}/postalCode/get
```

Description

Get a Postal Code

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**postalCodeId** *required*	postalCodeId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	PostalCodeRespons e

6.49.4. Search Postal Codes

```
POST /api/{version}/postalCode/search
```

Description

Get the list of regions. If latitude or longitude is null, will return all postal codes in the system with paginated response.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**descending** *required*	descending	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**limit** *optional*	limit	integer(int32)	`"20"`
Query	**longitude** *optional*	longitude	number(double)	
Query	**miles** *optional*	miles	number(double)	`"10.0"`
Query	**sortField** *required*	sortField	string	`"id"`
Query	**start** *optional*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< PostalCodeResponse > array

6.49.5. Update Postal Code

```
POST /api/{version}/postalCode/update
```

Description

Update a Postal Code

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**active** *optional*	active	boolean
Query	**city** *optional*	city	string
Query	**code** *optional*	code	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)
Query	**postalCodeId** *required*	postalCodeId	integer(int64)
Query	**stateCode** *optional*	stateCode	string

Responses

HTTP Code	Description	Schema
200	successful operation	PostalCodeResponse

6.50. PreviewPersona

The Preview Personas API allows for setting up and running tests on various Personas on a live server without actually being “live”. For example, you could set up an ad campaign and run it to all “test accounts” to see the results as if you were an actual user. Only your test accounts would see the ad campaign. This is a great tool to use for previewing ad campaigns or other data before fully launching it within an application. The Preview Personas API can perform the following functions: delete, search, get, and update.

Below are some example scenarios as to how the Preview Personas service can be used:

- Retail: Test push notifications targeted at a specific geo-fence to your Test Personas to make sure

they send when entering a geo-fenced area.

- Medical: Test wearable device notification to a certain population such as “Doctors” “Nurses” or “Patients” before setting it live.
- Smart Home: Test IoT device automation using a mobile app interface to a Test Persona before releasing a feature to every device.
- Automotive: Run updated features and diagnostics on Test Persona vehicles before pushing a new feature live to every vehicle in a fleet.
- Robotics: Run various automated functions to test machines to see results before deploying to every machine in a factory setting.

The Preview Personas API is important because it shows you see live data in a test group or population before it is set live to make sure that everything is operating properly within your application. Preview Personas is particularly important because it runs as a test using your live application as data to gain insights.

Below are great example services that could be combined with the Preview Persona service:

- Notifications Service: Send push notifications only to Test users within a specific geo-fence, region, or within a specific class of device.
- Offers: See real-time offer information to check pricing, offer text, and test purchasing before setting it live.
- Offer Status: Ensure that your offer status is valid before setting it live.
- Application Configurations: See updates to your applications targeted at a specific user set, such as “Nurses” or “Doctors” before setting the update live.

6.50.1. Create Persona

```
POST /api/{version}/persona/create
```

Description

Creates a new persona. If the given params are null those attributes will be override by null.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)

Type	Name	Description	Schema
Query	**age** *optional*	age	integer(int32)
Query	**date** *optional*	date	integer(int64)
Query	**gameExperienceLevel** *optional*	gameExperienceLevel	string
Query	**gender** *optional*	gender	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)
Query	**metaData** *optional*	metaData	string
Query	**previewAccounts** *optional*	previewAccounts	string
Query	**title** *required*		string

Responses

HTTP Code	Description	Schema
200	successful operation	PreviewPersonaResponse

6.50.2. Delete Persona

```
POST /api/{version}/persona/delete
```

Description

Mark the persona for deletion.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**personaId** *required*		integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.50.3. Get Persona

```
POST /api/{version}/persona/get
```

Description

Get the persona by the given persona ID. If the persona cannot be found, a invalid response is returned.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**personaId** *required*		integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	PreviewPersonaResponse

6.50.4. Search Personas

```
POST /api/{version}/persona/search
```

Description

Search for persona that the account owns by the given account ID.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	PreviewPersonaResponse

6.50.5. Update Persona

```
POST /api/{version}/persona/update
```

Description

Update the persona by the given personaId. If the given params are null those attributes will be override by null. If active is assigned, all other params will be ignored.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)

Type	Name	Description	Schema
Query	**accountId** *required*	accountId	integer(int64)
Query	**active** *optional*	active	boolean
Query	**age** *optional*	age	integer(int32)
Query	**date** *optional*	date	integer(int64)
Query	**gameExperienceLevel** *optional*	gameExperienceLevel	string
Query	**gender** *optional*	gender	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)
Query	**metaData** *optional*	metaData	string
Query	**personaId** *required*		integer(int64)
Query	**previewAccounts** *optional*	previewAccounts	string
Query	**title** *optional*	title	string

Responses

HTTP Code	Description	Schema
200	successful operation	PreviewPersonaResponse

6.51. Program

The program service allows an organization to set up different overarching products that utilize Route 'N' Roll. These programs can be targeted towards different demographics, locations, and more. This

service allows creation, updates, and getting information of an account's programs.

Below are some example scenarios as to how these services can be used:

- Ride share: A ride share company may have multiple programs based on the client. They could have one program for giving rides to kids that have been signed up by their parents, another program for elderly clients getting to their social programs, and one more program for elite clients that require top of the line luxury vehicles.
- Medical: A hospital might use one program for scheduled ambulance rides, and another program for emergency pick ups. With these two programs, the hospital can provide the best service for both types of ride.
- Education: Schools could have a program that handles day to day school bus routes, and another program that handles field trips, sports games, etc.

Some organizations will only need one program, and some will need multiple. Using the Program service, companies have the flexibility to support many different types of Route 'N' Roll services.

Below are great example services that could be combined with this service:

- Notification Service: Send a different notification to the customer, dispatcher, or driver based on the program they are currently using.
- Offer Service: Offer different rates, eligibility, and more using the offer service. Offer the right products and deals to the right demographic for your program.
- Dashboards: Handle your account's various programs right on the dashboard.

6.51.1. Create Program

```
POST /api/{version}/program
```

Description

Create a new program

Parameters

Type	Name	Schema
Path	**version** *required*	string
Body	**body** *optional*	Program

Responses

HTTP Code	Description	Schema
200	successful operation	Program

6.51.2. Search Programs

```
GET /api/{version}/program
```

Description

Search for programs

Parameters

Type	Name	Schema	Default
Path	**version** *required*	number(double)	
Query	**activeOnly** *required*	boolean	`"true"`
Query	**descending** *required*	boolean	`"false"`
Query	**keyword** *optional*	string	
Query	**limit** *required*	integer(int32)	`"20"`
Query	**sortField** *required*	string	`"id"`
Query	**start** *required*	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< Program > array

6.51.3. Update Program

```
POST /api/{version}/program/{id}
```

Description

Update an existing program

Parameters

Type	Name	Schema
Path	**id** *required*	integer(int64)
Path	**version** *required*	string
Body	**body** *optional*	Program

Responses

HTTP Code	Description	Schema
200	successful operation	Program

6.51.4. Get Program

```
GET /api/{version}/program/{id}
```

Description

Get an existing program

Parameters

Type	Name	Schema
Path	**id** *required*	integer(int64)
Path	**version** *required*	string

Responses

HTTP Code	Description	Schema
200	successful operation	Program

6.51.5. Update Program

```
PUT /api/{version}/program/{id}
```

Description

Update an existing program

Parameters

Type	Name	Schema
Path	**id** *required*	integer(int64)
Path	**version** *required*	string
Body	**body** *optional*	Program

Responses

HTTP Code	Description	Schema
200	successful operation	Program

6.51.6. Delete Program

```
DELETE /api/{version}/program/{id}
```

Description

Delete an existing program

Parameters

Type	Name	Schema
Path	**id** *required*	integer(int64)
Path	**version** *required*	string

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

6.52. PurchaseItem

The Purchase Item API allows for setting up “digital goods” for sale within your application. App users can purchase these items using various digital currencies. Setup your Purchase Item API using points or real digital currencies that can be mapped to via iOS, Google Play, Amazon and Mac in-App purchases. Detailed information regarding Purchases can be tracked, reported and analyzed.

Below are some example scenarios as to how the Purchase Item service can be used:

- Retail: Setup extended warranties for electronic goods or other hard items purchased within your mobile application.
- Medical: Setup digital gift cards that can be purchased for patients for items such as flowers, balloons, and other gift-shop type goods.
- Smart Home: Setup digital goods for services such as snow shoveling, chimney cleaning, yard work, and more when various time and date or weather triggers occur.
- Automotive: Setup digital goods for recurring services items such as oil changes, maintenance checks, detailing, and more.
- Robotics: Setup digital goods that can be purchased and delivered by drone to the end-user.

The Purchase Item API is important because it allows you to monetize your application by giving your app users a convenient method for making purchases.

Below are great example services that could be combined with the Purchase Item service:

- Digital Currency: Use digital currency services for monetizing your application and creating ease of use and convenience for your app users.
- Offers: Create offers and allow for the purchase of digital goods at a discount.
- Push Notifications: Trigger push notifications when a user is within a geo-fence to notify them of certain digital goods for sale.

- Packs: Create Game Packs that can be purchased by users such as “extra lives” or “boosters” so that they continue to play your games.

6.52.1. Create Purchase

```
POST /api/{version}/purchase/create
```

Description

Creates a purchase item for in app purchases

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The version of the api to respond with	number(double)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**allocateTickets** *optional*	Flag to indicate owner should receive tickets for completed missions	boolean	`"false"`
Query	**appKey** *required*	The application key that the purchase can be used in	string	
Query	**assetable** *optional*	Determines whether users can attach their own media/asset to the purchase item	boolean	
Query	**coverAssetId** *optional*	The cover image of the purchase item	integer(int64)	
Query	**description** *optional*	The description of the purchase item	string	
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string	
Query	**giftable** *optional*	Determines whether the purchase item can be gifted to other users	boolean	
Query	**name** *required*	The name of the purchase item	string	
Query	**points** *optional*	The number of points to award for completing a mission	integer(int64)	
Query	**price** *optional*	How much the purchase item will cost in real money	number(float)	

Type	Name	Description	Schema	Default
Query	**promoAssetId** *optional*	An application specific asset that can be used to store/provide additional data	integer(int64)	
Query	**purchaseCode** *optional*	The unique identifier used by purchase providers to identify in-app-purchases	string	
Query	**purchaseLimit** *optional*	How many times a user acquire the same purchase item	integer(int32)	`"0"`
Query	**purchaseType** *required*	The purchase provider - SIRQUL - the Sirqul store to make purchases using tickets - IOS - the iTunes store for iPhone, iPod, iPod Touch - GOOGLE - the Google Play store - AMAZON - the Amazon Android store - MAC - the iTunes store for OSX - WP8 - the Windows Phone 8 store - FREE - used for purchase items that are free (can be used for development/testing purposes)	enum (SIRQUL, IOS, GOOGLE, AMAZON, MAC, WP8, FREE)	
Query	**secretKey** *optional*	A secret key from purchase providers that would be used for validation	string	
Query	**serviceAction** *optional*	Determines whether the purchase item will enable certain features - DAY_PREMIUM - subscribes a user for a day of membership - WEEK_PREMIUM - subscribes a user for a week of membership - MONTH_PREMIUM - subscribes a user for a month of membership - ADD_TICKET - allows a user to add more tickets to their account - ADD_GIFT - allows a user to send/recieve the purchase item as a gift	enum (DAY_PREMIUM, WEEK_PREMIUM, MONTH_PREMIUM, YEAR_PREMIUM, LIFETIME_PREMIUM, ADD_TICKET, ADD_GIFT, ADD_POINTS)	
Query	**ticketType** *optional*	The type of ticket to reward, null means default type	string	
Query	**tickets** *optional*	How much the purchase item is worth in tickets	integer(int32)	

Responses

HTTP Code	Description	Schema
200	successful operation	PurchaseItemFullResponse

6.52.2. Delete Purchase

```
POST /api/{version}/purchase/delete
```

Description

Marks the purchase item as deleted

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The version of the api to respond with	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string
Query	**purchaseItemId** *required*	The purchase item id	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.52.3. Get Purchase

```
POST /api/{version}/purchase/get
```

Description

Get detailed information about a purchase item

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The version of the api to respond with	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string
Query	**purchaseItemId** *required*	The purchase item id	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	PurchaseItemFullResponse

6.52.4. Search Purchases

```
POST /api/{version}/purchase/search
```

Description

Search for purchasable items from the system

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The version of the api to respond with	number(double)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**activeOnly** *optional*	Return only active results	boolean	`"false"`
Query	**appKey** *optional*	The application key to filter results by application	string	

Type	Name	Description	Schema	Default
Query	**descending** *optional*	The order to return the search results	boolean	`"false"`
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string	
Query	**filterByBillable** *optional*	Determines whether to filter results by the user's billable entity	boolean	`"false"`
Query	**keyword** *optional*	The keyword used to search	string	
Query	**limit** *optional*	The number of records to return	integer(int32)	`"20"`
Query	**purchaseType** *optional*	A comma separated list of purchase providers to filter by. Possible values include: - SIRQUL - purchases from the Sirqul store using tickets - IOS - purchases from the iTunes store for iPhone, iPod, iPod Touch - GOOGLE - purchases from the Google Play store - AMAZON - purchases from the Amazon Android store - MAC - purchases from the iTunes store for OSX - WP8 - purchases from the Windows Phone 8 store - FREE - purchases that are free (can be used for development/testing purposes)	string	
Query	**serviceAction** *optional*	A comma separated list of service actions to filter results by. Possible values include: - DAY_PREMIUM - purchases that subscribes a user for a day of membership - WEEK_PREMIUM - purchases that subscribes a user for a week of membership - MONTH_PREMIUM - purchases that subscribes a user for a month of membership - ADD_TICKET - purchases that allow users to add more tickets - ADD_GIFT - purchases that allow users to recieve gifts	string	

Type	Name	Description	Schema	Default
Query	**sortField** *optional*	The column to sort the search on. Possible values include: ID, CREATED, UPDATED, ACTIVE, ACTIVATED, EXPIRES, TITLE, SUBTITLE, DETAILS, OFFER_TYPE, SPECIAL_OFFER_TYPE, OFFER_VISIBILITY, ESTIMATED_VALUE, VOUCHER_PRICE, RETAILER_ID, RETAILER_NAME, RETAILER_LOCATION_ID, RETAILER_LOCATION_NAME, BILLABLE_ENTITY_ID, BILLABLE_ENTITY_NAME, RESPONSIBLE_DISPLAY	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, NAME, DESCRIPTION, TICKETS, PRICE, PURCHASE_TYPE, PURCHASE_CODE, PURCHASE_LIMIT, SERIVCE_ACTION, GIFTABLE, ASSETABLE, APPLICATION_ID, APPLICATION_NAME)	`"NAME"`
Query	**start** *optional*	The record to begin the return set on	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< PurchaseItemResponse > array

6.52.5. Update Purchase

```
POST /api/{version}/purchase/update
```

Description

Updates a purchase item for in app purchases

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The version of the api to respond with	number(double)	

Type	Name	Description	Schema	Default
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**active** *optional*	Sets whether the purchase item is active or inactive (hidden from consumers)	boolean	
Query	**allocateTickets** *optional*	Flag to indicate owner should receive tickets for completed missions	boolean	`"false"`
Query	**assetable** *optional*	Determines whether users can attach their own media/asset to the purchase item	boolean	
Query	**coverAssetId** *optional*	The cover image of the purchase item	integer(int64)	
Query	**description** *optional*	The description of the purchase item	string	
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string	
Query	**giftable** *optional*	Determines whether the purchase item can be gifted to other users	boolean	
Query	**name** *optional*	The name of the purchase item	string	
Query	**points** *optional*	The number of points to award for completing a mission	integer(int64)	
Query	**price** *optional*	How much the purchase item will cost in real money	number(float)	
Query	**promoAssetId** *optional*	An application specific asset that can be used to store/provide additional data	integer(int64)	
Query	**purchaseCode** *optional*	The unique identifier used by purchase providers to identify in-app-purchases	string	
Query	**purchaseItemId** *required*	The purchase item id	integer(int64)	
Query	**purchaseLimit** *optional*	How many times a user acquire the same purchase item	integer(int32)	

Type	Name	Description	Schema	Default
Query	**purchaseType** *optional*	The purchase provider - SIRQUL - the Sirqul store to make purchases using tickets - IOS - the iTunes store for iPhone, iPod, iPod Touch - GOOGLE - the Google Play store - AMAZON - the Amazon Android store - MAC - the iTunes store for OSX - WP8 - the Windows Phone 8 store - FREE - used for purchase items that are free (can be used for development/testing purposes)	enum (SIRQUL, IOS, GOOGLE, AMAZON, MAC, WP8, FREE)	
Query	**secretKey** *optional*	A secret key from purchase providers that would be used for validation	string	
Query	**serviceAction** *optional*	Determines whether the purchase item will enable certain features - DAY_PREMIUM - subscribes a user for a day of membership - WEEK_PREMIUM - subscribes a user for a week of membership - MONTH_PREMIUM - subscribes a user for a month of membership - ADD_TICKET - allows a user to add more tickets to their account - ADD_GIFT - allows a user to send/recieve the purchase item as a gift	enum (DAY_PREMIUM, WEEK_PREMIUM, MONTH_PREMIUM, YEAR_PREMIUM, LIFETIME_PREMIUM, ADD_TICKET, ADD_GIFT, ADD_POINTS)	
Query	**ticketType** *optional*	The type of ticket to reward, null means default type	string	
Query	**tickets** *optional*	How much the purchase item is worth in tickets	integer(int32)	

Responses

HTTP Code	Description	Schema
200	successful operation	PurchaseItemFullResponse

6.53. PurchaseOrder

The Purchase Order service allows users to group multiple purchasable items in a single transaction. This is especially helpful for retailers that want to offer a 'Shopping Cart' type feature in their product offering. This is beneficial because each transaction will cost the merchant money, since all payment processing systems like Authorize.net, Chase Paymentech, etc. all charge on a per transaction basis. If your product allows users to make purchases, you will want to think about incorporating the Purchase

Order service, since it will reduce your costs per transaction, allow users to group their items into a shopping cart for single purchase, and lastly be able to track all of the items that were purchased from a single transaction ID reference.

Below are some example scenarios as to how these services can be used:

- Retail: Customers inside of a retail store who have the shopping app installed should be able to find items they want to purchase, add them to a 'shopping cart', and then making a single transaction
- Healthcare: If a patient needs to purchase multiple medications, and doctors want to group all of the items under a single purchase order, this can be possible with the Purchase Order service
- Stadiums: Fans that are attending a game will likely visit one of the retail shops at the stadium. The retail shop will have purchasable items where the fan can group multiple items to a single purchase
- Buildings & Spaces: If the refrigerator of your home can make purchases at Amazon, use the Purchase Order service to group multiple items in a single transaction

The Purchase Order service is important as it is a way to reduce costs by reducing the number of transactions that have to occur for users to purchase their items. In addition, it is a secure way of assigning items purchased to a single transaction record. This allows you to track invoices in case users have to make a return on one of the items purchased for the entire order. You can then identify which of the items on a purchase order was refunded or subtracted from the total cost of the order.

Below are example services that could be combined with the Purchase Order service:

- Wallets Service: Wallets can be used to group all of the items in a single purchase order
- Billable Entity: The user that will be charged for making the purchase of items for the transaction
- Billing: The payment method (e.g. credit card, digital currency, eCheck) that is used to purchase items of an order

Charges Description

The Charges service allows a business to charge the credit card, bank account, tickets, bucks, and so forth, of a user in our system. This is helpful when you are running a business where users can purchase items, and you need to charge them because of penalties or fees that need to be assessed. For example, if a user makes a late payment on their subscription of a service, you can charge their credit card saved on file x amount per the agreed terms and conditions. Charges can be done both manually and automatically, giving business owners the flexibility on how they want to handle each customer use case.

Below are some example scenarios as to how these services can be used:

- Retail: Customers are charged money on their credit cards because of a late subscription payment
- Healthcare: Hospital can charge patients their credit card if they do not cancel their appointment in a timely manner

- Smart Vehicle: For speeding infractions applied to a driver, charges on their credit card or credit balance can happen

The Charges service is important because it prevents users from being able to violate the terms and conditions of your product. For example, if a user signs up and subscribes for a service offered by your product, but does not pay on-time per the agreed upon terms, this allows you to charge them for the amount owed to them. This is also important so that you are not liable for users that are trying to game a system. There are malicious people that could intentionally try to hurt your business by registering hundreds or thousands of accounts, making many purchases, and then canceling their subscription right before it's time to pay at the end of the month.

Below are example services that could be combined with the Charges service:

- Wallets Service: The wallet can track the items that a user has added
- Notifications Service: Users receive notifications when they have not yet paid their monthly subscription on time
- Billable Entity: The billable entity is the user that will be charged
- Billing: The payment method that the user saved
- Reports: Generate reports for ip addresses that have unusually large amount of account creations

Refunds Description

The Refunds service allows a business to refund money back to the credit card, bank account, tickets, bucks, and so forth, of a user in our system. This is helpful when you are running a business where users can purchase items, and you need to refund them money because they either were not satisfied with their purchased item or need to make a return. For example, if a user purchased multiple items in a single transaction, but decides they want to return one of the items, they can do so and will be refunded money for that specific item. Refunds can be done both manually and automatically, giving business owners the flexibility on how they want to handle each customer use case.

Below are some example scenarios as to how these services can be used:

- Retail: Customers can get a refund for money if they return an item they purchased
- Healthcare: Patients can get refunded money or credits if they were overcharged
- Stadiums: Fans get refunded money if they return an item they purchased during a game
- Commercial Spaces: Property owners can refund money to their tenants if they are overcharged on their monthly lease

The Refunds service is important because it allows users to be refunded money if they are either unhappy with their service received or need to return an item. This service is necessary to ensure that your users continue using your products. Without the ability to refund money back to users, users could write negative reviews that will prevent other users from potentially signing up and/or continuing to use your product. In addition, refunds can be used to track money that goes in and out of your system. This is important so that there is accountability for where money is moving. Lastly,

refunds are important so that users can get a chance to use your service and are guaranteed their money back if they don't feel like using your service any more. This allows users to try your service without feeling locked in.

Below are example services that could be combined with the Refunds service:

- Purchase Order: The item that was purchased together in a Purchase Order can be tracked and refunded specifically
- Notifications Service: Personalized push notifications that are targeted to user accounts that fit in a particular audience
- Billable Entity: The user that will be refunded the money
- Billing: The payment method that the user will be refunded money onto, whether it's back to their credit card or bank account

6.53.3. Create Order

```
POST /api/{version}/order/create
```

Description

Creates a new purchase with some number of items associated with it. The purchase is added to the order that was created

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The version of the api to response with	number(double)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**appKey** *required*	The application requesting the purchase	string	

Type	Name	Description	Schema	Default
Query	**cart** *required*	A JSON list of items to purchase: <code><pre> [{ 'orderItemType': 'OFFER', 'orderItemId': 234, 'orderCustomType':'OfferLocation', 'orderCustomId':123, 'retailerLocationId':1234, 'quantity': 2 , { 'orderItemType': 'OFFER', 'orderItemId': 235, 'quantity': 2 , { 'orderItemType': 'CUSTOM', 'amount': 10.50, 'orderCustomType': 'ServiceFee', , { 'orderItemType': 'CUSTOM', 'amount': 25.10, 'quantity': 2 'orderCustomType': 'Hat', 'orderCustomId': 123,] </pre></code>	string	
Query	**currencyType** *optional*	Determines the method of purchasing offer. <code><pre> VOID: used for when there is no payment involved (ie. for updating the status of a purchase order) CASH: use card on file POINTS: use account balance TICKETS: use tickets REFUND: not allowed for a create, requires a paymentTransactionId. CREDIT: add to the account balance RELOAD: charge a credit card then add to the account balance </pre></code>	enum (VOID, CASH, POINTS, TICKETS, REFUND, CREDIT, RELOAD)	`"CASH"`
Query	**description** *optional*	A description of the purchase	string	
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string	
Query	**externalDate** *optional*	External Date	integer(int64)	
Query	**externalOrderId** *optional*	Store identifier from external system	string	
Query	**externalPaymentId** *optional*	Store identifier from external system	string	
Query	**paymentMethodId** *optional*	Use a specific payment method (CASH), if not provided use default	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	OrderResponse

6.53.4. Delete Order

```
POST /api/{version}/order/delete
```

Description

Removes the transaction from the wallet by setting the deleted date to the current date/time. Requires a valid account and transactionId.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The version of the api to response with	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string
Query	**orderId** *required*	Order Id	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.53.5. Get Order

```
POST /api/{version}/order/get
```

Description

Get an order record

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The version of the api to response with	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string
Query	**externalOrderId** *optional*	The external order id to get details of, either orderId or externalOrderId must be provided	string
Query	**orderId** *optional*	The order id to get details of, either orderId or externalOrderId must be provided	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	OrderResponse

6.53.6. Search Orders

```
POST /api/{version}/order/search
```

Description

Search on active orders by customer

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The version of the api to response with	number(double)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**activeOnly** *optional*	Only return active orders	boolean	`"false"`
Query	**appKey** *required*	The application requesting the purchase	string	

Type	Name	Description	Schema	Default
Query	**descending** *optional*	Determines whether to return the resulting list in descending or ascending order	boolean	`"true"`
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string	
Query	**limit** *optional*	The limit for pagination	integer(int32)	`"20"`
Query	**start** *optional*	The start index for pagination	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< OrderResponse > array

6.53.7. Update Order

```
POST /api/{version}/order/update
```

Description

Updates new purchase with some number of items associated with it. The orderId provided is used to retrieve the record and the payment is added to it.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The version of the api to response with	number(double)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**appKey** *required*	The application requesting the purchase	string	

Type	Name	Description	Schema	Default
Query	**cart** *required*	A JSON list of items to purchase: <code><pre> [{ 'orderItemType': 'OFFER', 'orderItemId': 234, 'orderCustomType':'OfferLocation', 'orderCustomId':123, 'retailerLocationId':1234, 'quantity': 2 , { 'orderItemType': 'OFFER', 'orderItemId': 235, 'quantity': 2 , { 'orderItemType': 'CUSTOM', 'amount': 10.50, 'orderCustomType': 'ServiceFee', , { 'orderItemType': 'CUSTOM', 'amount': 25.10, 'quantity': 2 'orderCustomType': 'Hat', 'orderCustomId': 123,] </pre></code>	string	
Query	**currencyType** *optional*	Determines the method of purchasing offer. <code><pre> VOID: used for when there is no payment involved (ie. for updating the status of a purchase order) CASH: use card on file POINTS: use account balance TICKETS: use tickets REFUND: return value to the original transaction, requires paymentTransactionId to be provided. ignores paymentMethodId. May refund POINTS, CREDIT, CASH, and RELOAD types. CREDIT: add to the account balance RELOAD: charge a credit card then add to the account balance </pre></code>	enum (VOID, CASH, POINTS, TICKETS, REFUND, CREDIT, RELOAD)	`"CASH"`
Query	**description** *optional*	A description of the purchase	string	
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string	
Query	**externalDate** *optional*	External Date	integer(int64)	
Query	**externalPaymentId** *optional*	Store identifier from external system	string	
Query	**orderId** *required*	The order to add the purchase to, leave null for new order.	integer(int64)	
Query	**paymentMethodId** *optional*	Use a specific payment method (CASH), if not provided use default	integer(int64)	
Query	**paymentTransactionId** *optional*	The payment transaction to apply the refund to	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	OrderResponse

6.54. Question

The Question API allows you to setup Questions or Quizzes within your application. These can be setup as packets to create levels within Trivia Game applications, or, as lesson categories for educational applications. Questions can be setup and managed in real time by your administrators through an online dashboard.

Below are some example scenarios as to how the Question service can be used:

- Retail: Setup employee training programs to educate and improve the knowledge base of your workforce.
- Medical: Create various categories for online education to nurses and doctors so they are always up to date with the latest information.
- Smart Home: Create quizzes about how to “reduce, reuse, & recycle” goods within the home to educate people using Smart Devices to be more energy efficient.
- Automotive: Train employees on various aspects of services vehicles or a fleet. Create questions that disseminate important information in a Q&A format for greater retention.
- Colleges/Universities: Setup quizzes and exams across a multitude of subjects to help students learn better, faster, and easier online.

The Question API is important because it allows you get creative with Trivia Game style content and/or Educational Information within your application.

Below are great example services that could be combined with the Question service:

- Score: Use Score so users earn points for questions they answer correctly and or within a certain amount of time.
- Ticket: Create tickets that application users can win and redeem for Digital Goods at various point levels. This is especially useful as an employee incentive tool.
- Push Notifications: Use push notifications to alert users of new Questions and content within your application. User push notifications as a reminder to finish certain levels before a certain deadline in educational settings.
- Tournaments: Create Tournaments to allow users to compete against each other for added incentives and rewards such as “Smartest Employee”, “World Trivia Champion” and more.

6.54.1. Create Question

```
POST /api/{version}/game/question/create
```

Description

Create a question and related answers by the given params.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *required*	active	boolean	`"false"`
Query	**allocateTickets** *required*	allocateTickets	boolean	`"false"`
Query	**answers** *required*	answers	string	
Query	**assetId** *optional*	assetId	integer(int64)	
Query	**metaData** *optional*	metaData	string	
Query	**points** *optional*	points	integer(int64)	
Query	**question** *required*	question	string	
Query	**tags** *optional*	tags	string	
Query	**ticketCount** *required*	ticketCount	integer(int64)	`"0"`
Query	**ticketType** *optional*	ticketType	string	
Query	**videoURL** *optional*	videoURL	string	

Responses

HTTP Code	Description	Schema
200	successful operation	QuestionResponse

6.54.2. Delete Question

```
POST /api/{version}/game/question/delete
```

Description

Delete a question by the given questionId. The accountId given needs to be the owner or executive to delete.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**questionId** *required*	questionId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.54.3. Get Question

```
POST /api/{version}/game/question/get
```

Description

Get a question by the given id.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**questionId** *required*	questionId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	QuestionResponse

6.54.4. Search Questions

```
POST /api/{version}/game/question/search
```

Description

Search for questions by the given params.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activeOnly** *required*	activeOnly	boolean	`"false"`
Query	**descending** *required*	descending	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`

Type	Name	Description	Schema	Default
Query	**sortField** *required*	sortField	string	`"id"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< QuestionResponse > array

6.54.5. Update Question

```
POST /api/{version}/game/question/update
```

Description

Update a question and related answers.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	
Query	**allocateTickets** *optional*	allocateTickets	boolean	
Query	**answers** *optional*	answers	string	
Query	**assetId** *optional*	assetId	integer(int64)	
Query	**metaData** *optional*	metaData	string	

Type	Name	Description	Schema	Default
Query	**points** *optional*	points	integer(int64)	
Query	**question** *optional*	question	string	
Query	**questionId** *required*	questionId	integer(int64)	
Query	**tags** *optional*	tags	string	
Query	**ticketCount** *required*	ticketCount	integer(int64)	`"0"`
Query	**ticketType** *optional*	ticketType	string	
Query	**videoURL** *optional*	videoURL	string	

Responses

HTTP Code	Description	Schema
200	successful operation	QuestionResponse

6.55. Ranking

The Ranking API allows you to setup various “Leaderboards” within your application. Some examples of Ranking can be “highest score” or “most shares” or “most games played”. Ranking allows your users to compete against each other which drives more usage of your application. The Ranking(s) can be displayed across various time ranges such as “daily, weekly, or monthly.”

Below are some example scenarios as to how the Rating service can be used:

- Retail: Setup Ranking so your customers can compete against each other to win various prizes or recognition such as “most store visits” on a monthly business.
- Medical: Create an internal Leaderboard to show leading employees in various categories such as “patients served”.
- Smart Home: Use Ranking to get feedback on power usage peak times during the week, month or year. Use the data to adjust settings on devices to maximize energy efficiencies.
- Automotive: Use Ranking to track miles driven by drivers in a fleet to show most miles driven per week, per month, etc.
- Robotics: Use Ranking to keep track of uptime or number of products produced by various

assembly lines.

The Ranking API is important because it allows you to see and share scoring standings across time ranges.

Below are great example services that could be combined with the Ranking service:

- Missions: Use Missions in combination with Ranking to create a Leaderboard for most Missions accomplished by application users.
- Invites: Create a Leaderboard using Ranking to display a listing of users who successfully invite the most other users to download and use your application.
- Favorites: Use Ranking with Favorites to display the rank of the favorite items in your application.
- Retailers: Use Ranking in combination with Retailers to show a list of the most visited Retail for your business locations.
- Purchase Item: Use Rankings in combination with Purchase Item to create a list of most purchased items within your application.

6.55.1. Override User Rank

```
POST /api/{version}/ranking/override
```

Description

Allows an admin of an application to override a user's scores for a leaderboard.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *required*	appKey	string
Query	**dailyCount** *optional*	dailyCount	integer(int64)
Query	**dailyScore** *optional*	dailyScore	integer(int64)
Query	**dailyTime** *optional*	dailyTime	integer(int64)

Type	Name	Description	Schema
Query	**endDate** *optional*	endDate	integer(int64)
Query	**lowestScore** *optional*	lowestScore	integer(int64)
Query	**monthlyCount** *optional*	monthlyCount	integer(int64)
Query	**monthlyScore** *optional*	monthlyScore	integer(int64)
Query	**monthlyTime** *optional*	monthlyTime	integer(int64)
Query	**ownerAccountId** *required*	ownerAccountId	integer(int64)
Query	**rankType** *required*	rankType	string
Query	**startDate** *optional*	startDate	integer(int64)
Query	**streakBestCount** *optional*	streakBestCount	integer(int64)
Query	**streakCount** *optional*	streakCount	integer(int64)
Query	**topScore** *optional*	topScore	integer(int64)
Query	**totalCount** *optional*	totalCount	integer(int64)
Query	**totalScore** *optional*	totalScore	integer(int64)
Query	**totalTime** *optional*	totalTime	integer(int64)
Query	**weeklyCount** *optional*	weeklyCount	integer(int64)
Query	**weeklyScore** *optional*	weeklyScore	integer(int64)
Query	**weeklyTime** *optional*	weeklyTime	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.55.2. POST /api/{version}/ranking/personal/ranks

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**descending** *required*	descending	boolean	`"true"`
Query	**deviceId** *optional*	deviceId	string	
Query	**keyword** *optional*	keyword	string	
Query	**leaderboardMode** *required*	leaderboardMode	string	`"GLOBAL"`
Query	**limit** *required*	limit	integer(int32)	`"100"`
Query	**rankType** *optional*	rankType	string	
Query	**returnUserRank** *required*	returnUserRank	boolean	`"false"`
Query	**sortField** *required*	sortField	string	`"TOTAL"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.55.3. Search Rankings

```
POST /api/{version}/ranking/search
```

Description

Get leader board rankings. This is an all in one endpoint that can return multiple ranking types and also the current user rank.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**_i** *optional*	_i	integer(int32)	
Query	**_l** *optional*	_l	integer(int32)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**descending** *required*	descending	boolean	`"true"`
Query	**deviceId** *optional*	deviceId	string	
Query	**gameType** *optional*	gameType	string	
Query	**keyword** *optional*	keyword	string	
Query	**leaderboardMode** *required*	leaderboardMode	string	`"GLOBAL"`

Type	Name	Description	Schema	Default
Query	**limit** *required*	limit	integer(int32)	`"100"`
Query	**q** *optional*	q	string	
Query	**rankType** *required*	rankType	string	`"POINTS,DOWNLOADS,INVITATIONS"`
Query	**returnUserRank** *required*	returnUserRank	boolean	`"true"`
Query	**sortField** *required*	sortField	string	`"TOTAL"`
Query	**start** *required*	start	integer(int32)	`"0"`
Query	**withinAccountIds** *optional*	withinAccountIds	string	

Responses

HTTP Code	Description	Schema
200	successful operation	RankFullResponse

6.55.4. Update Ranking

```
POST /api/{version}/ranking/update
```

Description

Update the rank value

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	

Type	Name	Description	Schema	Default
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *required*	appKey	string	
Query	**createLeaderboard** *required*	createLeaderboard	boolean	`"false"`
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**increment** *required*	increment	integer(int64)	`"1"`
Query	**rankType** *required*	rankType	string	
Query	**startDate** *optional*	startDate	integer(int64)	
Query	**tag** *optional*	tag	string	
Query	**timeIncrement** *optional*	timeIncrement	integer(int64)	
Query	**updateGlobal** *required*	updateGlobal	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.56. Rating

The Rating API allows you to setup Ratings on a custom defined scale within your application. Some examples of Ratings can be “five stars” or a scale of “1 to 5” or “not likely, somewhat likely, very likely” or even “bad” “neutral” and “great”. Rating allows your users to give you feedback on assets, content or even other users within your application. This is valuable information that you can use to improve your application and user experience resulting in increased user interaction and overall application usage.

Below are some example scenarios as to how the Rating service can be used:

- Retail: Setup Ratings for various products and services in your application, similar to Amazon.com or Yelp.
- Medical: Create a rating system to collect feedback from Patients regarding services received.
- Smart Home: Use Rating to get
- Automotive: Use a real-time rating service to get feedback on traffic such as “light, medium, and heavy.”
- Robotics: Use a rating system in tandem with temperature gauges to keep track of temperature ranges, “cold, warm, hot” or to monitor other performance levels in a category “empty, half-full, full”.

The Rating API is important because it allows you get user feedback about various content, assets, and users within your application.

Below are great example services that could be combined with the Ratings service:

- Albums: Use Albums so users give Ratings for collections of certain content such as “autos” or “landscapes.”
- Creatives: Creatives (Advertisements) can be presented to users in specific geographies or user groups – use Ratings to get feedback on certain Ads seen by your users.
- Retailers: Use Ratings in combination with Retailers locations so users can give you valuable feedback about various business locations you operate.
- Purchase Item: Use Ratings in combination with Purchase Items so that customers can give ratings about various products in your digital catalog.
- Events: Use Ratings in combination with Events (concerts, giveaways, VIP Only Sales) to gather user feedback on what types of events are most liked.

6.56.1. Search Location Rating Indexes

```
POST /api/{version}/location/rating/index/search
```

Description

Search for retailer locations by averages near you.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**categoryIds** *optional*	categoryIds	string	

Type	Name	Description	Schema	Default
Query	**descending** *required*	descending	boolean	`"true"`
Query	**distanceUnit** *required*	distanceUnit	enum (MILES, KILOMETERS)	`"MILES"`
Query	**keyword** *optional*	keyword	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**longitude** *optional*	longitude	number(double)	
Query	**returnAssets** *required*	returnAssets	boolean	`"true"`
Query	**returnCategories** *required*	returnCategories	boolean	`"true"`
Query	**returnFilters** *required*	returnFilters	boolean	`"true"`
Query	**returnOffers** *required*	returnOffers	boolean	`"true"`
Query	**returnOverallRating** *required*	returnOverallRating	boolean	`"false"`
Query	**returnRetailer** *required*	returnRetailer	boolean	`"true"`
Query	**searchRange** *required*	searchRange	number(double)	`"5.0"`

Type	Name	Description	Schema	Default
Query	**sortField** *required*	sortField	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, RATABLE_TYPE, RATABLE_ID, RATABLE_DISPLAY, CATEGORY_ID, CATEGORY_NAME, CATEGORY_SHORT_NAME, CATEGORY_DISPLAY, COUNT, SUMMATION, AVERAGE, VALUE)	`"VALUE"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< RatingIndexResponse > array

6.56.2. Create Rating

```
POST /api/{version}/rating/create
```

Description

This is used to leave rating on a ratable object (i.e. retailer locations). Each user can only rate on a ratable object once per category. If a user rates on the same object and category, the previous rating will be overwritten. Leaving a rating on a ratable object will be visible to everyone who has access to view the object.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**categoryId** *optional*	categoryId	integer(int64)
Query	**description** *optional*	description	string
Query	**deviceId** *optional*	deviceId	string
Query	**display** *optional*	display	string
Query	**latitude** *optional*	latitude	number(double)
Query	**locationDescri ption** *optional*	locationDescription	string
Query	**longitude** *optional*	longitude	number(double)
Query	**ratableId** *required*	ratableId	integer(int64)
Query	**ratableType** *required*	ratableType	string
Query	**ratingValue** *required*	ratingValue	integer(int32)

Responses

HTTP Code	Description	Schema
200	successful operation	RatingResponse

6.56.3. Delete Rating

```
POST /api/{version}/rating/delete
```

Description

Sets a rating as deleted.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**ratingId** *required*	ratingId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.56.4. Search Rating Indexes

```
POST /api/{version}/rating/index/search
```

Description

Search for ratable items by averages.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**categoryIds** *optional*	categoryIds	string	
Query	**descending** *required*	descending	boolean	`"true"`

Type	Name	Description	Schema	Default
Query	**keyword** *optional*	keyword	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**longitude** *optional*	longitude	number(double)	
Query	**ratableIds** *optional*	ratableIds	string	

Type	Name	Description	Schema	Default
Query	**ratableType** *required*	ratableType	enum (LOCATABLE, RESERVABLE, PERMISSIONABLE, NOTABLE, ASSETABLE, LIKABLE, FLAGABLE, FAVORITABLE, RATABLE, ALBUM, COLLECTION, APPLICATION, APPLICATION_SETTING, APPLICATION_CERT, APPLICATION_PLACEMENT, ACCOUNT, ACCOUNT_SETTING, GAME_LEVEL, PACK, MISSION, TOURNAMENT, ASSET, ALBUM_CONTEST, THEME_DESCRIPTOR, OFFER, OFFER_LOCATION, EVENT, RETAILER, RETAILER_LOCATION, NOTE, CREATIVE, FAVORITE, LIKE, RATING, ANALYTIC, THIRD_PARTY_CREDENTIAL, THIRD_PARTY_NETWORK, REGION)	
Query	**returnOverallRating** *required*	returnOverallRating	boolean	`"false"`
Query	**returnRatable** *required*	returnRatable	boolean	`"false"`

Type	Name	Description	Schema	Default
Query	**secondaryType** *optional*	secondaryType	string	
Query	**sortField** *required*	sortField	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, RATABLE_TYPE, RATABLE_ID, RATABLE_DISPLAY, CATEGORY_ID, CATEGORY_NAME, CATEGORY_SHORT_NAME, CATEGORY_DISPLAY, COUNT, SUMMATION, AVERAGE, VALUE)	`"VALUE"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< RatingIndexResponse > array

6.56.5. Search Ratings

```
POST /api/{version}/rating/search
```

Description

Search for ratings on a ratable object.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**categoryIds** *optional*	categoryIds	string	
Query	**descending** *required*	descending	boolean	`"true"`
Query	**deviceId** *optional*	deviceId	string	
Query	**filterAccountId** *optional*	filterAccountId	integer(int64)	
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**ratableId** *optional*	ratableId	integer(int64)	
Query	**ratableType** *optional*	ratableType	string	
Query	**sortField** *required*	sortField	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, OWNER_DISPLAY, RATABLE_TYPE, RATABLE_ID, RATABLE_DISPLAY, CATEGORY_ID, CATEGORY_NAME, CATEGORY_SHORT_NAME, CATEGORY_DISPLAY, VALUE)	`"ID"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< RatingResponse > array

6.56.6. Update Rating

```
POST /api/{version}/rating/update
```

Description

Update an existing rating. Only the creator of the rating have permission to update.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**categoryId** *optional*	categoryId	integer(int64)
Query	**description** *optional*	description	string
Query	**deviceId** *optional*	deviceId	string
Query	**display** *optional*	display	string
Query	**latitude** *optional*	latitude	number(double)
Query	**locationDescription** *optional*	locationDescription	string
Query	**longitude** *optional*	longitude	number(double)
Query	**ratingId** *required*	ratingId	integer(int64)

Type	Name	Description	Schema
Query	**ratingValue** *optional*	ratingValue	integer(int32)

Responses

HTTP Code	Description	Schema
200	successful operation	RatingResponse

6.57. Region

The Region API allows you to group Postal Codes to create a geographic area for any business purposes such as service areas, or sales areas. Additionally, Region can be used along with Push Notifications or for setting up areas to serve specific advertisements within your application. Regions are hierarchical and custom taxonomies can be created, for example by most important to least important.

Below are some example scenarios as to how the Rating service can be used:

- Retail: Setup Region so you can track store visits, sales and more through your application by users located within various geographic regions.
- Medical: Create a Region in order to see number of users of your application within various geographic area around your multiple locations within that region. Or, use analytics to track billable services by Region.
- Smart Home: Use Region to track usage of certain devices within groups of Postal Codes to gain insights into marketing more devices to new potential users within popular Regions.
- Automotive: Use Region to track vehicles driven within Regions and setup notifications if any vehicles travel outside of assigned regions.
- Robotics: Use Region to keep track of numbers of trips made by drones flying within a group of postal codes in order to determine usage and adjustments to numbers of drones used in each Region.

The Region API is important because it allows you to more easily manage a larger geographic area by combining multiple Postal Codes. For example, a Region can be large groups of Postal Codes such as “Pacific”, “Midwest” and “Atlantic”.

Below are great example services that could be combined with the Region service:

- Ranking: Use Ranking in combination with Region to see a listed ranking of application usage, sales and more by region.
- Push Notifications: Create and deploy Push Notifications to certain regions to promote sales and offers.

- Audience: Use Region with Audience to create promotions, offers, missions and more to particular user groups within a particular Region
- Retailers: Use Region in combination with Retailers to show a list of the most visited Retail locations for your business locations inside of a Region.
- Purchase Item: Use Regions in combination with Purchase Item to create specific items for sale within your application by geographical Region.

6.57.1. Create Region

```
POST /api/{version}/region/create
```

Description

Create a region.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	`"true"`
Query	**categoryIds** *optional*	categoryIds	string	
Query	**childrenIds** *optional*	childrenIds	string	
Query	**end** *optional*	end	integer(int64)	
Query	**filterIds** *optional*	filterIds	string	
Query	**fullName** *optional*	fullName	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**locations** *optional*	locations	string	

Type	Name	Description	Schema	Default
Query	**longitude** *optional*	longitude	number(double)	
Query	**metaData** *optional*	metaData	string	
Query	**parentIds** *optional*	parentIds	string	
Query	**polygon** *optional*	polygon	string	
Query	**postalCodeIds** *optional*	postalCodeIds	string	
Query	**regionClass** *required*	regionClass	string	
Query	**retailerLocationId** *optional*	retailerLocationId	integer(int64)	
Query	**root** *optional*	root	boolean	`"false"`
Query	**shortName** *required*	shortName	string	
Query	**start** *optional*	start	integer(int64)	
Query	**versionCode** *optional*	versionCode	integer(int32)	
Query	**visibility** *optional*	visibility	enum (PUBLIC, PRIVATE, FRIENDS)	

Responses

HTTP Code	Description	Schema
200	successful operation	RegionResponse

6.57.2. Delete Region

```
POST /api/{version}/region/delete
```

Description

Delete a region.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**regionId** *required*	regionId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	RegionResponse

6.57.3. Get Region

```
POST /api/{version}/region/get
```

Description

Get a region.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**regionId** *required*	regionId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	RegionResponse

6.57.4. Search Regions

```
POST /api/{version}/region/search
```

Description

Get the list of regions.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activeOnly** *required*	activeOnly	boolean	`"true"`
Query	**categoryIds** *optional*	categoryIds	string	
Query	**descending** *required*	descending	boolean	`"true"`
Query	**filterIds** *optional*	filterIds	string	
Query	**includeChildren** *required*	includeChildren	boolean	`"false"`
Query	**includeParent** *required*	includeParent	boolean	`"false"`
Query	**includePostalCodes** *required*	includePostalCodes	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	

Type	Name	Description	Schema	Default
Query	**lastUpdatedSince** *optional*	lastUpdatedSince	integer(int64)	
Query	**latitude** *optional*	latitude	number(double)	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**longitude** *optional*	longitude	number(double)	
Query	**query** *optional*	query	string	
Query	**range** *required*	range	number(double)	`"50.0"`
Query	**regionClass** *optional*	regionClass	enum (NATIONAL, STATE, COUNTY, METRO_AREA, CITY, NEIGHBORHOOD, TERRITORY, CUSTOM, ZONE)	
Query	**searchMode** *optional*	searchMode	enum (RDS, LUCENE, CLOUDINDEX)	
Query	**showDeleted** *required*	showDeleted	boolean	`"false"`
Query	**sortField** *optional*	sortField	enum (ID, UPDATED, NAME, DISTANCE)	
Query	**start** *required*	start	integer(int32)	`"0"`
Query	**versionCode** *optional*	versionCode	integer(int32)	
Query	**visibility** *optional*	visibility	enum (PUBLIC, PRIVATE, FRIENDS)	

Responses

HTTP Code	Description	Schema
200	successful operation	< RegionResponse > array

6.57.5. Update Region

```
POST /api/{version}/region/update
```

Description

Update a region.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	
Query	**categoryIds** *optional*	categoryIds	string	
Query	**childrenIds** *optional*	childrenIds	string	
Query	**clearLists** *optional*	clearLists	boolean	`"false"`
Query	**end** *optional*	end	integer(int64)	
Query	**filterIds** *optional*	filterIds	string	
Query	**fullName** *optional*	fullName	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**locations** *optional*	locations	string	

Type	Name	Description	Schema	Default
Query	**longitude** *optional*	longitude	number(double)	
Query	**metaData** *optional*	metaData	string	
Query	**parentIds** *optional*	parentIds	string	
Query	**polygon** *optional*	polygon	string	
Query	**postalCodeIds** *optional*	postalCodeIds	string	
Query	**regionClass** *optional*	regionClass	string	
Query	**regionId** *required*	regionId	integer(int64)	
Query	**retailerLocationId** *optional*	retailerLocationId	integer(int64)	
Query	**root** *optional*	root	boolean	
Query	**shortName** *optional*	shortName	string	
Query	**start** *optional*	start	integer(int64)	
Query	**versionCode** *optional*	versionCode	integer(int32)	
Query	**visibility** *optional*	visibility	enum (PUBLIC, PRIVATE, FRIENDS)	

Responses

HTTP Code	Description	Schema
200	successful operation	RegionResponse

6.58. Reporting

The Reports API helps you understand your user behavior as well as trends within your application. Reports allow for the creation of custom ad hoc reports or you can use a set of predefined reports to see important data about your application quickly. The Reports API includes information on data such as time spent in your application as well as time spent per page. You can also see abandonment rates on various pages to gain a better understanding of how users interact with your application so that you can make improvements and increase usage.

Below are some example scenarios as to how the Rating service can be used:

- Retail: Setup Reports so you can view the time of day in app while in-store or, nearby your stores to help drive sales and revenue in combination with Push Notifications.
- Medical: Use Reports to see most visited pages and time spent on those pages within your application in order to make adjustments to your content by highlighting most viewed content or Medical services.
- Smart Home: Use Reporting to track the time spent using certain devices, the most used services or actions on those devices, and more.
- Automotive: Use Reports to track vehicle usage and time spent in transit. This is important for things such as scheduled vehicle maintenance or for tracking hours spent driving by your fleets drivers.
- Robotics: Use Reports to keep track of numbers of hours spent in production by various machines in order to optimize assembly lines and more.

The Reports API is important because it allows you know detailed information about the overall usage of your application and get details about various aspects for making improvements for your applications users.

Below are great example services that could be combined with the Reporting service:

- Region: Use Region in combination with Reports to get detailed information on the usage of your application by Region.
- Audience: Use Reports with Audience to quickly learn and make informed decisions about the types of people using your application in order to target and market to future potential users.
- Push Notifications: Use Reports to determine the best time of day to send Push Notifications to your users.
- Packs: Use Packs with Reporting in order to see which game levels are most popular with your audience, or, to see where your users fall-off in terms of playing your game.

6.58.1. Create Offline Report

```
POST /api/{version}/report/batch/create
```

Description

Create an entry for the batch for offline report

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**description** *optional*	description	string	
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**endpoint** *optional*	endpoint	string	
Query	**name** *optional*	name	string	
Query	**pageUrl** *optional*	pageUrl	string	
Query	**parameters** *optional*	parameters	string	
Query	**previewLimit** *required*	previewLimit	integer(int32)	`"0"`
Query	**startDate** *optional*	startDate	integer(int64)	
Query	**status** *required*	status	enum (NEW, ERROR, COMPLETE, PARSE_ERROR, PROCESSING, DUPLICATE, SAVEONLY)	`"NEW"`

Responses

HTTP Code	Description	Schema
200	successful operation	ReportBatchResponse

6.58.2. Delete Offline Report

```
POST /api/{version}/report/batch/delete
```

Description

Deletes a batch report.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**batchId** *required*		integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.58.3. Get Offline Report

```
POST /api/{version}/report/batch/get
```

Description

Checks status of batch report.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**allResults** *required*	allResults	boolean	`"false"`
Query	**batchId** *required*		integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	ReportBatchResponse

6.58.4. Search Offline Reports

```
POST /api/{version}/report/batch/search
```

Description

Retrieves batches for a user..

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *optional*	appKey	string	
Query	**endDate** *optional*	endDate	integer(int64)	

Type	Name	Description	Schema	Default
Query	**globalAppSearch** *optional*	globalAppSearch	boolean	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**names** *optional*	names	string	
Query	**start** *required*	start	integer(int32)	`"0"`
Query	**startDate** *optional*	startDate	integer(int64)	
Query	**status** *optional*	status	enum (NEW, ERROR, COMPLETE, PARSE_ERROR, PROCESSING, DUPLICATE, SAVEONLY)	

Responses

HTTP Code	Description	Schema
200	successful operation	< ReportBatchResponse > array

6.58.5. Run Report

```
POST /api/{version}/report/run
```

Description

This endpoint allows you to run a set of predefined reports that can be used to understand your users' behavior as well as trends within your application.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**desc** *required*	desc	boolean	`"false"`
Query	**limit** *optional*	limit	integer(int64)	
Query	**order** *optional*	order	string	
Query	**parameters** *optional*	parameters	string	
Query	**query** *optional*	query	string	
Query	**responseFormat** *optional*	responseFormat	enum (HTML, XML, JSON, CSV)	
Query	**start** *optional*	start	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	ReportResponse

6.59. Reservation

The Reservations service allows your users to make reservations for an object or a space, where the user has a specified period of time where they can occupy it before it is available again to be reserved by another user. For example, allow your residents of a highrise building to reserve an amenity like the communal kitchen. The resident can reserve the space for 4 hours on a specific day, and when the reservation is completed, the resident no longer has exclusive access to the space and other residents can now reserve the space.

Below are some example scenarios as to how these services can be used:

- Restaurants: Allow customers to reserve a table in advance
- Commercial Building: Use the Reservations service to allow tenants to reserve a space temporarily

inside of your commercial building

- Hospital: Allow patients to make an appointment by reserving a time/day to see the doctor
- High Rise Residential Building: Allow residents to reserve an amenity
- Routing & Logistics: Allow customers to reserve a moving truck for a specified period of time

The Reservations service is important because it gives your business the opportunity to rent your products or services to customers for a specified period of time and through agreed upon payment. For example, if your business is about renting vehicles for moving services, the reservations service will be the core service used. Allow customers to reserve the vehicle for a specified period of time, and charge them a flat fee, in addition, per mile and fuel costs. In addition, the customer may also need to rent equipment such as dollies or tarps, which can also be reserved for a specified period of time and through agreed upon payment.

Below are example services that could be combined with the Reservations service:

- Billable Entity Service: The billable entity will be the user that is charged the amount of money based upon their reservation, for example if they're reserving a rental truck for $x amount of money
- Billing Service: This is the payment method that the user will be charged on when they're confirming their reservation of an object or space
- Events Service: User can make a reservation for an event that they're attending. This will notify the event planner and identify the user that is attending
- Offers Service: An offer could be a product, where the user is reserving that product for a specified period of time

6.59.1. Search Availability

```
POST /api/{version}/reservable/availability/search
```

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	Version	number(double)	
Query	**accountId** *required*	Account Id	integer(int64)	
Query	**deviceId** *optional*	Device Id	string	
Query	**endDate** *optional*	End Date	integer(int64)	

Type	Name	Description	Schema	Default
Query	**limit** *optional*	Limit	integer(int32)	`"100"`
Query	**reservableId** *required*	Reservable Id	integer(int64)	
Query	**reservableType** *required*	Reservable Type	enum (LOCATABLE, RESERVABLE, PERMISSIONABLE, NOTABLE, ASSETABLE, LIKABLE, FLAGABLE, FAVORITABLE, RATABLE, ALBUM, COLLECTION, APPLICATION, APPLICATION_SETTING, APPLICATION_CERT, APPLICATION_PLACEMENT, ACCOUNT, ACCOUNT_SETTING, GAME_LEVEL, PACK, MISSION, TOURNAMENT, ASSET, ALBUM_CONTEST, THEME_DESCRIPTOR, OFFER, OFFER_LOCATION, EVENT, RETAILER, RETAILER_LOCATION, NOTE, CREATIVE, FAVORITE, LIKE, RATING, ANALYTIC, THIRD_PARTY_CREDENTIAL, THIRD_PARTY_NETWORK, REGION)	
Query	**start** *optional*	Start	integer(int32)	`"0"`

Type	Name	Description	Schema	Default
Query	**startDate** *optional*	Start Date	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	< AvailabilityResponse > array

6.59.2. Update Availability

```
POST /api/{version}/reservable/availability/update
```

Parameters

Type	Name	Description	Schema
Path	**version** *required*	Version	number(double)
Query	**accountId** *required*	Account Id	integer(int64)
Query	**availability** *optional*	Availability	string
Query	**availabilitySummary** *optional*	Availability Summary	string
Query	**deviceId** *optional*	Device Id	string
Query	**reservableId** *required*	Reservable Id	integer(int64)

Type	Name	Description	Schema
Query	**reservableType** *required*	Reservable Type	enum (LOCATABLE, RESERVABLE, PERMISSIONABLE, NOTABLE, ASSETABLE, LIKABLE, FLAGABLE, FAVORITABLE, RATABLE, ALBUM, COLLECTION, APPLICATION, APPLICATION_SETTING, APPLICATION_CERT, APPLICATION_PLACEMENT, ACCOUNT, ACCOUNT_SETTING, GAME_LEVEL, PACK, MISSION, TOURNAMENT, ASSET, ALBUM_CONTEST, THEME_DESCRIPTOR, OFFER, OFFER_LOCATION, EVENT, RETAILER, RETAILER_LOCATION, NOTE, CREATIVE, FAVORITE, LIKE, RATING, ANALYTIC, THIRD_PARTY_CREDENTIAL, THIRD_PARTY_NETWORK, REGION)

Responses

HTTP Code	Description	Schema
200	successful operation	< AvailabilityResponse > array

6.59.3. Search Schedule

```
POST /api/{version}/reservable/schedule/search
```

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	Version	number(double)	
Query	**accountId** *required*	Account Id	integer(int64)	
Query	**deviceId** *optional*	Device Id	string	
Query	**endDate** *required*	End Date	integer(int64)	
Query	**reservableId** *required*	Reservable Id	integer(int64)	

Type	Name	Description	Schema	Default
Query	**reservableType** *required*	Reservable Type	enum (LOCATABLE, RESERVABLE, PERMISSIONABLE, NOTABLE, ASSETABLE, LIKABLE, FLAGABLE, FAVORITABLE, RATABLE, ALBUM, COLLECTION, APPLICATION, APPLICATION_SETTING, APPLICATION_CERT, APPLICATION_PLACEMENT, ACCOUNT, ACCOUNT_SETTING, GAME_LEVEL, PACK, MISSION, TOURNAMENT, ASSET, ALBUM_CONTEST, THEME_DESCRIPTOR, OFFER, OFFER_LOCATION, EVENT, RETAILER, RETAILER_LOCATION, NOTE, CREATIVE, FAVORITE, LIKE, RATING, ANALYTIC, THIRD_PARTY_CREDENTIAL, THIRD_PARTY_NETWORK, REGION)	
Query	**startDate** *required*	Start Date	integer(int64)	
Query	**timeBucketMins** *optional*	Time Bucket Mins	integer(int32)	`"30"`

Responses

HTTP Code	Description	Schema
200	successful operation	< TimeSlotResponse > array

6.59.4. Create Reservation

```
POST /api/{version}/reservation/create
```

Description

Creates a reservation on an offer object

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The api version	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**appKey** *optional*	The application requesting the reservation	string
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string
Query	**endDate** *required*	The end date	integer(int64)
Query	**metaData** *optional*	External custom client defined data	string
Query	**offerId** *optional*	The id of the offer being added (offerId or offeLocationId required)	integer(int64)
Query	**offerLocationId** *optional*	The id of the offer location being added (offerId or offeLocationId required)	integer(int64)
Query	**startDate** *required*	The start date	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	ReservationResponse

6.59.5. Delete Reservation

```
POST /api/{version}/reservation/delete
```

Description

Deleted a reservation on a reservation object

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The api version	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string
Query	**reservationId** *required*	The reservation id	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.59.6. Search Reservations

```
POST /api/{version}/reservation/search
```

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	Version	number(double)	
Query	**accountId** *required*	Account Id	integer(int64)	
Query	**appKey** *optional*	Appilcation Key	string	
Query	**deviceId** *optional*	Device Id	string	
Query	**endDate** *optional*	End Date	integer(int64)	
Query	**filterAccountId** *optional*	Filter Account Id	integer(int64)	
Query	**keyword** *optional*	Keyword	string	
Query	**limit** *optional*	Limit	integer(int32)	`"100"`
Query	**reservableId** *optional*	Reservable Id	integer(int64)	

Type	Name	Description	Schema	Default
Query	**reservableType** *optional*	Reservable Type	enum (LOCATABLE, RESERVABLE, PERMISSIONABLE, NOTABLE, ASSETABLE, LIKABLE, FLAGABLE, FAVORITABLE, RATABLE, ALBUM, COLLECTION, APPLICATION, APPLICATION_SETTING, APPLICATION_CERT, APPLICATION_PLACEMENT, ACCOUNT, ACCOUNT_SETTING, GAME_LEVEL, PACK, MISSION, TOURNAMENT, ASSET, ALBUM_CONTEST, THEME_DESCRIPTOR, OFFER, OFFER_LOCATION, EVENT, RETAILER, RETAILER_LOCATION, NOTE, CREATIVE, FAVORITE, LIKE, RATING, ANALYTIC, THIRD_PARTY_CREDENTIAL, THIRD_PARTY_NETWORK, REGION)	
Query	**start** *optional*	Start	integer(int32)	`"0"`
Query	**startDate** *optional*	Start Date	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	< ReservationResponse > array

6.60. Retailer

The Retailers API allows for the defining of business and locations of those businesses within your application. This can be used in two ways. First, it can be used for the input of physical retail or service locations into the application owned by the company. Second, an application owner can allow other businesses to input their Retail locations into the database. For example, an application offering city-wide coupons for goods and services purchased at physical location can allow any business in a specific city or region to input their information for users to discover and buy.

Below are some example scenarios as to how the Rating service can be used:

- Retail: Setup Retailers for every store location owned by your company. These can be displayed on a map and include information such as store hours and phone number. Include Coupons to help drive customers to your Retail locations.
- Medical: Use Retailers to display all the Hospital or Medical Center locations owned by your company then list services plus include the ability to schedule appointments.
- Smart Home: Use Retailers to allow other businesses to list their goods and services in your application for the purchase of goods, maintenance or service items needed in the home that are supplied by third parties.
- Automotive: Use Retailers to show fueling stations, service centers, parking lots and more within your application .
- Robotics: Use Retailers to input all the potential pickup locations for drones to pickup Purchased Item packages for delivery to customers.

The Retailers API is important because it allows you specifically input the information and data for physical locations, or for third parties to input their locations into your application.

Below are great example services that could be combined with the Reporting service:

- Location: Use Location in combination with Retailers to allow users to search and find various businesses close to their current position.
- Audience: Use Retailers with Audience to recommend certain businesses and services to a particular Audience.
- Push Notifications: Allow Retailers to purchase Push Notifications that can be sent to your applications users. This lets you monetize your user base in a unique way.

- Missions: Create Missions for your application users to visit specific Retailers and get various rewards for Missions completed to increase customer value and application usage.

6.60.1. Create Retailer

```
POST /api/{version}/retailer/create
```

Description

Create a retailer record. A billable entity must be created first before a retailer record can be made.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**businessPhone** *optional*	businessPhone	string	
Query	**businessPhoneExt** *optional*	businessPhoneExt	string	
Query	**categoryIds** *optional*	categoryIds	string	
Query	**city** *optional*	city	string	
Query	**country** *optional*	country	string	
Query	**createDefaultLocation** *optional*	createDefaultLocation	boolean	
Query	**deviceId** *optional*	deviceId	string	
Query	**email** *optional*	email	string	
Query	**facebookUrl** *optional*	facebookUrl	string	

Type	Name	Description	Schema	Default
Query	**filterIds** *optional*	filterIds	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**logo** *optional*	logo	ref	
Query	**logoAssetId** *optional*	logoAssetId	integer(int64)	
Query	**longitude** *optional*	longitude	number(double)	
Query	**metaData** *optional*	metaData	string	
Query	**name** *required*	name	string	
Query	**picture1** *optional*	picture1	ref	
Query	**picture1AssetId** *optional*	picture1AssetId	integer(int64)	
Query	**picture2** *optional*	picture2	ref	
Query	**picture2AssetId** *optional*	picture2AssetId	integer(int64)	
Query	**postalCode** *optional*	postalCode	string	
Query	**responseFormat** *required*	responseFormat	enum (HTML, XML, JSON, CSV)	`"JSON"`
Query	**state** *optional*	state	string	
Query	**streetAddress** *optional*	streetAddress	string	
Query	**streetAddress2** *optional*	streetAddress2	string	

Type	Name	Description	Schema	Default
Query	**twitterUrl** *optional*	twitterUrl	string	
Query	**visibility** *optional*	visibility	enum (PUBLIC, PRIVATE, FRIENDS)	
Query	**website** *optional*	website	string	

Responses

HTTP Code	Description	Schema
200	successful operation	RetailerFullResponse

6.60.2. Delete Retailer

```
POST /api/{version}/retailer/delete
```

Description

Set the deleted timestamp to current time.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**retailerId** *optional*	retailerId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.60.3. Get Retailer

```
POST /api/{version}/retailer/get
```

Description

Gets a retailer. Only the owner and the employees of a retailer have access to view its information.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**includeCounts** *required*	includeCounts	boolean	"1"
Query	**retailerId** *required*	retailerId	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	RetailerFullResponse

6.60.4. Login Retailer

```
POST /api/{version}/retailer/login
```

Description

Retailer login check.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**appKey** *optional*	appKey	string
Query	**deviceId** *optional*	deviceId	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)
Query	**password** *required*	password	string
Query	**username** *required*	username	string

Responses

HTTP Code	Description	Schema
200	successful operation	AccountLoginResponse

6.60.5. Search Retailers

```
POST /api/{version}/retailer/search
```

Description

earches on retailers that the account has access to.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**_i** *optional*	_i	integer(int32)	

Type	Name	Description	Schema	Default
Query	**_l** *optional*	_l	integer(int32)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activeOnly** *required*	activeOnly	boolean	`"false"`
Query	**categoryIds** *optional*	categoryIds	string	
Query	**descending** *required*	descending	boolean	`"false"`
Query	**deviceId** *optional*	deviceId	string	
Query	**filterIds** *optional*	filterIds	string	
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**q** *optional*	q	string	

Type	Name	Description	Schema	Default
Query	**sortField** *required*	sortField	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, BILLABLE_ENTITY_ID, BILLABLE_ENTITY_NAME, RESPONSIBLE_DISPLAY, ADDRESS_STREET, ADDRESS_CITY, ADDRESS_STATE, ADDRESS_POSTAL_CODE, ADDRESS_COUNTRY, FILTERS, CATEGORIES, VISIBILITY, NAME)	`"NAME"`
Query	**start** *required*	start	integer(int32)	`"0"`
Query	**visibility** *required*	visibility	enum (PUBLIC, PRIVATE, FRIENDS)	`"PRIVATE"`

Responses

HTTP Code	Description	Schema
200	successful operation	< RetailerResponse > array

6.60.6. Update Retailer

```
POST /api/{version}/retailer/update
```

Description

Update a retailer record. Only the owner and the employees of the retailer have access to update its information.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	
Query	**businessPhone** *optional*	businessPhone	string	
Query	**businessPhoneExt** *optional*	businessPhoneExt	string	
Query	**categoryIds** *optional*	categoryIds	string	
Query	**city** *optional*	city	string	
Query	**country** *optional*	country	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**email** *optional*	email	string	
Query	**facebookUrl** *optional*	facebookUrl	string	
Query	**filterIds** *optional*	filterIds	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**logo** *optional*	logo	ref	
Query	**logoAssetId** *optional*	logoAssetId	integer(int64)	
Query	**longitude** *optional*	longitude	number(double)	

Type	Name	Description	Schema	Default
Query	**metaData** *optional*	metaData	string	
Query	**name** *optional*	name	string	
Query	**picture1** *optional*	picture1	ref	
Query	**picture1AssetId** *optional*	picture1AssetId	integer(int64)	
Query	**picture2** *optional*	picture2	ref	
Query	**picture2AssetId** *optional*	picture2AssetId	integer(int64)	
Query	**postalCode** *optional*	postalCode	string	
Query	**responseFormat** *required*	responseFormat	enum (HTML, XML, JSON, CSV)	`"JSON"`
Query	**retailerId** *required*	retailerId	integer(int64)	
Query	**state** *optional*	state	string	
Query	**streetAddress** *optional*	streetAddress	string	
Query	**streetAddress2** *optional*	streetAddress2	string	
Query	**twitterUrl** *optional*	twitterUrl	string	
Query	**visibility** *optional*	visibility	enum (PUBLIC, PRIVATE, FRIENDS)	
Query	**website** *optional*	website	string	

Responses

HTTP Code	Description	Schema
200	successful operation	RetailerFullResponse

6.61. RetailerLocation

The Locations service enables your business or entity to identify and manage physical locations that you own or operate. For example, if you have multiple medical clinics under a single parent entity, then you have the ability to establish each of these clinics as a 'location'. With this, you have the ability to modify your locations information such as the name, address, employees & roles, operating hours, and much more.

Below are some example scenarios as to how these services can be used:

- Retail: The mobile application presents information about your store location such as the operating hours or address. The administrator who manages the location can update this information through the internal retailer web dashboard.
- Medical: The hospital is operating multiple mobile health check vehicles throughout the city. Each of the vehicles can be configured as a mobile 'location', and the administrators can push out content and messaging to patients that are nearby based upon using a geo-fence trigger.
- Real Estate: As part of a 'smart school' initiative, each elementary school within a county district can be established as a location. Through the county district web dashboard, administrators can manage all configurations of the smart devices that are used at each school location.

The locations service is important when being able to customize the configurations and information for each individual location, as one may not have the same characteristics as the other. For example, your retail locations may have the ability to support in-door positioning of products for customers to more easily locate through the mobile app shopping experience. Since each store location may have different floor plans, it's important to identify the discrepancies and apply the adjustments that must be made to each location. In this case, the administrator must set the correct coordinates of each beacon in accordance to the floor plan of the particular location.

Below are great example services that could be combined with the Locations service:

- Retailers Service: There can be one or many locations, but they must all be created or assigned under one parent location called a retailer.
- Offers Service: Each location can have unique offers (coupons, vouchers, products, etc.) assigned to it. These same offers may not be available at another location.
- Employees Service: Employees are employed by a single parent retailer, but they may be assigned to specific locations that they work at.

6.61.1. Create Retailer Location (Consumer)

```
POST /api/{version}/location/create
```

Description

Creates a location record for an application that can support crowd sourced locations.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	
Query	**appKey** *required*	appKey	string	
Query	**businessPhone** *optional*	businessPhone	string	
Query	**businessPhoneExt** *optional*	businessPhoneExt	string	
Query	**categoryIds** *optional*	categoryIds	string	
Query	**city** *optional*	city	string	
Query	**country** *optional*	country	string	
Query	**detailsBody** *optional*	detailsBody	string	
Query	**detailsHeader** *optional*	detailsHeader	string	
Query	**deviceId** *optional*	deviceId	string	

Type	Name	Description	Schema	Default
Query	**email** *optional*	email	string	
Query	**filterIds** *optional*	filterIds	string	
Query	**hours** *optional*	hours	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**logoAssetId** *optional*	logoAssetId	integer(int64)	
Query	**longitude** *optional*	longitude	number(double)	
Query	**metaData** *optional*	metaData	string	
Query	**name** *required*	name	string	
Query	**picture1AssetId** *optional*	picture1AssetId	integer(int64)	
Query	**picture2AssetId** *optional*	picture2AssetId	integer(int64)	
Query	**postalCode** *optional*	postalCode	string	
Query	**publicLocation** *required*	publicLocation	boolean	`"false"`
Query	**state** *optional*	state	string	
Query	**streetAddress** *optional*	streetAddress	string	
Query	**streetAddress2** *optional*	streetAddress2	string	
Query	**tags** *optional*	tags	string	

Type	Name	Description	Schema	Default
Query	**website** *optional*	website	string	

Responses

HTTP Code	Description	Schema
200	successful operation	RetailerLocationResponse

6.61.2. POST /api/{version}/location/details/get

CAUTION	operation.deprecated

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**distance** *optional*	distance	number(double)
Query	**locationId** *required*	locationId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.61.3. Get Retailer Location (Consumer)

```
POST /api/{version}/location/get
```

Description

Gets the details of a retailer location as a consumer.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**retailerLocationId** *required*	retailerLocationId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	RetailerLocationResponse

6.61.4. Create Retailer Location

```
POST /api/{version}/retailer/location/create
```

Description

Creates a location record for a retailer. Only the owner and the employees of the retailer have access to do this.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	

Type	Name	Description	Schema	Default
Query	**active** *optional*	active	boolean	
Query	**building** *optional*	building	string	
Query	**businessPhone** *optional*	businessPhone	string	
Query	**businessPhoneExt** *optional*	businessPhoneExt	string	
Query	**categoryIds** *optional*	categoryIds	string	
Query	**city** *required*	city	string	
Query	**country** *optional*	country	string	
Query	**detailsBody** *optional*	detailsBody	string	
Query	**detailsHeader** *optional*	detailsHeader	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**email** *optional*	email	string	
Query	**filterIds** *optional*	filterIds	string	
Query	**googlePlaceId** *optional*	googlePlaceId	string	
Query	**hours** *optional*	hours	string	
Query	**internalId** *optional*	internalId	string	
Query	**latitude** *optional*	latitude	number(double)	

Type	Name	Description	Schema	Default
Query	**logo** *optional*	logo	ref	
Query	**logoAssetId** *optional*	logoAssetId	integer(int64)	
Query	**longitude** *optional*	longitude	number(double)	
Query	**metaData** *optional*	metaData	string	
Query	**name** *required*	name	string	
Query	**paymentProvider** *optional*	paymentProvider	string	
Query	**picture1** *optional*	picture1	ref	
Query	**picture1AssetId** *optional*	picture1AssetId	integer(int64)	
Query	**picture2** *optional*	picture2	ref	
Query	**picture2AssetId** *optional*	picture2AssetId	integer(int64)	
Query	**postalCode** *required*	postalCode	string	
Query	**publicLocation** *required*	publicLocation	boolean	`"false"`
Query	**responseFormat** *required*	responseFormat	enum (HTML, XML, JSON, CSV)	`"JSON"`
Query	**responseIncludes** *required*		string	
Query	**retailerId** *required*	retailerId	integer(int64)	

Type	Name	Description	Schema	Default
Query	**state** *required*	state	string	
Query	**streetAddress** *required*	streetAddress	string	
Query	**streetAddress2** *optional*	streetAddress2	string	
Query	**tags** *optional*	tags	string	
Query	**website** *optional*	website	string	
Query	**yelpId** *optional*	yelpId	string	

Responses

HTTP Code	Description	Schema
200	successful operation	RetailerLocationResponse

6.61.5. Delete Retailer Location

```
POST /api/{version}/retailer/location/delete
```

Description

Set the deleted timestamp to current time. This effectively deletes the retailer location since all queries should ignore any records with a deleted time stamp.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)

Type	Name	Description	Schema
Query	**deviceId** *optional*	deviceId	string
Query	**retailerLocationId** *optional*	retailerLocationId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.61.6. Get Retailer Location

```
POST /api/{version}/retailer/location/get
```

Description

Gets a retailer location. Only the owner and the employees of the retailer have access to view its information.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**retailerLocationId** *required*	retailerLocationId	integer(int64)
Query	**retailerLocationToken** *optional*	retailerLocationToken	string

Responses

HTTP Code	Description	Schema
200	successful operation	RetailerLocationResponse

6.61.7. Distance Search Retailer Locations (Indexed)

```
POST /api/{version}/retailer/location/idistancesearch
```

Description

Retailer location indexed search by distance. This searches on any retailer location with location data and returns the results sorted by distance.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**address** *optional*	address	string	
Query	**categories** *optional*	categories	string	
Query	**distanceUnit** *required*	distanceUnit	enum (MILES, KILOMETERS)	`"MILES"`
Query	**filters** *optional*	filters	string	
Query	**hasOffers** *required*	hasOffers	boolean	`"true"`
Query	**keyword** *optional*	keyword	string	
Query	**latitude** *required*	latitude	number(double)	
Query	**limit** *required*	limit	integer(int32)	`"10"`
Query	**longitude** *required*	longitude	number(double)	

Type	Name	Description	Schema	Default
Query	**q** *optional*	q	string	
Query	**returnFavorited** *optional*	returnFavorited	boolean	
Query	**searchRange** *required*	searchRange	number(double)	`"5.0"`
Query	**sortField** *optional*	sortField	string	
Query	**start** *required*	start	integer(int32)	`"0"`
Query	**tags** *optional*	tags	string	

Responses

HTTP Code	Description	Schema
200	successful operation	< RetailerLocationResponse > array

6.61.8. Keyword Search Retailer Locations (Indexed)

```
POST /api/{version}/retailer/location/isearch
```

Description

Retailer location (faster) indexed search. This searches all retailer locations.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**bq** *optional*	bq	string	
Query	**categories** *optional*	categories	string	

Type	Name	Description	Schema	Default
Query	**filters** *optional*	filters	string	
Query	**hasOffers** *required*	hasOffers	boolean	`"true"`
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"10"`
Query	**q** *optional*	q	string	
Query	**sortField** *required*	sortField	string	`"RETAILER_LOCATION_NAME"`
Query	**start** *required*	start	integer(int32)	`"0"`
Query	**tags** *optional*	tags	string	

Responses

HTTP Code	Description	Schema
200	successful operation	< RetailerLocationResponse > array

6.61.9. Search Retailer Locations (Owned)

```
POST /api/{version}/retailer/location/search
```

Description

Searches on retailer locations that the account has access to.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	

Type	Name	Description	Schema	Default
Query	**_i** *optional*	_i	integer(int32)	
Query	**_l** *optional*	_l	integer(int32)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activeOnly** *required*	activeOnly	boolean	`"false"`
Query	**descending** *required*	descending	boolean	`"false"`
Query	**deviceId** *optional*	deviceId	string	
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**q** *optional*	q	string	
Query	**retailerIds** *optional*	retailerIds	string	
Query	**showPublicLocations** *required*	showPublicLocations	boolean	`"false"`

Type	Name	Description	Schema	Default
Query	**sortField** *required*	sortField	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, RETAILER_ID, BILLABLE_ENTITY_ID, BILLABLE_ENTITY_NAME, RESPONSIBLE_DISPLAY, ADDRESS_STREET, ADDRESS_CITY, ADDRESS_STATE, ADDRESS_POSTAL_CODE, ADDRESS_COUNTRY, NAME, CODE)	`"NAME"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< RetailerLocationResponse > array

6.61.10. Update Retailer Location

```
POST /api/{version}/retailer/location/update
```

Description

Updates a location record for a retailer. Only the owner and the employees of the retailer have access to do this.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	
Query	**building** *optional*	building	string	
Query	**businessPhone** *optional*	businessPhone	string	
Query	**businessPhoneExt** *optional*	businessPhoneExt	string	
Query	**categoryIds** *optional*	categoryIds	string	
Query	**city** *optional*	city	string	
Query	**country** *optional*	country	string	
Query	**detailsBody** *optional*	detailsBody	string	
Query	**detailsHeader** *optional*	detailsHeader	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**email** *optional*	email	string	
Query	**filterIds** *optional*	filterIds	string	
Query	**googlePlaceId** *optional*	googlePlaceId	string	
Query	**hours** *optional*	hours	string	

Type	Name	Description	Schema	Default
Query	**internalId** *optional*	internalId	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**logo** *optional*	logo	ref	
Query	**logoAssetId** *optional*	logoAssetId	integer(int64)	
Query	**longitude** *optional*	longitude	number(double)	
Query	**metaData** *optional*	metaData	string	
Query	**name** *optional*	name	string	
Query	**paymentProvider** *optional*	paymentProvider	string	
Query	**picture1** *optional*	picture1	ref	
Query	**picture1AssetId** *optional*	picture1AssetId	integer(int64)	
Query	**picture2** *optional*	picture2	ref	
Query	**picture2AssetId** *optional*	picture2AssetId	integer(int64)	
Query	**postalCode** *optional*	postalCode	string	
Query	**publicLocation** *required*	publicLocation	boolean	`"false"`
Query	**responseFormat** *required*	responseFormat	enum (HTML, XML, JSON, CSV)	`"JSON"`

Type	Name	Description	Schema	Default
Query	**retailerLocationId** *required*	retailerLocationId	integer(int64)	
Query	**state** *optional*	state	string	
Query	**streetAddress** *optional*	streetAddress	string	
Query	**streetAddress2** *optional*	streetAddress2	string	
Query	**tags** *optional*	tags	string	
Query	**website** *optional*	website	string	
Query	**yelpId** *optional*	yelpId	string	

Responses

HTTP Code	Description	Schema
200	successful operation	RetailerLocationResponse

6.62. RetailerV2

The Retailers API allows for the defining of business and locations of those businesses within your application. This can be used in two ways. First, it can be used for the input of physical retail or service locations into the application owned by the company. Second, an application owner can allow other businesses to input their Retail locations into the database. For example, an application offering city-wide coupons for goods and services purchased at physical location can allow any business in a specific city or region to input their information for users to discover and buy.

Below are some example scenarios as to how the Rating service can be used:

- Retail: Setup Retailers for every store location owned by your company. These can be displayed on a map and include information such as store hours and phone number. Include Coupons to help drive customers to your Retail locations.
- Medical: Use Retailers to display all the Hospital or Medical Center locations owned by your

company then list services plus include the ability to schedule appointments.

- Smart Home: Use Retailers to allow other businesses to list their goods and services in your application for the purchase of goods, maintenance or service items needed in the home that are supplied by third parties.
- Automotive: Use Retailers to show fueling stations, service centers, parking lots and more within your application .
- Robotics: Use Retailers to input all the potential pickup locations for drones to pickup Purchased Item packages for delivery to customers.

The Retailers API is important because it allows you specifically input the information and data for physical locations, or for third parties to input their locations into your application.

Below are great example services that could be combined with the Reporting service:

- Location: Use Location in combination with Retailers to allow users to search and find various businesses close to their current position.
- Audience: Use Retailers with Audience to recommend certain businesses and services to a particular Audience.
- Push Notifications: Allow Retailers to purchase Push Notifications that can be sent to your applications users. This lets you monetize your user base in a unique way.
- Missions: Create Missions for your application users to visit specific Retailers and get various rewards for Missions completed to increase customer value and application usage.

6.62.1. Search Retailers

```
GET /api/{version}/retailer
```

Description

Search on Retailers

Parameters

Type	Name	Schema
Path	**version** *required*	number(double)
Query	**activeOnly** *required*	boolean
Query	**descending** *required*	boolean

Type	Name	Schema
Query	**keyword** *optional*	string
Query	**limit** *required*	integer(int32)
Query	**sortField** *required*	string
Query	**start** *required*	integer(int32)

Responses

HTTP Code	Description	Schema
200	successful operation	< Retailer > array

6.63. Route

Routes are the core piece to any transportation based service. They enable an organization to have an efficient, organized way of storing where drivers need to go on any particular day. Drivers then have up to the minute information on where they need to go for the day, who or what they will be picking up and dropping off, and when all of these tasks need to happen.

Below are some example scenarios as to how these services can be used:

- Retail: A warehouse makes deliveries to many retailer locations throughout the week. Using Routes, everyone (drivers, dispatchers, recipients) has a clear overview of when and what is being picked up and dropped off.
- Robotics: A robot in a manufacturing plant could follow a specified route to deliver goods to the employees..
- Real Estate: A real estate agent who is showing houses to a potential buyer can follow a pre-planned, optimized route to show the buyers the most houses in the given time.

Routes are essential in managing and executing a successful transportation based service. They are extremely detailed, yet very flexible.

Below are great example services that could be combined with this service:

- Notification Service: Contact a dispatcher or recipient when a driver reaches a specified stop on the route.
- Analytics Service: Track how many times a particular stop is used on a route.

- Dashboards: Easily create, modify, and delete routes on the dashboard, drivers can get instant updates for any changes made.

6.63.1. Create Route

```
POST /api/{version}/route
```

Description

Create new route

Parameters

Type	Name	Schema
Path	**version** *required*	string
Body	**body** *optional*	Route

Responses

HTTP Code	Description	Schema
200	successful operation	Route

6.63.2. Search Routes

```
GET /api/{version}/route
```

Description

Search for routes.

Parameters

Type	Name	Schema	Default
Path	**version** *required*	number(double)	
Query	**activeOnly** *required*	boolean	`"true"`

Type	Name	Schema	Default
Query	**approved** *optional*	boolean	
Query	**completed** *optional*	boolean	
Query	**descending** *required*	boolean	`"false"`
Query	**featured** *optional*	boolean	
Query	**hasDriver** *optional*	boolean	
Query	**hubId** *optional*	integer(int64)	
Query	**includesEmpty** *required*	boolean	`"true"`
Query	**limit** *required*	integer(int32)	`"20"`
Query	**parentId** *optional*	integer(int64)	
Query	**programId** *optional*	integer(int64)	
Query	**rootOnly** *required*	boolean	`"false"`
Query	**scheduledEnd** *optional*	integer(int64)	
Query	**scheduledStart** *optional*	integer(int64)	
Query	**seatCount** *optional*	integer(int32)	
Query	**showInheritedProperties** *required*	boolean	`"true"`
Query	**sortField** *required*	string	`"id"`
Query	**start** *required*	integer(int32)	`"0"`
Query	**started** *optional*	boolean	

Type	Name	Schema	Default
Query	**updatedEnd** *optional*	integer(int64)	
Query	**updatedStart** *optional*	integer(int64)	
Query	**valid** *optional*	boolean	

Responses

HTTP Code	Description	Schema
200	successful operation	< Route > array

6.63.3. Set Driver

```
POST /api/{version}/route/{id}/driver/{driverId}
```

Description

Update the driver of the route.

Parameters

Type	Name	Schema
Path	**driverId** *required*	integer(int64)
Path	**id** *required*	integer(int64)
Path	**version** *required*	number(double)

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

6.63.4. Get Route

```
GET /api/{version}/route/{routeId}
```

Description

Get an existing route

Parameters

Type	Name	Schema	Default
Path	**routeId** *required*	integer(int64)	
Path	**version** *required*	string	
Query	**showInheritedProperties** *required*	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	Route

6.63.5. Update Route

```
PUT /api/{version}/route/{routeId}
```

Description

Update an existing route

Parameters

Type	Name	Schema
Path	**routeId** *required*	integer(int64)
Path	**version** *required*	string

Type	Name	Schema
Body	**body** *optional*	Route

Responses

HTTP Code	Description	Schema
200	successful operation	Route

6.63.6. Delete Route

```
DELETE /api/{version}/route/{routeId}
```

Description

Delete an existing route

Parameters

Type	Name	Schema
Path	**routeId** *required*	integer(int64)
Path	**version** *required*	string

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

6.63.7. Approve Route

```
POST /api/{version}/route/{routeId}/approve
```

Description

Approve a route

Parameters

Type	Name	Schema
Path	**routeId** *required*	integer(int64)
Path	**version** *required*	string

Responses

HTTP Code	Description	Schema
200	successful operation	Route

6.63.8. Copy Route

```
POST /api/{version}/route/{routeId}/copy
```

Description

Make an copy of the given route with optional overriding properties

Parameters

Type	Name	Schema
Path	**routeId** *required*	integer(int64)
Path	**version** *required*	string
Body	**body** *optional*	Route

Responses

HTTP Code	Description	Schema
200	successful operation	Route

6.63.9. Get Route Directions

```
GET /api/{version}/route/{routeId}/directions
```

Description

Get the directions of a route

Parameters

Type	Name	Schema
Path	**routeId** *required*	integer(int64)
Path	**version** *required*	string

Responses

HTTP Code	Description	Schema
200	successful operation	< Direction > array

6.63.10. Update Route Directions

```
PUT /api/{version}/route/{routeId}/directions
```

Description

Regenerate the directions of a route

Parameters

Type	Name	Schema
Path	**routeId** *required*	integer(int64)
Path	**version** *required*	string

Responses

HTTP Code	Description	Schema
200	successful operation	< Direction > array

6.63.11. Disapprove Route

```
POST /api/{version}/route/{routeId}/disapprove
```

Description

Disapprove a route

Parameters

Type	Name	Schema
Path	**routeId** *required*	integer(int64)
Path	**version** *required*	string

Responses

HTTP Code	Description	Schema
200	successful operation	Route

6.63.12. Optimize Route

```
POST /api/{version}/route/{routeId}/optimize
```

Description

Optimize a route. The optimization method based on how the server is configured.

Parameters

Type	Name	Schema
Path	**routeId** *required*	integer(int64)

Type	Name	Schema
Path	**version** *required*	string

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

6.63.13. Create Route Polyline

```
PUT /api/{version}/route/{routeId}/polyline
```

Description

Update the polyline of the requested route

Parameters

Type	Name	Schema
Path	**routeId** *required*	integer(int64)
Path	**version** *required*	string

Responses

HTTP Code	Description	Schema
200	successful operation	Route

6.63.14. Get Route Shipments

```
GET /api/{version}/route/{routeId}/shipments
```

Description

Get the shipments on the requested route

Parameters

Type	Name	Schema
Path	**routeId** *required*	integer(int64)
Path	**version** *required*	string

Responses

HTTP Code	Description	Schema
200	successful operation	< Shipment > array

6.63.15. Get Route Stop

```
GET /api/{version}/route/{routeId}/stop/{stopId}
```

Description

Get the specific stop on a route

Parameters

Type	Name	Schema
Path	**routeId** *required*	integer(int64)
Path	**stopId** *required*	integer(int64)
Path	**version** *required*	string

Responses

HTTP Code	Description	Schema
200	successful operation	Stop

6.63.16. Update Route Stop

```
PUT /api/{version}/route/{routeId}/stop/{stopId}
```

Description

Update a stop on a specified route

Parameters

Type	Name	Schema
Path	**routeId** *required*	integer(int64)
Path	**stopId** *required*	integer(int64)
Path	**version** *required*	string
Body	**body** *optional*	Stop

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

6.63.17. Delete Stop

```
DELETE /api/{version}/route/{routeId}/stop/{stopId}
```

Description

Delete a stop on a route

Parameters

Type	Name	Description	Schema
Path	**routeId** *required*		integer(int64)
Path	**stopId** *required*		integer(int64)

Type	Name	Description	Schema
Path	**version** *required*	version	string

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

6.63.18. Get Shipments At Stop

```
GET /api/{version}/route/{routeId}/stop/{stopId}/shipments
```

Description

Get the list of shipments on the requested route at a stop

Parameters

Type	Name	Schema
Path	**routeId** *required*	integer(int64)
Path	**stopId** *required*	integer(int64)
Path	**version** *required*	string

Responses

HTTP Code	Description	Schema
200	successful operation	< Shipment > array

6.63.19. Get Route Stops

```
GET /api/{version}/route/{routeId}/stops
```

Description

The stops of the route requested

Parameters

Type	Name	Schema	Default
Path	**routeId** *required*	integer(int64)	
Path	**version** *required*	string	
Query	**confirmedOnly** *required*	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	< Stop > array

6.63.20. Reorder Route Stops

```
POST /api/{version}/route/{routeId}/stops/reorder
```

Description

Reordering the stops on the route with and update route distance, time, direction, and polyline

Parameters

Type	Name	Schema
Path	**routeId** *required*	integer(int64)
Path	**version** *required*	string
Body	**body** *optional*	< Stop > array

Responses

HTTP Code	Description	Schema
200	successful operation	< Stop > array

6.63.21. Reorder Route Stops

```
PATCH /api/{version}/route/{routeId}/stops/reorder
```

Description

Reordering the stops on the route with and update route distance, time, direction, and polyline

Parameters

Type	Name	Schema
Path	**routeId** *required*	integer(int64)
Path	**version** *required*	string
Body	**body** *optional*	< Stop > array

Responses

HTTP Code	Description	Schema
200	successful operation	< Stop > array

6.64. RouteSetting

Route settings are global settings for a hub/program pair to help configure how routes should be generated. Settings can be created, updated, deleted, and viewed. Use these settings for any hub/program specific values or options for that hub/program's routes. For example, you could choose to set the maximum time that a rider can be on a vehicle

Below are some example scenarios as to how these services can be used:

- Retail: Route settings could be used to limit how much time perishable foods are on a delivery truck.
- Medical: Route settings could be used to send an abmulance on a road that is otherwise unaccessable.

- Robotics: Using route settings, robots in motion can limit their speed based upon the given setting for their hub and program.

Ride share, delivery, and other transportation based programs and their associated hubs all need various settings based on which program or hub they are a part of at any given time. Using route settings enables differences between an organization's various hubs and programs that allow the organization to run efficiently and safely.

Below are great example services that could be combined with this service:

- Notification Service: Notify a driver if a setting is changed.
- Routing: Find the best route for multiple stops, use settings to fine tune your preferences for the route..
- Dashboards: Easily create, update, or remove route settings on a dashboard.

6.64.1. Create Route Setting

```
POST /api/{version}/route/setting
```

Description

Create a new route setting

Parameters

Type	Name	Schema
Path	**version** *required*	string
Body	**body** *optional*	RouteSettings

Responses

HTTP Code	Description	Schema
200	successful operation	RouteSettings

6.64.2. Search Route Settings

```
GET /api/{version}/route/setting
```

Description

Search for route settings

Parameters

Type	Name	Schema	Default
Path	**version** *required*	number(double)	
Query	**activeOnly** *required*	boolean	`"true"`
Query	**descending** *required*	boolean	`"false"`
Query	**hubId** *optional*	integer(int64)	
Query	**keyword** *optional*	string	
Query	**limit** *required*	integer(int32)	`"20"`
Query	**programId** *optional*	integer(int64)	
Query	**sortField** *required*	string	`"id"`
Query	**start** *required*	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< RouteSettings > array

6.64.3. Get Route Setting

```
GET /api/{version}/route/setting/{routeSettingsId}
```

Description

Get an existing route settings

Parameters

Type	Name	Schema
Path	**routeSettingsId** *required*	integer(int64)
Path	**version** *required*	string

Responses

HTTP Code	Description	Schema
200	successful operation	RouteSettings

6.64.4. Update Route Setting

```
PUT /api/{version}/route/setting/{routeSettingsId}
```

Description

Update an existing route setting

Parameters

Type	Name	Schema
Path	**routeSettingsId** *required*	integer(int64)
Path	**version** *required*	string
Body	**body** *optional*	RouteSettings

Responses

HTTP Code	Description	Schema
200	successful operation	RouteSettings

6.64.5. Delete Route Setting

```
DELETE /api/{version}/route/setting/{routeSettingsId}
```

Description

Delete an existing route setting

Parameters

Type	Name	Schema
Path	**routeSettingsId** *required*	integer(int64)
Path	**version** *required*	string

Responses

HTTP Code	Description	Schema
200	successful operation	object

6.65. Routing

The Routing Simulation service takes a set of resources like vehicles, drivers, and passengers, and uses the Sirqul routing algorithm to generate the most optimal recommendations on how passengers or objects should be delivered or fulfilled. The best scenario to use this tool is when you are running a business that involves hundreds or thousands of shipments that need to be delivered the same day they're requested. The simulation can be used to determine which shipments should be assigned to available drivers and vehicles. You can generate hundreds to thousands of vehicles from a single hub, to a city, to even the entire United States. The simulator makes it easy to learn how you plan to scale your business and understanding the risks and costs involved before actually committing to shipment deliveries.

Below are some example scenarios as to how these services can be used:

- There is a 1,000 shipment order that is requested, and each shipment has a different time window delivery requirement, different drop off location, and there are mixed cargo types. The routing simulator allows you to upload the shipments and the algorithm will take the required time windows, different drop off locations, and mixed cargo types, and generate suggestions on how the shipments should be added onto routes, and assigned to the appropriate drivers and vehicles
- Use the Routing simulator when you are trying to optimize the floorplan of your warehouse. If you have machine that is used to help move around cargo inside of a building, you can use the simulator to determine how shipments should be moved based upon their delivery requirement,

weight, size, shape, driver/vehicle assigned, and more.

The Routing Simulation service is important as it allows your businesses to plan and scale your business without having to do any guess work. It allows you to load up many vehicles, drivers, and passengers/objects needing to be picked up/dropped off into a simulator, and will be able to get suggestions on how the passengers/objects should be routed.

Other services that work great with the simulator is Audience (Geofence) service, which allows you to designated specific locations to determine a clustered drop off area for one to many passengers/objects. We typically call these 'Zones'. Another great service to pair with the routing simulator is the Assignments (Tasks) service, as a route suggested by the simulator could be assigned to a driver as a 'task' that needs to be completed.

6.65.1. Compute Route

```
POST /api/{version}/routing/compute
```

Description

This service finds the most optimal routes for delivering items between locations (reducing transit time/resources). It can take in a list of vehicles and a list of items (to be transported).All load items have pick-up and drop-off locations with time windows for when the item is expected to be picked-up and dropped-off.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The api version	number(double)
Query	**data** *required*	Json object containing inputs for generating the routes. See description for more info. Also see RoutingRequest	string

Responses

HTTP Code	Description	Schema
200	successful operation	RoutingListResponse

6.66. ScheduledNotification

The Scheduled Notifications API allows for you to easily create a single or set of Push Notifications to

be sent through your application. Scheduled Notifications can be sent to a single user, a group, or your entire user base. This feature helps you create set days and times for your Push Notifications to automatically be sent. This eliminates the tedious work of having to manually send each Push Notification. With this service, you can set up all the Push Notifications for your application for the week, month, quarter or the year.

Below are some example scenarios as to how the Rating service can be used:

- Retail: Setup Scheduled Notifications to send out a Push Notification every Friday of the year at 3 p.m. with a Coupon for extra savings to drive customers to your stores.
- Medical: Use Scheduled Notifications to automatically send Push Notifications to specific users for follow up doctor visits.
- Smart Home: Use Scheduled Notifications to automatically alert users of important dates and times such as National Holidays or Daylight Savings Time.
- Automotive: Use Scheduled Notifications to automatically alert drivers of scheduled road closures, or timed service schedule alerts such as “Monthly Service Inspection”.
- Robotics: Use Scheduled Notifications to alert specific service personnel of maintenance checks that need to be performed on specific days and times on certain robotics.

The Scheduled Notifications API is important because it allows you automate specific messaging to your application users on the days and times needed.

Below are great example services that could be combined with the Reporting service:

- Coupons or Offers: Use Coupons or Offers scheduled at the end of each month to drive sales of Purchase Items within your application for added revenues.
- Groups: Use Scheduled Notifications to automatically notify a specific Group of an event particular to that Group, such as an upcoming game for a team.
- Events: Combine Scheduled Notifications with Events such as concerts, sporting events, and more to alert users to purchase tickets in advance.
- Missions: Send Scheduled Notification to your application users to notify of them of new Missions as they are released each month.

6.66.1. Create Scheduled Notification

```
POST /api/{version}/notification/schedule/create
```

Description

This endpoint creates a Scheduled Notification message that can be configured to process and send periodically at set time periods

The processing schedule is configured by passing in a UNIX cron expression into the cronExpression

parameter. In addition, you can also specify when it's available to start processing using the startDate parameter, and when it expires using the endDate parameter. If no cronExpression is specified, then the message will only be sent once at the scheduledDate.</p>

Scheduling Example:

```
// starts on Tue, 01 Nov 2016 00:00:00 GMT
startDate = 1477958400000
//ends on Thu, 01 Dec 2016 00:00:00 GMT
endDate = 1480550400000
// every Monday at midnight UTC
cronExpression = 0 0 * * 1
```

The parameters above will configure the notifications to process and send at the following times:

```
Mon, 07 Nov 2016 00:00:00 GMT
Mon, 14 Nov 2016 00:00:00 GMT
Mon, 21 Nov 2016 00:00:00 GMT
Mon, 28 Nov 2016 00:00:00 GMT
```

The recipient list can be configured by using the following parameters:

- connectionGroupIds - Which is a comma separated list of connection group IDs and will send to all users in the list of groups (A connection group is a private group of accounts that the user creates).
- connectionAccountIds - Which is a comma separated list of account IDs and will send to those individual users
- audienceIds - Which is a comma separated list of audience IDs and will send to any user who match the audience criteria. For example, an audience of "males in Seattle" will only send to male users who are currently in Seattle.
- appKey - Which only sends to users who have used the following application. If only the appKey is set, it will send to all users who have used the application. This parameter is also required when sending push notifications via the MOBILE_NOTIFICATION type.

Note that at least one of the four parameters (connectionGroupIds, connectionAccountIds, audienceIds, appKey) must have a value. Also note, the recipient list is exclusive. This means that if an audience of "males" and a connection group is used, then only the users in the group who are male will receive the notification messages.</p>

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *required*	active	boolean	`"true"`
Query	**albumIds** *optional*	albumIds	string	
Query	**appKey** *optional*	appKey	string	
Query	**audienceId** *optional*	audienceId	integer(int64)	
Query	**audienceIds** *optional*	audienceIds	string	
Query	**conditionalInput** *optional*	conditionalInput	string	
Query	**connectionAccountIds** *optional*	connectionAccountIds	string	
Query	**connectionGroupIds** *optional*	connectionGroupIds	string	
Query	**contentId** *optional*	contentId	integer(int64)	
Query	**contentName** *optional*	contentName	string	
Query	**contentType** *optional*	contentType	string	
Query	**cronExpression** *optional*	cronExpression	string	
Query	**cronType** *optional*	cronType	string	
Query	**deepLinkURI** *optional*	deepLinkURI	string	

Type	Name	Description	Schema	Default
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**endpointURL** *optional*	endpointURL	string	
Query	**eventType** *required*	eventType	string	`"CUSTOM"`
Query	**groupingId** *optional*	groupingId	string	
Query	**message** *required*	message	string	
Query	**metaData** *optional*	metaData	string	
Query	**name** *required*	name	string	
Query	**parentId** *optional*	parentId	integer(int64)	
Query	**parentType** *optional*	parentType	string	
Query	**payload** *optional*	payload	string	
Query	**reportId** *optional*	reportId	integer(int64)	
Query	**reportParams** *optional*	reportParams	string	
Query	**scheduledDate** *optional*	scheduledDate	integer(int64)	
Query	**sendNow** *optional*	sendNow	boolean	
Query	**startDate** *optional*	startDate	integer(int64)	
Query	**templateType** *optional*	templateType	string	
Query	**type** *required*	type	string	

Type	Name	Description	Schema	Default
Query	**visibility** *optional*	visibility	enum (PUBLIC, PRIVATE, FRIENDS)	

Responses

HTTP Code	Description	Schema
200	successful operation	ScheduledNotificationFullResponse

6.66.2. Delete Scheduled Notification

```
POST /api/{version}/notification/schedule/delete
```

Description

This endpoint deletes a Scheduled Notification. Only the original owner of the Scheduled Notification or someone with write permissions can use this endpoint. Permissions can be granted to other users by using the UserPermissionsApi.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**deleteByGroupingId** *required*	deleteByGroupingId	boolean	`"false"`
Query	**scheduledNotificationId** *required*	scheduledNotificationId	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	ScheduledNotificationFullResponse

6.66.3. Generate Schedule Notifications

```
POST /api/{version}/notification/schedule/generate
```

Description

Use a report to identify events that are starting soon and then create a scheduled notification to push a message to matching users.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *required*	appKey	string	
Query	**message** *required*	message	string	
Query	**offset** *required*	offset	integer(int32)	
Query	**recipientReportId** *required*	recipientReportId	integer(int64)	
Query	**reportName** *required*	reportName	string	
Query	**reportParams** *optional*	reportParams	string	
Query	**type** *required*	type	string	`"MOBILE_NOTIFICATION"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.66.4. Get Scheduled Notification

```
POST /api/{version}/notification/schedule/get
```

Description

Get a ScheduledNotification

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**scheduledNotificationId** *required*	scheduledNotificationId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	ScheduledNotificationFullResponse

6.66.5. Search Scheduled Notifications

```
POST /api/{version}/notification/schedule/search
```

Description

This endpoint searches on Scheduled Notifications. If a scheduled notification was created with the visibility parameter set to PUBLIC, then anyone can search on it if the filter parameter includes the PUBLIC value. PRIVATE visibility means that it can only be searched on by the owner or if it has been shared to the user using the UserPermissionsApi.

In addition, if a PUBLIC Scheduled Notification was created for an application that requires content approval (using the publicContentApproval parameter), then an administrator of the application needs to approve it before it can be search on by other users. Before this happens, it is in a PENDING state, and only the original creator or the owner of the application can search and see it. Also, only the owner of the application can use the UserPermissionsApi to approve or reject it. Scheduled

notifications that have been rejected are only visible to the original creators.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activeOnly** *required*	activeOnly	boolean	`"true"`
Query	**appKey** *optional*	appKey	string	
Query	**audienceId** *optional*	audienceId	integer(int64)	
Query	**contentIds** *optional*	contentIds	string	
Query	**contentTypes** *optional*	contentTypes	string	
Query	**descending** *required*	descending	boolean	`"true"`
Query	**filter** *required*	filter	string	`"MINE"`
Query	**groupByGroupingId** *required*	groupByGroupingId	boolean	`"false"`
Query	**groupingId** *optional*	groupingId	string	
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**parentIds** *optional*	parentIds	string	
Query	**parentTypes** *optional*	parentTypes	string	
Query	**sortField** *required*	sortField	string	`"CREATED"`

Type	Name	Description	Schema	Default
Query	**start** *required*	start	integer(int32)	`"0"`
Query	**statuses** *required*	statuses	string	`"NEW,ERROR,COMPLETE,PROCESSING"`
Query	**templateTypes** *optional*	templateTypes	string	
Query	**types** *optional*	types	string	

Responses

HTTP Code	Description	Schema
200	successful operation	ScheduledNotificationFullResponse

6.66.6. Update Scheduled Notification

```
POST /api/{version}/notification/schedule/update
```

Description

This endpoint updates a Scheduled Notification message that can be configured to process and send periodically at set time periods. Please see createScheduledNotification for more details.

Only the original owner of the Scheduled Notification or someone with write permissions can use this endpoint. Permissions can be granted to other users by using theUserPermissionsApi.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	

Type	Name	Description	Schema	Default
Query	**albumIds** *optional*	albumIds	string	
Query	**appKey** *optional*	appKey	string	
Query	**audienceId** *optional*	audienceId	integer(int64)	
Query	**audienceIds** *optional*	audienceIds	string	
Query	**conditionalInput** *optional*	conditionalInput	string	
Query	**connectionAccountIds** *optional*	connectionAccountIds	string	
Query	**connectionGroupIds** *optional*	connectionGroupIds	string	
Query	**contentId** *optional*	contentId	integer(int64)	
Query	**contentName** *optional*	contentName	string	
Query	**contentType** *optional*	contentType	string	
Query	**cronExpression** *optional*	cronExpression	string	
Query	**cronType** *optional*	cronType	string	
Query	**deepLinkURI** *optional*	deepLinkURI	string	
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**endpointURL** *optional*	endpointURL	string	
Query	**errorMessage** *optional*	errorMessage	string	

Type	Name	Description	Schema	Default
Query	**eventType** *required*	eventType	string	`"CUSTOM"`
Query	**groupingId** *optional*	groupingId	string	
Query	**message** *optional*	message	string	
Query	**metaData** *optional*	metaData	string	
Query	**name** *optional*	name	string	
Query	**parentId** *optional*	parentId	integer(int64)	
Query	**parentType** *optional*	parentType	string	
Query	**payload** *optional*	payload	string	
Query	**reportId** *optional*	reportId	integer(int64)	
Query	**reportParams** *optional*	reportParams	string	
Query	**scheduledDate** *optional*	scheduledDate	integer(int64)	
Query	**scheduledNotificationId** *required*	scheduledNotificationId	integer(int64)	
Query	**sendNow** *optional*	sendNow	boolean	
Query	**startDate** *optional*	startDate	integer(int64)	
Query	**status** *optional*	status	string	
Query	**templateType** *optional*	templateType	string	

Type	Name	Description	Schema	Default
Query	**type** *optional*	type	string	
Query	**updateByGroupingId** *required*	updateByGroupingId	boolean	`"false"`
Query	**visibility** *optional*	visibility	enum (PUBLIC, PRIVATE, FRIENDS)	

Responses

HTTP Code	Description	Schema
200	successful operation	ScheduledNotificationFullResponse

6.67. Score

The Score API lets you assign points for certain actions that users take within your application. The most obvious example is creating a game and awarding points to users for finishing each level of the game. Score can be combined with Ranking (Leaderboards) to show overall rank of players in a game or application. Score can also be used as an indicator of digital currency earned by an application user. For example, for every store visit or level completed, a user can earn 10 points that can eventually be redeemed for savings on Purchase Items.

Below are some example scenarios as to how the Rating service can be used:

- Retail: Setup Score to reward your customers for each physical visit to your locations. Add a Ranking (Leaderboard) so your customers compete with each other for Badges.
- Medical: Use Score to keep track of employee work days and display a Leaderboard (or Badges) for high-performing employees.
- Smart Home: Use Score to keep track of the number of times a device is used. Keep track of specific users in your database to reward heavy users of your Smart Home device with free upgrades or special offers.
- Automotive: Use Score to keep track of miles driven and display a Leaderboard of top drivers in your fleet.
- Robotics: Use Score to keep track of hours of uptime, or units produced for a specific assembly line.

The Score API is important because it allows you to creatively track various interactions and events on a user by user basis within your application and then display results in a Ranking (Leaderboard).

Below are great example services that could be combined with the Reporting service:

- Coupons or Offers: Use Coupons or Offers to be awarded to your application users when they reach certain Scores.
- Groups: Use Score to let teams compete against each other and see their Ranking against other teams to foster productivity or increase usage of your application.
- Retailers: Award points to your application users when they visit your physical locations. Award extra points for visiting new locations or Bonus Points for visiting a specific location more than a certain number of times.
- Missions: Reward your application users with various amounts of points for completing Missions. Easier Missions get lower points, more difficult Missions result in a higher amount of points earned.

6.67.1. Create Score

```
POST /api/{version}/score/create
```

Description

Create a score. The response object will contain a series of coded messages detailing what items were completed, the score registered, and any tickets allocated. Scoring a level could complete the pack it is in, completing that pack could complete the game, which in turn could complete the mission. This completion chain is indicated to the client via
a list of {@link MessageResponse}.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *required*	appKey	string	
Query	**gameId** *required*	gameId	integer(int64)	`"0"`
Query	**gameLevelId** *required*	gameLevelId	integer(int64)	`"0"`
Query	**gameObjectId** *required*	gameObjectId	integer(int64)	`"0"`

Type	Name	Description	Schema	Default
Query	**highest** *required*	highest	boolean	`"true"`
Query	**missionId** *required*	missionId	integer(int64)	`"0"`
Query	**packId** *required*	packId	integer(int64)	`"0"`
Query	**points** *required*	points	integer(int32)	
Query	**timeTaken** *required*	timeTaken	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	ScoreResponse

6.67.2. Get Score

```
POST /api/{version}/score/get
```

Description

Get the high score for an item. Pass in the full path IDs for the score.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *required*	appKey	string
Query	**gameId** *optional*	gameId	integer(int64)
Query	**gameLevelId** *optional*	gameLevelId	integer(int64)

Type	Name	Description	Schema
Query	**gameObjectId** *optional*	gameObjectId	integer(int64)
Query	**missionId** *optional*	missionId	integer(int64)
Query	**packId** *optional*	packId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	ScoreResponse

6.67.3. Search Score

```
POST /api/{version}/score/search
```

Description

Search the scores for an item. Pass in the full path IDs for the scores.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *required*	appKey	string
Query	**gameId** *optional*	gameId	integer(int64)
Query	**gameLevelId** *optional*	gameLevelId	integer(int64)
Query	**gameObjectId** *optional*	gameObjectId	integer(int64)
Query	**missionId** *optional*	missionId	integer(int64)

Type	Name	Description	Schema
Query	**packId** *optional*	packId	integer(int64)
Query	**scoreObjectType** *optional*	scoreObjectType	enum (GAME_OBJECT, GAME_LEVEL, PACK, GAME, MISSION, PROFILE, APPLICATION, TICKETS, ASSET, CUSTOM)
Query	**scoreStatus** *optional*	scoreStatus	enum (COMPLETED, OWNER_CREDIT, SUMMATION, TOURNAMENT)

Responses

HTTP Code	Description	Schema
200	successful operation	< ScoreResponse > array

6.68. SecureApp

The SecureApp service enables your application, website, or service to enable users to identify or authenticate themselves using facial or fingerprinting technology. Facial and fingerprint authentication is increasingly becoming a feature that many users desire when they want to have a quick way to authenticate themselves either when they are on the go and don't want to have to enter information. A great use case for this is using fingerprinting or facial technology when making a purchase transaction. Because facial and fingerprinting is unique per person, there is an extremely low likelihood that your information will ever get compromised, which is why this type of authentication is becoming increasingly in demand.

Below are some example scenarios as to how these services can be used:

- There is a long line to get into a football stadium and fans are frustrated with having to wait. Setup kiosks at the stadium and use either fingerprinting or facial recognition technology to help expedite the process so that fans can enter into the stadium without ever having to stop walking. We would be able to verify that the user can enter the stadium because they have scanned their face and/or fingerprint when they setup their account
- Implement fingerprinting or facial recognition scanning for residents who live in a high-rise building. They can authenticate themselves when using the elevator, access the front entrance after

hours, and even access their own home unit, without ever having to use a card or dongle to scan.

The SecureApp service is important because it is much more accurate and secure than traditional methods of authenticating like entering a username and password, and in addition, can also reduce costs by not requiring items such as ID cards, dongles, etc. Facial and fingerprinting technology is already being applied in major industries like ticketing systems, security gates at airports, and even accessing a laptop. It's increasingly becoming the next method of authenticating users because it provides convenience, reduces costs by replacing/reducing papertrail and ID cards, and is a much more secure way of providing users with access.

Below are example services that could be combined with the SecureApp service:

- The Accounts, Categories, Events, and Locations service works well with the SecureApp service because you need an account that is associated with the valid face or fingerprint of the user, your system could categorize the user into a certain category based upon their identity, associate the identity of the user to specific events that they can attend, and then only grant this user so that they have access to a specific location

6.68.1. Create Secure Application

```
POST /api/{version}/secure/application/create
```

Description

Create a secure application record.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version being used	number(double)	
Query	**accountId** *required*	The unique id of the user making the request	integer(int64)	
Query	**active** *optional*	Active	boolean	`"true"`
Query	**appKey** *required*	The application to secure	string	

Type	Name	Description	Schema	Default
Query	**biometricPosition** *optional*	The position for the biometric file uploaded, options are: - UNKNOWN - LEFT_INDEX - LEFT_MIDDLE - LEFT_RING - LEFT_LITTLE - LEFT_THUMB - RIGHT_INDEX - RIGHT_MIDDLE - RIGHT_RING - RIGHT_LITTLE - RIGHT_THUMB - LEFT - RIGHT	enum (UNKNOWN, LEFT_INDEX, LEFT_MIDDLE, LEFT_RING, LEFT_LITTLE, LEFT_THUMB, RIGHT_INDEX, RIGHT_MIDDLE, RIGHT_RING, RIGHT_LITTLE, RIGHT_THUMB, LEFT, RIGHT)	`"UNKNOWN"`
Query	**biometricPosition2** *optional*	The position for each the biometric2 file uploaded, options are: - UNKNOWN - LEFT_INDEX - LEFT_MIDDLE - LEFT_RING - LEFT_LITTLE - LEFT_THUMB - RIGHT_INDEX - RIGHT_MIDDLE - RIGHT_RING - RIGHT_LITTLE - RIGHT_THUMB - LEFT - RIGHT	enum (UNKNOWN, LEFT_INDEX, LEFT_MIDDLE, LEFT_RING, LEFT_LITTLE, LEFT_THUMB, RIGHT_INDEX, RIGHT_MIDDLE, RIGHT_RING, RIGHT_LITTLE, RIGHT_THUMB, LEFT, RIGHT)	`"UNKNOWN"`
Query	**biometricType** *optional*	The type of biometric, options are: - FINGERPRINT - IRIS - FACIAL	enum (FINGERPRINT, IRIS, FACIAL)	`"FACIAL"`
Query	**keyCert** *required*	Key Certificate	ref	
Query	**password** *required*	Password	string	
Query	**trustStore** *required*	Trust Store	ref	
Query	**username** *required*	Username	string	

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.68.2. Delete Secure Application

```
POST /api/{version}/secure/application/delete
```

Description

Delete a secure application record.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version being used	number(double)
Query	**accountId** *required*	The unique id of the user making the request	integer(int64)
Query	**appKey** *required*	The application to secure	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.68.3. Rest Secure Application

```
POST /api/{version}/secure/application/reset
```

Description

Reset a secure application client.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version being used	number(double)
Query	**accountId** *required*	The unique id of the user making the request	integer(int64)

Type	Name	Description	Schema
Query	**appKey** *required*	The application to secure	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.68.4. Update Secure Application

```
POST /api/{version}/secure/application/update
```

Description

Update a secure application record.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version being used	number(double)
Query	**accountId** *required*	The unique id of the user making the request	integer(int64)
Query	**active** *optional*	Active	boolean
Query	**appKey** *required*	The application to secure	string
Query	**biometricPosition** *optional*	The position for the biometric file uploaded, options are: - UNKNOWN - LEFT_INDEX - LEFT_MIDDLE - LEFT_RING - LEFT_LITTLE - LEFT_THUMB - RIGHT_INDEX - RIGHT_MIDDLE - RIGHT_RING - RIGHT_LITTLE - RIGHT_THUMB - LEFT - RIGHT	enum (UNKNOWN, LEFT_INDEX, LEFT_MIDDLE, LEFT_RING, LEFT_LITTLE, LEFT_THUMB, RIGHT_INDEX, RIGHT_MIDDLE, RIGHT_RING, RIGHT_LITTLE, RIGHT_THUMB, LEFT, RIGHT)

Type	Name	Description	Schema
Query	**biometricPosition2** *optional*	The position for each the biometric2 file uploaded, options are: - UNKNOWN - LEFT_INDEX - LEFT_MIDDLE - LEFT_RING - LEFT_LITTLE - LEFT_THUMB - RIGHT_INDEX - RIGHT_MIDDLE - RIGHT_RING - RIGHT_LITTLE - RIGHT_THUMB - LEFT - RIGHT	enum (UNKNOWN, LEFT_INDEX, LEFT_MIDDLE, LEFT_RING, LEFT_LITTLE, LEFT_THUMB, RIGHT_INDEX, RIGHT_MIDDLE, RIGHT_RING, RIGHT_LITTLE, RIGHT_THUMB, LEFT, RIGHT)
Query	**biometricType** *optional*	The type of biometric, options are: - FINGERPRINT - IRIS - FACIAL	enum (FINGERPRINT, IRIS, FACIAL)
Query	**keyCert** *optional*	Key Certificate	ref
Query	**password** *optional*	Password	string
Query	**trustStore** *optional*	Trust Store	ref
Query	**username** *optional*	Username	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.68.5. Login Clear

```
POST /api/{version}/secure/login
```

Description

Login via Clear.me. Creates a new account if logging in for the first time.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version being used	number(double)	
Query	**ageRestriction** *optional*	Checks user's birthday to see if they meet an age requirement. If the user is under age, an error message is returned.	integer(int32)	`"0"`
Query	**appKey** *required*	The application making the request, defines what type and position is required to make a secure login the request.	string	
Query	**biometricFile** *required*	The data file used to perform authentication	ref	
Query	**biometricFile2** *optional*	The data file used to perform authentication	ref	
Query	**deviceId** *optional*	The unique id of the device making the request	string	
Query	**latitude** *optional*	Used to update the user's current location	number(double)	
Query	**longitude** *optional*	Used to update the user's current location	number(double)	
Query	**responseFilters** *optional*	If returnProfile is set to true, this determines how much of the profile should be returned, see ProfileFilters	string	`"PROFILE"`
Query	**returnProfile** *optional*	Determines whether to return a ProfileResponse on success, or a PaymentTypesResponse	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	ProfileResponse

6.68.6. Purchase Clear

```
POST /api/{version}/secure/purchase
```

Description

Purchase via Clear.me. Creates a new account if purchasing for the first time.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version being used	number(double)
Body	**body** *required*	The payment request object	PaymentRequest

Responses

HTTP Code	Description	Schema
200	successful operation	ProfileResponse

Consumes

- `application/json`

Produces

- `application/json`

6.69. ServiceHub

The service hub represents a service region, this usually means where the vehicle is deployed from. A delivery or ride share organization might many one or many service hubs depending on the type and scope of their business.

Below are some example scenarios as to how these services can be used:

- Retail: A retailer that sells food, clothing, electronics, and home decor may get its products delivered from different locations. The retailer can use service hubs to easily handle multiple shipments from these various shipment centers.
- Medical: Ambulances can be dispatched at the proper service hub based on the location of the emergency.
- Automotive: A taxi service could use service hubs in order to dispatch taxi's in areas that have the highest concentration of potential passengers.

Organizations need a way to determine from which service hub they will be dispatching a vehicle from. Using this service, the organization will be able to quickly and efficiently send the right vehicle to and from the right location, based on the needs of the customer.

Below are great example services that could be combined with this service:

- Notification Service: Send a notification any time a vehicle is dispatched from a service hub.
- Routing Service: After a vehicle is dispatched, use the routing service to find the most efficient route to complete the job.
- Dashboards: Enable or dispable service hubs on the fly based upon availability.

6.69.1. Create Service Hub

```
POST /api/{version}/hub
```

Description

Create new service hub

Parameters

Type	Name	Schema
Path	**version** *required*	number(double)
Body	**body** *optional*	ServiceHub

Responses

HTTP Code	Description	Schema
200	successful operation	ServiceHub

6.69.2. Search Service Hubs

```
GET /api/{version}/hub
```

Description

Search for service hubs.

Parameters

Type	Name	Schema	Default
Path	**version** *required*	number(double)	
Query	**activeOnly** *required*	boolean	`"false"`
Query	**descending** *required*	boolean	`"false"`
Query	**keyword** *optional*	string	
Query	**limit** *required*	integer(int32)	`"20"`
Query	**retailerId** *optional*	integer(int64)	
Query	**sortField** *required*	string	`"name"`
Query	**start** *required*	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< ServiceHub > array

6.69.3. Update Service Hub

```
POST /api/{version}/hub/{id}
```

Description

Update an existing service hub

Parameters

Type	Name	Schema
Path	**id** *required*	integer(int64)

Type	Name	Schema
Path	**version** *required*	number(double)
Body	**body** *optional*	ServiceHub

Responses

HTTP Code	Description	Schema
200	successful operation	ServiceHub

6.69.4. Get Service Hub

```
GET /api/{version}/hub/{id}
```

Description

Get an existing service hub

Parameters

Type	Name	Schema
Path	**id** *required*	integer(int64)
Path	**version** *required*	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	object

6.69.5. Update Service Hub

```
PUT /api/{version}/hub/{id}
```

Description

Update an existing service hub

Parameters

Type	Name	Schema
Path	**id** *required*	integer(int64)
Path	**version** *required*	number(double)
Body	**body** *optional*	ServiceHub

Responses

HTTP Code	Description	Schema
200	successful operation	ServiceHub

6.69.6. Delete Service Hub

```
DELETE /api/{version}/hub/{id}
```

Description

Delete an existing service hub

Parameters

Type	Name	Schema
Path	**id** *required*	integer(int64)
Path	**version** *required*	number(double)

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

6.70. Shipment

A shipment represents a package or passenger on a route. A shipment has a pickup stop and a drop off stop. A shipment can be assigned to a route, and the driver assigned to that route will see that it has been added and handle the shipment when they start working that route.

Below are some example scenarios as to how these services can be used:

- Retail: A retailer could receive multiple shipments in a week. Using this service, they can know exactly what type of package is coming and prepare to handle it effectively when it arrives.
- Delivery: A delivery service can use Shipments to keep an organized list of what needs to be delivered, what is en route, and what has already been delivered for that day.
- Real Estate: A condo can use Shipments to determine what type of package is being delivered, and whether or not it is for a resident or for the condo itself.

Shipments are a very simple way to show what is being delivered. Both the sender and recipient will know what type of package to expect, and when to expect it.

Below are great example services that could be combined with this service:

- Notification Service: Send a notification when a certain type of shipment is being delivered, perhaps one that requires some preparation before it arrives.
- Analytics Service: Record what types of shipments are delivered more than others, or record which shipment types tend to have problems or issues.
- Scheduled Task Service: Automatically create a recurring shipment, to reduce the manual workload.

6.70.1. Create Shipment

```
POST /api/{version}/shipment
```

Description

Create new shipment

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version	number(double)
Body	**body** *optional*		Shipment

Responses

HTTP Code	Description	Schema
200	successful operation	Shipment

6.70.2. Search Shipments

```
GET /api/{version}/shipment
```

Description

Search for shipments

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version	number(double)	
Query	**activeOnly** *required*	Return only active results	boolean	`"true"`
Query	**descending** *required*	Determines whether the sorted list is in descending or ascending order	boolean	`"false"`
Query	**limit** *required*	The limit for pagination	integer(int32)	`"20"`
Query	**ownerId** *optional*	The owner of the shipment	integer(int64)	
Query	**riderId** *optional*	The rider associate to this shipment	integer(int64)	
Query	**routeId** *optional*	The route associate to this shipment	integer(int64)	
Query	**sortField** *required*	The field to sort by	string	`"id"`
Query	**start** *required*	The start index for pagination	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< Shipment > array

6.70.3. Get Shipment

```
GET /api/{version}/shipment/{id}
```

Description

Get an existing shipment

Parameters

Type	Name	Description	Schema
Path	**id** *required*	The requested shipment	integer(int64)
Path	**version** *required*	The API version	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	Shipment

6.70.4. Update Shipment

```
PUT /api/{version}/shipment/{id}
```

Description

Update an existing shipment

Parameters

Type	Name	Description	Schema
Header	**Application-Key** *required*		string

Type	Name	Description	Schema
Path	**id** *required*	The shipment to update	integer(int64)
Path	**version** *required*	The API version	number(double)
Body	**body** *optional*		Shipment

Responses

HTTP Code	Description	Schema
200	successful operation	Shipment

6.70.5. Delete Shipment

```
DELETE /api/{version}/shipment/{id}
```

Description

Delete an existing shipment

Parameters

Type	Name	Description	Schema
Header	**Application-Key** *required*		string
Path	**id** *required*	The shipment to delete	integer(int64)
Path	**version** *required*	The API version	number(double)

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

6.70.6. Cancel Shipment

```
POST /api/{version}/shipment/{id}/cancel
```

Description

Remove shipment from route

Parameters

Type	Name	Description	Schema	Default
Path	**id** *required*	The shipment to delete	integer(int64)	
Path	**version** *required*	The API version	number(double)	
Query	**accountId** *optional*	The account id of the requester	integer(int64)	
Query	**rematch** *required*	Determines whether to rematch shipment with another driver	boolean	`"true"`

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

6.70.7. Uupdate Shipment Status

```
POST /api/{version}/shipment/{id}/status
```

Description

Update status of an existing shipment

Parameters

Type	Name	Description	Schema
Path	**id** *required*	The shipment to update status	integer(int64)

Type	Name	Description	Schema
Path	**version** *required*	The API version	number(double)
Body	**body** *optional*		< string, boolean > map

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

6.71. ShipmentBatch

A Shipment Batch represents an upload of a whole batch of shipments to be routed and optimized. This reduces the effort of handling shipments as they do not have to be uploaded one by one.

Below are some example scenarios as to how these services can be used:

- Retail: A retailer could receive multiple shipments in a week. Using this service, they can know exactly what type of package is coming and prepare to handle it effectively when it arrives. Additionally, they can look at just one shipment batch to learn about the whole set of shipments.
- Delivery: A delivery service can use a Shipment Batch to keep an organized set of shipments to be delivered, en route, or that have already been delivered for that day.
- Real Estate: A condo can use a Shipment Batch to determine what type of packages are being delivered, and whether or not they are for a resident or for the condo itself.

Shipments are a very simple way to show what is being delivered. Both the sender and recipient will know what type of package to expect, and when to expect it.

Below are great example services that could be combined with this service:

- Notification Service: Send a notification when a certain type of shipment batch is being delivered, perhaps one that requires some preparation before it arrives.
- Analytics Service: Record what types of shipment batches are delivered more than others, or record which shipment types tend to have problems or issues.
- Scheduled Task Service: Automatically create a recurring shipment batch, to reduce the manual workload.

6.71.1. Create Shipment Batch

```
POST /api/{version}/shipment/batch
```

Description

Create a new shipment batch

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version	string
Body	**body** *optional*		ShipmentBatch

Responses

HTTP Code	Description	Schema
200	successful operation	ShipmentBatch

6.71.2. Search Shipment Batch

```
GET /api/{version}/shipment/batch
```

Description

Search for shipment batches

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version	number(double)	
Query	**descending** *required*	Determines whether the sorted list is in descending or ascending order	boolean	`"false"`
Query	**hubId** *required*	The associated service hub	integer(int64)	
Query	**limit** *required*	The limit for pagination	integer(int32)	`"20"`

Type	Name	Description	Schema	Default
Query	**sortField** *required*	The field to sort by	string	`"id"`
Query	**start** *required*	The start index for pagination	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< ShipmentBatch > array

6.71.3. Get Shipment Batch

```
GET /api/{version}/shipment/batch/{batchId}
```

Description

Get an existing shipment batch

Parameters

Type	Name	Description	Schema
Path	**batchId** *required*	The id of the requested shipment batch	integer(int64)
Path	**version** *required*	The API version	string

Responses

HTTP Code	Description	Schema
200	successful operation	ShipmentBatch

6.71.4. Delete Shipment Batch

```
DELETE /api/{version}/shipment/batch/{batchId}
```

Description

Search for shipment batches

Parameters

Type	Name	Description	Schema
Path	**batchId** *required*	The id of the shipment batch to delete	integer(int64)
Path	**version** *required*	The API version	string

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

6.71.5. Get Shipment Batch Status

```
GET /api/{version}/shipment/batch/{batchId}/status
```

Description

Get the import status list of the import shipment batch

Parameters

Type	Name	Description	Schema	Default
Path	**batchId** *required*	The id of the requested shipment batch	integer(int64)	
Path	**version** *required*	The API version	number(double)	
Query	**accountId** *required*	The account id of the requester	integer(int64)	
Query	**completed** *optional*	Completed import status only	boolean	
Query	**descending** *required*	Determines whether the sorted list is in descending or ascending order	boolean	`"false"`

Type	Name	Description	Schema	Default
Query	**hasRoute** *optional*	Has route associate to the status	boolean	
Query	**hasShipment** *optional*	Has shipment associate to the status	boolean	
Query	**keyword** *optional*	The keyword to search for	string	
Query	**limit** *required*	The limit for pagination	integer(int32)	`"20"`
Query	**sortField** *required*	The field to sort by	string	`"id"`
Query	**start** *required*	The start index for pagination	integer(int32)	`"0"`
Query	**started** *optional*	Started import status only	boolean	
Query	**valid** *optional*	Valid import status only or not	boolean	

Responses

HTTP Code	Description	Schema
200	successful operation	< ShipmentImportStatus > array

6.72. Simulation

The Routing Simulation service takes a set of resources like vehicles, drivers, and passengers, and uses the Sirqul routing algorithm to generate the most optimal recommendations on how passengers or objects should be delivered or fulfilled. The best scenario to use this tool is when you are running a business that involves hundreds or thousands of shipments that need to be delivered the same day they're requested. The simulation can be used to determine which shipments should be assigned to available drivers and vehicles. You can generate hundreds to thousands of vehicles from a single hub, to a city, to even the entire United States. The simulator makes it easy to learn how you plan to scale your business and understanding the risks and costs involved before actually committing to shipment deliveries.

Below are some example scenarios as to how these services can be used:

- There is a 1,000 shipment order that is requested, and each shipment has a different time window

delivery requirement, different drop off location, and there are mixed cargo types. The routing simulator allows you to upload the shipments and the algorithm will take the required time windows, different drop off locations, and mixed cargo types, and generate suggestions on how the shipments should be added onto routes, and assigned to the appropriate drivers and vehicles

- Use the Routing simulator when you are trying to optimize the floorplan of your warehouse. If you have machine that is used to help move around cargo inside of a building, you can use the simulator to determine how shipments should be moved based upon their delivery requirement, weight, size, shape, driver/vehicle assigned, and more.

The Routing Simulation service is important as it allows your businesses to plan and scale your business without having to do any guess work. It allows you to load up many vehicles, drivers, and passengers/objects needing to be picked up/dropped off into a simulator, and will be able to get suggestions on how the passengers/objects should be routed.

Other services that work great with the simulator is Audience (Geofence) service, which allows you to designated specific locations to determine a clustered drop off area for one to many passengers/objects. We typically call these 'Zones'. Another great service to pair with the routing simulator is the Assignments (Tasks) service, as a route suggested by the simulator could be assigned to a driver as a 'task' that needs to be completed.

6.72.1. Routing Simulation

```
POST /api/{version}/simulation/routing
```

Description

Simulates routing requests.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**data** *required*	data	string	
Query	**realTime** *required*	realTime	boolean	`"true"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.73. Stop

Stops are what make up a route. Stops can also be associated to a shipment, that shipment has a pickup stop and a drop off stop. A route can have one or many stops. This service allows you to get or update a stop.

Below are some example scenarios as to how these services can be used:

- Retail: A warehouse ships goods to multiple retailers. Each retailer that has a delivery would be considered a stop on that route.
- Automotive: In a ride share scenario, each person that is picked up and dropped off would have two stops - one for pickup and one for drop off.
- Real Estate: When showing houses for sale along a pre-optimized route, each house the group parks at to view would be considered a stop.

Stops are the core of any transportation based route. They handle the pickup and drop off of people, items, and more.

Below are great example services that could be combined with this service:

- Notification Service: Send a notification when a stop has been completed (passenger dropped off, package delivered, etc).
- Analytics Service: Track which stops or stop types are being used more that others for later optimization.
- Dashboards: Add or remove stops from a route on the fly, driver's would see this update immediately.

6.73.1. Get Stop

```
GET /api/{version}/stop/{id}
```

Description

Get an existing stop

Parameters

Type	Name	Description	Schema
Path	**id** *required*	The id of the requested stop	integer(int64)
Path	**version** *required*	The API version	string

Responses

HTTP Code	Description	Schema
200	successful operation	Stop

6.73.2. Update Stop

```
PUT /api/{version}/stop/{id}
```

Description

Update an existing stop

Parameters

Type	Name	Description	Schema
Path	**id** *required*	The id of the stop to update	integer(int64)
Path	**version** *required*	The API version	string
Body	**body** *optional*		Stop

Responses

HTTP Code	Description	Schema
200	successful operation	Stop

6.74. StopLocation

The Location V2 service is used to create and modify both public and private locations that can be used for pickup and drop off waypoints. Using this, consumers can create locations to be used for ride

sharing. A user creating a Location to go to the mall would use a public location, but a user creating a location to get to their house would use a private location.

Below are some example scenarios as to how these services can be used:

- Automotive: A consumer or driver could request a mobile technician to fix something on their vehicle. They would create a custom private location for the technician to drive to and assist the caller.
- Medical: A person calls for an ambulance, the hospital can take their location and automatically create a private Location in their system for data, route optimization, etc.
- Robotics: A robot could create a Location to let its operator know of an issue in that area. For instance, a chemical spill.

Using Locations, both customer and service providers can optimize a program to work for them. Both sides are able to create public and private locations.

Below are great example services that could be combined with this service:

- Notification Service: Send a notification to the program admin when a new location is created.
- Analytic Service: Track where the majority of locations are being created.
- Dashboards: Easily create, update, or delete a location on the dashboard.

6.74.1. Create Location

```
POST /api/{version}/location
```

Description

Create new stop location

Parameters

Type	Name	Schema
Path	**version** *required*	string
Body	**body** *optional*	Location

Responses

HTTP Code	Description	Schema
200	successful operation	Location

6.74.2. Update Location

```
PUT /api/{version}/location/{id}
```

Description

Update an existing stop location

Parameters

Type	Name	Schema
Path	**id** *required*	integer(int64)
Path	**version** *required*	string
Body	**body** *optional*	Location

Responses

HTTP Code	Description	Schema
200	successful operation	Location

6.75. Subscription

The Subscriptions service allows you to manage and customize recurring billable events to fit your business needs. Whether it is managing subscriptions for monthly charges or yearly memberships, use our services in conjunction with our billing services to ensure that recurring payments are made on time.

Below are some example scenarios as to how these services can be used:

- Retail: Setup and manage yearly memberships for added discounts, expedited shipping bonuses, and other incentives for shoppers that use the mobile app.
- Medical: Setup a recurring billing plan for medical expenses that give users the ability to make reasonable payments on time.

- Real Estate: Setup a recurring billing plan for utilities and other monthly expenses to plan out a budget. Have the budget be automatically modified based upon trends and overall usage.
- Automotive: Setup and manage yearly memberships for recurring, routine maintenance for vehicles. Offer discounts and other perks to get users to subscribe.

The benefit of using Sirqul's subscription services is so that you can seamlessly integrate it into the rest of Sirqul's platform and services. Have everything work hand-in-hand with no speed bumps or needs to work with other third parties.

Below are great example services that could be combined with this service:

- Offer services: Setup subscription plans via offers. Use this as a gateway of what services are offered for the subscription plan.
- Billing services: The other have of subscription management where charges are made.
- Tickets services: Instead of using a credit card on file, use digital currency equivalents to pay for the subscription.

6.75.1. Create Subscription

```
POST /api/{version}/subscription/create
```

Description

Create a subscription for a billable entity. Provide a planId, if not provided then the base plan will be assigned.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**accountId** *required*	The account used to perform the create, must be the responsible manager	integer(int64)
Query	**metaData** *optional*	Set metaData	string
Query	**planId** *optional*	The plan to subscribe to, if null use default plan	integer(int64)
Query	**promoCode** *optional*	Set a promo code for a discount.	string

Responses

HTTP Code	Description	Schema
200	successful operation	SubscriptionResponse

6.75.2. Delete Subscription

```
POST /api/{version}/subscription/delete
```

Description

Suspend the current subscription for the billable entity managed by the account. The account must be the responsible manager to perform this action

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**accountId** *required*	The account used to perform the delete, must be the responsible manager	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.75.3. Get Subscription

```
POST /api/{version}/subscription/get
```

Description

Use the accountId to determine the associated BillableEntity. Then get the subscription.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**accountId** *required*	The account used to perform the lookup	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SubscriptionResponse

6.75.4. Get Subscription Plan

```
POST /api/{version}/subscription/plan/get
```

Description

Get the matched subscription plan

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**planId** *required*	The ID of the plan to get	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SubscriptionPlanResponse

6.75.5. List Subscription Plans

```
POST /api/{version}/subscription/plan/list
```

Description

Get the matched subscription plan

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**role** *optional*	The role the plan is targeted for, values are: DEVELOPER, RETAILER, ADVERTISER	string
Query	**visible** *optional*	Include visible only (true), hidden only (false), or all (null)	boolean

Responses

HTTP Code	Description	Schema
200	successful operation	< SubscriptionPlanResponse > array

6.75.6. Update Subscription

```
POST /api/{version}/subscription/update
```

Description

Updates the subscription for the billable entity for an account

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**accountId** *required*	The account used to perform the update, must be the responsible manager	integer(int64)
Query	**active** *optional*	Set active status	boolean
Query	**metaData** *optional*	Set metaData.	string

Type	Name	Description	Schema
Query	**planId** *optional*	The plan to subscribe to	integer(int64)
Query	**promoCode** *optional*	Set a promo code for a discount.	string

Responses

HTTP Code	Description	Schema
200	successful operation	SubscriptionResponse

6.75.7. Get Subscription Usage

```
POST /api/{version}/subscription/usage/get
```

Description

Use the accountId to determine the associated BillableEntity. Then get the application usage.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version to use to make the request	number(double)
Query	**accountId** *required*	The account used to perform the lookup	integer(int64)
Query	**applicationId** *optional*	Get for just 1 application instead of the BillableEntity	integer(int64)
Query	**end** *optional*	The end time frame	integer(int64)
Query	**start** *optional*	The start time frame	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	ApplicationUsageResponse

6.76. Task

The scheduled tasks service is a service that enables to you set specific actions and trigger them based upon certain times and actions. It allows you to create 'recipes' that can automate workflows and processes. You can even create and customize recipes to occur every N timeframe that you desire (e.g. minutes, days, weeks, months, etc.). This service is great for people who are looking increase overall efficiency and effectiveness of their day to day activities.

Below are some example scenarios as to how these services can be used:

- Retail: Schedule a notification to be sent every morning of daily tasks and assignments for the day.
- Automotive: Schedule a notification when the 30k, 60k, and 90k milestones are coming up so that the driver can schedule and budget their finances appropriately for the upcoming maintenance cost.
- Smart Home: Have the coffee pot brew every morning at 7am and configure this from your smart device.
- Healthcare: Schedule autonomous delivery carts that can delivery prescriptions to patients rooms at specific times in the day.

This service is primarily used to make manual processes more efficient and effective. Once the recipes are set, there is no need to change anything unless an adjustment needs to be made. The recipes are managed via web dashboard that and can be saved and managed at any point in time.

Below are great example services that could be combined with this service:

- Notifications service: Set up the task to send yourself a push notification when some specific action happens (e.g. reminders).
- Album service: Set up a security camera to snap a video when it detects motion and save it to an album that can be reviewed later.
- Offers service: Set up a scheduled task to generate an offer when someone who matches an Audience enters the geofence.
- Missions (tasks) service: Once a user starts and completes a mission, queue up the next mission that is on the list for the user and automatically notify the mission creator that they are moving on to the next mission.

6.76.1. Create Task

```
POST /api/{version}/task/create
```

Description

Create a Task

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The api version to use.	number(double)	
Query	**accountId** *required*	The logged in user.	integer(int64)	
Query	**active** *optional*	Sets whether the Task is active or not (inactive Tasks are not processed)	boolean	`"true"`
Query	**appKey** *optional*	The application to target	string	
Query	**cronExpression** *optional*	The cron expression that represents the task's schedule	string	
Query	**cronType** *optional*	cronType	string	
Query	**endDate** *optional*	The ending date of the task	integer(int64)	
Query	**endpointURL** *optional*	The URL for making an HTTP call	string	
Query	**groupingId** *optional*	Client defined identifier for grouping tasks	string	
Query	**metaData** *optional*	Set metaData	string	
Query	**name** *required*	The name of the task	string	
Query	**payload** *optional*	The parameters for making an HTTP call	string	
Query	**scheduledDate** *optional*	The date and time of the task	integer(int64)	

Type	Name	Description	Schema	Default
Query	**startDate** *optional*	The starting date of the task	integer(int64)	
Query	**visibility** *optional*	The determines the scope of who is able to find and view the scheduled notification (PUBLIC - openly available to all Sirqul users, PRIVATE - only available to users that have been invited)	enum (PUBLIC, PRIVATE, FRIENDS)	

Responses

HTTP Code	Description	Schema
200	successful operation	TaskResponse

6.76.2. Delete Task

```
POST /api/{version}/task/delete
```

Description

Delete a Task

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The api version to use.	number(double)
Query	**accountId** *required*	The logged in user.	integer(int64)
Query	**taskId** *required*	The id of the Task to delete.	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.76.3. Get Task

```
POST /api/{version}/task/get
```

Description

Get a Task

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The api version to use.	number(double)
Query	**accountId** *required*	The logged in user.	integer(int64)
Query	**taskId** *required*	The id of the Task to return.	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	TaskResponse

6.76.4. Search Tasks

```
POST /api/{version}/task/search
```

Description

Search on Tasks

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The api version to use.	number(double)	
Query	**accountId** *required*	The logged in user.	integer(int64)	

Type	Name	Description	Schema	Default
Query	**activeOnly** *optional*	Determines whether to return only active results	boolean	`"true"`
Query	**appKey** *optional*	Filter the list by a specific application	string	
Query	**descending** *optional*	Determines whether the sorted list is in descending or ascending order	boolean	`"true"`
Query	**filter** *optional*	A comma separated list of filters: - MINE - Return tasks that the user has created. - SHARED - Return tasks that have been shared to the user via com.sirqul.geoapps.common.service.UserPermissionsApi#addUsersToPermissionable addUsersToPermissionable. - FOLLOWER - Return tasks that have been created by the user's followers (the content needs to have been APPROVED or FEATURED). - FOLLOWING - Return tasks that have been created by people who the user is following (the content needs to have been APPROVED or FEATURED). - PUBLIC - Return all PUBLIC tasks that have been APPROVED or FEATURED. - ALL_PUBLIC - Return all PUBLIC tasks regardless of whether they are approved or not (ignores the approval status). - LIKED - Return all tasks that the user has liked. - FEATURED - Return all tasks that have been featured. - PENDING - Return all pending tasks.	string	`"MINE"`
Query	**groupingId** *optional*	Filter results by a grouping identifier defined by the client	string	
Query	**keyword** *optional*	Keyword search on the task names.	string	
Query	**limit** *optional*	Limit the result to some number.	integer(int32)	`"20"`
Query	**sortField** *optional*	The field to sort by. Possible values include: {ID, CREATED, UPDATED, ACTIVE, NAME, SCHEDULED_DATE, START_DATE, END_DATE	string	`"CREATED"`
Query	**start** *optional*	Start the result set at some index.	integer(int32)	`"0"`

Type	Name	Description	Schema	Default
Query	**statuses** *optional*	Filter results by status (comma separated list). Values include: NEW, ERROR, COMPLETE, PROCESSING, TEMPLATE	string	`"NEW,ERROR,COMPLETE,PROCESSING"`
Query	**templateTypes** *optional*	Template Types	string	

Responses

HTTP Code	Description	Schema
200	successful operation	< TaskResponse > array

6.76.5. Update Task

```
POST /api/{version}/task/update
```

Description

Update a Task

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The api version to use.	number(double)
Query	**accountId** *required*	The logged in user.	integer(int64)
Query	**active** *optional*	Sets whether the Task is active or not (inactive Tasks are not processed)	boolean
Query	**appKey** *optional*	The application to target	string
Query	**cronExpression** *optional*	The cron expression that represents the task's schedule	string
Query	**cronType** *optional*	cronType	string

Type	Name	Description	Schema
Query	**endDate** *optional*	The ending date of the task	integer(int64)
Query	**endpointURL** *optional*	The URL for making an HTTP call	string
Query	**groupingId** *optional*	Client defined identifier for grouping tasks	string
Query	**metaData** *optional*	Set metaData	string
Query	**name** *optional*	The name of the task	string
Query	**payload** *optional*	The parameters for making an HTTP call	string
Query	**scheduledDate** *optional*	The date and time of the task	integer(int64)
Query	**startDate** *optional*	The starting date of the task	integer(int64)
Query	**taskId** *required*	Task Id	integer(int64)
Query	**visibility** *optional*	The determines the scope of who is able to find and view the scheduled notification (PUBLIC - openly available to all Sirqul users, PRIVATE - only available to users that have been invited)	enum (PUBLIC, PRIVATE, FRIENDS)

Responses

HTTP Code	Description	Schema
200	successful operation	TaskResponse

6.77. Territory

Territories are defined by a list of postal codes without a hierarchy. They are considered “areas” that are managed by an account. The account could manage the territories for any purpose.

This service would primarily be used by managers who manage teams who manage defined territories. For example, think of a delivery service that has a fleet of vehicles that need to delivery packages that meet their ETA’s and quotas. One vehicle is not going to service all areas, this would be inefficient and costly. Imagine multiple vehicles covering multiple territories. This would get the job

done more efficiently and may even allow the business to grow and expand more than anticipated.

Below are some example scenarios as to how these services can be used:

- Retail: Sales representatives could be managing specific retail outlets in a defined territory.
- Medical: Medical delivery services manage and deliver supplies to specific hospitals.
- Real Estate: Agents and brokers could cover specific territories and be considered “recommended” agents or brokers to customers.
- Automotive: Automated delivery cars could be managed to delivery and service specific territories in an area.

This service is simply used to manage areas that are comprised of postal codes.

Below are great example services that could be combined with this service:

- Routing service: Use the routing service to factor in the most efficient route to deliver items to and from N locations.
- Tracking service: Use the tracking service to ensure that deliveries are made appropriately to and from N locations. Send an alert if ETA’s are missed by alot or if driers are veering off course.

6.77.1. Create Territory

```
POST /api/{version}/territory/create
```

Description

Creates a territory.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	`"true"`
Query	**name** *required*	name	string	

Responses

HTTP Code	Description	Schema
200	successful operation	TerritoryResponse

6.77.2. Delete Territory

```
POST /api/{version}/territory/delete
```

Description

Deletes a territory.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**territoryId** *required*	territoryId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.77.3. Get Territory

```
POST /api/{version}/territory/get
```

Description

Get a territory.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**territoryId** *required*	territoryId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	TerritoryResponse

6.77.4. Search Territories

```
POST /api/{version}/territory/search
```

Description

Searches on territories.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**descending** *required*	descending	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**limit** *optional*	limit	integer(int32)	`"20"`
Query	**sortField** *required*	sortField	enum (ID, UPDATED, CREATED, NAME)	`"id"`
Query	**start** *optional*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< TerritoryResponse > array

Territories are defined by a list of postal codes without a hierarchy. They are considered “areas” that are managed by an account. The account could manage the territories for any purpose.

This service would primarily be used by managers who manage teams who manage defined territories. For example, think of a delivery service that has a fleet of vehicles that need to delivery packages that meet their ETA’s and quotas. One vehicle is not going to service all areas, this would be inefficient and costly. Imagine multiple vehicles covering multiple territories. This would get the job done more efficiently and may even allow the business to grow and expand more than anticipated.

Below are some example scenarios as to how these services can be used:

- Retail: Sales representatives could be managing specific retail outlets in a defined territory.
- Medical: Medical delivery services manage and deliver supplies to specific hospitals.
- Real Estate: Agents and brokers could cover specific territories and be considered “recommended” agents or brokers to customers.
- Automotive: Automated delivery cars could be managed to delivery and service specific territories in an area.

This service is simply used to manage areas that are comprised of postal codes.

Below are great example services that could be combined with this service:

- Routing service: Use the routing service to factor in the most efficient route to deliver items to and from N locations.
- Tracking service: Use the tracking service to ensure that deliveries are made appropriately to and from N locations. Send an alert if ETA’s are missed by alot or if driers are veering off course.

6.77.5. Update Territory

```
POST /api/{version}/territory/update
```

Description

Updates a territory.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**active** *optional*	active	boolean
Query	**name** *optional*	name	string
Query	**territoryId** *required*	territoryId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	TerritoryResponse

6.78. ThemeDescriptor

The Theme service allows you to make periodic updates to your application without any software changes or server deployments. This is extremely useful for doing promotions and advertisements during events or holidays. For example, if you have a game application and you want to make graphical changes to match the holiday season, you can schedule an update using the Theme service by uploading graphics, messages, and different icons that will get applied to your main menu. You can also schedule different themes within game levels that changes the look and feel for objects, character designs, and rewards as the user progresses through the story.

Below are some example scenarios as to how these services can be used:

- There is a Kiosk on display inside of a mall. The content, images, and graphics on the display screen will change periodically. These have been pre-scheduled to update dynamically without an admin having to manually configure the updates
- For the on-screen dash inside of a vehicle, if the dash is connected to the internet, it can periodically update the menu graphics view to improve the user experience

The Theme Service is important because it allows you to update images, text, media files, etc. without any need for software changes or server deployments. These updates can be scheduled in advance so that the user does not have to manually make these modifications at the time they need these changes deployed. This type of service helps retain users by providing a pleasing experience and reminds users to revisit an application if they have not used the application for a while.

Below are example services that could be combined with the Theme Service:

- Scheduled Tasks and Assets services are great to use with the Theme service because an admin can schedule every single theme update for the entire year in advance so that they don't have to make the updates manually, and upload the images and videos using Assets that will be the graphical changes for the theme

6.78.1. Create/Update Theme

```
POST /api/{version}/consumer/theme
```

Description

Creates or updates a theme descriptor that can be used to give applications a customized look and feel. The theme can be created by consumers and shared to other users, allowing them to use and/or collaborate on making the theme.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appVersion** *optional*	appVersion	string	
Query	**bgImage** *optional*	bgImage	ref	
Query	**bgSound** *optional*	bgSound	ref	
Query	**colorValueJson** *optional*	colorValueJson	string	
Query	**completeWithDefaultValues** *required*	completeWithDefaultValues	boolean	`"0"`
Query	**connectionGroupIdsToAdd** *optional*	connectionGroupIdsToAdd	string	

Type	Name	Description	Schema	Default
Query	**connectionIdsToAdd** *optional*	connectionIdsToAdd	string	
Query	**customJsonObjects** *optional*	customJsonObjects	string	
Query	**description** *optional*	description	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**gameType** *optional*	gameType	string	
Query	**iconImage** *optional*	iconImage	ref	
Query	**includeFriendGroup** *required*	includeFriendGroup	boolean	`"0"`
Query	**latitude** *optional*	latitude	number(double)	
Query	**locationDescription** *optional*	locationDescription	string	
Query	**longitude** *optional*	longitude	number(double)	
Query	**musicSelection** *optional*	musicSelection	string	
Query	**publicAdd** *required*	publicAdd	boolean	`"0"`
Query	**publicDelete** *required*	publicDelete	boolean	`"0"`
Query	**publicRead** *required*	publicRead	boolean	`"1"`
Query	**publicWrite** *required*	publicWrite	boolean	`"0"`

Type	Name	Description	Schema	Default
Query	**sceneAtlasImage** *optional*	sceneAtlasImage	ref	
Query	**stringReplacerJson** *optional*	stringReplacerJson	string	
Query	**themeDescriptorId** *optional*	themeDescriptorId	integer(int64)	
Query	**title** *optional*	title	string	
Query	**visibility** *required*	visibility	enum (PUBLIC, PRIVATE, FRIENDS)	`"PRIVATE"`

Responses

HTTP Code	Description	Schema
200	successful operation	ThemeDescriptorResponse

6.78.2. Get Theme

```
POST /api/{version}/consumer/theme/get
```

Description

Gets a theme.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**deviceId** *optional*	deviceId	string

Type	Name	Description	Schema
Query	**gameType** *optional*	gameType	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)
Query	**themeDescriptorId** *required*	themeDescriptorId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	PurchaseItemListResponse

6.78.3. Delete Theme

```
POST /api/{version}/consumer/theme/remove
```

Description

Removes a theme.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**gameType** *optional*	gameType	string
Query	**latitude** *optional*	latitude	number(double)

Type	Name	Description	Schema
Query	**longitude** *optional*	longitude	number(double)
Query	**themeDescriptorId** *required*	themeDescriptorId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.78.4. Search Themes

```
POST /api/{version}/consumer/theme/search
```

Description

Searches for themes.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**_i** *optional*	_i	integer(int32)	
Query	**_l** *optional*	_l	integer(int32)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appVersion** *optional*	appVersion	string	
Query	**contestType** *optional*	contestType	string	
Query	**dateCreated** *optional*	dateCreated	integer(int64)	

Type	Name	Description	Schema	Default
Query	**descending** *required*	descending	boolean	`"true"`
Query	**deviceId** *optional*	deviceId	string	
Query	**filter** *required*	filter	string	`"MINE"`
Query	**gameType** *optional*	gameType	string	
Query	**keyword** *optional*	keyword	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**limit** *required*	limit	integer(int32)	`"0"`
Query	**longitude** *optional*	longitude	number(double)	
Query	**ownerId** *optional*	ownerId	integer(int64)	
Query	**q** *optional*	q	string	
Query	**sortField** *required*	sortField	string	`"CREATED"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	PurchaseItemListResponse

6.79. ThirdPartyCredentials

Authenticate users in Sirqul by leveraging a variety of third party credentials. Whether it is through Facebook, Twitter, or even your own social network, Sirqul makes authentication via a third party seamless.

This service is great for those who want to support users who authenticate through popular social networks like Facebook and Twitter. But it is also great for those want to integrate their own social network into Sirqul authentication.

Below are some example scenarios as to how these services can be used:

- Retail: Existing retail stores have their own corporate authentication system and want to leverage this information to create an account on the stack running Sirqul
- Medical: Have a custom healthcare authentication system that is specific to surgeons, nurses, etc.
- Automotive: Integrate a custom authentication system that is unique to the user who owns the vehicle. Only allow certain actions and events occur only if the driver's profile matches what was originally set in the beginning.

This service is great if you want users to authenticate via an existing popular social network, or if you have your own authentication system that can be used on the Sirqul stack.

6.79.1. Create Credential

```
POST /api/{version}/thirdparty/credential/create
```

Description

This endpoint creates a third-party login for a Sirqul account. A third party login is a way for external systems (Third Party Networks) to link their own user accounts with a Sirqul account.

The thirdPartyId parameter is used to determine if the user already exists in Sirqul or not. This parameter needs to be unique for each user in the Third Party Network (identified by the networkUID parameter). Note that subsequent calls will update the user's third-party login credentials for the user with the same thirdPartyId and networkUID combination.

The thirdPartyToken parameter acts as a shared secret and used by client applications to log users into Sirqul without providing a Sirqul username and password.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *required*	appKey	string	

Type	Name	Description	Schema	Default
Query	**deviceId** *optional*	deviceId	string	
Query	**emailAddress** *optional*	emailAddress	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**longitude** *optional*	longitude	number(double)	
Query	**metaData** *optional*	metaData	string	
Query	**networkUID** *required*	networkUID	string	
Query	**responseFilters** *required*	responseFilters	string	`"PROFILE"`
Query	**sessionId** *optional*	sessionId	string	
Query	**signinOnlyMode** *required*	signinOnlyMode	boolean	`"false"`
Query	**thirdPartyId** *required*	thirdPartyId	string	
Query	**thirdPartyName** *optional*	thirdPartyName	string	
Query	**thirdPartyRefreshToken** *optional*	thirdPartyRefreshToken	string	
Query	**thirdPartyToken** *required*	thirdPartyToken	string	

Responses

HTTP Code	Description	Schema
200	successful operation	ProfileResponse

6.79.2. Delete Credential

```
POST /api/{version}/thirdparty/credential/delete
```

Description

Delete a third party network on a Sirqul account.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *required*		string
Query	**networkUID** *required*		string
Query	**thirdPartyId** *required*	thirdPartyId	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.79.3. Get Credential

```
POST /api/{version}/thirdparty/credential/get
```

Description

Gets the account information given a third party token.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *required*	appKey	string	
Query	**createNewAccount** *required*	createNewAccount	boolean	`"false"`
Query	**deviceId** *optional*	deviceId	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**longitude** *optional*	longitude	number(double)	
Query	**networkUID** *required*	networkUID	string	
Query	**referralAccountId** *optional*	referralAccountId	integer(int64)	
Query	**responseFilters** *required*	responseFilters	string	`"PROFILE"`
Query	**sessionId** *optional*	sessionId	string	
Query	**thirdPartyCredentialId** *optional*	thirdPartyCredentialId	integer(int64)	
Query	**thirdPartySecret** *optional*	thirdPartySecret	string	
Query	**thirdPartyToken** *optional*	thirdPartyToken	string	

Responses

HTTP Code	Description	Schema
200	successful operation	ProfileResponse

6.79.4. POST /api/{version}/thirdparty/credential/mfa/send

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**appKey** *required*	appKey	string
Query	**networkUID** *required*	networkUID	string
Query	**thirdPartyCredentialId** *optional*	thirdPartyCredentialId	integer(int64)
Query	**thirdPartyToken** *optional*	thirdPartyToken	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.79.5. Search Credentials

```
POST /api/{version}/thirdparty/credential/search
```

Description

Search on a user's linked third party networks.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**descending** *required*	descending	boolean	`"true"`
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**networkUID** *optional*	networkUID	string	
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< ThirdPartyCredentialResponse > array

6.79.6. Update Credential

```
POST /api/{version}/thirdparty/credential/update
```

Description

Updates a third-party login for an account.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**appKey** *required*	appKey	string	

Type	Name	Description	Schema	Default
Query	**deviceId** *optional*	deviceId	string	
Query	**metaData** *optional*	metaData	string	
Query	**networkUID** *required*	networkUID	string	
Query	**responseFilters** *required*	responseFilters	string	`"PROFILE"`
Query	**thirdPartyId** *required*	thirdPartyId	string	
Query	**thirdPartyName** *optional*	thirdPartyName	string	
Query	**thirdPartyRefreshToken** *optional*	thirdPartyRefreshToken	string	
Query	**thirdPartyToken** *optional*	thirdPartyToken	string	

Responses

HTTP Code	Description	Schema
200	successful operation	ProfileResponse

6.79.7. Create Network

```
POST /api/{version}/thirdparty/network/create
```

Description

Creates a custom third party network.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**description** *optional*	description	string	
Query	**enableIntrospection** *required*	enableIntrospection	boolean	`"true"`
Query	**introspectionParams** *optional*	introspectionParams	string	
Query	**introspectionURL** *optional*	introspectionURL	string	
Query	**name** *required*		string	
Query	**requiredRootField** *optional*	requiredRootField	string	
Body	**body** *optional*	introspectionMethod	string	

Responses

HTTP Code	Description	Schema
200	successful operation	ThirdPartyNetworkResponse

6.79.8. Delete Network

```
POST /api/{version}/thirdparty/network/delete
```

Description

Marks a custom third party network as deleted. Only the network owners and managers have access to this.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**networkUID** *required*		string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.79.9. Get Network

```
POST /api/{version}/thirdparty/network/get
```

Description

Get the details of a third party network. Only the network owners and managers have access to this.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**networkUID** *required*		string

Responses

HTTP Code	Description	Schema
200	successful operation	ThirdPartyNetworkResponse

6.79.10. Search Networks

```
POST /api/{version}/thirdparty/network/search
```

Description

Search on supported third party networks and custom networks from external users.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activeOnly** *required*	activeOnly	boolean	`"true"`
Query	**descending** *required*	descending	boolean	`"false"`
Query	**filterBillable** *optional*	filterBillable	boolean	
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**sortField** *required*	sortField	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, BILLABLE_ENTITY_ID, BILLABLE_ENTITY_NAME, RESPONSIBLE_DISPLAY, NAME, DESCRIPTION)	`"CREATED"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< ThirdPartyNetworkShortResponse > array

6.79.11. Update Network

```
POST /api/{version}/thirdparty/network/update
```

Description

Updates a custom third party network. Only the network owners and managers have access to this.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**description** *optional*	description	string
Query	**enableIntrospection** *optional*	enableIntrospection	boolean
Query	**introspectionParams** *optional*	introspectionParams	string
Query	**introspectionURL** *optional*	introspectionURL	string
Query	**name** *optional*	name	string
Query	**networkUID** *required*		string

Type	Name	Description	Schema
Query	**requiredRootField** *optional*	requiredRootField	string
Body	**body** *optional*	introspectionMethod	string

Responses

HTTP Code	Description	Schema
200	successful operation	ThirdPartyNetworkResponse

6.79.12. Search User Credentials by Network

```
POST /api/{version}/thirdparty/network/user/search
```

Description

Search users by third party networks.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**cellPhone** *optional*	cellPhone	string	
Query	**emailAddress** *optional*	emailAddress	string	
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**networkUID** *required*	networkUID	string	

Type	Name	Description	Schema	Default
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< ThirdPartyCredentialResponse > array

6.80. Ticket

The Tickets service is a way to represent digital currency in the Sirqul ecosystem. Tickets can be called whatever you wish for it to be called such as coins, bucks, etc. Tickets are used to purchase offers (products, vouchers, and packs) which can be redeemed and used at a later point in time.

Tickets can be earned via in-app purchases or rewarded for completing tasks. You get to define the real world dollar to tickets ratio in your ecosystem.

All transactions that occur with tickets are recorded so that you can build reports on everything that happens in your ecosystem to determine the next steps in improving your business needs.

Below are some example scenarios as to how these services can be used:

- Retail: Can be used as a form of digital currency that is used to purchase a digital offer that can be redeemed for a physical product at a retail location.
- Medical: Can be used as a form of credits that are used when billed for medical services.
- Automotive: Can be used as a form of credits that are used when parking at a smart garage. Automated deposits, etc.
- Robotics: Can be used as a form of credits that are used when needing to rent or use robotics services.

Use tickets to generate your own organic ecosystem of value for your users. Whether they are used as credits for services or used as a way of means to purchase products, think of tickets as your own digital currency.

Below are great example services that could be combined with this service:

- Offers service: Tickets can be used to purchase Offers (products and vouchers), which could translate to be a digital or physical good that can be redeemed at a later point in time.
- Missions service (Tasks): Reward users with credits based upon completion of missions (e.g. chores

around the house, meeting a daily quota, etc.).

6.80.1. Gift Tickets

```
POST /api/{version}/purchase/gift
```

Description

Gift tickets to another user.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *optional*	appKey	string
Query	**assetId** *optional*	assetId	integer(int64)
Query	**customMessage** *optional*	customMessage	string
Query	**deviceId** *optional*	deviceId	string
Query	**gameType** *optional*	gameType	string
Query	**receiverAccountId** *required*	receiverAccountId	integer(int64)
Query	**ticketId** *required*	ticketId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.80.2. POST /api/{version}/purchase/list

CAUTION	operation.deprecated

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**gameType** *optional*	gameType	string	
Query	**purchaseType** *optional*	purchaseType	string	`"SIRQUL"`
Query	**serviceAction** *optional*	serviceAction	string	

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.80.3. Get Ticket Count

```
POST /api/{version}/ticket/count
```

Description

Gets the ticket count.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *optional*	appKey	string
Query	**deviceId** *optional*	deviceId	string
Query	**gameType** *optional*	gameType	string
Query	**ticketType** *optional*	ticketType	string

Responses

HTTP Code	Description	Schema
200	successful operation	CountResponse

6.80.4. Get Ticket List

```
POST /api/{version}/ticket/getList
```

Description

Gets the list of tickets.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**actionType** *optional*	actionType	string
Query	**appKey** *optional*	appKey	string

Type	Name	Description	Schema
Query	**deviceId** *optional*	deviceId	string
Query	**gameType** *optional*	gameType	string
Query	**objectIds** *optional*	objectIds	string
Query	**receiptTokens** *optional*	receiptTokens	string
Query	**ticketIds** *optional*	ticketIds	string
Query	**ticketObjectType** *optional*	ticketObjectType	string

Responses

HTTP Code	Description	Schema
200	successful operation	TicketListResponse

6.80.5. Save Ticket

```
POST /api/{version}/ticket/save
```

Description

Allow user to acquire a purchase item and generate a ticket record. Used to redeem tickets or add tickets to the system.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**actionType** *required*	actionType	string	

Type	Name	Description	Schema	Default
Query	**appKey** *optional*	appKey	string	
Query	**appVersion** *optional*	appVersion	string	
Query	**count** *optional*	count	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**gameType** *optional*	gameType	string	
Query	**includeProfileResponse** *required*	includeProfileResponse	boolean	`"false"`
Query	**objectId** *optional*	objectId	integer(int64)	
Query	**purchaseCode** *optional*	purchaseCode	string	
Query	**purchaseProvider** *optional*	purchaseProvider	string	
Query	**purchaseType** *optional*	purchaseType	string	`"SIRQUL"`
Query	**receiptData** *optional*	receiptData	string	
Query	**receiptToken** *optional*	receiptToken	string	
Query	**returnNulls** *required*	returnNulls	boolean	`"1"`
Query	**returnProfileResponse** *required*	returnProfileResponse	boolean	`"true"`
Query	**ticketObjectType** *required*	ticketObjectType	string	

Type	Name	Description	Schema	Default
Query	**ticketType** *optional*	ticketType	string	

Responses

HTTP Code	Description	Schema
200	successful operation	ProfileResponse

6.80.6. Save Ticket with Reciept

```
POST /api/{version}/ticket/save/fileUpload
```

Description

Similar to the Save Ticket endpoint but allows the receiptData to be in binary format. This must be a multi-part post

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**actionType** *required*	actionType	string	
Query	**appKey** *optional*	appKey	string	
Query	**appVersion** *optional*	appVersion	string	
Query	**count** *optional*	count	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**gameType** *optional*	gameType	string	

Type	Name	Description	Schema	Default
Query	**includeProfileResponse** *required*	includeProfileResponse	boolean	`"false"`
Query	**objectId** *optional*	objectId	integer(int64)	
Query	**purchaseCode** *optional*	purchaseCode	string	
Query	**purchaseProvider** *optional*	purchaseProvider	string	
Query	**purchaseType** *optional*	purchaseType	string	`"SIRQUL"`
Query	**receiptData** *required*	receiptData	ref	
Query	**receiptToken** *optional*	receiptToken	string	
Query	**returnNulls** *required*	returnNulls	boolean	`"1"`
Query	**returnProfileResponse** *required*	returnProfileResponse	boolean	`"true"`
Query	**ticketObjectType** *required*	ticketObjectType	string	
Query	**ticketType** *optional*	ticketType	string	

Responses

HTTP Code	Description	Schema
200	successful operation	ProfileResponse

6.80.7. Get Ticket Offers

```
POST /api/{version}/ticket/ticketoffers
```

Description

Get a list offers for tickets owned by sirqul. Purchasing these will add the number of tickets to the account specified by the offer.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	TicketOfferResponse

6.81. Tournament

The Tournament service enables you to create scheduled tournaments that allow users to join together and compete against each other. The tournament creator can specify the start times, the times between the matches and the times between each round. Tournaments can span from 1 day to multiple weeks. Tournaments are one user versus another user. The user who scores higher moves to the next round until there is one person left standing. Tournaments can also be applied to non-user based systems to score the best performing tasks, routes, or anything else that requires measurement based upon results.

Below are some example scenarios as to how these services can be used:

- Retail: Employees can compete against each other to see who can meet and exceed their quota for the week.
- Automotive: Autonomous delivery vehicles can compete against each other to determine the most optimal route for efficiency.
- Robotics: A fleet of autonomous machinery can be paired against each other for the week. The machinery that performs the best will earn the highest score and will move on. This information can be used as a way to measure overall performance of various setups to determine which setup is the most optimal.

Use tournaments as a way to measure overall performance and filter out inefficient processes and methods. Use it also as an engagement tool amongst your team to foster competition and productivity.

Below are great example services that could be combined with this service:

- Notifications service: Get notified each time a match starts, ends, and when the tournament is complete.
- Scheduled tasks service: Each time a winner is determined for a tournament round, this can trigger an analytic to be sent to record the results.

6.81.1. Create Tournament

```
POST /api/{version}/tournament/create
```

Description

Create a tournament.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	
Query	**allocateTickets** *required*	allocateTickets	boolean	`"false"`
Query	**appKey** *required*	appKey	string	
Query	**audienceIds** *optional*	audienceIds	string	
Query	**availableLimit** *optional*	availableLimit	integer(int32)	
Query	**costToPlay** *required*	costToPlay	integer(int32)	
Query	**costToPlayType** *optional*	costToPlayType	string	
Query	**description** *optional*	description	string	

Type	Name	Description	Schema	Default
Query	**enableBuyBack** *required*	enableBuyBack	boolean	`"false"`
Query	**enableMultipleEntries** *required*	enableMultipleEntries	boolean	`"false"`
Query	**enableMultipleVotes** *required*	enableMultipleVotes	boolean	`"false"`
Query	**featured** *required*	featured	boolean	`"false"`
Query	**fixedReward** *required*	fixedReward	boolean	`"false"`
Query	**imageAssetId** *optional*	imageAssetId	integer(int64)	
Query	**maximumLevelLength** *required*	maximumLevelLength	integer(int32)	`"1800"`
Query	**metaData** *optional*	metaData	string	
Query	**minimumToPlay** *required*	minimumToPlay	integer(int32)	`"1"`
Query	**missionType** *required*	missionType	enum (PUBLIC, SHARED, TOURNAMENT, POOLPLAY, MULTISTAGE)	`"MULTISTAGE"`
Query	**offerAssetId** *optional*	offerAssetId	integer(int64)	
Query	**offerIds** *optional*	offerIds	string	
Query	**preliminaryGroupAdvancements** *required*	preliminaryGroupAdvancements	string	`"1"`
Query	**preliminaryGroups** *required*	preliminaryGroups	integer(int32)	`"1"`

Type	Name	Description	Schema	Default
Query	**secondsBetweenLevels** *required*	secondsBetweenLevels	integer(int32)	`"600"`
Query	**secondsBetweenPacks** *required*	secondsBetweenPacks	integer(int32)	`"86400"`
Query	**secondsForTieBreaker** *required*	secondsForTieBreaker	integer(int32)	`"600"`
Query	**splitReward** *required*	splitReward	enum (EVEN, ALL, FIRST, RANDOM)	`"ALL"`
Query	**startDate** *required*	startDate	integer(int64)	
Query	**startingLimit** *optional*	startingLimit	integer(int32)	
Query	**subType** *optional*	subType	string	
Query	**tieTag** *optional*	tieTag	string	
Query	**title** *required*	title	string	
Query	**tournamentData** *optional*	tournamentData	string	
Query	**visibility** *required*	visibility	enum (PUBLIC, LISTABLE, REWARDABLE, TRIGGERABLE, PRIVATE)	`"PUBLIC"`
Query	**winnerTag** *optional*	winnerTag	string	

Responses

HTTP Code	Description	Schema
200	successful operation	TournamentResponse

6.81.2. Delete Tournament

```
POST /api/{version}/tournament/delete
```

Description

Delete a tournament.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**missionId** *required*	missionId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.81.3. Get Tournament

```
POST /api/{version}/tournament/get
```

Description

Get a tournament.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	

Type	Name	Description	Schema	Default
Query	**includeScores** *optional*	includeScores	string	
Query	**joinCode** *optional*	joinCode	string	
Query	**missionId** *optional*	missionId	integer(int64)	
Query	**objectPreviewSize** *required*	objectPreviewSize	integer(int32)	`"50"`

Responses

HTTP Code	Description	Schema
200	successful operation	TournamentResponse

6.81.4. POST /api/{version}/tournament/object/search

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**descending** *required*	descending	boolean	`"true"`
Query	**gameLevelId** *required*	gameLevelId	integer(int64)	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**sortField** *required*	sortField	enum (ID, PLAYER_SCORE_COUNT)	`"PLAYER_SCORE_COUNT"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.81.5. Search Tournament Rounds

```
POST /api/{version}/tournament/round/search
```

Description

Search for the user's tournament games.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *required*	appKey	string	
Query	**currentOnly** *required*	currentOnly	boolean	`"true"`
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**missionType** *optional*	missionType	enum (PUBLIC, SHARED, TOURNAMENT, POOLPLAY, MULTISTAGE)	
Query	**start** *required*	start	integer(int32)	`"0"`
Query	**status** *required*	status	string	`"ACCEPTED,ACTIVE"`
Query	**visibilities** *required*	visibilities	string	`"PUBLIC"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.81.6. Submit Tournament Score

```
POST /api/{version}/tournament/score
```

Description

Submit an array of scores for a tournament match.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *required*	appKey	string
Query	**gameId** *required*	gameId	integer(int64)
Query	**gameLevelId** *optional*	gameLevelId	integer(int64)
Query	**missionId** *required*	missionId	integer(int64)
Query	**packId** *required*	packId	integer(int64)
Query	**scores** *required*	scores	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.81.7. Search Tournaments

```
POST /api/{version}/tournament/search
```

Description

Search for tournaments

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**appKey** *required*	appKey	string	
Query	**descending** *optional*	descending	boolean	
Query	**filter** *required*	filter	string	`"UPCOMING"`
Query	**includeInactive** *required*	includeInactive	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**missionTypes** *required*	missionTypes	string	`"MULTISTAGE,TOURNAMENT,POOLPLAY"`
Query	**sortField** *required*	sortField	string	`"START_DATE"`
Query	**start** *required*	start	integer(int32)	`"0"`
Query	**subType** *optional*	subType	string	
Query	**visibility** *required*	visibility	string	`"PUBLIC"`

Responses

HTTP Code	Description	Schema
200	successful operation	MissionShortResponse

6.81.8. Substitute Tournament Player

```
POST /api/{version}/tournament/substitute
```

Description

Service to replace the user's opponent in the current level - pack - mission with an AI account.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**gameLevelId** *required*	gameLevelId	integer(int64)
Query	**missionId** *required*	missionId	integer(int64)
Query	**packId** *required*	packId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.81.9. Update Tournament

```
POST /api/{version}/tournament/update
```

Description

Update a tournament.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**active** *optional*	active	boolean
Query	**allocateTickets** *optional*	allocateTickets	boolean
Query	**audienceIds** *optional*	audienceIds	string
Query	**availableLimit** *optional*	availableLimit	integer(int32)
Query	**costToPlay** *optional*	costToPlay	integer(int32)
Query	**costToPlayType** *optional*	costToPlayType	string
Query	**description** *optional*	description	string
Query	**enableBuyBack** *optional*	enableBuyBack	boolean
Query	**enableMultipleEntries** *optional*	enableMultipleEntries	boolean
Query	**enableMultipleVotes** *optional*	enableMultipleVotes	boolean
Query	**featured** *optional*	featured	boolean

Type	Name	Description	Schema
Query	**fixedReward** *optional*	fixedReward	boolean
Query	**imageAssetId** *optional*	imageAssetId	integer(int64)
Query	**maximumLevelLength** *optional*	maximumLevelLength	integer(int32)
Query	**metaData** *optional*	metaData	string
Query	**minimumToPlay** *optional*	minimumToPlay	integer(int32)
Query	**missionId** *required*	missionId	integer(int64)
Query	**offerAssetId** *optional*	offerAssetId	integer(int64)
Query	**offerIds** *optional*	offerIds	string
Query	**preliminaryGroupAdvancements** *optional*	preliminaryGroupAdvancements	string
Query	**preliminaryGroups** *optional*	preliminaryGroups	integer(int32)
Query	**secondsBetweenLevels** *optional*	secondsBetweenLevels	integer(int32)
Query	**secondsBetweenPacks** *optional*	secondsBetweenPacks	integer(int32)
Query	**secondsForTieBreaker** *optional*	secondsForTieBreaker	integer(int32)
Query	**splitReward** *optional*	splitReward	enum (EVEN, ALL, FIRST, RANDOM)
Query	**startDate** *optional*	startDate	integer(int64)

Type	Name	Description	Schema
Query	**startingLimit** *optional*	startingLimit	integer(int32)
Query	**subType** *optional*	subType	string
Query	**tieTag** *optional*	tieTag	string
Query	**title** *optional*	title	string
Query	**tournamentData** *optional*	tournamentData	string
Query	**visibility** *optional*	visibility	enum (PUBLIC, LISTABLE, REWARDABLE, TRIGGERABLE, PRIVATE)
Query	**winnerTag** *optional*	winnerTag	string

Responses

HTTP Code	Description	Schema
200	successful operation	TournamentResponse

6.81.10. Submit Tournament Score

```
POST /api/{version}/tournament/vote
```

Description

Submit a vote for a multi-stage album tournament.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)

Type	Name	Description	Schema
Query	**accountId** *required*	accountId	integer(int64)
Query	**appKey** *required*	appKey	string
Query	**deviceId** *optional*	deviceId	string
Query	**gameObjectId** *required*	gameObjectId	integer(int64)
Query	**missionId** *required*	missionId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.82. Tracking

The Tracking service allows you to get the predicted location for a customer based on previous behavior. If a customer resides in a place for a period of time this is marked as a preferred location. We look back over the previous few days and the previous days of the week from the day specified. If for instance the day was a Wednesday then the would check the days before Tuesday, Monday, Sunday, etc. It will also check some number of previous Wednesdays. Using that information we then determine location matches for each hour of the day. These are ordered by the number of matches and the location with the most matches is returned for that hour as the most like location the customer will be.

Below are some example scenarios as to how these services can be used:

- Retail: Track employees and their whereabouts in the retail location to ensure that everyone is utilizing their time correctly. Send a notification if they are spending time in a spot that they are not assigned to.
- Medical: Track medical equipment in the hospital to ensure correct delivery, placement, and location of each device. Send a notification to the administrator if a piece of medical equipment is not where it should be.
- Automotive: Integrate location tracking in fleet vehicles as a standard option for manned and/or autonomous vehicles. Send a notification if a vehicle has been parked in an area where it should not be.
- Robotics: Track the location and whereabouts of autonomous robots in a warehouse. Send a

notification to the administrator if the robot is not in a location where it should be.

The Tracking service can be used by dispatchers and administrators alike to ensure that employees and/or objects are on track for their intended destination. Alternatively, the tracking service can be used to gather data to provide recommendations for users based upon where they are located at any given time.

Below are great example services that could be combined with this service:

- Routing service: In addition to tracking the path of a user or object, associate pick-up and drop off times as well as ETA's to ensure a successful delivery.
- Retailer service: Depending on where users have been and what their demographic information is, a retailer can be recommended to the user.
- Offers service: Depending on where users have been and what their demographic information is, an offer (product, voucher, coupon, person, etc.) can be recommended to the user to be interacted with, etc.

6.82.1. Create Batch Tracking

```
POST /api/{version}/tracking/batch/create
```

Description

Batch create tracking legs

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**data** *required*	data	string	
Query	**defaultTag** *required*	defaultTag	string	`"PASSIVE"`
Query	**deviceId** *optional*	deviceId	string	
Query	**generateAccounts** *optional*	generateAccounts	boolean	

Type	Name	Description	Schema	Default
Query	**slaveUID** *optional*	slaveUID	string	
Query	**updateAccountLocations** *optional*	updateAccountLocations	boolean	

Responses

HTTP Code	Description	Schema
200	successful operation	< Leg > array

6.82.2. Create Tracking Leg

```
POST /api/{version}/tracking/leg/create
```

Description

Send tracking points to be able to generate pathing data

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**distance** *optional*	distance	number(double)
Query	**duration** *optional*	duration	integer(int64)
Query	**endDate** *required*	endDate	integer(int64)
Query	**endLat** *required*	endLat	number(double)

Type	Name	Description	Schema
Query	**endLng** *required*	endLng	number(double)
Query	**startDate** *required*	startDate	integer(int64)
Query	**startLat** *required*	startLat	number(double)
Query	**startLng** *required*	startLng	number(double)
Query	**steps** *optional*	steps	string
Query	**tags** *optional*	tags	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.82.3. List Tracking

```
POST /api/{version}/tracking/list
```

Description

Search for all accounts that have tracking legs data by the given constraints.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activeOnly** *required*	activeOnly	boolean	`"false"`
Query	**audienceIds** *optional*	audienceIds	string	

Type	Name	Description	Schema	Default
Query	**descending** *required*	descending	boolean	`"true"`
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**keyword** *optional*	keyword	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**longitude** *optional*	longitude	number(double)	
Query	**range** *required*	range	number(double)	`"5.0"`
Query	**sortField** *required*	sortField	string	`"LEG_START_DATE"`
Query	**start** *required*	start	integer(int32)	`"0"`
Query	**startDate** *optional*	startDate	integer(int64)	
Query	**tags** *optional*	tags	string	

Responses

HTTP Code	Description	Schema
200	successful operation	< AccountMiniResponse > array

6.82.4. Get Tracking Path

```
POST /api/{version}/tracking/path/get
```

Description

Get the path (lat/long coordinates) between 2 steps previously logged for a customer.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**endStepId** *required*	endStepId	integer(int64)
Query	**startStepId** *required*	startStepId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	< StepResponse > array

6.82.5. Get Predicted Locations

```
POST /api/{version}/tracking/predicted/get
```

Description

Get the predicted location for a customer based on previous behavior. If a customer resides in a place for a period of time this is marked as a preferred location. We look back over the previous few days and the previous days of the week from the day specified. If for instance the day was a Wednesday then this would check the days before, including: Tuesday, Monday, Sunday, etc. It will also check some number of previous Wednesdays in the past few weeks.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**dateCheck** *optional*	dateCheck	integer(int64)	

Type	Name	Description	Schema	Default
Query	**distanceUnit** *required*	distanceUnit	enum (MILES, KILOMETERS)	`"MILES"`
Query	**hourCheck** *optional*	hourCheck	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**longitude** *optional*	longitude	number(double)	
Query	**searchRange** *required*	searchRange	number(double)	`"0.0"`
Query	**sortOrder** *required*	sortOrder	enum (MATCHES, DISTANCE, WEIGHTED)	`"MATCHES"`
Query	**threshold** *required*	threshold	integer(int64)	`"1"`

Responses

HTTP Code	Description	Schema
200	successful operation	PredictedLocationResponse

6.82.6. Search Preferred Locations

```
POST /api/{version}/tracking/preferred/search
```

Description

Search on preferred locations for a user, which is created when a customer resides in a place for a period of time.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	

Type	Name	Description	Schema	Default
Query	**dateCheck** *optional*	dateCheck	integer(int64)	
Query	**descending** *required*	descending	boolean	`"true"`
Query	**distanceUnit** *required*	distanceUnit	enum (MILES, KILOMETERS)	`"MILES"`
Query	**hourCheck** *optional*	hourCheck	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**longitude** *optional*	longitude	number(double)	
Query	**searchRange** *required*	searchRange	number(double)	`"0.0"`
Query	**sortField** *required*	sortField	string	`"PREFERRED_DATE"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< PreferredLocationResponse > array

6.82.7. Search Tracking

```
POST /api/{version}/tracking/search
```

Description

Retrieve tracking data to be able to show where a user has been.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**endDate** *optional*	endDate	integer(int64)	
Query	**getLastPoint** *required*	getLastPoint	boolean	`"false"`
Query	**ownerId** *optional*	ownerId	integer(int64)	
Query	**startDate** *optional*	startDate	integer(int64)	
Query	**tags** *optional*	tags	string	
Query	**trackingDeviceId** *optional*	trackingDeviceId	string	

Responses

HTTP Code	Description	Schema
200	successful operation	< LegResponse > array

6.82.8. Create Tracking Step

```
POST /api/{version}/tracking/step/create
```

Description

Send tracking points to be able to generate pathing data

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**distance** *optional*	distance	number(double)
Query	**duration** *optional*	duration	integer(int64)
Query	**endDate** *required*	endDate	integer(int64)
Query	**endLat** *required*	endLat	number(double)
Query	**endLng** *required*	endLng	number(double)
Query	**legId** *required*	legId	integer(int64)
Query	**startDate** *required*	startDate	integer(int64)
Query	**startLat** *required*	startLat	number(double)
Query	**startLng** *required*	startLng	number(double)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.83. Trigger

The Triggers service is similar to IFTTT. Applying conditional logic where when a certain action occurs (e.g. user lands on a specific screen, user clicks on a specific button, etc.) this will cause an event (e.g. send an SMS alert to the user, add credits to an account, etc.). For example, if you are running an uber-

like business, and you need to send an alert to the passenger when the driver is nearing their pickup location, there needs to be an event to trigger the notification alert to be sent to the user. By setting up a geofence on the pickup location, once the driver enters within proximity of the location's geofence, it should trigger an action to send an SMS alert to the passenger indicating that the driver is nearing their location. This is just one of many ways a trigger can be used to enhance communication to your users.

Below are some example scenarios as to how these services can be used:

- Retail: Merchant is notified on their device via trigger when a customer adds an offer to their wallet
- Healthcare: When a patient arrives at the hospital, trigger an alert to the doctor that their next appointment is starting shortly
- Smart Home/Communities: When a package is accepted by the concierge of a high-rise apartment building, this will trigger an SMS alert to the resident
- Smart Vehicle: Passenger being picked up by an uber-like service is triggered with an alert on their phone when the driver is arriving at their location

The Triggers service is important because it can be used to communicate when there are problems that arise without intervention of manual human investigation. For example, if you have a CO2 sensor inside of your building and it detects high readings of CO2, when it hits a certain threshold it will trigger an alarm or notification to be sent to the resident/owner's device, building owner's device, and even the fire department system. The Triggers service is also important because it can reduce costs. For example, you have a sprinkler system on your front lawn and it's scheduled to water the grass every day at a specific time. Implement sensors so that when it rains, it will prevent the scheduled system to not enable the sprinkler to turn on.

Below are example services that could be combined with the Triggers service:

- Locations Service: Setup a geofence at a hospital location, and when patients enter the proximity of this location through the geofence, trigger an alert to the doctors who have an appointment scheduled with that patient
- Notifications Service: Triggers can enable notifications to be sent via email, SMS, push notifications, etc.
- Weather Service: If the weather service detects that rain will occur during the day, turn off the automated scheduled sprinkler system

6.83.1. Create Trigger

```
POST /api/{version}/trigger/create
```

Description

Create a trigger

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The api version to use	number(double)	
Query	**accountId** *required*	The logged in user	integer(int64)	
Query	**active** *optional*	Sets whether the Trigger is active or not (inactive Triggers are not processed)	boolean	`"true"`
Query	**appKey** *optional*	The application to target	string	
Query	**conditionalInput** *optional*	Json input representing conditional logic that has to be met before running the trigger	string	
Query	**cronExpression** *optional*	The cron expression that represents the trigger's schedule	string	
Query	**cronType** *optional*	cronType	string	
Query	**endDate** *optional*	The ending date of the trigger	integer(int64)	
Query	**endpointURL** *optional*	The URL for making an HTTP call	string	
Query	**groupingId** *optional*	Client defined identifier for grouping triggers	string	
Query	**name** *required*	The name of the trigger	string	
Query	**payload** *optional*	The parameters for making an HTTP call	string	
Query	**scheduledDate** *optional*	The date and time of the next trigger	integer(int64)	
Query	**startDate** *optional*	The starting date of the trigger	integer(int64)	

Type	Name	Description	Schema	Default
Query	**visibility** *optional*	The determines the scope of who is able to find and view the scheduled notification (PUBLIC - openly available to all Sirqul users, PRIVATE - only available to users that have been invited)	enum (PUBLIC, PRIVATE, FRIENDS)	

Responses

HTTP Code	Description	Schema
200	successful operation	TriggerResponse

6.83.2. Delete Trigger

```
POST /api/{version}/trigger/delete
```

Description

Mark a trigger as deleted.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The api version to use.	number(double)
Query	**accountId** *required*	The logged in user.	integer(int64)
Query	**triggerId** *required*	The id of the trigger to delete.	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.83.3. Get Trigger

```
POST /api/{version}/trigger/get
```

Description

Get a trigger

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The api version to use.	number(double)
Query	**accountId** *required*	The logged in user.	integer(int64)
Query	**triggerId** *required*	The id of the Trigger to return.	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	TriggerResponse

6.83.4. Search Triggers

```
POST /api/{version}/trigger/search
```

Description

Search for triggers

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The api version to use.	number(double)	
Query	**accountId** *required*	The logged in user.	integer(int64)	
Query	**activeOnly** *optional*	Determines whether to return only active results	boolean	`"true"`
Query	**appKey** *optional*	Filter the list by a specific application	string	

Type	Name	Description	Schema	Default
Query	**descending** *optional*	Determines whether the sorted list is in descending or ascending order	boolean	`"true"`
Query	**filter** *optional*	A comma separated list of filters: - MINE - Return triggers that the user has created. - SHARED - Return triggers that have been shared to the user via com.sirqul.geoapps.common.service.UserPermissionsApi#addUsersToPermissionable addUsersToPermissionable. - FOLLOWER - Return triggers that have been created by the user's followers (the content needs to have been APPROVED or FEATURED). - FOLLOWING - Return triggers that have been created by people who the user is following (the content needs to have been APPROVED or FEATURED). - PUBLIC - Return all PUBLIC triggers that have been APPROVED or FEATURED. - ALL_PUBLIC - Return all PUBLIC triggers regardless of whether they are approved or not (ignores the approval status). - LIKED - Return all triggers that the user has liked. - FEATURED - Return all triggers that have been featured. - PENDING - Return all pending triggers.	string	`"MINE"`
Query	**groupingId** *optional*	Filter results by a grouping identifier defined by the client	string	
Query	**keyword** *optional*	Keyword search on the trigger names.	string	
Query	**limit** *optional*	Limit the result to some number.	integer(int32)	`"20"`
Query	**sortField** *optional*	The field to sort by. Possible values include: {ID, CREATED, UPDATED, ACTIVE, NAME, SCHEDULED_DATE, START_DATE, END_DATE	string	`"CREATED"`
Query	**start** *optional*	Start the result set at some index.	integer(int32)	`"0"`
Query	**statuses** *optional*	Filter results by status (comma separated list). Values include: NEW, ERROR, COMPLETE, PROCESSING, TEMPLATE	string	`"NEW,ERROR,COMPLETE,PROCESSING"`

Type	Name	Description	Schema	Default
Query	**templateTypes** *optional*	Template Types	string	

Responses

HTTP Code	Description	Schema
200	successful operation	< TriggerResponse > array

6.83.5. Update Trigger

```
POST /api/{version}/trigger/update
```

Description

Update a trigger

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The api version to use	number(double)
Query	**accountId** *required*	The logged in user	integer(int64)
Query	**active** *optional*	Sets whether the Trigger is active or not (inactive Triggers are not processed)	boolean
Query	**appKey** *optional*	The application to target	string
Query	**conditionalInput** *optional*	Json input representing conditional logic that has to be met before running the trigger	string
Query	**cronExpression** *optional*	The cron expression that represents the trigger's schedule	string
Query	**cronType** *optional*	cronType	string

Type	Name	Description	Schema
Query	**endDate** *optional*	The ending date of the trigger	integer(int64)
Query	**endpointURL** *optional*	The URL for making an HTTP call	string
Query	**groupingId** *optional*	Client defined identifier for grouping triggers	string
Query	**name** *optional*	The name of the trigger	string
Query	**payload** *optional*	The parameters for making an HTTP call	string
Query	**scheduledDate** *optional*	The date and time of the next trigger	integer(int64)
Query	**startDate** *optional*	The starting date of the trigger	integer(int64)
Query	**triggerId** *required*	The trigger to update	integer(int64)
Query	**visibility** *optional*	The determines the scope of who is able to find and view the scheduled notification (PUBLIC - openly available to all Sirqul users, PRIVATE - only available to users that have been invited)	enum (PUBLIC, PRIVATE, FRIENDS)

Responses

HTTP Code	Description	Schema
200	successful operation	TriggerResponse

6.84. Trip

A trip is a request for being picked up at a location with an optional time, and dropped off at a location with an optional time (these pickups and drop offs are called 'stops'). A trip can also be recurring, which will create repeating instances with gapped frequency like calendar events.

Below are some example scenarios as to how these services can be used:

- Automotive: A user can hail a ride from an app. This creates a trip from the user's exact location to wherever they specify the drop off point.
- Medical: Trips can be used to move patients to a new room. Their starting location would be their

current room or waiting area, and the ending location is the new room.

- Robotics: A trip can be a starting point and an ending point for a robot. Perhaps the robot is delivering a parcel or room service.

Trips are the backbone of a ride sharing service. Everybody that needs to be picked up also needs to be dropped off, and trips handle just that.

Below are great example services that could be combined with this service:

- Notification Service: Automatically notify the passenger when the vehicle is nearby for pickup.
- Tracking Service: Track the driver's vehicle so it can be viewed on a dashboard.
- Dashboards: Add, remove, or update trips based upon new feedback or updates.

6.84.1. Create Trip

```
POST /api/{version}/trip
```

Description

Create a new trip

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version	string
Body	**body** *optional*		Trip

Responses

HTTP Code	Description	Schema
200	successful operation	Trip

6.84.2. Search Trips

```
GET /api/{version}/trip
```

Description

Search for trips

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version	number(double)	
Query	**accountId** *required*	The owner of the trips	integer(int64)	
Query	**activeOnly** *required*	Return only active results	boolean	`"true"`
Query	**descending** *required*	Determines whether the sorted list is in descending or ascending order	boolean	`"false"`
Query	**endDate** *optional*	The upper bound limit of time	integer(int64)	
Query	**hasDriver** *optional*	Return only trips with drivers	boolean	
Query	**hasNotifications** *optional*	If trip has pending notifications to send or not	boolean	
Query	**hasRoute** *optional*	Return only trips with routes	boolean	
Query	**limit** *required*	The limit for pagination	integer(int32)	`"20"`
Query	**sortField** *required*	The field to sort by	string	`"id"`
Query	**start** *required*	The start index for pagination	integer(int32)	`"0"`
Query	**startDate** *optional*	The lower bound limit of time	integer(int64)	
Query	**timeZone** *optional*	The timeZone string, e.g. "America/Los_Angeles"	string	

Responses

HTTP Code	Description	Schema
200	successful operation	< Trip > array

6.84.3. Search Trips

```
GET /api/{version}/trip/match
```

Description

Search for trips with matching information.

Parameters

Type	Name	Schema	Default
Path	**version** *required*	number(double)	
Query	**accountId** *required*	integer(int64)	
Query	**activeOnly** *required*	boolean	`"true"`
Query	**descending** *required*	boolean	`"false"`
Query	**endDate** *optional*	integer(int64)	
Query	**limit** *required*	integer(int32)	`"20"`
Query	**matchedHasDriver** *optional*	boolean	
Query	**matchedHasRoute** *optional*	boolean	
Query	**sortField** *required*	string	`"id"`
Query	**start** *required*	integer(int32)	`"0"`
Query	**startDate** *optional*	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	< Trip > array

6.84.4. Process Trip Matches

```
POST /api/{version}/trip/match/process
```

Description

Process trip matching, assign trips with no route to matched trips with route.

Parameters

Type	Name	Schema
Path	**version** *required*	string
Query	**endDate** *optional*	integer(int64)
Query	**startDate** *optional*	integer(int64)
Query	**tripId** *optional*	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	< Trip > array

6.84.5. Trip Notifications

```
POST /api/{version}/trip/notifications
```

Description

Update the trip notifications

Parameters

Type	Name	Schema
Path	**version** *required*	number(double)
Query	**id** *required*	integer(int64)
Query	**notifications** *optional*	string

Responses

HTTP Code	Description	Schema
200	successful operation	Trip

6.84.6. Get Trip

```
GET /api/{version}/trip/{id}
```

Description

Get an existing trip

Parameters

Type	Name	Description	Schema
Path	**id** *required*		integer(int64)
Path	**version** *required*	The API version	string

Responses

HTTP Code	Description	Schema
200	successful operation	Trip

6.84.7. Update Trip

```
PUT /api/{version}/trip/{id}
```

Description

Update an existing trip. Does not support recurring trip update.

Parameters

Type	Name	Schema
Path	**id** *required*	integer(int64)
Path	**version** *required*	string
Body	**body** *optional*	Trip

Responses

HTTP Code	Description	Schema
200	successful operation	Trip

6.84.8. Delete Trip

```
DELETE /api/{version}/trip/{id}
```

Description

Delete an existing trip

Parameters

Type	Name	Schema
Path	**id** *required*	integer(int64)
Path	**version** *required*	string
Query	**deleteOriginal** *optional*	boolean

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

6.84.9. Cancel Trip Preference

```
POST /api/{version}/trip/{id}/cancel
```

Description

Cancel a trip (removes riders from trip) and set preferences back to original state.

Parameters

Type	Name	Schema	Default
Path	**id** *required*	integer(int64)	
Path	**version** *required*	number(double)	
Query	**recurrence** *required*	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	Trip

6.84.10. Set Trip Preference Driver

```
POST /api/{version}/trip/{id}/drive
```

Description

Update trip preference to drive, also create a route and assign the trip to the route

Parameters

Type	Name	Schema	Default
Path	**id** *required*	integer(int64)	
Path	**version** *required*	number(double)	
Query	**recurrence** *required*	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	Trip

6.84.11. Set Trip Preference Flexible

```
POST /api/{version}/trip/{id}/flexible
```

Description

Update trip preference to flexible.

Parameters

Type	Name	Schema	Default
Path	**id** *required*	integer(int64)	
Path	**version** *required*	number(double)	
Query	**recurrence** *required*	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	Trip

6.84.12. Update Trip Locations

```
POST /api/{version}/trip/{id}/locations
```

Parameters

Type	Name	Schema
Path	**id** *required*	integer(int64)
Path	**version** *required*	string
Body	**body** *optional*	Trip

Responses

HTTP Code	Description	Schema
200	successful operation	Trip

6.84.13. Update Recurrence Locations

```
POST /api/{version}/trip/{id}/locations/recurrence
```

Parameters

Type	Name	Schema
Path	**id** *required*	integer(int64)
Path	**version** *required*	string
Body	**body** *optional*	Trip

Responses

HTTP Code	Description	Schema
200	successful operation	< Trip > array

6.84.14. Get Trip Matches

```
GET /api/{version}/trip/{id}/match
```

Description

Get matching trips of specific trip

Parameters

Type	Name	Schema	Default
Path	**id** *required*	integer(int64)	
Path	**version** *required*	number(double)	
Query	**activeOnly** *required*	boolean	`"true"`
Query	**descending** *required*	boolean	`"false"`
Query	**limit** *required*	integer(int32)	`"20"`
Query	**matchedHasDriver** *optional*	boolean	
Query	**matchedHasRoute** *optional*	boolean	
Query	**sortField** *required*	string	`"id"`
Query	**start** *required*	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< Trip > array

6.84.15. Set Trip Preference Rider

```
POST /api/{version}/trip/{id}/ride
```

Description

Update trip preference to ride.

Parameters

Type	Name	Schema	Default
Path	**id** *required*	integer(int64)	
Path	**version** *required*	number(double)	
Query	**recurrence** *required*	boolean	`"false"`

Responses

HTTP Code	Description	Schema
200	successful operation	Trip

6.84.16. Update Trip Shipments

```
POST /api/{version}/trip/{id}/shipments
```

Parameters

Type	Name	Schema
Path	**id** *required*	integer(int64)
Path	**version** *required*	string
Body	**body** *optional*	Trip

Responses

HTTP Code	Description	Schema
200	successful operation	Trip

6.84.17. Update Recurrence Shipments

```
POST /api/{version}/trip/{id}/shipments/recurrence
```

Parameters

Type	Name	Schema
Path	**id** *required*	integer(int64)
Path	**version** *required*	string
Body	**body** *optional*	Trip

Responses

HTTP Code	Description	Schema
200	successful operation	< Trip > array

6.85. Twilio

Already a Twilio user? Integrate it into the Sirqul platform to enhance and augment your existing notification system. Leverage the power of Twilio with Sirqul's scheduled notifications and scheduled tasks to create notification campaigns that can be easily managed via a robust web dashboard.

Even if you aren't an existing Twilio subscriber, you can easily integrate Twilio into Sirqul using our easy to use integration tools.

Below are some example scenarios as to how these services can be used:

- Retail: Notify existing subscribers of deals and events happening at the mall.
- Medical: Notify Doctors, Nurses, and other important people in relation to a patient's current stay at the hospital.
- Real Estate: Notify customers of new interest rate updates, property changes, and anything to help the customer be kept up to date with the latest information.
- Automotive: Send and receive ETA notifications for deliveries, pick ups, and drop offs that are based upon proximity and location.
- Robotics: Receive notifications in real-time for status updates, failures, and any other things that you set to send as a notification for various devices and machinery.

Use Twilio to ensure successful delivery of SMS/MMS to registered phone numbers.

Below are great example services that could be combined with this service:

- Scheduled Notifications service: Define and set when the SMS/MMS messages are sent to the specified phone numbers.
- Scheduled Tasks service: Set specific triggers that send notifications when a particular action happens.
- Offers service: Send users product information, deals, and other digital goods that can be redeemed at a later point in time.

6.85.1. Buy Offer by SMS

```
POST /api/{version}/sms/buyoffer/{appKey}
```

Description

Recieve an SMS payload from Twillio to purchase an offer.

Parameters

Type	Name	Description	Schema	Default
Path	**appKey** *required*	appKey	string	
Path	**version** *required*	version	number(double)	
Query	**Body** *required*	Body	string	
Query	**From** *required*	From	string	
Query	**currencyType** *required*	currencyType	string	`"CASH"`

Responses

HTTP Code	Description	Schema
200	successful operation	TwiMLResponse

Produces

- `application/xml`

6.86. Twitter

Authenticate users in Sirqul by leveraging Twitter. Sirqul makes authentication via the third party social network seamlessly. It allows you to pull information from their Twitter account such as page follows, hashtags, and their public posts, as long as the user grants permission to do so. The service is secure so that this information cannot be accessed by other users unless they are granted permission by both the user and also through the application.

Below are some example scenarios as to how these services can be used:

- Retail: Allow users to login through their Twitter account so that all of their follows data can be leveraged to match them with products in your store
- Smart Home: Allow your user to login via Twitter to a custom music application installed in your home, which can then pull information from your Twitter account to recommend music based upon their likes and interests

The Twitter service is important because it allows your app to leverage additional information about your user that may already be available through their Twitter account. This means that you don't have to do anything within your application to build features in order to obtain information such as page follows, keywords from public posts, hashtags, and so forth. This is an extremely easy and seamless way for your application to ask permission to the user to obtain this information.

- The Categories and Filters service works well with the Twitter service because it can provide your users with relevant content based upon the information that is pulled in from Twitter such as hashtags, keywords posted on tweets, follows, etc. which can all be managed as categories and filters.

6.86.1. Authorize Twitter

```
POST /api/{version}/twitter/authorize
```

Description

Makes an authorization call to twitter for a user to login and allow any app permissions.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)

Type	Name	Description	Schema
Query	**appKey** *required*	appKey	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.86.2. Login Twitter

```
POST /api/{version}/twitter/login
```

Description

Returns the user profile information given an access token and the secret access token. This call verifies the tokens with twitter and creates a Sirqul account for the user if its their first time logging in.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accessToken** *required*	accessToken	string	
Query	**accessTokenSecret** *required*	accessTokenSecret	string	
Query	**appKey** *required*	appKey	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**longitude** *optional*	longitude	number(double)	
Query	**responseFilters** *required*	responseFilters	string	`"PROFILE"`

Responses

HTTP Code	Description	Schema
200	successful operation	ProfileResponse

6.87. UserPermissions

Define user and group permissions for various Sirqul services. Choose who has access to read, write, update, and/or delete content to the items they have access to. Services that use permissions:

- Accounts
- Albums
- Album Contests
- Collection
- Game levels
- Theme descriptors

Below are some example scenarios as to how these services can be used:

- Retail: Internal employee dashboard that grant access to various parts of the dashboard depending on the employee's role type.
- Medical: Different levels of access to a patient's x-ray files. Only the approved practitioners and their patients have access to access and comment on the files.
- Real Estate: Agents and their clients accessing the same home information when looking to buy and/or sell a home.
- Automotive: Family members will have a different profile for the vehicle to personalize their experience.
- Robotics: Certain smart devices (e.g. Rumba) can be controlled by specific users in the home.

User permissions are key a key service to ensure that you can share and interact with content that are authorized to. It's important to maintain privacy among users and to respect the information that is exchanged between them.

Below are great example services that could be combined with this service:

- Albums service: Pick and choose which person or group of people can see the content that you upload to an album (collection of assets). Define if the users can simply view content versus adding their own content.
- Game levels service: Pick and choose which smart devices can participate in a set of rules and commands that are defined by you. For example, a Rumba will execute the commands defined in

the game level, but the Air Conditioner will not have access to these commands.

- Theme descriptors: As an application owner, choose who can content that is distributed in your mobile application or website experience. For example, the look and feel of the dashboard.

6.87.1. Add User

```
POST /api/{version}/consumer/permissions/add
```

Description

Adds a user to a permissionable object.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**add** *optional*	add	boolean	`"0"`
Query	**admin** *optional*	admin	boolean	
Query	**audienceIds** *optional*	audienceIds	string	
Query	**connectionAccountIds** *optional*	connectionAccountIds	string	
Query	**connectionGroupIds** *optional*	connectionGroupIds	string	
Query	**connectionIds** *optional*	connectionIds	string	
Query	**delete** *optional*	delete	boolean	`"0"`
Query	**deviceId** *optional*	deviceId	string	

Type	Name	Description	Schema	Default
Query	**includeFriendGroup** *required*	includeFriendGroup	boolean	`"0"`
Query	**latitude** *optional*	latitude	number(double)	
Query	**longitude** *optional*	longitude	number(double)	
Query	**pending** *required*	pending	boolean	`"false"`
Query	**permissionableId** *required*	permissionableId	integer(int64)	
Query	**permissionableType** *required*	permissionableType	string	
Query	**read** *optional*	read	boolean	`"1"`
Query	**write** *optional*	write	boolean	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.87.2. Leave

```
POST /api/{version}/consumer/permissions/leave
```

Description

Used when the user wants to leave from someone else's permissionable object

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)

Type	Name	Description	Schema
Query	**accountId** *required*	accountId	integer(int64)
Query	**deviceId** *optional*	deviceId	string
Query	**latitude** *optional*	latitude	number(double)
Query	**longitude** *optional*	longitude	number(double)
Query	**permissionableId** *required*	permissionableId	integer(int64)
Query	**permissionableType** *required*	permissionableType	string

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.87.3. Remove User

```
POST /api/{version}/consumer/permissions/remove
```

Description

Used to remove someone (assuming they have permission) from a permissionable object

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**audienceIds** *optional*	audienceIds	string	

Type	Name	Description	Schema	Default
Query	**connectionAccountIds** *optional*	connectionAccountIds	string	
Query	**connectionGroupIds** *optional*	connectionGroupIds	string	
Query	**connectionIds** *optional*	connectionIds	string	
Query	**deviceId** *optional*	deviceId	string	
Query	**latitude** *optional*	latitude	number(double)	
Query	**longitude** *optional*	longitude	number(double)	
Query	**permissionableId** *required*	permissionableId	integer(int64)	
Query	**permissionableType** *required*	permissionableType	string	
Query	**removeFriendGroup** *required*	removeFriendGroup	boolean	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.87.4. Approve Permissionable

```
POST /api/{version}/permissionable/approve
```

Description

Sets the approval status of a permissionable object.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**approvalStatus** *required*	approvalStatus	enum (PENDING, REJECTED, APPROVED, FEATURED)	`"APPROVED"`
Query	**deviceId** *optional*	deviceId	string	
Query	**permissionableId** *required*	permissionableId	integer(int64)	
Query	**permissionableType** *required*	permissionableType	string	

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.87.5. Search Permissionables Following Distance

```
POST /api/{version}/permissions/distancesearch
```

Description

Search on UserPermissions Distance

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	

Type	Name	Description	Schema	Default
Query	**accountId** *required*	accountId	integer(int64)	
Query	**admin** *optional*	admin	boolean	
Query	**connectionAccountId** *optional*	connectionAccountId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**latitude** *required*	latitude	number(double)	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**longitude** *required*	longitude	number(double)	
Query	**pending** *optional*	pending	boolean	
Query	**permissionableId** *optional*	permissionableId	integer(int64)	
Query	**permissionableType** *optional*	permissionableType	string	
Query	**searchRange** *required*	searchRange	number(double)	`"5.0"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< UserPermissionsResponse > array

6.87.6. Search Permissionables

```
POST /api/{version}/permissions/search
```

Description

Search on UserPermissions

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**admin** *optional*	admin	boolean	
Query	**connectionAccountId** *optional*	connectionAccountId	integer(int64)	
Query	**deviceId** *optional*	deviceId	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**pending** *optional*	pending	boolean	
Query	**permissionableId** *optional*	permissionableId	integer(int64)	
Query	**permissionableType** *optional*	permissionableType	string	
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< UserPermissionsResponse > array

6.88. Vehicle

A vehicle is assigned to a route. The vehicle has a vehicle type which determines the constraint of what and how much a vehicle can carry when doing routing and optimization. This can be an actual vehicle, or even an autonomous machine or robot.

Below are some example scenarios as to how these services can be used:

- Retail: A vehicle could be assigned to a service hub for deliveries to the various retail stores. A vehicle could also be a piece of heavy machinery in the warehouse.
- Medical: Vehicles would be ambulances being dispatched to emergencies and other issues. These vehicles could be tracked and have their routes optimized to have the quickest response time.
- Robotics: Robots moving along a track and performing tasks could be considered vehicles. A dispatcher could send out specific robots based upon their vehicle type to have the right robot for the job.

Vehicles are the core of any transportation service. Using our services, an organization can have each vehicle they operate listed and organized in a database and able to make changes and updates on the fly. With this, a dispatcher would know exactly how many and which types of vehicles are still available for dispatch and which vehicles are already out on the road.

Below are great example services that could be combined with this service:

- Notification Service: Notify the owner of a vehicle when it is time for scheduled maintenance.
- Tracking Service: Extremely useful for dispatchers. Track vehicles on a map and have a birds eye view of where each vehicle is. Parent's could also use this to see their child on a school bus.
- Account Service: Assign each vehicle to a user's account.

6.88.1. Create Vehicle

```
POST /api/{version}/vehicle
```

Description

Create new vehicle

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version	string

Type	Name	Description	Schema
Body	**body** *optional*		Vehicle

Responses

HTTP Code	Description	Schema
200	successful operation	Vehicle

6.88.2. Search Vehicle

```
GET /api/{version}/vehicle
```

Description

Search for vehicles

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version	number(double)
Query	**activeOnly** *required*	Return only active results	boolean
Query	**descending** *required*	Determines whether the sorted list is in descending or ascending order	boolean
Query	**hubId** *required*	Filter by service hub	integer(int64)
Query	**keyword** *optional*	The keyword to search for	string
Query	**limit** *required*	The limit for pagination	integer(int32)
Query	**sortField** *required*	The field to sort by	string
Query	**start** *required*	The start index for pagination	integer(int32)

Responses

HTTP Code	Description	Schema
200	successful operation	< Vehicle > array

6.88.3. Get Vehicle

```
GET /api/{version}/vehicle/{id}
```

Description

Get an existing vehicle

Parameters

Type	Name	Description	Schema
Path	**id** *required*	The id of the vehicle requested	integer(int64)
Path	**version** *required*	The API version	string

Responses

HTTP Code	Description	Schema
200	successful operation	Vehicle

6.88.4. Update Vehicle

```
PUT /api/{version}/vehicle/{id}
```

Description

Update an existing vehicle

Parameters

Type	Name	Description	Schema
Path	**id** *required*	The id of the vehicle to update	integer(int64)

Type	Name	Description	Schema
Path	**version** *required*	The API version	string
Body	**body** *optional*		Vehicle

Responses

HTTP Code	Description	Schema
200	successful operation	Vehicle

6.88.5. Delete Vehicle

```
DELETE /api/{version}/vehicle/{id}
```

Description

Delete an existing vehicle

Parameters

Type	Name	Description	Schema
Path	**id** *required*	The id of the vehicle to delete	integer(int64)
Path	**version** *required*	The API version	string

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

6.89. VehicleType

A vehicle type is associated to a specific service hub. A vehicle type specifies properties of a vehicle, including dimensions, max weight, and supported cargo types to carry. This service allows you to create, delete, get, or update a vehicle type.

Below are some example scenarios as to how these services can be used:

- Retail: A retail warehouse could use different vehicle types for different types of shipments. They could get hot items delivered daily with a smaller vehicle, and larger - but slower to sell - products could be delivered on a large semi truck.
- Medical: Medical dispatchers could send out a smaller medical van for medical issues, or a full on ambulance for medical emergencies. With Vehicle Types, dispatchers will know the exact capabilities of the vehicle and make the best decision on which to dispatch.
- Robotics: A manufacturing plant has robots that perform various tasks. Using Vehicle Types, a robot manager could see the max carry weight and size of each robot and be able to choose the most efficient robot for the job.

Vehicle Type brings a new level of granularity in regards to vehicles. Dispatchers, or even an automatic dispatch system, will use the best vehicle for the job. Choosing the right vehicle saves time, money, and effort.

Below are great example services that could be combined with this service:

- Notification Service: Notify the recipient of which vehicle type is delivering their product. The recipient can then make the proper preparations to accept the delivery smoothly.
- Tracking Service: Only track certain types of vehicles. A company could only track semi trucks, if they don't care to track their pickup trucks.
- Dashboards: Add or update vehicle types on the fly.

6.89.1. Create Vehicle Type

```
POST /api/{version}/vehicle/type
```

Description

Create a new vehicle type

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version	string
Body	**body** *optional*		VehicleType

Responses

HTTP Code	Description	Schema
200	successful operation	VehicleType

6.89.2. Search Vehicle Type

```
GET /api/{version}/vehicle/type
```

Description

Search for types of vehicles

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The API version	number(double)	
Query	**activeOnly** *required*	Return only active results	boolean	`"true"`
Query	**descending** *required*	Determines whether the sorted list is in descending or ascending order	boolean	`"false"`
Query	**hubId** *optional*	Filter by service hub	integer(int64)	
Query	**limit** *required*	The limit for pagination	integer(int32)	`"20"`
Query	**retailerId** *optional*	Filter by retailer	integer(int64)	
Query	**sortField** *required*	The field to sort by	string	`"id"`
Query	**start** *required*	The start index for pagination	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< VehicleType > array

6.89.3. Get Vehicle Type

```
GET /api/{version}/vehicle/type/{vehicleTypeId}
```

Description

Get a vehicle type

Parameters

Type	Name	Description	Schema
Path	**vehicleTypeId** *required*	The id of the requested vehicle type	integer(int64)
Path	**version** *required*	The API version	string

Responses

HTTP Code	Description	Schema
200	successful operation	VehicleType

6.89.4. Update Vehicle Type

```
PUT /api/{version}/vehicle/type/{vehicleTypeId}
```

Description

Update a vehicle type

Parameters

Type	Name	Description	Schema
Path	**vehicleTypeId** *required*	The id of the vehicle type to update	integer(int64)
Path	**version** *required*	The API version	string
Body	**body** *optional*		VehicleType

Responses

HTTP Code	Description	Schema
200	successful operation	VehicleType

6.89.5. Delete Vehicle Type

```
DELETE /api/{version}/vehicle/type/{vehicleTypeId}
```

Description

Delete a vehicle type

Parameters

Type	Name	Description	Schema
Path	**vehicleTypeId** *required*	The id of the vehicle type to update	integer(int64)
Path	**version** *required*	The API version	string

Responses

HTTP Code	Description	Schema
default	successful operation	No Content

6.90. Wallet

The Wallet service acts as a container to store various digital items such as offers, products, coupons, vouchers, etc. that have been purchased or otherwise obtained by the user. Items in the wallet can be used whenever the user wishes to use them given that it has not expired yet.

The Wallet service is primarily for consumers who are collecting items or objects by using some form of a consumer facing application.

Below are some example scenarios as to how these services can be used:

- Retail: Macy's creates an offer that is discoverable by consumers using the Macy's shopping app. Consumers add the deal to their Wallet that can be redeemed at a later point in time.
- Medical: A hospital bill can be considered an item that can be added to the Wallet to be paid at a

later point in time.

- Real Estate: Users can add buyer bonuses and other incentives that help in the final cost of a home purchase.
- Automotive: Users can add offers for routine maintenance and gas rewards to be used at some point in time.
- Robotics: Work orders can be put into the Wallet for robotics to perform on a daily, weekly, or monthly basis.

This service is specifically used as a container for objects that can be used at some point in the future. Without it, you will have to build your own custom solution to interact with these objects.

Below are great example services that could be combined with this service:

- Billing service: Many items need to be purchased in order to be put into the wallet. Some can be purchased after the fact. But, users will need a valid card on file to properly transact the purchase.
- Analytics service: When money is involved, analytics are very important. Purchase success and failure rates are very important to know. You don't want to be losing money without knowing it.
- User transaction service: Items from the wallet all have a status. These are statuses that represent if the item is in a redeemable, unredeemable, valid, or invalid state.
- Offers service: The key item or object that goes into the wallet. An offer can be considered a coupon, voucher, product, event, person, etc.

6.90.1. Create Wallet Offers

```
POST /api/{version}/wallet/create
```

Description

Adds offers to the wallet

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The version of the api to response with	number(double)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**appKey** *optional*	The application requesting the purchase, required when currencyType is TICKETS	string	

Type	Name	Description	Schema	Default
Query	**currencyType** *optional*	Determines the method of purchasing offer. CASH = use card on file, POINTS = use points, TICKETS = use tickets	string	`"CASH"`
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string	
Query	**metaData** *optional*	External custom client defined data	string	
Query	**offerCart** *optional*	A JSON list of offers to purchase: <code><pre> [{ 'offerId': 123, 'offerLocationId': 234, 'quantity': 2 , { offerId': 456, offerLocationId': 567, quantity': 1] </pre></code>	string	
Query	**offerId** *optional*	The id of the offer being added (offerId or offeLocationId required)	integer(int64)	
Query	**offerLocationId** *optional*	The id of the offer location being added (offerId or offeLocationId required)	integer(int64)	
Query	**usePoints** *optional*	Sets the currencyType to POINTS	boolean	

Responses

HTTP Code	Description	Schema
200	successful operation	< OfferTransactionResponse > array

6.90.2. Delete Wallet Offer

```
POST /api/{version}/wallet/delete
```

Description

Removes the transaction from the wallet by setting the deleted date to the current date/time. Requires a valid account and transactionId.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The version of the api to response with	number(double)
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string
Query	**transactionId** *required*	The offer transaction id to remove	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.90.3. Get Wallet Offer

```
POST /api/{version}/wallet/get
```

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The version of the api to response with	number(double)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string	
Query	**includeMission** *optional*	If true then include mission data, false to not include	boolean	`"false"`
Query	**latitude** *optional*	The latitude location of the user	number(double)	
Query	**longitude** *optional*	The latitude location of the user	number(double)	

Type	Name	Description	Schema	Default
Query	**returnFullResponse** *optional*	Determines whether to return a detailed version of the response	boolean	`"true"`
Query	**transactionId** *required*	The offer transaction id to get details of	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	OfferTransactionResponse

6.90.4. Search Wallet Offers

```
POST /api/{version}/wallet/search
```

Description

Search on active offers currently in the user's wallet, or past offers the user has already redeemed.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The version of the api to response with	number(double)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**activeOnly** *optional*	Active Only	boolean	`"false"`
Query	**categoryIds** *optional*	Category Ids	string	
Query	**descending** *optional*	Determines whether the results are in descending order	boolean	`"true"`
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string	
Query	**endedBefore** *optional*	Filter results by the offer end date	integer(int64)	

Type	Name	Description	Schema	Default
Query	**endedSince** *optional*	Filter results by the offer end date	integer(int64)	
Query	**filterByParentOffer** *optional*	Apply params to offer's parent	boolean	
Query	**keyword** *optional*	The keyword to search for	string	
Query	**latitude** *optional*	The latitude location of the user	number(double)	
Query	**limit** *optional*	The limit for pagination	integer(int32)	`"20"`
Query	**longitude** *optional*	The latitude location of the user	number(double)	
Query	**offerId** *optional*	Filter results for this offer	integer(int64)	
Query	**offerLocationId** *optional*	Filter results for this offer location	integer(int64)	
Query	**offerType** *optional*	Filter results to only return COUPON or VOUCHER offers	enum (VOUCHER, COUPON, PRODUCT, MEDIA, EVENT, DEVICE)	
Query	**redeemableEndDate** *optional*	Filter results by the offer redeemable date	integer(int64)	
Query	**redeemableStartDate** *optional*	Filter results by the offer redeemable date	integer(int64)	
Query	**redeemed** *optional*	If true return redeemed transactions (status 2), otherwise return active transactions (status 0 or 1)	boolean	`"false"`
Query	**reservationsOnly** *optional*	Returns only reservation transactions if true	boolean	`"false"`
Query	**retailerId** *optional*	Filter results for this retailer	integer(int64)	

Type	Name	Description	Schema	Default
Query	**retailerLocationId** *optional*	Filter results for this retailer location	integer(int64)	
Query	**returnFullResponse** *optional*	Determines whether to return a detailed version of the response	boolean	`"false"`
Query	**sortField** *optional*	Determines what to sort the results by {CREATED, UPDATED, SEARCH_TAGS, ACTIVE, ACTIVATED, EXPIRES, TITLE, SUBTITLE, OFFER_TYPE, SPECIAL_OFFER_TYPE, OFFER_VISIBILITY, CUSTOMER_ID, CUSTOMER_DISPLAY, RETAILER_ID, RETAILER_NAME, RETAILER_LOCATION_ID, RETAILER_LOCATION_NAME, BILLABLE_ENTITY_ID, BILLABLE_ENTITY_NAME, RESPONSIBLE_DISPLAY	enum (ID, CREATED, UPDATED, DELETED, SEARCH_TAGS, ACTIVE, ACTIVATED, EXPIRES, TITLE, SUBTITLE, DETAILS, OFFER_TYPE, SPECIAL_OFFER_TYPE, OFFER_VISIBILITY, REDEEMABLE_START, REDEEMABLE_END, CUSTOMER_ID, CUSTOMER_DISPLAY, RETAILER_ID, RETAILER_NAME, RETAILER_LOCATION_ID, RETAILER_LOCATION_NAME, BILLABLE_ENTITY_ID, BILLABLE_ENTITY_NAME, RESPONSIBLE_DISPLAY)	`"CREATED"`
Query	**start** *optional*	The start index for pagination	integer(int32)	`"0"`
Query	**startedBefore** *optional*	Filter results by the offer start date	integer(int64)	
Query	**startedSince** *optional*	Filter results by the offer start date	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	< OfferTransactionResponse > array

6.90.5. Update Wallet Offer

```
POST /api/{version}/wallet/update
```

Description

Update offer status. The status values are: 0 - not redeemable, 1 - redeemable.

Not redeemable means the customer has received the offer but has not decided to use (or print) it yet. Until they choose to do this the merchant cannot redeem the offer (has not been given permission yet).

Redeemable means the customer has chosen to use the offer and wishes to redeem it.

Redeemed means the merchant has accepted the offer and the given the customer its value, then marked it a used in the system. This status change is handled by a merchant end point.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	The version of the api to response with	number(double)	
Query	**accountId** *required*	The account id of the user (deviceId or accountId required)	integer(int64)	
Query	**appKey** *optional*	App Key	string	
Query	**currencyType** *optional*	Currency Type	string	`"CASH"`
Query	**deviceId** *optional*	The device id (deviceId or accountId required)	string	
Query	**latitude** *optional*	The latitude location of the user	number(double)	

Type	Name	Description	Schema	Default
Query	**longitude** *optional*	The latitude location of the user	number(double)	
Query	**metaData** *optional*	External custom client defined data	string	
Query	**offerLocationId** *optional*	Offer Location Id	integer(int64)	
Query	**returnFullResponse** *optional*	Determines whether to return a detailed version of the response	boolean	`"false"`
Query	**status** *required*	The status value to change to (0 or 1)	integer(int32)	
Query	**transactionId** *required*	The offer transaction id to remove	integer(int64)	
Query	**usePoints** *optional*	Use Points	boolean	

Responses

HTTP Code	Description	Schema
200	successful operation	OfferTransactionResponse

6.91. Weather

The Weather service is used to get the most up to date weather information for your Sirqul powered experience. This service is also cacheable so that in the case of an outage, you still have the information that you need.

In a world of home automation and smart devices, the Weather service can act as a gauge of when to perform certain actions. For example, if the the forecast calls for rain today, you can be pushed a notification to bring a raincoat. While many services can do this already, integrating it with Sirqul can bring much more depth to the experience. The system will also take into consideration any events or locations that you are going to today and analyze the information even further for more suggestions.

Below are some example scenarios as to how these services can be used:

- Retail: If the outdoor temperature reaches over 80 degrees fahrenheit (26.67 celsius), automatically generate an offer for a buy one, get one free ice cream treat at Baskin Robbins.

- Medical: In a temperature controlled environment
- Real Estate: If the temperature is going to be greater than 60 degrees fahrenheit (15.56 celsius) and it is not going to rain, push notifications out to users for an open house.
- Automotive: If the outdoor temperature reaches a certain threshold, begin adjusting the climate inside of the car to a setting that the user has used often.

The Weather service should be used in conjunction with the Sirqul Scheduled Task service to ensure that you can leverage the most benefit out of automated triggers and actions.

Below are great example services that could be combined with this service:

- Scheduled Tasks service: When certain temperature and weather conditions are met, trigger some action from that.
- Analytics service: Track overall trend of weather data and how it impacts the Scheduled Tasks assigned to it.

6.91.1. Search Weather

```
POST /api/{version}/weather/search
```

Description

Search the weather forcast for the next 5 days

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	Version	number(double)	
Query	**regionId** *required*	Region Id	integer(int64)	
Query	**timezoneOffset** *optional*	Timezone Offset	integer(int64)	`"-6"`

Responses

HTTP Code	Description	Schema
200	successful operation	WeatherResponse

6.92. Word

The Words service enables you to create custom words with definitions that can be thought of like a dictionary. This service would be used in conjunction with other services to realize its full potential. Using Words with our Scheduled Tasks service would empower users to set “Words” (commands) that can be triggered based upon voice recognition cues. For example, I could set up the Words “Open” and “Garage” as triggers and then create a Scheduled Task to then communicate with a central gateway that would trigger the appropriate action. In this case, opening the garage.

This service is not limited to voice recognition, it could also be text based such that applications and scripts can execute the desired commands based upon values that are returned from this service.

Below are some example scenarios as to how these services can be used:

- Retail: If a status report indicates that there has been a sharp increase in sales of a particular item, it could detect the Word that is specified and suggest to the user to put in an order for more inventory or have it automatically done.
- Medical: Keywords specified in medical reports could trigger instantaneous recommendations for patients in regards to prescriptions and other such needs.
- Robotics: Use custom work orders with commands that smart machines can understand and execute.

Use Words as a way to trigger specific actions and automate process you would otherwise do manually. Manage them remotely via a web dashboard or mobile app and make updates on the fly.

Below are great example services that could be combined with this service:

- Scheduled Task service: When certain words are called, trigger some action.
- Game level service: Build a custom set of ‘rules’ that are defined by the words that are added to a game level. This could be an order of operations based upon the triggers that are defined.
- Pack service: Group multiple game levels to create a more complex automation system defined by each game level.
- Analytic service: Every time an action occurs, record an analytic so that you know how often something happens and if there are any failure points along the way.

6.92.1. Create Word

```
POST /api/{version}/game/word/create
```

Description

Create a word by the given params.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *required*	active	boolean	`"false"`
Query	**allocateTickets** *required*	allocateTickets	boolean	`"false"`
Query	**assetId** *optional*	assetId	integer(int64)	
Query	**definition** *required*	definition	string	
Query	**metaData** *optional*	metaData	string	
Query	**points** *optional*	points	integer(int64)	
Query	**ticketCount** *required*	ticketCount	integer(int64)	`"0"`
Query	**ticketType** *optional*	ticketType	string	
Query	**word** *required*	word	string	

Responses

HTTP Code	Description	Schema
200	successful operation	WordzWordResponse

6.92.2. Delete Word

```
DELETE /api/{version}/game/word/delete
```

Description

Delete a word by the given id. The accountId given needs to be the owner or executive to delete.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**wordId** *required*	wordId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

6.92.3. Get Word

```
POST /api/{version}/game/word/get
```

Description

Get a word by the given id.

Parameters

Type	Name	Description	Schema
Path	**version** *required*	version	number(double)
Query	**accountId** *required*	accountId	integer(int64)
Query	**wordId** *required*	wordId	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	WordzWordResponse

6.92.4. Search Words

```
POST /api/{version}/game/word/search
```

Description

Search for words by the given params.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**activeOnly** *required*	activeOnly	boolean	`"false"`
Query	**descending** *required*	descending	boolean	`"false"`
Query	**keyword** *optional*	keyword	string	
Query	**limit** *required*	limit	integer(int32)	`"20"`
Query	**sortField** *required*	sortField	string	`"id"`
Query	**start** *required*	start	integer(int32)	`"0"`

Responses

HTTP Code	Description	Schema
200	successful operation	< WordzWordResponse > array

6.92.5. Update Word

```
POST /api/{version}/game/word/update
```

Description

Update a word by the given params.

Parameters

Type	Name	Description	Schema	Default
Path	**version** *required*	version	number(double)	
Query	**accountId** *required*	accountId	integer(int64)	
Query	**active** *optional*	active	boolean	
Query	**allocateTickets** *optional*	allocateTickets	boolean	
Query	**assetId** *optional*	assetId	integer(int64)	
Query	**definition** *optional*	definition	string	
Query	**metaData** *optional*	metaData	string	
Query	**points** *optional*	points	integer(int64)	
Query	**ticketCount** *required*	ticketCount	integer(int64)	`"0"`
Query	**ticketType** *optional*	ticketType	string	
Query	**word** *optional*	word	string	
Query	**wordId** *required*	wordId	integer(int64)	

Responses

HTTP Code	Description	Schema
200	successful operation	WordzWordResponse

6.93. Workflow

The Workflows service allows you to combine multiple services together to create an entire user flow for a website or application. For example, you have a retail store and you want to create an experience for your customer that will engage them by personalizing their experience. You can create a workflow where when the customer first enters the store, they receive a push notification alert on their device that greets them with a message, shares that there is a promotion on some products that they have purchased in the past. The user then can open up the app, and can use turn-by-turn directions to get to that specific product. In addition, the user adds the offer to their wallet, which also sends an alert to an employee that a customer inside of the store has added an offer to their wallet. The merchant sends a message to the customer, asking if they need any help finding their product or if they have any additional questions. This workflow involves using the Notifications service (Push notification), Employees Service (Notifying the employee that is nearest to the customer), Triggers Service (The trigger mechanism to send the greeting message to the customer when they first walked into the store), Offers Service (when the customer saw the offer), and Wallet Service (When the customer added the offer to their wallet). Combining all of these services created has created a seamless and engaging experience for the customer.

The Workflows service is important because it allows you to configure and customize the best experience for your users. Not all services work best together, so it's recommended that you use our workflows dashboard called Magellan to adjust and optimize the user experience and features you want to define for your product. The dashboard is a 'drag and drop' tool that allows you to visually map each API together to create the desired experience. It's very easy to to add, remove, and modify and feel this is important because it allows you to plan out the types of features that you may want to build into your application or website without yet making any code changes. Another benefit of using the Workflow service is that it will provide you with generated code examples that you can insert directly into your programming environment, rather than having to write the code from scratch. We have setup example workflows on our website to help you get started. Please visit our Developer site to learn more.

6.93.1. Run Workflow

```
POST /api/{version}/workflow/run
```

Description

Runs a published executable workflow

Parameters

Type	Name	Description	Schema
Path	**version** *required*	The API version	number(double)
Query	**accountId** *required*	The account ID of the user	integer(int64)
Query	**parameters** *optional*	Override parameters in Json format. Example: { 'arguments_81': { 'filter': 'PUBLIC' , 'arguments_87': { 'tag': 'custom_tag' } } }	string
Query	**skuId** *optional*	This runs a particular sku on the workflow	integer(int64)
Query	**versionCode** *optional*	This runs a particular sku version on the workflow	integer(int32)
Query	**workflowId** *required*	The workflow to run	integer(int64)

Responses

HTTP Code	Description	Schema
200	successful operation	SirqulResponse

Chapter 7. Definitions

7.1. Account

Name	Schema
aboutUs *optional*	string
accessLevel *optional*	integer(int32)
accountProfile *optional*	AccountProfileInfo
accountType *optional*	string
active *optional*	boolean
address *optional*	Address
appSettings *optional*	< AccountAppSettings > array
application *optional*	Application
audienceInfo_gender *optional*	enum (MALE, FEMALE, ANY)
audiences *optional*	< Audience > array
businessName *optional*	string
businessPhone *optional*	string
businessPhoneExt *optional*	string
cellCarrier *optional*	CellCarrier
cellPhone *optional*	string

Name	Schema
code501c3 *optional*	string
consumersAboutUs *optional*	string
consumersNotes *optional*	string
contentName *optional*	string
contentType *optional*	string
created *optional*	string(date-time)
currentPostalCode *optional*	string
currentPostalCodeLatitude *optional*	number(double)
currentPostalCodeLongitude *optional*	number(double)
currentRegion *optional*	Region
customType *optional*	string
deactivated *optional*	boolean
defaultThreshold *optional*	integer(int64)
dislikeCount *optional*	integer(int64)
display *optional*	string
emailAddress *optional*	string
executive *optional*	boolean
faxNumber *optional*	string

Name	Schema
firstName *optional*	string

Name	Schema
flagEvent *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
flagThresholdEvent *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
flagThreshold_id *optional*	integer(int64)
homePhone *optional*	string
id *optional*	integer(int64)
inviteCode *optional*	string
lastName *optional*	string
latitude *optional*	number(double)
likeCount *optional*	integer(int64)
locale *optional*	string
locationAccuracy *optional*	number(double)
locationDescription *optional*	string
locationLastUpdated *optional*	string(date-time)
locationUpdated *optional*	boolean
longitude *optional*	number(double)
media *optional*	< Asset > array
middleName *optional*	string
partnerActive *optional*	boolean
picture *optional*	Asset
prefix *optional*	string

Name	Schema
premigrationId *optional*	integer(int64)
profileImage *optional*	string
resetToken *optional*	string
retailer *optional*	boolean
retailerGroup *optional*	boolean
retailerInviteCode *optional*	string
retailerLimited *optional*	boolean
retailerLocations *optional*	< RetailerLocation > array
rssiLastReceived *optional*	string(date-time)
suffix *optional*	string
thirdPartyCredentials *optional*	< ThirdPartyCredentials > array
timeZone *optional*	string
title *optional*	string
updated *optional*	string(date-time)
username *optional*	string
utcOffset *optional*	string
valid *optional*	boolean
web *optional*	string

Name	Schema
website *optional*	string

7.2. AccountAppSettings

Name	Schema
account *optional*	Account
accountLinkedToApp *optional*	boolean
active *optional*	boolean
appBlob *optional*	string
application *optional*	Application
created *optional*	string(date-time)
deactivated *optional*	boolean
enableEmail *optional*	boolean
enablePush *optional*	boolean
enableSMS *optional*	boolean
id *optional*	integer(int64)
indexStatus *optional*	enum (NEW, UPDATING, INDEXED)
latitude *optional*	number(double)
locationDescription *optional*	string
locationVisibility *optional*	enum (PUBLIC, PRIVATE, FRIENDS)

Name	Schema
longitude *optional*	number(double)
termsAcceptedDate *optional*	string(date-time)
updated *optional*	string(date-time)
valid *optional*	boolean

7.3. AccountListResponse

Name	Schema
count *optional*	integer(int32)
countTotal *optional*	integer(int64)
errorCode *optional*	string
hasMoreResults *optional*	boolean
items *optional*	< AccountResponse > array
limit *optional*	integer(int32)
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
start *optional*	integer(int32)
startTimeLog *optional*	integer(int64)

Name	Schema
valid *optional*	boolean
version *optional*	number(double)

7.4. AccountLoginResponse

Name	Schema
accountId *optional*	integer(int64)
accountType *optional*	string
display *optional*	string
errorCode *optional*	string
locationDisplay *optional*	string
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
username *optional*	string
valid *optional*	boolean
validated *optional*	boolean
version *optional*	number(double)

7.5. AccountMiniResponse

Name	Schema
accountId *optional*	integer(int64)
accountType *optional*	string
display *optional*	string
locationDisplay *optional*	string
username *optional*	string

7.6. AccountProfileInfo

Name	Schema
account *optional*	Account
active *optional*	boolean
ageGroup *optional*	enum (AGE_0_13, AGE_14_17, AGE_18_22, AGE_23_30, AGE_31_54, AGE_55_PLUS)
birthday *optional*	string(date-time)
bodyType *optional*	string
children *optional*	string
companionship *optional*	string
companionshipIndex *optional*	integer(int32)
created *optional*	string(date-time)
deactivated *optional*	boolean

Name	Schema
developmentPlatforms *optional*	< enum (MAC, WINDOWS, IOS, ANDROID, WINDOWSPHONE, KINDLE, UNITY3D, COCOS2D, HTML5, FACEBOOK) > array
drink *optional*	string
education *optional*	string
educationIndex *optional*	integer(int32)
ethnicity *optional*	string
gender *optional*	enum (MALE, FEMALE, ANY)
height *optional*	string
heightIndex *optional*	integer(int32)
homeLatitude *optional*	number(double)
homeLongitude *optional*	number(double)
id *optional*	integer(int64)
maritalStatus *optional*	string
matchString *optional*	string
matchToken *optional*	integer(int64)
noneZeroMatchToken *optional*	integer(int64)
preferredBodyType *optional*	string
preferredEducation *optional*	string

Name	Schema
preferredEducationIndex *optional*	integer(int32)
preferredEthnicity *optional*	string
preferredGender *optional*	enum (MALE, FEMALE, ANY)
preferredLocation *optional*	string
preferredLocationRange *optional*	number(double)
preferredMaxAge *optional*	integer(int32)
preferredMaxBirthday *optional*	string(date-time)
preferredMaxHeight *optional*	integer(int32)
preferredMinAge *optional*	integer(int32)
preferredMinBirthday *optional*	string(date-time)
preferredMinHeight *optional*	integer(int32)
religion *optional*	string
smoke *optional*	string
updated *optional*	string(date-time)
valid *optional*	boolean

7.7. AccountResponse

Name	Schema
accountId *optional*	integer(int64)

Name	Schema
accountType *optional*	string
active *optional*	boolean
businessName *optional*	string
contact *optional*	ContactResponse
contactEmail *optional*	string
display *optional*	string
gender *optional*	enum (MALE, FEMALE, ANY)
locationDisplay *optional*	string
manager *optional*	AccountMiniResponse
profileImage *optional*	string
username *optional*	string

7.8. AccountShortResponse

Name	Schema
accountId *optional*	integer(int64)
accountType *optional*	string
businessName *optional*	string
contactEmail *optional*	string
display *optional*	string

Name	Schema
gender *optional*	enum (MALE, FEMALE, ANY)
locationDisplay *optional*	string
profileImage *optional*	string
username *optional*	string

7.9. AchievementProgressResponse

Name	Schema
accumulativeCount *optional*	integer(int64)
achievementId *optional*	integer(int64)
achievementMetaData *optional*	string
achievementTierId *optional*	integer(int64)
achievementTitle *optional*	string
achievementType *optional*	string
bestMonthlyStreakCount *optional*	integer(int64)
bestStreakCount *optional*	integer(int64)
bestWeeklyStreakCount *optional*	integer(int64)
completed *optional*	boolean
currentMonthlyStreakCount *optional*	integer(int64)
currentStreakCount *optional*	integer(int64)

Name	Schema
currentWeeklyStreakCount *optional*	integer(int64)
dateUpdated *optional*	integer(int64)
description *optional*	string
endDate *optional*	integer(int64)
errorCode *optional*	string
goalCount *optional*	integer(int64)
icon *optional*	AssetShortResponse
incrementRule *optional*	string
message *optional*	string
progressCount *optional*	integer(int64)
rankType *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startDate *optional*	integer(int64)
startTimeLog *optional*	integer(int64)
tierMetaData *optional*	string
tierType *optional*	string

Name	Schema
title *optional*	string
valid *optional*	boolean
version *optional*	number(double)

7.10. AchievementResponse

Name	Schema
achievementId *optional*	integer(int64)
achievementType *optional*	string
active *optional*	boolean
analyticsTag *optional*	string
appName *optional*	string
created *optional*	integer(int64)
description *optional*	string
errorCode *optional*	string
incrementRule *optional*	string
limitPerDay *optional*	integer(int32)
maxIncrement *optional*	integer(int32)
message *optional*	string
metaData *optional*	string

Name	Schema
minIncrement *optional*	integer(int32)
rankIncrement *optional*	integer(int32)
rankType *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
tierCount *optional*	integer(int32)
tiers *optional*	< AchievementTierResponse > array
title *optional*	string
updated *optional*	integer(int64)
valid *optional*	boolean
validate *optional*	boolean
version *optional*	number(double)

7.11. AchievementShortResponse

Name	Schema
achievementId *optional*	integer(int64)
achievementType *optional*	string

Name	Schema
active *optional*	boolean
analyticsTag *optional*	string
appName *optional*	string
created *optional*	integer(int64)
description *optional*	string
errorCode *optional*	string
incrementRule *optional*	string
limitPerDay *optional*	integer(int32)
maxIncrement *optional*	integer(int32)
message *optional*	string
metaData *optional*	string
minIncrement *optional*	integer(int32)
rankIncrement *optional*	integer(int32)
rankType *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)

Name	Schema
tierCount *optional*	integer(int32)
title *optional*	string
updated *optional*	integer(int64)
valid *optional*	boolean
validate *optional*	boolean
version *optional*	number(double)

7.12. AchievementTierResponse

Name	Schema
achievementTierId *optional*	integer(int64)
created *optional*	integer(int64)
description *optional*	string
gameId *optional*	integer(int64)
gameLevelId *optional*	integer(int64)
gameObjectId *optional*	integer(int64)
goalCount *optional*	integer(int64)
groupAlbumId *optional*	integer(int64)
icon *optional*	AssetShortResponse
metaData *optional*	string

Name	Schema
missionId *optional*	integer(int64)
packId *optional*	integer(int64)
tierType *optional*	string
title *optional*	string
updated *optional*	integer(int64)

7.13. Address

Name	Schema
city *optional*	string
country *optional*	string
countryCode *optional*	string
display *optional*	string
postalCode *optional*	string
state *optional*	string
stateCode *optional*	string
streetAddress *optional*	string
streetAddress2 *optional*	string

7.14. AddressResponse

Name	Schema
city *optional*	string
country *optional*	string
countryCode *optional*	string
display *optional*	string
postalCode *optional*	string
state *optional*	string
stateCode *optional*	string
streetAddress *optional*	string
streetAddress2 *optional*	string

7.15. AgeGroupResponse

Name	Schema
key *optional*	string
toString *optional*	string

7.16. Album

Name	Schema
active *optional*	boolean
albumType *optional*	string
albumTypeId *optional*	integer(int64)

Name	Schema
anonymous *optional*	boolean
appKey *optional*	string
application *optional*	Application
applicationTitle *optional*	string
approvalStatus *optional*	enum (PENDING, REJECTED, APPROVED, FEATURED)
assetCount *optional*	integer(int64)
assets *optional*	< Asset > array
audiences *optional*	< Audience > array
billableId *optional*	integer(int64)
categories *optional*	< Category > array
clickedCount *optional*	integer(int64)
contactInfo *optional*	ContactInfo
contentAsset *optional*	Asset
contentName *optional*	string
coverAsset *optional*	Asset
created *optional*	string(date-time)
customType *optional*	string
deactivated *optional*	boolean

Name	Schema
defaultThreshold *optional*	integer(int64)
description *optional*	string
dislikeCount *optional*	integer(int64)
display *optional*	string
endDate *optional*	string(date-time)
favoriteCount *optional*	integer(int64)
filters *optional*	< Filter > array
flagCount *optional*	integer(int64)

Name	Schema
flagEvent *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
flagThreshold *optional*	FlagThreshold

Name	Schema
flagThresholdEvent *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
flags *optional*	< Flag > array
hasRatings *optional*	boolean
id *optional*	integer(int64)
indexStatus *optional*	enum (NEW, UPDATING, INDEXED)
latitude *optional*	number(double)
likeCount *optional*	integer(int64)
likes *optional*	< YayOrNay > array
locationDescription *optional*	string
longitude *optional*	number(double)
metaData *optional*	string
noteCount *optional*	integer(int64)
notes *optional*	< Note > array
owner *optional*	Account
ownerUsername *optional*	string

Name	Schema
permissionRequestEvent *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
permissionSharedEvent *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
publicPermissions *optional*	Permissions
secondaryType *optional*	string
sharedCount *optional*	integer(int64)
shortTitle *optional*	string
startDate *optional*	string(date-time)
subType *optional*	string
tags *optional*	string
title *optional*	string
updated *optional*	string(date-time)
userPermissions *optional*	< UserPermissions > array
valid *optional*	boolean
viewedCount *optional*	integer(int64)
visibility *optional*	enum (PUBLIC, PRIVATE, FRIENDS)

7.17. AlbumContestListResponse

Name	Schema
count *optional*	integer(int32)
countTotal *optional*	integer(int64)
errorCode *optional*	string

Name	Schema
hasMoreResults *optional*	boolean
items *optional*	< AlbumContestResponse > array
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.18. AlbumContestResponse

Name	Schema
album1 *optional*	AlbumFullResponse
album2 *optional*	AlbumFullResponse
albumContestId *optional*	integer(int64)
albums *optional*	< AlbumFullResponse > array
approvalStatus *optional*	enum (PENDING, REJECTED, APPROVED, FEATURED)
comments *optional*	< NoteFullResponse > array
commentsCount *optional*	integer(int32)

Name	Schema
connectionGroups *optional*	< ConnectionGroupShortResponse > array
connections *optional*	< ConnectionResponse > array
contestType *optional*	string
dateCreated *optional*	integer(int64)
dateUpdated *optional*	integer(int64)
description *optional*	string
dislikeCount *optional*	integer(int64)
distance *optional*	number(double)
endDate *optional*	integer(int64)
errorCode *optional*	string
hasLiked *optional*	boolean
latitude *optional*	number(double)
likeCount *optional*	integer(int64)
likes *optional*	< LikeResponse > array
locationDescription *optional*	string
longitude *optional*	number(double)
message *optional*	string
metaData *optional*	string

Name	Schema
owner *optional*	AccountShortResponse
publicPermissions *optional*	Permissions
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startDate *optional*	integer(int64)
startTimeLog *optional*	integer(int64)
title *optional*	string
userCount *optional*	integer(int32)
userPermissions *optional*	Permissions
valid *optional*	boolean
version *optional*	number(double)
visibility *optional*	enum (PUBLIC, PRIVATE, FRIENDS)
voteCount *optional*	integer(int64)

7.19. AlbumFullResponse

Name	Schema
albumAdmin *optional*	boolean
albumId *optional*	integer(int64)

Name	Schema
albumMember *optional*	boolean
albumType *optional*	string
albumTypeId *optional*	integer(int64)
anonymous *optional*	boolean
approvalStatus *optional*	enum (PENDING, REJECTED, APPROVED, FEATURED)
assetCount *optional*	integer(int32)
assets *optional*	< AssetResponse > array
audiences *optional*	< AudienceResponse > array
categories *optional*	< CategoryResponse > array
cellPhone *optional*	string
clickedCount *optional*	integer(int64)
comments *optional*	< NoteFullResponse > array
commentsCount *optional*	integer(int64)
completableAction *optional*	CompletableActionResponse
connectionGroups *optional*	< ConnectionGroupShortResponse > array
connections *optional*	< ConnectionResponse > array
coverAsset *optional*	AssetResponse
dateCreated *optional*	integer(int64)

Name	Schema
dateUpdated *optional*	integer(int64)
description *optional*	string
dislikeCount *optional*	integer(int64)
distance *optional*	number(double)
endDate *optional*	integer(int64)
errorCode *optional*	string
favoritableId *optional*	integer(int64)
favoritableType *optional*	string
favorite *optional*	boolean
favoriteCount *optional*	integer(int64)
favoriteId *optional*	integer(int64)
firstAlbumIdInStack *optional*	integer(int64)
firstFriend *optional*	AccountMiniResponse
flagCount *optional*	integer(int64)
fullAddress *optional*	string
hasDisliked *optional*	boolean
hasLiked *optional*	boolean
hasRatings *optional*	boolean

Name	Schema
lastAlbumIdInStack *optional*	integer(int64)
latitude *optional*	number(double)
likeCount *optional*	integer(int64)
liked *optional*	boolean
likes *optional*	< LikeResponse > array
locationDescription *optional*	string
longitude *optional*	number(double)
message *optional*	string
metFlagThreshold *optional*	boolean
metaData *optional*	string
notifications *optional*	boolean
owner *optional*	AccountShortResponse
ownerId *optional*	integer(int64)
pendingAlbumMember *optional*	boolean
publicPermissions *optional*	Permissions
request *optional*	< NameStringValueResponse > array
returning *optional*	string
rollingWindowEnd *optional*	integer(int64)

Name	Schema
rollingWindowNumber *optional*	integer(int64)
rollingWindowStart *optional*	integer(int64)
serializeNulls *optional*	boolean
sharedCount *optional*	integer(int64)
startDate *optional*	integer(int64)
startTimeLog *optional*	integer(int64)
subType *optional*	string
tags *optional*	string
title *optional*	string
totalAlbumsInStack *optional*	integer(int64)
totalFriendsCount *optional*	integer(int64)
totalPeopleCount *optional*	integer(int64)
userCount *optional*	integer(int32)
userPermissions *optional*	Permissions
valid *optional*	boolean
version *optional*	number(double)
viewedCount *optional*	integer(int64)
visibility *optional*	enum (PUBLIC, PRIVATE, FRIENDS)

7.20. AlbumResponse

Name	Schema
albumId *optional*	integer(int64)
albumType *optional*	string
albumTypeId *optional*	integer(int64)
anonymous *optional*	boolean
approvalStatus *optional*	enum (PENDING, REJECTED, APPROVED, FEATURED)
assetCount *optional*	integer(int32)
audiences *optional*	< AudienceResponse > array
categories *optional*	< CategoryResponse > array
cellPhone *optional*	string
clickedCount *optional*	integer(int64)
commentsCount *optional*	integer(int64)
coverAsset *optional*	AssetResponse
dateCreated *optional*	integer(int64)
dateUpdated *optional*	integer(int64)
description *optional*	string
dislikeCount *optional*	integer(int64)
distance *optional*	number(double)

Name	Schema
endDate *optional*	integer(int64)
errorCode *optional*	string
favoritableId *optional*	integer(int64)
favoritableType *optional*	string
favorite *optional*	boolean
favoriteCount *optional*	integer(int64)
favoriteId *optional*	integer(int64)
flagCount *optional*	integer(int64)
fullAddress *optional*	string
hasDisliked *optional*	boolean
hasLiked *optional*	boolean
hasRatings *optional*	boolean
latitude *optional*	number(double)
likeCount *optional*	integer(int64)
liked *optional*	boolean
locationDescription *optional*	string
longitude *optional*	number(double)
message *optional*	string

Name	Schema
metFlagThreshold *optional*	boolean
metaData *optional*	string
notifications *optional*	boolean
owner *optional*	AccountShortResponse
ownerId *optional*	integer(int64)
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
sharedCount *optional*	integer(int64)
startDate *optional*	integer(int64)
startTimeLog *optional*	integer(int64)
subType *optional*	string
tags *optional*	string
title *optional*	string
userCount *optional*	integer(int32)
valid *optional*	boolean
version *optional*	number(double)
viewedCount *optional*	integer(int64)

Name	Schema
visibility *optional*	enum (PUBLIC, PRIVATE, FRIENDS)

7.21. AnswerResponse

Name	Schema
answer *optional*	string
answerId *optional*	integer(int64)
correct *optional*	boolean
image *optional*	AssetShortResponse
videoURL *optional*	string

7.22. AppInfoResponse

Name	Schema
appBlob *optional*	string
completedAchievements *optional*	< AchievementProgressResponse > array
enableEmail *optional*	boolean
enablePush *optional*	boolean
enableSMS *optional*	boolean
locationVisibility *optional*	string
maxPoints *optional*	integer(int64)
maxTickets *optional*	integer(int64)

Name	Schema
pointToTicketModifier *optional*	number(float)
points *optional*	integer(int64)
purchaseItemListResponse *optional*	PurchaseItemListResponse
rank *optional*	string
requiresTermsAcceptance *optional*	boolean
scoringType *optional*	enum (GAME_LEVEL, GAME_OBJECT)
termsAcceptedDate *optional*	integer(int64)
ticketCounts *optional*	< TicketCountResponse > array
ticketsAvailable *optional*	integer(int64)
ticketsEarned *optional*	integer(int64)
ticketsPurchased *optional*	integer(int64)
ticketsUsed *optional*	integer(int64)
wipAchievements *optional*	< AchievementProgressResponse > array

7.23. AppResponse

Name	Schema
errorCode *optional*	string
message *optional*	string
messages *optional*	MessageListResponse

Name	Schema
missions *optional*	MissionListResponse
profile *optional*	ProfileResponse
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.24. AppVersion

Name	Schema
majorVersion *optional*	integer(int32)
minorVersion *optional*	integer(int32)
versionIndex *optional*	integer(int32)

7.25. Application

Name	Schema
about *optional*	string
active *optional*	boolean
activeServers *optional*	string

Name	Schema
amqpEnabled *optional*	boolean
appDescription *optional*	string
appIcon *optional*	Asset
appKey *optional*	string
appLogo *optional*	Asset
appName *optional*	string
appOwners *optional*	< Account > array
appRestKey *optional*	string
appTokenKey *optional*	string
appType *optional*	string
appleAppId *optional*	string
authorizedCount *optional*	integer(int32)
authorizedServers *optional*	string
baiduApiKey *optional*	string
baiduSecretKey *optional*	string
billableEntity *optional*	BillableEntity
bundleId *optional*	string
categories *optional*	< Category > array

Name	Schema
certificate *optional*	BlobFile
certificateLastUpdated *optional*	string(date-time)
certificateName *optional*	string
certificateUrl *optional*	string
created *optional*	string(date-time)
cssClass *optional*	string
currentApiVersion *optional*	string
currentApiVersionAsFloat *optional*	number(float)
currentBuildVersion *optional*	string
deactivated *optional*	boolean
defaultImage *optional*	string
downloadUrls *optional*	string
enableAPNSBadge *optional*	boolean
enableEmailValidation *optional*	boolean
eulaDate *optional*	string(date-time)
eulaVersion *optional*	string
facebookAppId *optional*	string
facebookAppSecret *optional*	string

Name	Schema
globalTickets *optional*	boolean
googleApiKey *optional*	string
googleAppId *optional*	string
googleOAuthAccessToken *optional*	string
googleOAuthClientId *optional*	string
googleOAuthRefreshToken *optional*	string
googleOAuthSecret *optional*	string
griplyBaseUrl *optional*	string
hasAdvertisements *optional*	boolean
hasGameData *optional*	boolean
hintCost *optional*	integer(int32)
id *optional*	integer(int64)
includeInReport *optional*	boolean
invitePageUrl *optional*	string
inviteWelcomeText *optional*	string
jpushAppKey *optional*	string
jpushMasterSecret *optional*	string
landingPageUrl *optional*	string

Name	Schema
licensePeriod *optional*	string(date-time)
localAdsEnabled *optional*	boolean
maxPointsPerScore *optional*	integer(int32)
metaData *optional*	string
minimumSessionLength *optional*	integer(int32)
modules *optional*	string
objectStoreSession *optional*	string
placements *optional*	< Placement > array
platforms *optional*	< Platform > array
processSessionData *optional*	boolean
productionMode *optional*	boolean
publicContentApproval *optional*	boolean
publicNotifications *optional*	boolean
retailer *optional*	Retailer
scoringType *optional*	enum (GAME_LEVEL, GAME_OBJECT)
sessionGapLength *optional*	integer(int32)
showInActivities *optional*	boolean
signupScheduledTask *optional*	integer(int64)

Name	Schema
sqootApiKey *optional*	string
ticketsPerPoint *optional*	number(float)
title *optional*	string
trilatSettings *optional*	TrilatAppSettings
twitterAppId *optional*	string
twitterAppSecret *optional*	string
twitterCallbackURL *optional*	string
uniqueUserTag *optional*	string
updated *optional*	string(date-time)
urlScheme *optional*	string
useAppCache *optional*	boolean
useMatchingAlgorithm *optional*	boolean
valid *optional*	boolean
xgPushSecretKey *optional*	string

7.26. ApplicationConfigResponse

Name	Schema
assetId *optional*	integer(int64)
configId *optional*	integer(int64)

Name	Schema
configJson *optional*	string
configJsonURL *optional*	string
configVersion *optional*	string
created *optional*	integer(int64)
errorCode *optional*	string
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
updated *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.27. ApplicationMiniResponse

Name	Schema
active *optional*	boolean
appIcon *optional*	AssetShortResponse
appKey *optional*	string

Name	Schema
appLogo *optional*	AssetShortResponse
applicationId *optional*	integer(int64)
currentApiVersion *optional*	string
currentBuildVersion *optional*	string
errorCode *optional*	string
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
title *optional*	string
valid *optional*	boolean
version *optional*	number(double)

7.28. ApplicationResponse

Name	Schema
about *optional*	string
active *optional*	boolean
activityDescription *optional*	string

Name	Schema
amqpEnabled *optional*	boolean
apnsCertificateFileName *optional*	string
appIcon *optional*	AssetShortResponse
appKey *optional*	string
appLogo *optional*	AssetShortResponse
appName *optional*	string
appRestKey *optional*	string
appToken *optional*	string
appType *optional*	string
applicationId *optional*	integer(int64)
authorizedCount *optional*	integer(int32)
authorizedServers *optional*	string
billable *optional*	BillableEntityShortResponse
billableEntityName *optional*	string
bundleId *optional*	string
categories *optional*	< CategoryResponse > array
created *optional*	integer(int64)
cssClass *optional*	string

Name	Schema
currentApiVersion *optional*	string
currentBuildVersion *optional*	string
defaultImageFullUrl *optional*	string
downloadUrls *optional*	string
enableAPNSBadge *optional*	boolean
errorCode *optional*	string
eulaDateUpdated *optional*	integer(int64)
eulaVersion *optional*	string
facebookAppId *optional*	string
facebookAppSecret *optional*	string
global *optional*	boolean
globalTickets *optional*	boolean
googleApiKey *optional*	string
hasAds *optional*	boolean
hasGameData *optional*	boolean
hintCost *optional*	integer(int32)
invitePageUrl *optional*	string
inviteWelcomeText *optional*	string

Name	Schema
landingPageFullUrl *optional*	string
landingPageUrl *optional*	string
localAdsEnabled *optional*	boolean
maxScore *optional*	integer(int32)
message *optional*	string
metaData *optional*	string
minimumSessionLength *optional*	integer(int32)
modules *optional*	string
owner *optional*	AccountMiniResponse
placementCount *optional*	integer(int32)
platforms *optional*	< PlatformResponse > array
productionMode *optional*	boolean
publicContentApproval *optional*	boolean
publicNotifications *optional*	boolean
request *optional*	< NameStringValueResponse > array
returning *optional*	string
scoringType *optional*	enum (GAME_LEVEL, GAME_OBJECT)
serializeNulls *optional*	boolean

Name	Schema
sessionGapLength *optional*	integer(int32)
showInActivities *optional*	boolean
sqootApiKey *optional*	string
startTimeLog *optional*	integer(int64)
ticketsPerPoint *optional*	number(float)
title *optional*	string
trilatSettings *optional*	TrilatAppSettings
updated *optional*	integer(int64)
urlScheme *optional*	string
valid *optional*	boolean
version *optional*	number(double)

7.29. ApplicationSettingsResponse

Name	Schema
application *optional*	ApplicationMiniResponse
created *optional*	integer(int64)
termsAcceptedDate *optional*	integer(int64)
updated *optional*	integer(int64)

7.30. ApplicationShortResponse

Name	Schema
about *optional*	string
active *optional*	boolean
activityDescription *optional*	string
appIcon *optional*	AssetShortResponse
appKey *optional*	string
appLogo *optional*	AssetShortResponse
applicationId *optional*	integer(int64)
billable *optional*	BillableEntityShortResponse
categories *optional*	< CategoryResponse > array
created *optional*	integer(int64)
currentApiVersion *optional*	string
currentBuildVersion *optional*	string
defaultImageFullUrl *optional*	string
downloadUrls *optional*	string
errorCode *optional*	string
global *optional*	boolean
hasAds *optional*	boolean

Name	Schema
landingPageFullUrl *optional*	string
landingPageUrl *optional*	string
localAdsEnabled *optional*	boolean
message *optional*	string
metaData *optional*	string
placementCount *optional*	integer(int32)
platforms *optional*	< PlatformResponse > array
publicNotifications *optional*	boolean
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
title *optional*	string
trilatSettings *optional*	TrilatAppSettings
updated *optional*	integer(int64)
urlScheme *optional*	string
valid *optional*	boolean
version *optional*	number(double)

7.31. ApplicationUsageResponse

Name	Schema
applicationId *optional*	integer(int64)
balance *optional*	number(double)
billableEntity *optional*	BillableEntityResponse
invoiceDate *optional*	string(date-time)
maxCoupons *optional*	integer(int64)
maxEvents *optional*	integer(int64)
maxLocations *optional*	integer(int64)
maxMedia *optional*	integer(int64)
maxMissions *optional*	integer(int64)
maxProducts *optional*	integer(int64)
maxVouchers *optional*	integer(int64)
totalDataStorage *optional*	integer(int64)
totalMediaStorage *optional*	integer(int64)
totalNotifications *optional*	integer(int64)
totalRequests *optional*	integer(int64)

7.32. Asset

Name	Schema
active *optional*	boolean
albums *optional*	< Album > array
appKey *optional*	string
application *optional*	Application
assignment *optional*	Assignment
attachedAssetId *optional*	integer(int64)
billableId *optional*	integer(int64)
caption *optional*	string
contentName *optional*	string
created *optional*	string(date-time)
customType *optional*	string
deactivated *optional*	boolean
defaultAlbumType *optional*	enum (PROFILE, DEFAULT_PROFILE_AVATAR, DEFAULT_APP_ICON, DEFAULT_APP_LOGO)
defaultThreshold *optional*	integer(int64)
dislikeCount *optional*	integer(int64)
display *optional*	string
fileName *optional*	string
flagCount *optional*	integer(int64)

Name	Schema
flagEvent *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
flagThreshold *optional*	FlagThreshold

Name	Schema
flagThresholdEvent *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
flags *optional*	< Flag > array
folderUri *optional*	string
id *optional*	integer(int64)
latitude *optional*	number(double)
likeCount *optional*	integer(int64)
likes *optional*	< YayOrNay > array
locationDescription *optional*	string
longitude *optional*	number(double)
mediaType *optional*	string
metaData *optional*	string
mimeType *optional*	string
newOwnerId *optional*	integer(int64)
noteCount *optional*	integer(int64)
notes *optional*	< Note > array
owner *optional*	Account
premigrationId *optional*	integer(int64)
subType *optional*	string
updated *optional*	string(date-time)

Name	Schema
urifolder *optional*	string(uri)
valid *optional*	boolean
versionCode *optional*	integer(int32)
versionName *optional*	string

7.33. AssetFullResponse

Name	Schema
assetId *optional*	integer(int64)
assetVersion *optional*	string
attachedAssetURL *optional*	string
attachedMediaType *optional*	string
caption *optional*	string
comments *optional*	< NoteFullResponse > array
commentsCount *optional*	integer(int32)
coverURL *optional*	string
created *optional*	integer(int64)
display *optional*	string
errorCode *optional*	string
fileName *optional*	string

Name	Schema
fullURL *optional*	string
hasDisliked *optional*	boolean
hasLiked *optional*	boolean
latitude *optional*	number(double)
likes *optional*	< LikeResponse > array
likesCount *optional*	integer(int64)
locationDescription *optional*	string
longitude *optional*	number(double)
mediaType *optional*	string
message *optional*	string
metFlagThreshold *optional*	boolean
metaData *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
searchTags *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
thumbnailURL *optional*	string

Name	Schema
updated *optional*	integer(int64)
uploader *optional*	AccountShortResponse
valid *optional*	boolean
version *optional*	number(double)
versionCode *optional*	integer(int32)
versionName *optional*	string

7.34. AssetListResponse

Name	Schema
count *optional*	integer(int32)
errorCode *optional*	string
hasMoreResults *optional*	boolean
items *optional*	< AssetResponse > array
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
valid *optional*	boolean

Name	Schema
version *optional*	number(double)

7.35. AssetResponse

Name	Schema
assetId *optional*	integer(int64)
assetVersion *optional*	string
attachedAssetURL *optional*	string
attachedMediaType *optional*	string
caption *optional*	string
commentsCount *optional*	integer(int32)
coverURL *optional*	string
created *optional*	integer(int64)
display *optional*	string
errorCode *optional*	string
fileName *optional*	string
fullURL *optional*	string
latitude *optional*	number(double)
likesCount *optional*	integer(int64)
locationDescription *optional*	string

Name	Schema
longitude *optional*	number(double)
mediaType *optional*	string
message *optional*	string
metaData *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
searchTags *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
thumbnailURL *optional*	string
updated *optional*	integer(int64)
uploader *optional*	AccountShortResponse
valid *optional*	boolean
version *optional*	number(double)
versionCode *optional*	integer(int32)
versionName *optional*	string

7.36. AssetShortResponse

Name	Schema
assetId *optional*	integer(int64)
assetVersion *optional*	string
caption *optional*	string
coverURL *optional*	string
errorCode *optional*	string
fileName *optional*	string
fullURL *optional*	string
mediaType *optional*	string
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
thumbnailURL *optional*	string
valid *optional*	boolean
version *optional*	number(double)
versionCode *optional*	integer(int32)
versionName *optional*	string

7.37. Assignment

Name	Schema
active *optional*	boolean
application *optional*	Application
approvalStatus *optional*	enum (PENDING, REJECTED, APPROVED, FEATURED)
assets *optional*	< Asset > array
assignee *optional*	Account
contentAsset *optional*	Asset
contentName *optional*	string
created *optional*	string(date-time)
creator *optional*	Account
currentStatus *optional*	AssignmentStatus
currentStatusType *optional*	enum (NEW, IN_PROGRESS, SUBSCRIBED, ARCHIVED)
customType *optional*	string
deactivated *optional*	boolean
description *optional*	string
id *optional*	integer(int64)
lastUpdatedBy *optional*	Account
metaData *optional*	string

Name	Schema
name *optional*	string
noteCount *optional*	integer(int64)
notes *optional*	< Note > array
owner *optional*	Account

Name	Schema
permissionRequestEvent *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
permissionSharedEvent *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
publicPermissions *optional*	Permissions
retailerLocation *optional*	RetailerLocation
statuses *optional*	< AssignmentStatus > array
updated *optional*	string(date-time)
userPermissions *optional*	< UserPermissions > array
valid *optional*	boolean
visibility *optional*	enum (PUBLIC, PRIVATE, FRIENDS)

7.38. AssignmentResponse

Name	Schema
active *optional*	boolean
assignee *optional*	AccountShortResponse
assignmentId *optional*	integer(int64)
created *optional*	integer(int64)
creator *optional*	AccountShortResponse
currentStatus *optional*	AssignmentStatusResponse
currentStatusType *optional*	enum (NEW, IN_PROGRESS, SUBSCRIBED, ARCHIVED)
description *optional*	string
lastReportedBy *optional*	AccountShortResponse

Name	Schema
location *optional*	RetailerLocationShortResponse
metaData *optional*	string
name *optional*	string
updated *optional*	integer(int64)

7.39. AssignmentStatus

Name	Schema
active *optional*	boolean
assignment *optional*	Assignment
closure *optional*	enum (PHONE, SITE_VISIT, PHONE_SITE, WEB)
connection *optional*	enum (INITIAL, FOLLOW_UP, DECLINED)
created *optional*	string(date-time)
creator *optional*	Account
deactivated *optional*	boolean
followUp *optional*	string(date-time)
id *optional*	integer(int64)
message *optional*	string
method *optional*	enum (PHONE, SITE_VISIT, EMAIL)
notification *optional*	ScheduledNotification

Name	Schema
status *optional*	enum (ARCHIVED, SUBSCRIBED, UNSUBSCRIBED, CONTACTED, DECLINED, NOT_CONTACTED)
toDo *optional*	enum (SITE_VISIT, PHONE, RECONTACT, RENEWAL, CREDIT)
updated *optional*	string(date-time)
valid *optional*	boolean

7.40. AssignmentStatusResponse

Name	Schema
active *optional*	boolean
assignmentStatusId *optional*	integer(int64)
closure *optional*	enum (PHONE, SITE_VISIT, PHONE_SITE, WEB)
connection *optional*	enum (INITIAL, FOLLOW_UP, DECLINED)
created *optional*	integer(int64)
followUp *optional*	integer(int64)
message *optional*	string
method *optional*	enum (PHONE, SITE_VISIT, EMAIL)
notification *optional*	ScheduledNotificationShortResponse
status *optional*	enum (ARCHIVED, SUBSCRIBED, UNSUBSCRIBED, CONTACTED, DECLINED, NOT_CONTACTED)
toDo *optional*	enum (SITE_VISIT, PHONE, RECONTACT, RENEWAL, CREDIT)
updated *optional*	integer(int64)

7.41. Audience

Name	Schema
active *optional*	boolean
ageGroups *optional*	< enum (AGE_0_13, AGE_14_17, AGE_18_22, AGE_23_30, AGE_31_54, AGE_55_PLUS) > array
application *optional*	Application
applications *optional*	< Application > array
associateContentName *optional*	string
associateId *optional*	integer(int64)
associateType *optional*	string
audienceType *optional*	string
categories *optional*	< Category > array
cohortRegions *optional*	< CohortRegion > array
created *optional*	string(date-time)
deactivated *optional*	boolean
description *optional*	string
deviceVersionRanges *optional*	< AudienceDeviceVersionRange > array
display *optional*	boolean
endTimeOffset *optional*	integer(int32)
gameExperienceLevel *optional*	string

Name	Schema
gender *optional*	enum (MALE, FEMALE, ANY)
groupingId *optional*	string
id *optional*	integer(int64)
latitude *optional*	number(double)
locationDescription *optional*	string
locations *optional*	< AudienceLocation > array
longitude *optional*	number(double)
metaData *optional*	string
name *optional*	string
owner *optional*	Account
polygon *optional*	string
radius *optional*	number(double)
sendSuggestion *optional*	boolean
startTimeOffset *optional*	integer(int32)
target *optional*	AudienceTargetType
trilaterationTypes *optional*	< enum (WIFI_PROBE_REQUEST, IBEACON, BLE, ESTIMOTE, EDDYSTONE_UID, EDDYSTONE_URL, EDDYSTONE_TLM, EDDYSTONE_EID, UNKNOWN) > array
updated *optional*	string(date-time)

Name	Schema
useOrder *optional*	boolean
valid *optional*	boolean
visibility *optional*	string

7.42. AudienceDevice

Name	Schema
active *optional*	boolean
created *optional*	string(date-time)
deactivated *optional*	boolean
id *optional*	integer(int64)
maximum *optional*	integer(int64)
minimum *optional*	integer(int64)
name *optional*	string
updated *optional*	string(date-time)
valid *optional*	boolean

7.43. AudienceDeviceResponse

Name	Schema
active *optional*	boolean
deviceId *optional*	integer(int64)

Name	Schema
display *optional*	string
maximum *optional*	integer(int64)
minimum *optional*	integer(int64)
name *optional*	string

7.44. AudienceDeviceVersionRange

Name	Schema
active *optional*	boolean
created *optional*	string(date-time)
deactivated *optional*	boolean
device *optional*	AudienceDevice
id *optional*	integer(int64)
maximum *optional*	integer(int64)
minimum *optional*	integer(int64)
updated *optional*	string(date-time)
valid *optional*	boolean

7.45. AudienceLocation

Name	Schema
active *optional*	boolean

Name	Schema
created *optional*	string(date-time)
deactivated *optional*	boolean
id *optional*	integer(int64)
latitude *optional*	number(double)
locationDescription *optional*	string
longitude *optional*	number(double)
radius *optional*	number(double)
updated *optional*	string(date-time)
valid *optional*	boolean

7.46. AudienceResponse

Name	Schema
accountCount *optional*	integer(int64)
active *optional*	boolean
ageGroups *optional*	< AgeGroupResponse > array
albumTypesCounts *optional*	< string, integer(int64) > map
applications *optional*	< ApplicationShortResponse > array
associateDescription *optional*	string
associateId *optional*	integer(int64)

Name	Schema
associateType *optional*	string
categories *optional*	< CategoryResponse > array
description *optional*	string
devices *optional*	< AudienceDeviceResponse > array
gameExperienceLevel *optional*	string
gender *optional*	string
groupingId *optional*	string
id *optional*	integer(int64)
latitude *optional*	number(double)
locations *optional*	< GeocodeLatLngResponse > array
longitude *optional*	number(double)
metaData *optional*	string
name *optional*	string
ownerId *optional*	integer(int64)
radius *optional*	number(double)
searchTags *optional*	string
sendSuggestion *optional*	boolean
useOrder *optional*	boolean

Name	Schema
visibility *optional*	string

7.47. AudienceTargetType

Name	Schema
account *optional*	Account
offerLocation *optional*	OfferLocation
region *optional*	Region
retailerLocation *optional*	RetailerLocation
targetDescription *optional*	string
targetType *optional*	enum (RETAILER_LOCATION, OFFER_LOCATION, ACCOUNT, REGION)

7.48. AvailabilityResponse

Name	Schema
availabilityId *optional*	integer(int64)
dayOfWeek *optional*	integer(int32)
deleted *optional*	boolean
endDate *optional*	integer(int64)
endTime *optional*	integer(int64)
startDate *optional*	integer(int64)
startTime *optional*	integer(int64)

Name	Schema
timeZone *optional*	string

7.49. BaseOfferResponse

Name	Schema
activated *optional*	integer(int64)
active *optional*	boolean
addedCount *optional*	integer(int32)
addedLimit *optional*	integer(int32)
availabilitySummary *optional*	string
barcode *optional*	string
billableEntityId *optional*	integer(int64)
clickedCount *optional*	integer(int32)
deleted *optional*	integer(int64)
details *optional*	string
discountPrice *optional*	number(double)
dislikeCount *optional*	integer(int64)
expiration *optional*	integer(int64)
externalId *optional*	string
externalRedeemOptions *optional*	string

Name	Schema
externalUrl *optional*	string
favorite *optional*	boolean
favoriteCount *optional*	integer(int64)
flagCount *optional*	integer(int64)
flagThreshold *optional*	integer(int64)
fullPrice *optional*	number(double)
image *optional*	string
image1 *optional*	string
image2 *optional*	string
image3 *optional*	string
image4 *optional*	string
image5 *optional*	string
images *optional*	< AssetResponse > array
likeCount *optional*	integer(int64)
metaData *optional*	string
noteCount *optional*	integer(int64)
offerType *optional*	string
offerVisibility *optional*	string

Name	Schema
parentOffer *optional*	BaseOfferResponse
publisher *optional*	string
redeemableEnd *optional*	integer(int64)
redeemableStart *optional*	integer(int64)
removedCount *optional*	integer(int32)
responsibleAccountId *optional*	integer(int64)
specialOfferType *optional*	enum (ALL, RESERVABLE, REGULAR_OFFER, ACT_NOW, GET_THERE_NOW, SQOOT, TICKETS, YIPIT)
subDetails *optional*	string
subTitle *optional*	string
ticketsReward *optional*	integer(int64)
ticketsRewardType *optional*	string
title *optional*	string
usedCount *optional*	integer(int32)
viewedCount *optional*	integer(int32)

7.50. BidResponse

Name	Schema
amountPerAction *optional*	number(double)
amountPerView *optional*	number(double)

Name	Schema
bidId *optional*	integer(int64)
biddableId *optional*	integer(int64)
biddableType *optional*	string
budgetAmount *optional*	number(double)
budgetSchedule *optional*	enum (DAILY, WEEKLY, MONTHLY, YEARLY)
currentBudget *optional*	number(double)
currentBudgetExpiration *optional*	integer(int64)
currentBudgetStart *optional*	integer(int64)
currentDailyBudget *optional*	number(double)

7.51. BillableEntity

Name	Schema
active *optional*	boolean
authorizeNetApiKey *optional*	string
authorizeNetTransactionKey *optional*	string
categories *optional*	< Category > array
contactInfo *optional*	ContactInfo
created *optional*	string(date-time)
deactivated *optional*	boolean

Name	Schema
disbursementIntrospect *optional*	ThirdPartyNetwork
filters *optional*	< Filter > array
id *optional*	integer(int64)
managers *optional*	< Account > array
media *optional*	< Asset > array
mediaAlbum *optional*	Album
name *optional*	string
responsible *optional*	Account
retailers *optional*	< Retailer > array
subscription *optional*	Subscription
updated *optional*	string(date-time)
valid *optional*	boolean

7.52. BillableEntityResponse

Name	Schema
billableEntityId *optional*	integer(int64)
contact *optional*	ContactInfoResponse
name *optional*	string
payment *optional*	PaymentTypesResponse

Name	Schema
responsible *optional*	ProfileShortResponse
subscription *optional*	SubscriptionResponse

7.53. BillableEntityShortResponse

Name	Schema
billableEntityId *optional*	integer(int64)
name *optional*	string
responsible *optional*	ProfileShortResponse

7.54. BiometricImage

Name	Schema
data *optional*	string
position *optional*	enum (UNKNOWN, LEFT_INDEX, LEFT_MIDDLE, LEFT_RING, LEFT_LITTLE, LEFT_THUMB, RIGHT_INDEX, RIGHT_MIDDLE, RIGHT_RING, RIGHT_LITTLE, RIGHT_THUMB, LEFT, RIGHT)

7.55. BiometricRequest

Name	Schema
face *optional*	BiometricImage
fingerprints *optional*	< FingerprintBiometricImage > array

7.56. BlobFile

Name	Schema
active *optional*	boolean
application *optional*	Application
created *optional*	string(date-time)
deactivated *optional*	boolean
file *optional*	< string(byte) > array
fileName *optional*	string
folderUri *optional*	string
id *optional*	integer(int64)
owner *optional*	Account
updated *optional*	string(date-time)
valid *optional*	boolean

7.57. Building

Name	Schema
active *optional*	boolean
buildingType *optional*	string
created *optional*	string(date-time)
deactivated *optional*	boolean
id *optional*	integer(int64)

Name	Schema
name *optional*	string
updated *optional*	string(date-time)
valid *optional*	boolean

7.58. CargoType

Name	Schema
active *optional*	boolean
adminOnly *optional*	boolean
business *optional*	Retailer
created *optional*	string(date-time)
deactivated *optional*	boolean
depth *optional*	number(double)
height *optional*	number(double)
hub *optional*	ServiceHub
id *optional*	integer(int64)
loadTime *optional*	integer(int64)
name *optional*	string
palletRatio *optional*	integer(int32)
palletizable *optional*	boolean

Name	Schema
updated *optional*	string(date-time)
valid *optional*	boolean
volume *optional*	number(double)
weight *optional*	number(double)
width *optional*	number(double)

7.59. Category

Name	Schema
active *optional*	boolean
application *optional*	Application
applicationId *optional*	integer(int64)
asset *optional*	Asset
children *optional*	< Category > array
childrenCount *optional*	integer(int64)
created *optional*	string(date-time)
deactivated *optional*	boolean
description *optional*	string
display *optional*	string
externalId *optional*	string

Name	Schema
externalType *optional*	string
favoriteCount *optional*	integer(int64)
id *optional*	integer(int64)
metaData *optional*	string
name *optional*	string
parent *optional*	Category
parentId *optional*	integer(int64)
parentName *optional*	string
root *optional*	Category
secondaryType *optional*	string
shortName *optional*	string
sqootSlug *optional*	string
tree *optional*	string
type *optional*	string
updated *optional*	string(date-time)
valid *optional*	boolean

7.60. CategoryResponse

Name	Schema
active *optional*	boolean
applicationId *optional*	integer(int64)
asset *optional*	AssetShortResponse
categoryId *optional*	integer(int64)
description *optional*	string
display *optional*	string
externalId *optional*	string
externalType *optional*	string
favorite *optional*	boolean
favoriteId *optional*	integer(int64)
name *optional*	string
parentId *optional*	integer(int64)
parentName *optional*	string
subCategoryCount *optional*	integer(int64)
type *optional*	string

7.61. CategoryTreeResponse

Name	Schema
active *optional*	boolean

Name	Schema
applicationId *optional*	integer(int64)
asset *optional*	AssetShortResponse
categoryId *optional*	integer(int64)
children *optional*	< CategoryTreeResponse > array
description *optional*	string
display *optional*	string
externalId *optional*	string
externalType *optional*	string
favorite *optional*	boolean
favoriteId *optional*	integer(int64)
name *optional*	string
parentId *optional*	integer(int64)
parentName *optional*	string
subCategoryCount *optional*	integer(int64)
type *optional*	string

7.62. CellCarrier

Name	Schema
active *optional*	boolean

Name	Schema
apiUrl *optional*	string
authUrl *optional*	string
clientId *optional*	string
clientSecret *optional*	string
created *optional*	string(date-time)
deactivated *optional*	boolean
display *optional*	string
emailFormat *optional*	string
id *optional*	integer(int64)
shortCode *optional*	string
uid *optional*	string
updated *optional*	string(date-time)
valid *optional*	boolean

7.63. CellCarrierResponse

Name	Schema
display *optional*	string
id *optional*	integer(int64)
uid *optional*	string

7.64. ChartData

Name	Schema
aggregatedItems *optional*	< object > array
alternateHGridAlpha *optional*	string
alternateHGridColor *optional*	string
anchorAlpha *optional*	string
caption *optional*	string
decimalPrecision *optional*	string
divLineAlpha *optional*	string
divLineColor *optional*	string
errorCode *optional*	string
formatNumberScale *optional*	string
getxAxisName *optional*	string
getyAxisMinValue *optional*	string
getyAxisName *optional*	string
idAttribute *optional*	string
identifier *optional*	string
items *optional*	< object > array
label *optional*	string

Name	Schema
maxValue *optional*	integer(int64)
message *optional*	string
numberPrefix *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
rotateNames *optional*	string
serializeNulls *optional*	boolean
showAlternateHGridColor *optional*	string
showAnchor *optional*	string
showNames *optional*	string
showValues *optional*	string
startTimeLog *optional*	integer(int64)
totalCount *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.65. Chronology

Name	Schema
zone *optional*	DateTimeZone

7.66. CohortRegion

Name	Schema
active *optional*	boolean
created *optional*	string(date-time)
deactivated *optional*	boolean
dwellSecs *optional*	integer(int32)
id *optional*	integer(int64)
orderIndex *optional*	integer(int32)
parentAudience *optional*	Audience
region *optional*	Region
updated *optional*	string(date-time)
valid *optional*	boolean

7.67. CompletableActionResponse

Name	Schema
action *optional*	string
completableActionId *optional*	integer(int64)
completableId *optional*	integer(int64)
completableType *optional*	string
createdDate *optional*	integer(int64)

Name	Schema
metaData *optional*	string
notificationCount *optional*	integer(int32)
updatedDate *optional*	integer(int64)

7.68. ConfigRoutingRequest

Name	Schema
averageCityTimePerMile *optional*	integer(int32)
averageFreewayTimePerMile *optional*	integer(int32)
debugMode *optional*	boolean
freewayTravelDistance *optional*	number(float)
maxItineraryTime *optional*	integer(int32)
maxRuntime *optional*	integer(int64)
maxStopsPerVehicle *optional*	integer(int32)
numberOfMatches *optional*	integer(int32)
orderAlgorithm *optional*	enum (CLOSEST_FIRST, FURTHEST_FIRST, PICKUPS_FIRST)
sameStopBuffer *optional*	number(double)
scoreAlgorithm *optional*	enum (FEWEST_VEHICLES, SHORTEST_PASSENGER_TIME)
threadType *optional*	string
threaded *optional*	boolean

7.69. ConnectedFriend

Name	Schema
active *optional*	boolean
avatar *optional*	Asset
avatarURL *optional*	string
created *optional*	string(date-time)
deactivated *optional*	boolean
display *optional*	string
id *optional*	integer(int64)
invited *optional*	boolean
locationDescription *optional*	string
platform *optional*	string
platformId *optional*	string
updated *optional*	string(date-time)
valid *optional*	boolean

7.70. Connection

Name	Schema
aboutUs *optional*	string
account *optional*	Account

Name	Schema
active *optional*	boolean
blocked *optional*	boolean
connection *optional*	Account
connectionTags *optional*	string
connectionType *optional*	enum (PRIMARY, SECONDARY, DEPENDENT, ACQUAINTANCE)
connectionViewedDate *optional*	string(date-time)
contact *optional*	boolean
created *optional*	string(date-time)
deactivated *optional*	boolean
deleted *optional*	string(date-time)
display *optional*	string
follower *optional*	boolean
following *optional*	boolean
friend *optional*	boolean
friendRequestDate *optional*	string(date-time)
friendRequestPending *optional*	boolean
friendRequested *optional*	boolean
friendResponseDate *optional*	string(date-time)

Name	Schema
id *optional*	integer(int64)
ignoreFriendRequest *optional*	boolean
latitude *optional*	number(double)
legacyId *optional*	string
locationDescription *optional*	string
longitude *optional*	number(double)
objectType *optional*	string
pendingConnection *optional*	ConnectedFriend
pendingConnection_id *optional*	integer(int64)
searchTags *optional*	string
title *optional*	string
trusted *optional*	boolean
updated *optional*	string(date-time)
valid *optional*	boolean

7.71. ConnectionGroup

Name	Schema
active *optional*	boolean
application *optional*	Application

Name	Schema
connections *optional*	< Connection > array
created *optional*	string(date-time)
deactivated *optional*	boolean
description *optional*	string
id *optional*	integer(int64)
image *optional*	Asset
metaData *optional*	string
name *optional*	string
owner *optional*	Account
permissions *optional*	GroupPermissions
subGroups *optional*	< ConnectionGroup > array
updated *optional*	string(date-time)
valid *optional*	boolean

7.72. ConnectionGroupResponse

Name	Schema
active *optional*	boolean
connectionCount *optional*	integer(int32)
connectionGroupId *optional*	integer(int64)

Name	Schema
connections *optional*	< ConnectionResponse > array
description *optional*	string
errorCode *optional*	string
groupPermissions *optional*	GroupPermissions
groupType *optional*	enum (PRIVATE, CONTACT, FRIENDS, TRUSTED, BLOCKED, FOLLOWERS, FOLLOWING, PENDING, REQUEST)
message *optional*	string
metaData *optional*	string
name *optional*	string
ownerId *optional*	integer(int64)
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
subGroups *optional*	< ConnectionGroupResponse > array
thumbnailURL *optional*	string
valid *optional*	boolean
version *optional*	number(double)

7.73. ConnectionGroupShortResponse

Name	Schema
connectionGroupId *optional*	integer(int64)
errorCode *optional*	string
groupType *optional*	enum (PRIVATE, CONTACT, FRIENDS, TRUSTED, BLOCKED, FOLLOWERS, FOLLOWING, PENDING, REQUEST)
message *optional*	string
name *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
thumbnailURL *optional*	string
valid *optional*	boolean
version *optional*	number(double)

7.74. ConnectionInfoResponse

Name	Schema
errorCode *optional*	string
followerCount *optional*	integer(int32)

Name	Schema
followingCount *optional*	integer(int32)
friendCount *optional*	integer(int32)
groups *optional*	< ConnectionGroupResponse > array
message *optional*	string
privateGroupCount *optional*	integer(int32)
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.75. ConnectionListResponse

Name	Schema
connections *optional*	< ConnectionResponse > array
errorCode *optional*	string
followingCount *optional*	integer(int32)
friendCount *optional*	integer(int32)
hasMoreResults *optional*	boolean

Name	Schema
itemCount *optional*	integer(int32)
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.76. ConnectionResponse

Name	Schema
aboutUs *optional*	string
blocked *optional*	boolean
connectionAccountId *optional*	string
connectionAccountType *optional*	string
connectionId *optional*	integer(int64)
connectionPendingId *optional*	string
connectionViewedDate *optional*	integer(int64)
contact *optional*	boolean

Name	Schema
display *optional*	string
follower *optional*	boolean
following *optional*	boolean
friend *optional*	boolean
friendRequestPending *optional*	boolean
friendRequested *optional*	boolean
ignoreFriendRequest *optional*	boolean
locationDescription *optional*	string
locationDisplay *optional*	string
profileImage *optional*	string
tags *optional*	string
title *optional*	string
trusted *optional*	boolean

7.77. ConsumerInviteResponse

Name	Schema
album *optional*	AlbumResponse
application *optional*	ApplicationShortResponse
connection *optional*	ConnectionResponse

Name	Schema
errorCode *optional*	string
inviteText *optional*	string
inviter *optional*	AccountShortResponse
inviterId *optional*	integer(int64)
message *optional*	string
offer *optional*	OfferShortResponse
request *optional*	< NameStringValueResponse > array
retailerLocation *optional*	RetailerLocationResponse
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
token *optional*	string
valid *optional*	boolean
version *optional*	number(double)
welcomeText *optional*	string

7.78. Contact

Name	Schema
address *optional*	Address

Name	Schema
businessPhone *optional*	string
businessPhoneExt *optional*	string
cellCarrier *optional*	CellCarrier
cellPhone *optional*	string
code501c3 *optional*	string
display *optional*	string
emailAddress *optional*	string
faxNumber *optional*	string
firstName *optional*	string
homePhone *optional*	string
lastName *optional*	string
middleName *optional*	string
prefix *optional*	string
suffix *optional*	string
timeZone *optional*	string
title *optional*	string
utcOffset *optional*	string
web *optional*	string

7.79. ContactInfo

Name	Schema
address *optional*	Address
businessPhone *optional*	string
businessPhoneExt *optional*	string
cellCarrier *optional*	CellCarrier
cellPhone *optional*	string
code501c3 *optional*	string
emailAddress *optional*	string
faxNumber *optional*	string
homePhone *optional*	string
timeZone *optional*	string
utcOffset *optional*	string
web *optional*	string

7.80. ContactInfoResponse

Name	Schema
address *optional*	AddressResponse
businessPhone *optional*	string
businessPhoneExt *optional*	string

Name	Schema
cellCarrier *optional*	CellCarrierResponse
cellPhone *optional*	string
cellPhoneCarrier *optional*	string
code501c3 *optional*	string
emailAddress *optional*	string
faxNumber *optional*	string
homePhone *optional*	string
timeZone *optional*	string
utcOffset *optional*	string
web *optional*	string

7.81. ContactResponse

Name	Schema
contactInfo *optional*	ContactInfoResponse
firstName *optional*	string
lastName *optional*	string
middleName *optional*	string
prefix *optional*	string
suffix *optional*	string

Name	Schema
title *optional*	string

7.82. CoordsResponse

Name	Schema
city *optional*	string
country *optional*	string
countryCode *optional*	string
errorCode *optional*	string
latitude *optional*	number(double)
longitude *optional*	number(double)
message *optional*	string
postalCode *optional*	string
region *optional*	string
regionCode *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
valid *optional*	boolean

Name	Schema
version *optional*	number(double)

7.83. CountResponse

Name	Schema
count *optional*	integer(int64)
errorCode *optional*	string
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.84. CreativeResponse

Name	Schema
action *optional*	string
active *optional*	boolean
appVersion *optional*	string
content *optional*	JsonElement

Name	Schema
creativeId *optional*	integer(int64)
currentBid *optional*	BidResponse
description *optional*	string
image *optional*	AssetShortResponse
metaData *optional*	string
name *optional*	string
owner *optional*	AccountShortResponse
preview *optional*	boolean
suffix *optional*	string
type *optional*	string

7.85. CsvImportResponse

Name	Schema
batchId *optional*	integer(int64)
csvImportBatchStatus *optional*	enum (NEW, ERROR, COMPLETE, PARSE_ERROR, PROCESSING, DUPLICATE)
errorCode *optional*	string
errorMessage *optional*	string
message *optional*	string
objectType *optional*	enum (OFFERS, RETAILERS, RETAILERLOCATIONS, CATEGORIES, FILTERS)

Name	Schema
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.86. DateTimeField

Name	Schema
durationField *optional*	DurationField
leapDurationField *optional*	DurationField
lenient *optional*	boolean
maximumValue *optional*	integer(int32)
minimumValue *optional*	integer(int32)
name *optional*	string
rangeDurationField *optional*	DurationField
supported *optional*	boolean
type *optional*	DateTimeFieldType

7.87. DateTimeFieldType

Name	Schema
durationType *optional*	DurationFieldType
name *optional*	string
rangeDurationType *optional*	DurationFieldType

7.88. DateTimeRange

Name	Schema
end *optional*	string(date-time)
interval *optional*	Interval
start *optional*	string(date-time)

7.89. DateTimeZone

Name	Schema
fixed *optional*	boolean
id *optional*	string

7.90. Device

Name	Schema
deviceInterference *optional*	number(double)
devicePower *optional*	number(double)
deviceType *optional*	string

Name	Schema
health *optional*	enum (RUNNING, WARNING, DOWN)
idleTimeoutInSecond *optional*	integer(int32)
lastDown *optional*	string(date-time)
lastHeartbeatSent *optional*	string(date-time)
lastNotificationSent *optional*	string(date-time)
lastUp *optional*	string(date-time)
rebootTimeHour *optional*	integer(int32)
rebootTimeMinute *optional*	integer(int32)
serialNumber *optional*	string
udid *optional*	string

7.91. DeviceResponse

Name	Schema
deviceInterference *optional*	number(double)
devicePower *optional*	number(double)
deviceType *optional*	string
health *optional*	enum (RUNNING, WARNING, DOWN)
idleTimeoutInSecond *optional*	integer(int32)
lastDown *optional*	integer(int64)

Name	Schema
lastHeartbeatSent *optional*	integer(int64)
lastNotificationSent *optional*	integer(int64)
lastUp *optional*	integer(int64)
rebootTimeHour *optional*	integer(int32)
rebootTimeMinute *optional*	integer(int32)
serialNumber *optional*	string
udid *optional*	string

7.92. Dimensions

Name	Schema
depth *optional*	number(double)
height *optional*	number(double)
volume *optional*	number(double)
width *optional*	number(double)

7.93. Direction

Name	Schema
active *optional*	boolean
created *optional*	string(date-time)
deactivated *optional*	boolean

Name	Schema
distance *optional*	string
duration *optional*	string
endLatitude *optional*	number(double)
endLongitude *optional*	number(double)
highlight *optional*	boolean
id *optional*	integer(int64)
instruction *optional*	string
polyline *optional*	string
progressStatus *optional*	enum (PENDING, COMPLETE, INVALID)
startLatitude *optional*	number(double)
startLongitude *optional*	number(double)
updated *optional*	string(date-time)
valid *optional*	boolean

7.94. DirectionResponse

Name	Schema
action *optional*	string
bearing *optional*	number(double)
bearingDifference *optional*	number(double)

Name	Schema
distance *optional*	number(double)
distanceString *optional*	string
distanceUnits *optional*	string
endLatitude *optional*	number(double)
endLongitude *optional*	number(double)
heading *optional*	string
instruction *optional*	string
progressStatus *optional*	string
startLatitude *optional*	number(double)
startLongitude *optional*	number(double)

7.95. DisbursementResponse

Name	Schema
amount *optional*	number
captureDate *optional*	integer(int64)
comment *optional*	string
created *optional*	integer(int64)
disbursementId *optional*	integer(int64)
errorCode *optional*	string

Name	Schema
errorMessage *optional*	string
estimatedDate *optional*	integer(int64)
externalId *optional*	string
introspectionParams *optional*	string
lastTransaction *optional*	PaymentTransactionResponse
provider *optional*	enum (AUTHORIZE_NET, AMAZON_FPS, BILL_COM, CLEAR, CLEAR_MERCHANT)
receiverBillable *optional*	BillableEntityShortResponse
scheduledDate *optional*	integer(int64)
senderBillable *optional*	BillableEntityShortResponse
settledDate *optional*	integer(int64)
status *optional*	enum (NEW, APPROVED, VALIDATING, ERROR, AUTHORIZED, CAPTURED, SETTLED)
title *optional*	string
updated *optional*	integer(int64)

7.96. DriverResponse

Name	Schema
availability *optional*	< AvailabilityResponse > array
depot *optional*	LocationResponse
id *optional*	string

Name	Schema
name *optional*	string

7.97. DurationField

Name	Schema
name *optional*	string
precise *optional*	boolean
supported *optional*	boolean
type *optional*	DurationFieldType
unitMillis *optional*	integer(int64)

7.98. DurationFieldType

Name	Schema
name *optional*	string

7.99. EmployeeResponse

Name	Schema
accountId *optional*	integer(int64)
accountType *optional*	string
active *optional*	boolean
appInfo *optional*	AppInfoResponse
assignedDeviceId *optional*	string

Name	Schema
businessName *optional*	string
contact *optional*	ContactResponse
contactEmail *optional*	string
currentRegion *optional*	RegionResponse
display *optional*	string
employer *optional*	AccountMiniResponse
gender *optional*	enum (MALE, FEMALE, ANY)
hasDelegateAccess *optional*	boolean
latitude *optional*	number(double)
locationCount *optional*	integer(int32)
locationDisplay *optional*	string
locationLastUpdated *optional*	integer(int64)
locations *optional*	< RetailerLocationShortResponse > array
longitude *optional*	number(double)
manager *optional*	AccountMiniResponse
profileImage *optional*	string
rssiLastReceived *optional*	integer(int64)
username *optional*	string

7.100. Event

Name	Schema
listing *optional*	Listing
sponsered *optional*	boolean

7.101. EventAttendanceResponse

Name	Schema
affiliatedCategories *optional*	< CategoryResponse > array
attendee *optional*	AccountShortResponse
event *optional*	OfferShortResponse
friend *optional*	boolean

7.102. EventResponse

Name	Schema
listing *optional*	ListingResponse

7.103. Filter

Name	Schema
active *optional*	boolean
application *optional*	Application
children *optional*	< Filter > array
created *optional*	string(date-time)

Name	Schema
deactivated *optional*	boolean
description *optional*	string
display *optional*	string
externalId *optional*	string
externalType *optional*	string
id *optional*	integer(int64)
metaData *optional*	string
name *optional*	string
parent *optional*	Filter
tree *optional*	string
updated *optional*	string(date-time)
valid *optional*	boolean

7.104. FilterResponse

Name	Schema
active *optional*	boolean
description *optional*	string
display *optional*	string
externalId *optional*	string

Name	Schema
externalType *optional*	string
filterId *optional*	integer(int64)
metaData *optional*	string
name *optional*	string
parentId *optional*	integer(int64)
subFilterCount *optional*	integer(int32)

7.105. FilterTreeResponse

Name	Schema
active *optional*	boolean
children *optional*	< FilterTreeResponse > array
description *optional*	string
display *optional*	string
externalId *optional*	string
externalType *optional*	string
filterId *optional*	integer(int64)
metaData *optional*	string
name *optional*	string
parentId *optional*	integer(int64)

Name	Schema
subFilterCount *optional*	integer(int32)

7.106. FingerprintBiometricImage

Name	Schema
data *optional*	string
imageFormat *optional*	string
position *optional*	enum (UNKNOWN, LEFT_INDEX, LEFT_MIDDLE, LEFT_RING, LEFT_LITTLE, LEFT_THUMB, RIGHT_INDEX, RIGHT_MIDDLE, RIGHT_RING, RIGHT_LITTLE, RIGHT_THUMB, LEFT, RIGHT)

7.107. Flag

Name	Schema
active *optional*	boolean
app *optional*	string
created *optional*	string(date-time)
deactivated *optional*	boolean
flagDescription *optional*	string
flagableObjectId *optional*	integer(int64)
flagableObjectType *optional*	string
id *optional*	integer(int64)
owner *optional*	Account

Name	Schema
updated *optional*	string(date-time)
valid *optional*	boolean

7.108. FlagResponse

Name	Schema
createdDate *optional*	integer(int64)
flagDescription *optional*	string
flagId *optional*	integer(int64)
flagableId *optional*	integer(int64)
flagableType *optional*	string
updatedDate *optional*	integer(int64)

7.109. FlagThreshold

Name	Schema
active *optional*	boolean
application *optional*	Application
created *optional*	string(date-time)
deactivated *optional*	boolean
flagableObjectType *optional*	string
id *optional*	integer(int64)

Name	Schema
threshold *optional*	integer(int64)
updated *optional*	string(date-time)
valid *optional*	boolean

7.110. Game

Name	Schema
active *optional*	boolean
allGameLevels *optional*	< GameLevel > array
allocateTickets *optional*	boolean
application *optional*	Application
authorOverride *optional*	string
contentName *optional*	string
contentType *optional*	string
created *optional*	string(date-time)
deactivated *optional*	boolean
description *optional*	string
endDate *optional*	string(date-time)
icon *optional*	Asset
id *optional*	integer(int64)

Name	Schema
image *optional*	Asset
metaData *optional*	string
owner *optional*	Account
packs *optional*	< Pack > array
points *optional*	integer(int64)
sequenceType *optional*	enum (FIRST_AVAILABLE, ALL_AVAILABLE)
startDate *optional*	string(date-time)
ticketCount *optional*	integer(int64)
ticketType *optional*	string
title *optional*	string
updated *optional*	string(date-time)
valid *optional*	boolean

7.111. GameLevel

Name	Schema
active *optional*	boolean
allocateTickets *optional*	boolean
application *optional*	Application
applicationTitle *optional*	string

Name	Schema
approvalStatus *optional*	enum (PENDING, REJECTED, APPROVED, FEATURED)
assignMission *optional*	boolean
authorOverride *optional*	string
balance *optional*	number(double)
contentAsset *optional*	Asset
contentName *optional*	string
contentType *optional*	string
created *optional*	string(date-time)
customType *optional*	string
deactivated *optional*	boolean
defaultThreshold *optional*	integer(int64)
description *optional*	string
dislikeCount *optional*	integer(int64)
downloadCount *optional*	integer(int64)
endDate *optional*	string(date-time)
flagCount *optional*	integer(int64)

Name	Schema
flagEvent *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
flagThreshold *optional*	FlagThreshold

Name	Schema
flagThresholdEvent *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
flags *optional*	< Flag > array
gameDifficulty *optional*	enum (VERY_EASY, EASY, MEDIUM, HARD, VERY_HARD)
gameObjectCount *optional*	integer(int32)
icon *optional*	Asset
id *optional*	integer(int64)
image *optional*	Asset
levelNumber *optional*	integer(int32)
likableNotificationModels *optional*	< < string, object > map > array
likeCount *optional*	integer(int64)
likes *optional*	< YayOrNay > array
metaData *optional*	string
name *optional*	string
newOwnerId *optional*	integer(int64)
nextLevel *optional*	GameLevel
notableNotificationModels *optional*	< < string, object > map > array
noteCount *optional*	integer(int64)
notes *optional*	< Note > array
offer *optional*	Offer

Name	Schema
owner *optional*	Account
ownerUsername *optional*	string

Name	Schema
permissionRequestEvent *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
permissionSharedEvent *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
playCount *optional*	integer(int64)
points *optional*	integer(int64)
publicPermissions *optional*	Permissions
published *optional*	boolean
randomizeGameObjects *optional*	boolean
splashMessage *optional*	string
splashTitle *optional*	string
startDate *optional*	string(date-time)
ticketCount *optional*	integer(int64)
ticketType *optional*	string
tutorial *optional*	Tutorial
updated *optional*	string(date-time)
userPermissions *optional*	< UserPermissions > array
valid *optional*	boolean
version *optional*	AppVersion
visibility *optional*	enum (PUBLIC, PRIVATE, FRIENDS)
winnerMessage *optional*	string

7.112. GameLevelListResponse

Name	Schema
count *optional*	integer(int32)
countTotal *optional*	integer(int64)
hasMoreResults *optional*	boolean
items *optional*	< GameLevelResponse > array
limit *optional*	integer(int32)
start *optional*	integer(int32)

7.113. GameLevelResponse

Name	Schema
action *optional*	enum (DATA, SAVE, DELETE, GET)
active *optional*	boolean
allocateTickets *optional*	boolean
appKey *optional*	string
appName *optional*	string
appVersion *optional*	string
assignMission *optional*	boolean
authorOverride *optional*	string
comments *optional*	< NoteResponse > array

Name	Schema
commentsCount *optional*	integer(int64)
completed *optional*	boolean
description *optional*	string
difficulty *optional*	string
dislikesCount *optional*	integer(int64)
downloadCount *optional*	integer(int64)
endDate *optional*	integer(int64)
gameData *optional*	string
gameDataSuffix *optional*	string
gameLevelId *optional*	integer(int64)
gameObjects *optional*	GameObjectListResponse
gameType *optional*	string
hasFlagged *optional*	boolean
hasLiked *optional*	boolean
icon *optional*	AssetShortResponse
image *optional*	AssetShortResponse
levelNumber *optional*	integer(int32)
likesCount *optional*	integer(int64)

Name	Schema
locked *optional*	boolean
metaData *optional*	string
name *optional*	string
nextLevelId *optional*	integer(int64)
offer *optional*	OfferShortResponse
owner *optional*	AccountShortResponse
points *optional*	integer(int64)
randomizeGameObjects *optional*	boolean
results *optional*	< NameStringValueResponse > array
scores *optional*	ScoreListResponse
scoringType *optional*	enum (GAME_LEVEL, GAME_OBJECT)
splashMessage *optional*	string
splashTitle *optional*	string
startDate *optional*	integer(int64)
ticketCount *optional*	integer(int64)
ticketType *optional*	string
tickets *optional*	TicketListResponse
ticketsEarned *optional*	integer(int64)

Name	Schema
tutorial *optional*	TutorialResponse
tutorialAlignment *optional*	string
tutorialImage *optional*	AssetShortResponse
tutorialMessage *optional*	string
tutorialTitle *optional*	string
updatedDate *optional*	integer(int64)
userPermissionsList *optional*	UserPermissionsListResponse
winnerMessage *optional*	string

7.114. GameListResponse

Name	Schema
count *optional*	integer(int32)
countTotal *optional*	integer(int64)
hasMoreResults *optional*	boolean
items *optional*	< GameResponse > array
limit *optional*	integer(int32)
start *optional*	integer(int32)

7.115. GameObjectListResponse

Name	Schema
count *optional*	integer(int32)
countTotal *optional*	integer(int64)
hasMoreResults *optional*	boolean
items *optional*	< GameObjectResponse > array
limit *optional*	integer(int32)
start *optional*	integer(int32)

7.116. GameObjectResponse

Name	Schema
action *optional*	enum (DATA, SAVE, DELETE, GET)
album *optional*	AlbumResponse
allocateTickets *optional*	boolean
gameObjectId *optional*	integer(int64)
gameType *optional*	string
image *optional*	AssetShortResponse
items *optional*	< NameStringValueResponse > array
owner *optional*	AccountShortResponse
playerScoreCount *optional*	integer(int64)
scores *optional*	ScoreListResponse

Name	Schema
tickets *optional*	TicketListResponse
ticketsEarned *optional*	integer(int64)

7.117. GameResponse

Name	Schema
action *optional*	enum (DATA, SAVE, DELETE, GET)
appKey *optional*	string
authorOverride *optional*	string
description *optional*	string
endDate *optional*	integer(int64)
gameId *optional*	integer(int64)
gameType *optional*	string
highestAppVersion *optional*	string
icon *optional*	AssetShortResponse
image *optional*	AssetShortResponse
metaData *optional*	string
owner *optional*	AccountShortResponse
packs *optional*	PackListResponse
results *optional*	< NameStringValueResponse > array

Name	Schema
rules *optional*	RuleListResponse
scores *optional*	ScoreListResponse
sequenceType *optional*	string
startDate *optional*	integer(int64)
ticketsEarned *optional*	integer(int64)
title *optional*	string
updatedDate *optional*	integer(int64)
userPermissionsList *optional*	UserPermissionsListResponse

7.118. GeoBox

Name	Schema
east *optional*	number(double)
north *optional*	number(double)
south *optional*	number(double)
west *optional*	number(double)

7.119. GeoPointResponse

Name	Schema
altitude *optional*	number(double)
asset *optional*	AssetShortResponse

Name	Schema
distance *optional*	number(double)
latitude *optional*	number(double)
locatableDisplay *optional*	string
locatableId *optional*	integer(int64)
locatableObjectId *optional*	string
locatableObjectName *optional*	string
locationDescription *optional*	string
longitude *optional*	number(double)
objectId *optional*	string

7.120. GeoResponse

Name	Schema
geoId *optional*	string
latitude *optional*	number(double)
longitude *optional*	number(double)

7.121. GeocodeLatLngResponse

Name	Schema
lat *optional*	number(double)
lng *optional*	number(double)

7.122. GroupPermissions

Name	Schema
canViewFriendInfo *optional*	boolean
canViewGameInfo *optional*	boolean
canViewProfileInfo *optional*	boolean

7.123. ImportStatuses

Name	Schema
statuses *optional*	< ShipmentImportStatus > array

7.124. Interval

Name	Schema
afterNow *optional*	boolean
beforeNow *optional*	boolean
chronology *optional*	Chronology
end *optional*	string(date-time)
endMillis *optional*	integer(int64)
start *optional*	string(date-time)
startMillis *optional*	integer(int64)

7.125. InviteResponse

Name	Schema
albumContestId *optional*	integer(int64)
albumId *optional*	integer(int64)
errorCode *optional*	string
inviteText *optional*	string
message *optional*	string
missionId *optional*	integer(int64)
offerId *optional*	integer(int64)
offerLocationId *optional*	integer(int64)
request *optional*	< NameStringValueResponse > array
retailerLocationId *optional*	integer(int64)
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
token *optional*	string
url *optional*	string
valid *optional*	boolean
version *optional*	number(double)
welcomeText *optional*	string

7.126. Invoice

Name	Schema
lineItems *optional*	< LineItem > array
location *optional*	string
tax *optional*	integer(int32)
total *optional*	integer(int32)

7.127. ItineraryResponse

Name	Schema
error *optional*	string
errorCode *optional*	string
estimatedDistance *optional*	number(double)
estimatedDuration *optional*	integer(int64)
estimatedEndTime *optional*	integer(int64)
estimatedStartTime *optional*	integer(int64)
generated *optional*	boolean
itemCount *optional*	integer(int32)
itineraryId *optional*	string
score *optional*	integer(int64)
stops *optional*	< StopResponse > array

Name	Schema
valid *optional*	boolean
vehicle *optional*	VehicleResponse
waitTime *optional*	integer(int64)
waypointCount *optional*	integer(int64)

7.128. JsonArray

Name	Schema
asBigDecimal *optional*	number
asBigInteger *optional*	integer
asBoolean *optional*	boolean
asByte *optional*	string(byte)
asCharacter *optional*	string
asDouble *optional*	number(double)
asFloat *optional*	number(float)
asInt *optional*	integer(int32)
asJsonArray *optional*	JsonArray
asJsonNull *optional*	JsonNull
asJsonObject *optional*	JsonObject
asJsonPrimitive *optional*	JsonPrimitive

Name	Schema
asLong *optional*	integer(int64)
asNumber *optional*	Number
asShort *optional*	integer(int32)
asString *optional*	string
jsonArray *optional*	boolean
jsonNull *optional*	boolean
jsonObject *optional*	boolean
jsonPrimitive *optional*	boolean

7.129. JsonElement

Name	Schema
asBigDecimal *optional*	number
asBigInteger *optional*	integer
asBoolean *optional*	boolean
asByte *optional*	string(byte)
asCharacter *optional*	string
asDouble *optional*	number(double)
asFloat *optional*	number(float)
asInt *optional*	integer(int32)

Name	Schema
asJsonArray *optional*	JsonArray
asJsonNull *optional*	JsonNull
asJsonObject *optional*	JsonObject
asJsonPrimitive *optional*	JsonPrimitive
asLong *optional*	integer(int64)
asNumber *optional*	Number
asShort *optional*	integer(int32)
asString *optional*	string
jsonArray *optional*	boolean
jsonNull *optional*	boolean
jsonObject *optional*	boolean
jsonPrimitive *optional*	boolean

7.130. JsonNull

Name	Schema
asBigDecimal *optional*	number
asBigInteger *optional*	integer
asBoolean *optional*	boolean
asByte *optional*	string(byte)

Name	Schema
asCharacter *optional*	string
asDouble *optional*	number(double)
asFloat *optional*	number(float)
asInt *optional*	integer(int32)
asJsonArray *optional*	JsonArray
asJsonNull *optional*	JsonNull
asJsonObject *optional*	JsonObject
asJsonPrimitive *optional*	JsonPrimitive
asLong *optional*	integer(int64)
asNumber *optional*	Number
asShort *optional*	integer(int32)
asString *optional*	string
jsonArray *optional*	boolean
jsonNull *optional*	boolean
jsonObject *optional*	boolean
jsonPrimitive *optional*	boolean

7.131. JsonObject

Name	Schema
asBigDecimal *optional*	number
asBigInteger *optional*	integer
asBoolean *optional*	boolean
asByte *optional*	string(byte)
asCharacter *optional*	string
asDouble *optional*	number(double)
asFloat *optional*	number(float)
asInt *optional*	integer(int32)
asJsonArray *optional*	JsonArray
asJsonNull *optional*	JsonNull
asJsonObject *optional*	JsonObject
asJsonPrimitive *optional*	JsonPrimitive
asLong *optional*	integer(int64)
asNumber *optional*	Number
asShort *optional*	integer(int32)
asString *optional*	string
jsonArray *optional*	boolean
jsonNull *optional*	boolean

Name	Schema
jsonObject *optional*	boolean
jsonPrimitive *optional*	boolean

7.132. JsonPrimitive

Name	Schema
asBigDecimal *optional*	number
asBigInteger *optional*	integer
asBoolean *optional*	boolean
asByte *optional*	string(byte)
asCharacter *optional*	string
asDouble *optional*	number(double)
asFloat *optional*	number(float)
asInt *optional*	integer(int32)
asJsonArray *optional*	JsonArray
asJsonNull *optional*	JsonNull
asJsonObject *optional*	JsonObject
asJsonPrimitive *optional*	JsonPrimitive
asLong *optional*	integer(int64)
asNumber *optional*	Number

Name	Schema
asShort *optional*	integer(int32)
asString *optional*	string
boolean *optional*	boolean
jsonArray *optional*	boolean
jsonNull *optional*	boolean
jsonObject *optional*	boolean
jsonPrimitive *optional*	boolean
number *optional*	boolean
string *optional*	boolean

7.133. LeaderboardResponse

Name	Schema
active *optional*	boolean
application *optional*	ApplicationMiniResponse
bannerAsset *optional*	AssetResponse
description *optional*	string
errorCode *optional*	string
iconAsset *optional*	AssetResponse
leaderboardId *optional*	integer(int64)

Name	Schema
limitation *optional*	integer(int32)
message *optional*	string
metaData *optional*	string
rankMode *optional*	string
rankType *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
sortType *optional*	string
startTimeLog *optional*	integer(int64)
title *optional*	string
valid *optional*	boolean
version *optional*	number(double)

7.134. Leg

Name	Schema
active *optional*	boolean
application *optional*	Application
created *optional*	string(date-time)

Name	Schema
deactivated *optional*	boolean
deviceId *optional*	string
distance *optional*	number(double)
duration *optional*	integer(int64)
endAltitude *optional*	number(double)
endDate *optional*	string(date-time)
endLatitude *optional*	number(double)
endLongitude *optional*	number(double)
floorId *optional*	string
id *optional*	integer(int64)
owner *optional*	Account
parentOwner *optional*	Account
retailerLocationId *optional*	integer(int64)
startAltitude *optional*	number(double)
startDate *optional*	string(date-time)
startLatitude *optional*	number(double)
startLongitude *optional*	number(double)
steps *optional*	< Step > array

Name	Schema
type *optional*	string
updated *optional*	string(date-time)
valid *optional*	boolean

7.135. LegResponse

Name	Schema
accountId *optional*	integer(int64)
appKey *optional*	string
created *optional*	integer(int64)
deviceId *optional*	string
deviceSignature *optional*	string
distance *optional*	number(double)
duration *optional*	integer(int64)
endAltitude *optional*	number(double)
endDate *optional*	integer(int64)
endLatitude *optional*	number(double)
endLongitude *optional*	number(double)
errorCode *optional*	string
message *optional*	string

Name	Schema
missingEndLocation *optional*	boolean
missingStartLocation *optional*	boolean
parentAccountId *optional*	integer(int64)
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startAltitude *optional*	number(double)
startDate *optional*	integer(int64)
startLatitude *optional*	number(double)
startLongitude *optional*	number(double)
startTimeLog *optional*	integer(int64)
steps *optional*	< StepResponse > array
tags *optional*	string
type *optional*	string
valid *optional*	boolean
version *optional*	number(double)

7.136. LikableResponse

Name	Schema
dislikeCount *optional*	integer(int64)
errorCode *optional*	string
likeCount *optional*	integer(int64)
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.137. LikeResponse

Name	Schema
account *optional*	AccountShortResponse

7.138. LineItem

Name	Schema
description *optional*	string
item *optional*	string
pricePerItem *optional*	integer(int32)

Name	Schema
quantity *optional*	integer(int32)
tax *optional*	integer(int32)
total *optional*	integer(int32)

7.139. ListCountResponse

Name	Schema
allOffersCount *optional*	integer(int64)
errorCode *optional*	string
instantOffersCount *optional*	integer(int64)
locationOffersCount *optional*	integer(int64)
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.140. ListResponse

Name	Schema
count *optional*	integer(int32)
errorCode *optional*	string
items *optional*	< object > array
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.141. Listing

Name	Schema
active *optional*	boolean
created *optional*	string(date-time)
deactivated *optional*	boolean
description *optional*	string
end *optional*	string(date-time)
externalGroupId *optional*	string

Name	Schema
externalId *optional*	string
filters *optional*	< Filter > array
id *optional*	integer(int64)
locationDescription *optional*	string
locationName *optional*	string
metaData *optional*	string
name *optional*	string
owner *optional*	Account
participants *optional*	< Participant > array
private *optional*	boolean
start *optional*	string(date-time)
updated *optional*	string(date-time)
valid *optional*	boolean

7.142. ListingFullResponse

Name	Schema
active *optional*	boolean
description *optional*	string
end *optional*	integer(int64)

Name	Schema
filters *optional*	< FilterResponse > array
id *optional*	integer(int64)
locationDescription *optional*	string
locationName *optional*	string
metaData *optional*	string
name *optional*	string
owner *optional*	AccountShortResponse
participants *optional*	< ParticipantResponse > array
privateListing *optional*	boolean
start *optional*	integer(int64)

7.143. ListingGroupResponse

Name	Schema
category *optional*	CategoryResponse
listings *optional*	< ListingResponse > array

7.144. ListingResponse

Name	Schema
active *optional*	boolean
description *optional*	string

Name	Schema
end *optional*	integer(int64)
id *optional*	integer(int64)
locationDescription *optional*	string
locationName *optional*	string
metaData *optional*	string
name *optional*	string
owner *optional*	AccountShortResponse
participants *optional*	< ParticipantResponse > array
privateListing *optional*	boolean
start *optional*	integer(int64)

7.145. LoadResponse

Name	Schema
dropoff *optional*	StopResponse
id *optional*	string
loadingTime *optional*	integer(int64)
name *optional*	string
pickup *optional*	StopResponse
separatePayloads *optional*	boolean

Name	Schema
size *optional*	integer(int64)

7.146. LoadShortResponse

Name	Schema
dropoff *optional*	StopResponse
id *optional*	string
loadingTime *optional*	integer(int64)
name *optional*	string
pickup *optional*	StopResponse
separatePayloads *optional*	boolean
size *optional*	integer(int64)

7.147. LocalTime

Name	Schema
chronology *optional*	Chronology
fieldTypes *optional*	< DateTimeFieldType > array
fields *optional*	< DateTimeField > array
hourOfDay *optional*	integer(int32)
millisOfDay *optional*	integer(int32)
millisOfSecond *optional*	integer(int32)

Name	Schema
minuteOfHour *optional*	integer(int32)
secondOfMinute *optional*	integer(int32)
values *optional*	< integer(int32) > array

7.148. Location

Name	Schema
active *optional*	boolean
address *optional*	Address
businessPhone *optional*	string
businessPhoneExt *optional*	string
cellCarrier *optional*	CellCarrier
cellPhone *optional*	string
code501c3 *optional*	string
contentName *optional*	string
created *optional*	string(date-time)
customType *optional*	string
deactivated *optional*	boolean
destination *optional*	boolean
dislikeCount *optional*	integer(int64)

Name	Schema
dockingTime *optional*	integer(int64)
emailAddress *optional*	string
faxNumber *optional*	string
groupingId *optional*	string
homePhone *optional*	string
hub *optional*	ServiceHub
id *optional*	integer(int64)
latitude *optional*	number(double)
likeCount *optional*	integer(int64)
likes *optional*	< YayOrNay > array
locationDescription *optional*	string
longitude *optional*	number(double)
name *optional*	string
noteCount *optional*	integer(int64)
notes *optional*	< Note > array
ownerDisplay *optional*	string
picture *optional*	Asset
program *optional*	Program

Name	Schema
territory *optional*	Territory
timeFramesAllowed *optional*	string
timeZone *optional*	string
updated *optional*	string(date-time)
usageCount *optional*	integer(int64)
utcOffset *optional*	string
valid *optional*	boolean
visibility *optional*	enum (PUBLIC, PRIVATE, FRIENDS)
web *optional*	string

7.149. LocationResponse

Name	Schema
errorCode *optional*	string
get_distance_miles *optional*	string
get_source *optional*	string
id *optional*	integer(int64)
latitude *optional*	string
longitude *optional*	string
message *optional*	string

Name	Schema
name *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
type *optional*	string
valid *optional*	boolean
version *optional*	number(double)
zip *optional*	string

7.150. LocationSearchResponse

Name	Schema
count *optional*	integer(int64)
errorCode *optional*	string
locations *optional*	< LocationResponse > array
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean

Name	Schema
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.151. Media

Name	Schema
asin *optional*	string
author *optional*	string
availabilityDate *optional*	integer(int64)
catalogNumbers *optional*	< string > array
collectionCount *optional*	integer(int32)
collections *optional*	< Album > array
condition *optional*	enum (NEW, USED, REFURBISHED, OTHER)
duration *optional*	integer(int32)
isbn *optional*	string
mediaType *optional*	enum (APK, APPLICATION, IMAGE, AUDIO, VIDEO, MULTIPART, BAR_CODE, TEXT, OTHER)
parentalRating *optional*	string
releaseDate *optional*	integer(int64)

7.152. MediaOfferResponse

Name	Schema
activated *optional*	integer(int64)
active *optional*	boolean
addedCount *optional*	integer(int32)
addedLimit *optional*	integer(int32)
availabilitySummary *optional*	string
barcode *optional*	string
billableEntityId *optional*	integer(int64)
categories *optional*	< CategoryResponse > array
clickedCount *optional*	integer(int32)
deleted *optional*	integer(int64)
details *optional*	string
discountPrice *optional*	number(double)
dislikeCount *optional*	integer(int64)
expiration *optional*	integer(int64)
externalId *optional*	string
externalRedeemOptions *optional*	string
externalUrl *optional*	string

Name	Schema
favorite *optional*	boolean
favoriteCount *optional*	integer(int64)
filters *optional*	< FilterResponse > array
flagCount *optional*	integer(int64)
flagThreshold *optional*	integer(int64)
fullPrice *optional*	number(double)
image *optional*	string
image1 *optional*	string
image2 *optional*	string
image3 *optional*	string
image4 *optional*	string
image5 *optional*	string
images *optional*	< AssetResponse > array
likeCount *optional*	integer(int64)
metaData *optional*	string
noteCount *optional*	integer(int64)
offerType *optional*	string
offerVisibility *optional*	string

Name	Schema
parentOffer *optional*	BaseOfferResponse
publisher *optional*	string
redeemableEnd *optional*	integer(int64)
redeemableStart *optional*	integer(int64)
removedCount *optional*	integer(int32)
responsibleAccountId *optional*	integer(int64)
specialOfferType *optional*	enum (ALL, RESERVABLE, REGULAR_OFFER, ACT_NOW, GET_THERE_NOW, SQOOT, TICKETS, YIPIT)
subDetails *optional*	string
subTitle *optional*	string
ticketsReward *optional*	integer(int64)
ticketsRewardType *optional*	string
title *optional*	string
usedCount *optional*	integer(int32)
viewedCount *optional*	integer(int32)

7.153. MediaResponse

Name	Schema
author *optional*	string
duration *optional*	integer(int32)

Name	Schema
mediaType *optional*	string
releaseDate *optional*	integer(int64)

7.154. MessageListResponse

Name	Schema
count *optional*	integer(int32)
countTotal *optional*	integer(int64)
hasMoreResults *optional*	boolean
items *optional*	< MessageResponse > array
limit *optional*	integer(int32)
start *optional*	integer(int32)

7.155. MessageResponse

Name	Schema
code *optional*	integer(int32)
ids *optional*	string
message *optional*	string
updateValue *optional*	string

7.156. Mission

Name	Schema
acceptedCount *optional*	integer(int32)
active *optional*	boolean
activePack *optional*	Pack
advancedReporting *optional*	boolean
allGameLevels *optional*	< GameLevel > array
allPacks *optional*	< Pack > array
allocateTickets *optional*	boolean
audiences *optional*	< Audience > array
authorOverride *optional*	string
availableLimit *optional*	integer(int32)
balance *optional*	number(double)
billableEntity *optional*	BillableEntity
childCount *optional*	integer(int64)
costToPlay *optional*	integer(int32)
costToPlayType *optional*	string
created *optional*	string(date-time)
deactivated *optional*	boolean
description *optional*	string

Name	Schema
enableBuyBack *optional*	boolean
enableMultipleEntries *optional*	boolean
enableMultipleVotes *optional*	boolean
endDate *optional*	string(date-time)
externalId *optional*	integer(int64)
featured *optional*	boolean
firstPack *optional*	Pack
fixedReward *optional*	boolean
games *optional*	< Game > array
icon *optional*	Asset
id *optional*	integer(int64)
image *optional*	Asset
inviteCount *optional*	integer(int32)
inviteStatus *optional*	enum (AVAILABLE, PENDING, UPDATED, ACCEPTED, ACTIVE, BUY_BACK, FAILED, QUIT, COMPLETED, PENDING_REVIEW, REJECTED)
joinCode *optional*	string
lastRound *optional*	integer(int32)
maximumLevelLength *optional*	integer(int32)

Name	Schema
metaData *optional*	string
minimumToPlay *optional*	integer(int32)
missionStatus *optional*	enum (NOT_STARTED, STARTED, TIE_BREAKER, PROCESSING, ENDED)
missionType *optional*	enum (PUBLIC, SHARED, TOURNAMENT, POOLPLAY, MULTISTAGE)
notificationsCreated *optional*	boolean
offers *optional*	< Offer > array
overallRankType *optional*	string
owner *optional*	Account
points *optional*	integer(int64)
preliminaryGroupAdvancementList *optional*	< integer(int32) > array
preliminaryGroupAdvancements *optional*	string
preliminaryGroups *optional*	integer(int32)
refunded *optional*	boolean
rewarded *optional*	boolean
roundCount *optional*	integer(int32)
roundType *optional*	enum (SUBMISSION, ELIMINATION, SEMIFINAL, FINAL, END)
secondsBetweenLevels *optional*	integer(int32)
secondsBetweenPacks *optional*	integer(int32)

Name	Schema
secondsForTieBreaker *optional*	integer(int32)
seedAlbum *optional*	Album
sequenceType *optional*	enum (FIRST_AVAILABLE, ALL_AVAILABLE)
splitReward *optional*	enum (EVEN, ALL, FIRST, RANDOM)
startDate *optional*	string(date-time)
startingLimit *optional*	integer(int32)
subType *optional*	string
task *optional*	MissionTask
ticketCount *optional*	integer(int64)
ticketType *optional*	string
tieTag *optional*	string
title *optional*	string
tournamentAlbumType *optional*	string
updated *optional*	string(date-time)
valid *optional*	boolean
visibility *optional*	enum (PUBLIC, LISTABLE, REWARDABLE, TRIGGERABLE, PRIVATE)
winnerTag *optional*	string

7.157. MissionFormatResponse

Name	Schema
active *optional*	boolean
description *optional*	string
format *optional*	string
formatType *optional*	string
missionFormatId *optional*	integer(int64)
name *optional*	string
suffix *optional*	string

7.158. MissionInviteResponse

Name	Schema
buyBackAvailable *optional*	boolean
description *optional*	string
endDate *optional*	integer(int64)
id *optional*	integer(int64)
inviteStatus *optional*	enum (AVAILABLE, PENDING, UPDATED, ACCEPTED, ACTIVE, BUY_BACK, FAILED, QUIT, COMPLETED, PENDING_REVIEW, REJECTED)
owner *optional*	AccountShortResponse
permissionableId *optional*	integer(int64)

Name	Schema
permissionableType *optional*	string
ranked *optional*	boolean
rewarded *optional*	boolean
startDate *optional*	integer(int64)
status *optional*	enum (AVAILABLE, PENDING, UPDATED, ACCEPTED, ACTIVE, BUY_BACK, FAILED, QUIT, COMPLETED, PENDING_REVIEW, REJECTED)
title *optional*	string
updated *optional*	integer(int64)

7.159. MissionListResponse

Name	Schema
count *optional*	integer(int32)
countTotal *optional*	integer(int64)
hasMoreResults *optional*	boolean
items *optional*	< MissionResponse > array
limit *optional*	integer(int32)
start *optional*	integer(int32)

7.160. MissionResponse

Name	Schema
acceptedCount *optional*	integer(int32)
action *optional*	enum (DATA, SAVE, DELETE, GET)
active *optional*	boolean
advancedReporting *optional*	boolean
allocateTickets *optional*	boolean
audiences *optional*	< AudienceResponse > array
authorOverride *optional*	string
availableLimit *optional*	integer(int32)
balance *optional*	number(double)
childCount *optional*	integer(int64)
costToPlay *optional*	integer(int32)
costToPlayType *optional*	string
creatives *optional*	< CreativeResponse > array
description *optional*	string
enableBuyBack *optional*	boolean
endDate *optional*	integer(int64)
fixedReward *optional*	boolean
formatType *optional*	string

Name	Schema
games *optional*	GameListResponse
highestAppVersion *optional*	string
icon *optional*	AssetShortResponse
image *optional*	AssetShortResponse
inviteCount *optional*	integer(int32)
inviteStatus *optional*	enum (AVAILABLE, PENDING, UPDATED, ACCEPTED, ACTIVE, BUY_BACK, FAILED, QUIT, COMPLETED, PENDING_REVIEW, REJECTED)
invitee *optional*	AccountShortResponse
joinCode *optional*	string
metaData *optional*	string
minimumToPlay *optional*	integer(int32)
missionId *optional*	integer(int64)
missionInvite *optional*	MissionInviteResponse
missionInviteId *optional*	integer(int64)
missionType *optional*	string
owner *optional*	AccountShortResponse
points *optional*	integer(int64)
refunded *optional*	boolean

Name	Schema
results *optional*	< NameStringValueResponse > array
rewarded *optional*	boolean
rewards *optional*	< RewardResponse > array
scores *optional*	ScoreListResponse
secondsBetweenLevels *optional*	integer(int32)
secondsBetweenPacks *optional*	integer(int32)
sequenceType *optional*	string
splitReward *optional*	string
startDate *optional*	integer(int64)
subType *optional*	string
ticketCount *optional*	integer(int64)
ticketType *optional*	string
ticketsEarned *optional*	integer(int64)
title *optional*	string
updatedDate *optional*	integer(int64)
userPermissionsList *optional*	UserPermissionsListResponse

7.161. MissionShortResponse

Name	Schema
acceptedCount *optional*	integer(int32)
action *optional*	enum (DATA, SAVE, DELETE, GET)
active *optional*	boolean
authorOverride *optional*	string
availableLimit *optional*	integer(int32)
balance *optional*	number(double)
childCount *optional*	integer(int64)
costToPlay *optional*	integer(int32)
costToPlayType *optional*	string
description *optional*	string
enableBuyBack *optional*	boolean
endDate *optional*	integer(int64)
highestAppVersion *optional*	string
icon *optional*	AssetShortResponse
image *optional*	AssetShortResponse
inviteCount *optional*	integer(int32)
joinCode *optional*	string
minimumToPlay *optional*	integer(int32)

Name	Schema
missionId *optional*	integer(int64)
missionType *optional*	string
owner *optional*	AccountShortResponse
refunded *optional*	boolean
rewarded *optional*	boolean
sequenceType *optional*	string
startDate *optional*	integer(int64)
ticketsEarned *optional*	integer(int64)
title *optional*	string
updatedDate *optional*	integer(int64)

7.162. MissionTask

Name	Schema
active *optional*	boolean
created *optional*	string(date-time)
deactivated *optional*	boolean
description *optional*	string
format *optional*	string
id *optional*	integer(int64)

Name	Schema
name *optional*	string
suffix *optional*	string
taskType *optional*	enum (PHOTO, VOTE, ASK, ANSWER, GROUP, INVITE, OFFER, AD, CUSTOM, GAME)
updated *optional*	string(date-time)
valid *optional*	boolean

7.163. NameStringValueResponse

Name	Schema
name *optional*	string
valueString *optional*	string

7.164. Node

Name	Schema
x *optional*	integer(int32)
y *optional*	integer(int32)
z *optional*	integer(int32)

7.165. NodeRequest

Name	Schema
accountId *optional*	integer(int64)
altitude *optional*	number(double)

Name	Schema
date *optional*	integer(int64)
isWaypoint *optional*	boolean
latitude *optional*	number(double)
longitude *optional*	number(double)
node *optional*	Node
x *optional*	integer(int32)
y *optional*	integer(int32)
z *optional*	integer(int32)

7.166. Note

Name	Schema
active *optional*	boolean
application *optional*	Application
assets *optional*	< Asset > array
contentName *optional*	string
created *optional*	string(date-time)
customType *optional*	string
deactivated *optional*	boolean
defaultThreshold *optional*	integer(int64)

Name	Schema
dislikeCount *optional*	integer(int64)
flagCount *optional*	integer(int64)

Name	Schema
flagEvent *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
flagThreshold *optional*	FlagThreshold

Name	Schema
flagThresholdEvent *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
flags *optional*	< Flag > array
id *optional*	integer(int64)
latitude *optional*	number(double)
likeCount *optional*	integer(int64)
likes *optional*	< YayOrNay > array
locationDescription *optional*	string
longitude *optional*	number(double)
metaData *optional*	string
notableId *optional*	integer(int64)
notableType *optional*	string
noteCount *optional*	integer(int64)
noteDescription *optional*	string
noteTag *optional*	string
notes *optional*	< Note > array
owner *optional*	Account
updated *optional*	string(date-time)
valid *optional*	boolean

7.167. NoteFullResponse

Name	Schema
active *optional*	boolean
application *optional*	ApplicationMiniResponse
attachedAssets *optional*	< AssetShortResponse > array
comment *optional*	string
createdDate *optional*	integer(int64)
dateCreated *optional*	integer(int64)
dislikesCount *optional*	integer(int64)
errorCode *optional*	string
flagCount *optional*	integer(int64)
flagsExceedThreshold *optional*	boolean
hasDisliked *optional*	boolean
hasLiked *optional*	boolean
latitude *optional*	number(double)
likes *optional*	< LikeResponse > array
likesCount *optional*	integer(int64)
locationDescription *optional*	string
longitude *optional*	number(double)

Name	Schema
message *optional*	string
metaData *optional*	string
noteId *optional*	integer(int64)
owner *optional*	AccountShortResponse
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
tags *optional*	string
updatedDate *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.168. NoteResponse

Name	Schema
active *optional*	boolean
application *optional*	ApplicationMiniResponse
comment *optional*	string
createdDate *optional*	integer(int64)

Name	Schema
dateCreated *optional*	integer(int64)
dislikesCount *optional*	integer(int64)
errorCode *optional*	string
flagCount *optional*	integer(int64)
flagsExceedThreshold *optional*	boolean
latitude *optional*	number(double)
likesCount *optional*	integer(int64)
locationDescription *optional*	string
longitude *optional*	number(double)
message *optional*	string
metaData *optional*	string
noteId *optional*	integer(int64)
owner *optional*	AccountShortResponse
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
tags *optional*	string

Name	Schema
updatedDate *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.169. NotificationMessageListResponse

Name	Schema
count *optional*	integer(int32)
errorCode *optional*	string
hasMoreResults *optional*	boolean
items *optional*	< NotificationMessageResponse > array
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.170. NotificationMessageResponse

Name	Schema
actionCategory *optional*	string
conduit *optional*	string
contentId *optional*	integer(int64)
contentName *optional*	string
contentType *optional*	string
coverURL *optional*	string
created *optional*	integer(int64)
deepLinkURI *optional*	string
event *optional*	string
hasRead *optional*	boolean
notificationMessage *optional*	string
notificationMessageId *optional*	integer(int64)
parentId *optional*	integer(int64)
parentType *optional*	string
sender *optional*	AccountShortResponse
thumbnailURL *optional*	string

7.171. NotificationRecipientResponse

Name	Schema
accountId *optional*	integer(int64)
accountType *optional*	string
appName *optional*	string
applicationId *optional*	integer(int64)
businessName *optional*	string
contactEmail *optional*	string
display *optional*	string
gender *optional*	enum (MALE, FEMALE, ANY)
hasApns *optional*	boolean
hasEmail *optional*	boolean
hasGcm *optional*	boolean
hasSms *optional*	boolean
locationDisplay *optional*	string
profileImage *optional*	string
username *optional*	string

7.172. NotificationSettingsResponse

Name	Schema
blockedNotifications *optional*	< enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

7.173. NotificationTemplateResponse

Name	Schema
application *optional*	ApplicationMiniResponse
body *optional*	string
conduit *optional*	enum (HTTP, SAVE, EMAIL, SMS, APNS, GCM, PUSH, MOBILE_NOTIFICATION)
created *optional*	integer(int64)
customType *optional*	string
disabled *optional*	boolean
event *optional*	string
notificationTemplateId *optional*	integer(int64)
publicBody *optional*	string
sendToAllParticipants *optional*	boolean
tags *optional*	string
title *optional*	string
updateNotificationCount *optional*	boolean
updated *optional*	integer(int64)

7.174. Number

Type : object

7.175. ObjectStoreResponse

Name	Schema
jsonObject *optional*	JsonObject
request *optional*	< NameStringValueResponse > array

7.176. Offer

Name	Schema
activated *optional*	string(date-time)
active *optional*	boolean
addedCount *optional*	integer(int32)
addedLimit *optional*	integer(int32)
allocateTickets *optional*	boolean
appKey *optional*	string
application *optional*	Application
artwork *optional*	Asset
availabilitySummary *optional*	string
availableLimit *optional*	integer(int32)
availableLimitPerUser *optional*	integer(int32)
barCode *optional*	Asset
barCodeEntry *optional*	string

Name	Schema
barCodeType *optional*	enum (NONE, UPC, CODE_128, QR, CUSTOM_MEDIA)
billableEntity *optional*	BillableEntity
billableId *optional*	integer(int64)
categories *optional*	< Category > array
categoryTree *optional*	string
clickStream *optional*	number(double)
clickedCount *optional*	integer(int32)
contentAsset *optional*	Asset
contentName *optional*	string
created *optional*	string(date-time)
customType *optional*	string
deactivated *optional*	boolean
defaultThreshold *optional*	integer(int64)
details *optional*	string
device *optional*	Device
discountPercentage *optional*	number(double)
discountPrice *optional*	number(double)
dislikeCount *optional*	integer(int64)

Name	Schema
display *optional*	string
endDate *optional*	string(date-time)
event *optional*	Event
expires *optional*	string(date-time)
externalId *optional*	string
externalUntrackedUrl *optional*	string
externalUrl *optional*	string
favoriteCount *optional*	integer(int64)
featured *optional*	boolean
filterTree *optional*	string
filters *optional*	< Filter > array
finePrint *optional*	string
flagCount *optional*	integer(int64)

Name	Schema
flagEvent *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
flagThreshold *optional*	FlagThreshold

Name	Schema
flagThresholdEvent *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
flags *optional*	< Flag > array
fullPrice *optional*	number(double)
getThereNowNotification *optional*	string
hasRatings *optional*	boolean
id *optional*	integer(int64)
indexable *optional*	boolean
likeCount *optional*	integer(int64)
likes *optional*	< YayOrNay > array
maxPrints *optional*	integer(int32)
media *optional*	Media
metaData *optional*	string
missions *optional*	< Mission > array
noteCount *optional*	integer(int64)
notes *optional*	< Note > array
offerAudiences *optional*	< Audience > array
offerImage1 *optional*	Asset
offerImage2 *optional*	Asset
offerImage3 *optional*	Asset

Name	Schema
offerImage4 *optional*	Asset
offerImage5 *optional*	Asset
offerLocations *optional*	< OfferLocation > array
offerProcessorId *optional*	enum (SQOOT, YIPIT, API, CSV)
offerVisibility *optional*	enum (PUBLIC, LISTABLE, REWARDABLE, TRIGGERABLE, PRIVATE)
owner *optional*	Account
parentOffer *optional*	Offer
product *optional*	Product
publisher *optional*	string
redeemAuthOptions *optional*	string
redeemableEnd *optional*	string(date-time)
redeemableStart *optional*	string(date-time)
removedCount *optional*	integer(int32)
replaced *optional*	boolean
savings *optional*	number(double)
secondaryType *optional*	string
showRedeemed *optional*	boolean
showRemaining *optional*	boolean

Name	Schema
specialOfferType *optional*	enum (ALL, RESERVABLE, REGULAR_OFFER, ACT_NOW, GET_THERE_NOW, SQOOT, TICKETS, YIPIT)
sqootCategoryName *optional*	string
sqootCategorySlug *optional*	string
sqootId *optional*	integer(int64)
sqootUrl *optional*	string
startDate *optional*	string(date-time)
subDetails *optional*	string
subTitle *optional*	string
subType *optional*	string
testField *optional*	string
ticketCount *optional*	integer(int64)
ticketPrice *optional*	integer(int64)
ticketPriceType *optional*	string
ticketType *optional*	string
title *optional*	string
transactionAudiences *optional*	< Audience > array
type *optional*	enum (VOUCHER, COUPON, PRODUCT, MEDIA, EVENT, DEVICE)
updated *optional*	string(date-time)

Name	Schema
usedCount *optional*	integer(int32)
valid *optional*	boolean
viewLimit *optional*	integer(int32)
viewedCount *optional*	integer(int32)
yipitId *optional*	integer(int64)

7.177. OfferListResponse

Name	Schema
count *optional*	integer(int32)
countTotal *optional*	integer(int64)
errorCode *optional*	string
hasMoreResults *optional*	boolean
items *optional*	< OfferResponse > array
limit *optional*	integer(int32)
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
start *optional*	integer(int32)

Name	Schema
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.178. OfferLocation

Name	Schema
active *optional*	boolean
addedCount *optional*	integer(int32)
altitude *optional*	number(double)
appKey *optional*	string
audiences *optional*	< Audience > array
billableId *optional*	integer(int64)
categoryIds *optional*	< integer(int64) > array
contentAsset *optional*	Asset
contentName *optional*	string
created *optional*	string(date-time)
customType *optional*	string
customValue *optional*	number(double)
deactivated *optional*	boolean

Name	Schema
devicePower *optional*	number(double)
display *optional*	string
distance *optional*	number(double)
favoriteCount *optional*	integer(int64)
filterIds *optional*	< integer(int64) > array
id *optional*	integer(int64)
inSearchIndex *optional*	boolean
latitude *optional*	number(double)
location *optional*	RetailerLocation
locationDescription *optional*	string
locationDetail *optional*	string
longitude *optional*	number(double)
noteCount *optional*	integer(int64)
notes *optional*	< Note > array
offer *optional*	Offer
owner *optional*	Account
powerLoss *optional*	number(double)
qrInviteCode *optional*	Asset

Name	Schema
searchIndexUpdated *optional*	string(date-time)
secondaryType *optional*	string
subType *optional*	string
udid *optional*	string
updated *optional*	string(date-time)
valid *optional*	boolean

7.179. OfferResponse

Name	Schema
activated *optional*	integer(int64)
active *optional*	boolean
addedCount *optional*	integer(int32)
addedLimit *optional*	integer(int32)
address *optional*	string
altitude *optional*	number(double)
audienceCount *optional*	integer(int32)
audiences *optional*	< AudienceResponse > array
availabilitySummary *optional*	string
barcode *optional*	string

Name	Schema
barcodeEntry *optional*	string
barcodeType *optional*	enum (NONE, UPC, CODE_128, QR, CUSTOM_MEDIA)
billableEntityId *optional*	integer(int64)
categories *optional*	< CategoryResponse > array
clickedCount *optional*	integer(int32)
created *optional*	integer(int64)
customValue *optional*	number(double)
deleted *optional*	integer(int64)
details *optional*	string
device *optional*	DeviceResponse
discountPercentage *optional*	number(double)
discountPrice *optional*	number(double)
dislikeCount *optional*	integer(int64)
distance *optional*	number(double)
endDate *optional*	integer(int64)
errorCode *optional*	string
estimatedValue *optional*	number(double)
event *optional*	EventResponse

Name	Schema
expiration *optional*	integer(int64)
expires *optional*	string(date-time)
externalId *optional*	string
externalRedeemAuth *optional*	integer(int32)
externalRedeemOptions *optional*	string
externalUrl *optional*	string
favorite *optional*	boolean
favoriteCount *optional*	integer(int64)
favoriteId *optional*	integer(int64)
filters *optional*	< FilterResponse > array
finePrint *optional*	string
fullPrice *optional*	number(double)
image *optional*	string
image1 *optional*	string
image2 *optional*	string
image3 *optional*	string
image4 *optional*	string
image5 *optional*	string

Name	Schema
images *optional*	< AssetResponse > array
latitude *optional*	number(double)
likeCount *optional*	integer(int64)
location *optional*	RetailerLocationResponse
locationAddedCount *optional*	integer(int32)
locationName *optional*	string
logId *optional*	integer(int64)
longitude *optional*	number(double)
media *optional*	MediaResponse
message *optional*	string
metaData *optional*	string
missionListResponse *optional*	MissionListResponse
noteCount *optional*	integer(int64)
offerId *optional*	integer(int64)
offerLocationId *optional*	integer(int64)
offerName *optional*	string
offerType *optional*	string
parentOffer *optional*	OfferShortResponse

Name	Schema
paymentFees *optional*	number(double)
product *optional*	ProductResponse
publisher *optional*	string
purchaseLimit *optional*	integer(int32)
purchaseLimitPerUser *optional*	integer(int32)
purchasesRemaining *optional*	integer(int32)
qrCodeUrl *optional*	string
redeemableEnd *optional*	integer(int64)
redeemableStart *optional*	integer(int64)
redemptionCode *optional*	string
redemptionLimit *optional*	integer(int32)
redemptionStatus *optional*	integer(int32)
redemptionsRemaining *optional*	integer(int32)
removedCount *optional*	integer(int32)
request *optional*	< NameStringValueResponse > array
responsibleAccountId *optional*	integer(int64)
retailerId *optional*	integer(int64)
retailerPhoneNumber *optional*	string

Name	Schema
returning *optional*	string
serializeNulls *optional*	boolean
showRedeemed *optional*	boolean
showRemaining *optional*	boolean
specialOfferType *optional*	enum (ALL, RESERVABLE, REGULAR_OFFER, ACT_NOW, GET_THERE_NOW, SQOOT, TICKETS, YIPIT)
sqootUrl *optional*	string
startDate *optional*	integer(int64)
startTimeLog *optional*	integer(int64)
starts *optional*	string(date-time)
subDetails *optional*	string
subTitle *optional*	string
tags *optional*	string
ticketPrice *optional*	integer(int64)
ticketPriceType *optional*	string
ticketsReward *optional*	integer(int64)
ticketsRewardType *optional*	string
transactionId *optional*	integer(int64)
updated *optional*	integer(int64)

Name	Schema
usedCount *optional*	integer(int32)
userRedemptionLimit *optional*	integer(int32)
valid *optional*	boolean
version *optional*	number(double)
viewedCount *optional*	integer(int32)
voucherDiscountPrice *optional*	number(double)
voucherPrice *optional*	number(double)
walletCount *optional*	integer(int32)

7.180. OfferShortResponse

Name	Schema
activated *optional*	integer(int64)
active *optional*	boolean
addedCount *optional*	integer(int32)
addedLimit *optional*	integer(int32)
altitude *optional*	number(double)
availabilitySummary *optional*	string
barcode *optional*	string
barcodeEntry *optional*	string

Name	Schema
barcodeType *optional*	enum (NONE, UPC, CODE_128, QR, CUSTOM_MEDIA)
billableEntityId *optional*	integer(int64)
clickedCount *optional*	integer(int32)
created *optional*	integer(int64)
customValue *optional*	number(double)
deleted *optional*	integer(int64)
device *optional*	DeviceResponse
discountPrice *optional*	number(double)
dislikeCount *optional*	integer(int64)
distance *optional*	number(double)
endDate *optional*	integer(int64)
errorCode *optional*	string
estimatedValue *optional*	number(double)
event *optional*	EventResponse
expiration *optional*	integer(int64)
expires *optional*	string(date-time)
externalId *optional*	string
externalRedeemAuth *optional*	integer(int32)

Name	Schema
externalRedeemOptions *optional*	string
externalUrl *optional*	string
favorite *optional*	boolean
favoriteCount *optional*	integer(int64)
favoriteId *optional*	integer(int64)
fullPrice *optional*	number(double)
image *optional*	string
image1 *optional*	string
image2 *optional*	string
image3 *optional*	string
image4 *optional*	string
image5 *optional*	string
images *optional*	< AssetResponse > array
latitude *optional*	number(double)
likeCount *optional*	integer(int64)
location *optional*	RetailerLocationResponse
locationAddedCount *optional*	integer(int32)
locationName *optional*	string

Name	Schema
logId *optional*	integer(int64)
longitude *optional*	number(double)
media *optional*	MediaResponse
message *optional*	string
metaData *optional*	string
missionListResponse *optional*	MissionListResponse
noteCount *optional*	integer(int64)
offerId *optional*	integer(int64)
offerLocationId *optional*	integer(int64)
offerName *optional*	string
offerType *optional*	string
parentOffer *optional*	OfferShortResponse
product *optional*	ProductResponse
publisher *optional*	string
redeemableEnd *optional*	integer(int64)
redeemableStart *optional*	integer(int64)
removedCount *optional*	integer(int32)
request *optional*	< NameStringValueResponse > array

Name	Schema
responsibleAccountId *optional*	integer(int64)
retailerId *optional*	integer(int64)
returning *optional*	string
serializeNulls *optional*	boolean
specialOfferType *optional*	enum (ALL, RESERVABLE, REGULAR_OFFER, ACT_NOW, GET_THERE_NOW, SQOOT, TICKETS, YIPIT)
sqootUrl *optional*	string
startDate *optional*	integer(int64)
startTimeLog *optional*	integer(int64)
starts *optional*	string(date-time)
subDetails *optional*	string
subTitle *optional*	string
tags *optional*	string
ticketsReward *optional*	integer(int64)
ticketsRewardType *optional*	string
transactionId *optional*	integer(int64)
updated *optional*	integer(int64)
usedCount *optional*	integer(int32)
valid *optional*	boolean

Name	Schema
version *optional*	number(double)
viewedCount *optional*	integer(int32)
voucherDiscountPrice *optional*	number(double)

7.181. OfferTransactionResponse

Name	Schema
activated *optional*	integer(int64)
active *optional*	boolean
addedCount *optional*	integer(int32)
addedLimit *optional*	integer(int32)
address *optional*	string
altitude *optional*	number(double)
audienceCount *optional*	integer(int32)
audiences *optional*	< AudienceResponse > array
availabilitySummary *optional*	string
barcode *optional*	string
barcodeEntry *optional*	string
barcodeType *optional*	enum (NONE, UPC, CODE_128, QR, CUSTOM_MEDIA)
billableEntityId *optional*	integer(int64)

Name	Schema
categories *optional*	< CategoryResponse > array
clickedCount *optional*	integer(int32)
created *optional*	integer(int64)
customValue *optional*	number(double)
customer *optional*	AccountShortResponse
deleted *optional*	integer(int64)
details *optional*	string
device *optional*	DeviceResponse
discountPercentage *optional*	number(double)
discountPrice *optional*	number(double)
dislikeCount *optional*	integer(int64)
distance *optional*	number(double)
endDate *optional*	integer(int64)
errorCode *optional*	string
estimatedValue *optional*	number(double)
event *optional*	EventResponse
expiration *optional*	integer(int64)
expires *optional*	string(date-time)

Name	Schema
externalId *optional*	string
externalRedeemAuth *optional*	integer(int32)
externalRedeemOptions *optional*	string
externalUrl *optional*	string
favorite *optional*	boolean
favoriteCount *optional*	integer(int64)
favoriteId *optional*	integer(int64)
filters *optional*	< FilterResponse > array
finePrint *optional*	string
fullPrice *optional*	number(double)
image *optional*	string
image1 *optional*	string
image2 *optional*	string
image3 *optional*	string
image4 *optional*	string
image5 *optional*	string
images *optional*	< AssetResponse > array
latitude *optional*	number(double)

Name	Schema
likeCount *optional*	integer(int64)
location *optional*	RetailerLocationResponse
locationAddedCount *optional*	integer(int32)
locationName *optional*	string
logId *optional*	integer(int64)
longitude *optional*	number(double)
media *optional*	MediaResponse
message *optional*	string
metaData *optional*	string
missionListResponse *optional*	MissionListResponse
noteCount *optional*	integer(int64)
offerId *optional*	integer(int64)
offerLocationId *optional*	integer(int64)
offerName *optional*	string
offerType *optional*	string
parentOffer *optional*	OfferShortResponse
paymentFees *optional*	number(double)
product *optional*	ProductResponse

Name	Schema
publisher *optional*	string
purchaseLimit *optional*	integer(int32)
purchaseLimitPerUser *optional*	integer(int32)
purchasedDate *optional*	integer(int64)
purchasesRemaining *optional*	integer(int32)
qrCodeUrl *optional*	string
redeemableEnd *optional*	integer(int64)
redeemableStart *optional*	integer(int64)
redemptionCode *optional*	string
redemptionLimit *optional*	integer(int32)
redemptionStatus *optional*	integer(int32)
redemptionsRemaining *optional*	integer(int32)
removedCount *optional*	integer(int32)
request *optional*	< NameStringValueResponse > array
responsibleAccountId *optional*	integer(int64)
retailerId *optional*	integer(int64)
retailerPhoneNumber *optional*	string
returning *optional*	string

Name	Schema
serializeNulls *optional*	boolean
showRedeemed *optional*	boolean
showRemaining *optional*	boolean
specialOfferType *optional*	enum (ALL, RESERVABLE, REGULAR_OFFER, ACT_NOW, GET_THERE_NOW, SQOOT, TICKETS, YIPIT)
sqootUrl *optional*	string
startDate *optional*	integer(int64)
startTimeLog *optional*	integer(int64)
starts *optional*	string(date-time)
subDetails *optional*	string
subTitle *optional*	string
tags *optional*	string
ticketPrice *optional*	integer(int64)
ticketPriceType *optional*	string
ticketsReward *optional*	integer(int64)
ticketsRewardType *optional*	string
transactionCreated *optional*	integer(int64)
transactionId *optional*	integer(int64)
transactionMetaData *optional*	string

Name	Schema
transactionUpdated *optional*	integer(int64)
updated *optional*	integer(int64)
usedCount *optional*	integer(int32)
usedDate *optional*	integer(int64)
userRedemptionLimit *optional*	integer(int32)
valid *optional*	boolean
version *optional*	number(double)
viewedCount *optional*	integer(int32)
voucherDiscountPrice *optional*	number(double)
voucherPrice *optional*	number(double)
walletCount *optional*	integer(int32)

7.182. OfferTransactionStatusResponse

Name	Schema
active *optional*	boolean
applications *optional*	< ApplicationShortResponse > array
code *optional*	integer(int32)
description *optional*	string
id *optional*	integer(int64)

Name	Schema
name *optional*	string
role *optional*	string

7.183. OrderItemRequest

Name	Schema
amount *optional*	number(double)
message *optional*	string
offerName *optional*	string
orderCustomId *optional*	string
orderCustomType *optional*	string
orderItemId *optional*	integer(int64)
orderItemType *optional*	enum (OFFER, CUSTOM)
quantity *optional*	integer(int32)
reserveEndDate *optional*	integer(int64)
reserveStartDate *optional*	integer(int64)
retailerLocationId *optional*	integer(int64)
tax *optional*	number(double)

7.184. OrderPackage

Name	Schema
CubicFeet *optional*	number(double)
PackageCount *optional*	integer(int32)
PackageTypeName *optional*	string
deactivated *optional*	boolean
valid *optional*	boolean

7.185. OrderResponse

Name	Schema
amount *optional*	number(double)
application *optional*	ApplicationMiniResponse
created *optional*	integer(int64)
credits *optional*	number(double)
customer *optional*	AccountShortResponse
description *optional*	string
errorCode *optional*	string
externalDate *optional*	integer(int64)
externalOrderId *optional*	string
message *optional*	string
orderId *optional*	integer(int64)

Name	Schema
payments *optional*	< PaymentTransactionResponse > array
purchases *optional*	number(double)
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
updated *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.186. Orders

Name	Schema
Orders *optional*	< ShipmentOrder > array

7.187. Pack

Name	Schema
active *optional*	boolean
allocateTickets *optional*	boolean
application *optional*	Application
applicationTitle *optional*	string

Name	Schema
authorOverride *optional*	string
background *optional*	Asset
created *optional*	string(date-time)
deactivated *optional*	boolean
description *optional*	string
downloaded *optional*	boolean
endDate *optional*	string(date-time)
gameLevels *optional*	< GameLevel > array
hasMoreItems *optional*	boolean
highest *optional*	boolean
id *optional*	integer(int64)
image *optional*	Asset
imageURL *optional*	string
inGame *optional*	boolean
levelNumberMap *optional*	< string, GameLevel > map
metaData *optional*	string
newOwnerId *optional*	integer(int64)
notificationCount *optional*	integer(int32)

Name	Schema
owner *optional*	Account
packOrder *optional*	integer(int64)
packType *optional*	enum (TUTORIAL, BUILTIN, DOWNLOAD, THRESHOLD, THEME, TOURNAMENT)
points *optional*	integer(int64)
price *optional*	integer(int32)
priceType *optional*	string
sequenceType *optional*	enum (FIRST_AVAILABLE, ALL_AVAILABLE)
startDate *optional*	string(date-time)
ticketCount *optional*	integer(int64)
ticketType *optional*	string
title *optional*	string
totalCount *optional*	integer(int64)
updated *optional*	string(date-time)
valid *optional*	boolean
winnersPerLevel *optional*	integer(int32)
winnersPerLevelWithDefault *optional*	integer(int32)

7.188. PackListResponse

Name	Schema
count *optional*	integer(int32)
countTotal *optional*	integer(int64)
hasMoreResults *optional*	boolean
items *optional*	< PackResponse > array
limit *optional*	integer(int32)
start *optional*	integer(int32)

7.189. PackResponse

Name	Schema
action *optional*	enum (DATA, SAVE, DELETE, GET)
active *optional*	boolean
allocateTickets *optional*	boolean
appKey *optional*	string
application *optional*	ApplicationShortResponse
authorOverride *optional*	string
background *optional*	AssetShortResponse
completed *optional*	boolean
costToRedeem *optional*	integer(int64)
description *optional*	string

Name	Schema
downloaded *optional*	boolean
endDate *optional*	integer(int64)
gameType *optional*	string
highest *optional*	boolean
highestAppVersion *optional*	string
icon *optional*	AssetShortResponse
inGame *optional*	boolean
levels *optional*	GameLevelListResponse
metaData *optional*	string
name *optional*	string
owner *optional*	AccountShortResponse
packId *optional*	integer(int64)
packOrder *optional*	integer(int64)
packType *optional*	enum (TUTORIAL, BUILTIN, DOWNLOAD, THRESHOLD, THEME, TOURNAMENT)
points *optional*	integer(int64)
price *optional*	integer(int64)
priceType *optional*	string
results *optional*	< NameStringValueResponse > array

Name	Schema
scores *optional*	ScoreListResponse
sequenceType *optional*	string
startDate *optional*	integer(int64)
ticketCount *optional*	integer(int64)
ticketType *optional*	string
tickets *optional*	TicketListResponse
ticketsEarned *optional*	integer(int64)
updatedDate *optional*	integer(int64)
userPermissionsList *optional*	UserPermissionsListResponse
winnersPerLevel *optional*	integer(int32)

7.190. Participant

Name	Schema
account *optional*	Account
active *optional*	boolean
availableEnd *optional*	string(date-time)
availableStart *optional*	string(date-time)
color1 *optional*	string
color2 *optional*	string

Name	Schema
created *optional*	string(date-time)
deactivated *optional*	boolean
id *optional*	integer(int64)
type *optional*	enum (TEAM, LEAGUE, SPORT)
updated *optional*	string(date-time)
valid *optional*	boolean

7.191. ParticipantResponse

Name	Schema
account *optional*	AccountMiniResponse
availableEnd *optional*	integer(int64)
availableStart *optional*	integer(int64)
color1 *optional*	string
color2 *optional*	string
id *optional*	integer(int64)
type *optional*	string

7.192. PathingResponse

Name	Schema
calibrationBearings *optional*	number(double)

Name	Schema
directionCount *optional*	integer(int32)
directions *optional*	< DirectionResponse > array
end *optional*	NodeRequest
exclusions *optional*	< NodeRequest > array
height *optional*	integer(int32)
metersPerX *optional*	number(double)
metersPerY *optional*	number(double)
necalibration *optional*	NodeRequest
northeast *optional*	NodeRequest
path *optional*	< NodeRequest > array
pathCount *optional*	integer(int32)
southwest *optional*	NodeRequest
start *optional*	NodeRequest
swcalibration *optional*	NodeRequest
waypoints *optional*	< NodeRequest > array
width *optional*	integer(int32)

7.193. PaymentRequest

Name	Schema
acceptedBilling *optional*	boolean
appKey *optional*	string
biometrics *optional*	BiometricRequest
clientPaymentId *optional*	string
invoice *optional*	Invoice
minAgeYears *optional*	integer(int32)
orderItems *optional*	< OrderItemRequest > array

7.194. PaymentTransactionResponse

Name	Schema
amount *optional*	number(double)
created *optional*	integer(int64)
currencyType *optional*	string
customer *optional*	AccountShortResponse
errorCode *optional*	string
externalPaymentId *optional*	string
message *optional*	string
metadata *optional*	string
offerTransactions *optional*	< OfferTransactionResponse > array

Name	Schema
orderItems *optional*	< PurchaseOrderItemResponse > array
paymentTransactionId *optional*	integer(int64)
promoCode *optional*	string
remoteRefId *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
tax *optional*	number(double)
updated *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.195. PaymentTypesResponse

Name	Schema
account *optional*	AccountShortResponse
accountName *optional*	string
accountNumber *optional*	string
address *optional*	string

Name	Schema
bankName *optional*	string
city *optional*	string
creditCardExpirationMonth *optional*	string
creditCardExpirationYear *optional*	string
creditCardNumber *optional*	string
creditCardOnFile *optional*	boolean
errorCode *optional*	string
firstName *optional*	string
isDefaultPaymentMethod *optional*	boolean
lastName *optional*	string
message *optional*	string
paymentMethodId *optional*	integer(int64)
paymentMethodNickname *optional*	string
phone *optional*	string
pointsAmount *optional*	number(double)
pointsOnFile *optional*	boolean
postalCode *optional*	string
provider *optional*	string

Name	Schema
request *optional*	< NameStringValueResponse > array
returning *optional*	string
routingNumber *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
state *optional*	string
taxId *optional*	string
valid *optional*	boolean
validated *optional*	boolean
version *optional*	number(double)

7.196. PermissionResponse

Name	Schema
add *optional*	boolean
delete *optional*	boolean
errorCode *optional*	string
message *optional*	string
read *optional*	boolean
request *optional*	< NameStringValueResponse > array

Name	Schema
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)
write *optional*	boolean

7.197. Permissions

Name	Schema
add *optional*	boolean
admin *optional*	boolean
delete *optional*	boolean
member *optional*	boolean
pending *optional*	boolean
read *optional*	boolean
write *optional*	boolean

7.198. PersonalProfileResponse

Name	Schema
birthday *optional*	integer(int64)

Name	Schema
bodyType *optional*	string
children *optional*	string
companionship *optional*	string
companionshipIndex *optional*	integer(int32)
drink *optional*	string
education *optional*	string
educationIndex *optional*	integer(int32)
errorCode *optional*	string
ethnicity *optional*	string
gender *optional*	enum (MALE, FEMALE, ANY)
height *optional*	string
heightIndex *optional*	integer(int32)
hometown *optional*	string
maritalStatus *optional*	string
message *optional*	string
preferredBodyType *optional*	string
preferredEducation *optional*	string
preferredEducationIndex *optional*	integer(int32)

Name	Schema
preferredEthnicity *optional*	string
preferredGender *optional*	enum (MALE, FEMALE, ANY)
preferredLocationRange *optional*	number(double)
preferredMaxAge *optional*	integer(int32)
preferredMaxHeight *optional*	integer(int32)
preferredMinAge *optional*	integer(int32)
preferredMinHeight *optional*	integer(int32)
religion *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
smoke *optional*	string
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.199. Placement

Name	Schema
active *optional*	boolean

Name	Schema
application *optional*	Application
clickType *optional*	enum (CONFIG, PURCHASE, BANNER, LEADERBOARD, SKYSCRAPER, VIDEO, ZIP, FULL, CUSTOM1, CUSTOM2, CUSTOM3, CUSTOM4, CUSTOM5, CUSTOM6, CUSTOM7, CUSTOM8, CUSTOM9, CUSTOM10)
created *optional*	string(date-time)
deactivated *optional*	boolean
defaultImage *optional*	Asset
description *optional*	string
height *optional*	integer(int32)
id *optional*	integer(int64)
metaData *optional*	string
name *optional*	string
refreshInterval *optional*	integer(int32)
size *optional*	enum (CONFIG, BANNER, LEADERBOARD, SKYSCRAPER, VIDEO, ZIP, INTERSTITIAL, CUSTOM1, CUSTOM2, CUSTOM3, CUSTOM4, CUSTOM5, CUSTOM6, CUSTOM7, CUSTOM8, CUSTOM9, CUSTOM10)
updated *optional*	string(date-time)
valid *optional*	boolean
width *optional*	integer(int32)

7.200. PlacementResponse

Name	Schema
active *optional*	boolean
appKey *optional*	string
appName *optional*	string
defaultImage *optional*	AssetShortResponse
description *optional*	string
height *optional*	integer(int32)
metaData *optional*	string
name *optional*	string
placementId *optional*	integer(int64)
refreshInterval *optional*	integer(int32)
size *optional*	enum (CONFIG, BANNER, LEADERBOARD, SKYSCRAPER, VIDEO, ZIP, INTERSTITIAL, CUSTOM1, CUSTOM2, CUSTOM3, CUSTOM4, CUSTOM5, CUSTOM6, CUSTOM7, CUSTOM8, CUSTOM9, CUSTOM10)
width *optional*	integer(int32)

7.201. Platform

Name	Schema
active *optional*	boolean
created *optional*	string(date-time)

Name	Schema
deactivated *optional*	boolean
description *optional*	string
device *optional*	AudienceDevice
downloadUrl *optional*	string
id *optional*	integer(int64)
maximum *optional*	integer(int64)
minimum *optional*	integer(int64)
updated *optional*	string(date-time)
valid *optional*	boolean

7.202. PlatformResponse

Name	Schema
active *optional*	boolean
description *optional*	string
deviceId *optional*	integer(int64)
display *optional*	string
downloadUrl *optional*	string
maximum *optional*	integer(int64)
minimum *optional*	integer(int64)

Name	Schema
name *optional*	string

7.203. PostalCode

Name	Schema
active *optional*	boolean
city *optional*	string
code *optional*	string
created *optional*	string(date-time)
deactivated *optional*	boolean
eliminated *optional*	boolean
id *optional*	integer(int64)
lastOfferImport *optional*	string(date-time)
latitude *optional*	number(double)
locationDescription *optional*	string
longitude *optional*	number(double)
plus4 *optional*	string
regions *optional*	< Region > array
searchPoint *optional*	boolean
stateCode *optional*	string

Name	Schema
timeZone *optional*	string
truncatedLatitude *optional*	number(double)
truncatedLongitude *optional*	number(double)
updated *optional*	string(date-time)
utcOffset *optional*	integer(int32)
valid *optional*	boolean

7.204. PostalCodeResponse

Name	Schema
city *optional*	string
code *optional*	string
latitude *optional*	number(double)
longitude *optional*	number(double)
stateCode *optional*	string
timezone *optional*	string
utcOffset *optional*	integer(int32)

7.205. PredictedLocationResponse

Name	Schema
averageDuration *optional*	number(double)

Name	Schema
distance *optional*	number(double)
hour *optional*	integer(int32)
latitude *optional*	number(double)
longitude *optional*	number(double)
matches *optional*	integer(int64)
onDate *optional*	string
onDateMilli *optional*	integer(int64)
ownerId *optional*	integer(int64)
weightedScore *optional*	number(double)

7.206. PreferredLocationResponse

Name	Schema
distance *optional*	number(double)
duration *optional*	integer(int64)
endStepId *optional*	integer(int64)
hour *optional*	integer(int32)
latitude *optional*	number(double)
longitude *optional*	number(double)
onDate *optional*	string

Name	Schema
onDateMilli *optional*	integer(int64)
ownerId *optional*	integer(int64)
startStepId *optional*	integer(int64)

7.207. PreviewPersonaResponse

Name	Schema
active *optional*	boolean
age *optional*	integer(int32)
date *optional*	integer(int64)
gameExperienceLevel *optional*	string
gender *optional*	string
latitude *optional*	number(double)
longitude *optional*	number(double)
metaData *optional*	string
personaId *optional*	integer(int64)
previewAccounts *optional*	< AccountMiniResponse > array
title *optional*	string

7.208. Product

Name	Schema
asin *optional*	string
availabilityDate *optional*	integer(int64)
brand *optional*	string
catalogNumbers *optional*	< string > array
condition *optional*	enum (NEW, USED, REFURBISHED, OTHER)
department *optional*	string
depth *optional*	number(double)
features *optional*	< string > array
height *optional*	number(double)
isbn *optional*	string
minimumPrice *optional*	number(double)
parentalRating *optional*	string
productType *optional*	enum (APPAREL, APPLIANCES, APPS, ARTS_CRAFTS, AUTOMOTIVE, BABY, BEAUTY, BOOKS, CLASSICAL, COLLECTIBLES, DVD, ELECTRONICS, GIFT_CARDS, GOURMET_FOOD, GROCERY, HEALTH_PERSONAL_CARE, HOME_GARDEN, INDUSTRIAL, JEWELRY, KITCHEN, LAWN_GARDEN, MAGAZINES, MISCELLANEOUS, MUSICAL_INSTRUMENTS, OFFICE_PRODUCTS, OTHER, OUTDOOR_LIVING, PC_HARDWARE, PET_SUPPLIES, SHOES, SOFTWARE, SPORTING_GOODS, TOOLS, TOYS, VIDEO_GAMES, WATCHES, WIRELESS, WIRELESS_ACCESSORIES)
publishDate *optional*	integer(int64)

Name	Schema
size *optional*	SizeGroup
studio *optional*	string
unit *optional*	enum (MILIMETER, CENTIMETER, METER, INCH, FOOT, YARD)
weight *optional*	number(double)
width *optional*	number(double)

7.209. ProductResponse

Name	Schema
asin *optional*	string
availabilityDate *optional*	integer(int64)
brand *optional*	string
catalogNumbers *optional*	< string > array
condition *optional*	string
department *optional*	string
depth *optional*	number(double)
features *optional*	< string > array
height *optional*	number(double)
isbn *optional*	string
minimumPrice *optional*	number(double)

Name	Schema
parentalRating *optional*	string
productType *optional*	string
publishDate *optional*	integer(int64)
size *optional*	SizeGroupResponse
studio *optional*	string
unit *optional*	string
weight *optional*	number(double)
width *optional*	number(double)

7.210. ProfileInfoResponse

Name	Schema
aboutUs *optional*	string
accountId *optional*	integer(int64)
accountType *optional*	string
age *optional*	string
categories *optional*	< CategoryResponse > array
contact *optional*	ContactResponse
currentPostalCode *optional*	PostalCodeResponse
display *optional*	string

Name	Schema
errorCode *optional*	string
flagCount *optional*	integer(int64)
gameExperience *optional*	string
gender *optional*	enum (MALE, FEMALE, ANY)
latitude *optional*	number(double)
locale *optional*	string
locationDisplay *optional*	string
longitude *optional*	number(double)
matchIndex *optional*	integer(int32)
message *optional*	string
personalProfile *optional*	PersonalProfileResponse
premiumExpiration *optional*	integer(int64)
profileAvatar *optional*	AssetShortResponse
profileImage *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)

Name	Schema
summaryTitle *optional*	string
tags *optional*	string
userSettings *optional*	UserSettingsResponse
username *optional*	string
valid *optional*	boolean
version *optional*	number(double)

7.211. ProfileResponse

Name	Schema
active *optional*	boolean
appInfo *optional*	AppInfoResponse
blocked *optional*	boolean
canViewAppInfo *optional*	boolean
canViewFriendInfo *optional*	boolean
canViewProfileInfo *optional*	boolean
contact *optional*	boolean
errorCode *optional*	string
flagCount *optional*	integer(int64)
flagged *optional*	boolean

Name	Schema
follower *optional*	boolean
following *optional*	boolean
friend *optional*	boolean
friendInfo *optional*	ConnectionInfoResponse
friendRequestPending *optional*	boolean
friendRequested *optional*	boolean
isBlocked *optional*	boolean
isProfileOwner *optional*	boolean
loginAccountId *optional*	integer(int64)
message *optional*	string
metFlagThreshold *optional*	boolean
profileInfo *optional*	ProfileInfoResponse
profileOwner *optional*	boolean
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
trusted *optional*	boolean

Name	Schema
valid *optional*	boolean
validated *optional*	boolean
version *optional*	number(double)

7.212. ProfileShortResponse

Name	Schema
aboutUs *optional*	string
accountId *optional*	integer(int64)
display *optional*	string
errorCode *optional*	string
flagCount *optional*	integer(int64)
locationDisplay *optional*	string
message *optional*	string
profileAvatar *optional*	AssetShortResponse
profileImage *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)

Name	Schema
username *optional*	string
valid *optional*	boolean
version *optional*	number(double)

7.213. Program

Name	Schema
active *optional*	boolean
costPerRide *optional*	integer(int32)
created *optional*	string(date-time)
deactivated *optional*	boolean
description *optional*	string
id *optional*	integer(int64)
name *optional*	string
updated *optional*	string(date-time)
valid *optional*	boolean

7.214. PromoCode

Name	Schema
active *optional*	boolean
code *optional*	string

Name	Schema
created *optional*	string(date-time)
days *optional*	integer(int32)
deactivated *optional*	boolean
id *optional*	integer(int64)
name *optional*	string
updated *optional*	string(date-time)
valid *optional*	boolean

7.215. PurchaseItemFullResponse

Name	Schema
allocateTickets *optional*	boolean
appKey *optional*	string
appName *optional*	string
assetable *optional*	boolean
coverAsset *optional*	AssetShortResponse
description *optional*	string
gameType *optional*	string
giftable *optional*	boolean
id *optional*	integer(int64)

Name	Schema
name *optional*	string
points *optional*	integer(int64)
price *optional*	number(float)
promoAsset *optional*	AssetShortResponse
purchaseCode *optional*	string
purchaseLimit *optional*	integer(int32)
purchaseType *optional*	string
quantity *optional*	integer(int32)
secretKey *optional*	string
serviceAction *optional*	enum (DAY_PREMIUM, WEEK_PREMIUM, MONTH_PREMIUM, YEAR_PREMIUM, LIFETIME_PREMIUM, ADD_TICKET, ADD_GIFT, ADD_POINTS)
ticketCount *optional*	integer(int64)
ticketType *optional*	string
tickets *optional*	integer(int32)

7.216. PurchaseItemListResponse

Name	Schema
count *optional*	integer(int32)
countTotal *optional*	integer(int64)

Name	Schema
errorCode *optional*	string
hasMoreResults *optional*	boolean
items *optional*	< PurchaseItemResponse > array
limit *optional*	integer(int32)
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
start *optional*	integer(int32)
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.217. PurchaseItemResponse

Name	Schema
appName *optional*	string
assetable *optional*	boolean
coverAsset *optional*	AssetShortResponse
description *optional*	string

Name	Schema
gameType *optional*	string
giftable *optional*	boolean
id *optional*	integer(int64)
name *optional*	string
price *optional*	number(float)
promoAsset *optional*	AssetShortResponse
purchaseCode *optional*	string
purchaseLimit *optional*	integer(int32)
purchaseType *optional*	string
quantity *optional*	integer(int32)
serviceAction *optional*	enum (DAY_PREMIUM, WEEK_PREMIUM, MONTH_PREMIUM, YEAR_PREMIUM, LIFETIME_PREMIUM, ADD_TICKET, ADD_GIFT, ADD_POINTS)
tickets *optional*	integer(int32)

7.218. PurchaseItemShortResponse

Name	Schema
assetable *optional*	boolean
coverAsset *optional*	AssetShortResponse
description *optional*	string

Name	Schema
giftable *optional*	boolean
id *optional*	integer(int64)
name *optional*	string
promoAsset *optional*	AssetShortResponse
tickets *optional*	integer(int32)

7.219. PurchaseOrderItemResponse

Name	Schema
amount *optional*	number(double)
comment *optional*	string
created *optional*	integer(int64)
customer *optional*	AccountShortResponse
errorCode *optional*	string
message *optional*	string
orderCustomId *optional*	string
orderCustomType *optional*	string
orderItemId *optional*	integer(int64)
orderItemType *optional*	string
purchaseOrderItemId *optional*	integer(int64)

Name	Schema
request *optional*	< NameStringValueResponse > array
retailerLocationId *optional*	integer(int64)
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
tax *optional*	number(double)
updated *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.220. QuestionResponse

Name	Schema
active *optional*	boolean
allocateTickets *optional*	boolean
answers *optional*	< AnswerResponse > array
authorOverride *optional*	string
createdDate *optional*	integer(int64)
endDate *optional*	integer(int64)
icon *optional*	AssetShortResponse

Name	Schema
id *optional*	integer(int64)
image *optional*	AssetShortResponse
metaData *optional*	string
owner *optional*	AccountShortResponse
points *optional*	integer(int64)
question *optional*	string
scores *optional*	ScoreListResponse
startDate *optional*	integer(int64)
tags *optional*	string
ticketCount *optional*	integer(int64)
ticketType *optional*	string
tickets *optional*	TicketListResponse
updatedDate *optional*	integer(int64)
videoURL *optional*	string

7.221. QueueResponse

Name	Schema
analyticTags *optional*	string
dataMapping *optional*	string

Name	Schema
exchanger *optional*	string
exchangerType *optional*	string
handleDelivery *optional*	boolean
host *optional*	string
name *optional*	string
password *optional*	string
port *optional*	integer(int32)
queueId *optional*	integer(int64)
username *optional*	string
virtualHost *optional*	string
workers *optional*	integer(int32)

7.222. RankFullResponse

Name	Schema
errorCode *optional*	string
message *optional*	string
rankings *optional*	< RankListResponse > array
request *optional*	< NameStringValueResponse > array
returning *optional*	string

Name	Schema
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.223. RankListResponse

Name	Schema
app *optional*	string
appKey *optional*	string
appName *optional*	string
applicationId *optional*	integer(int64)
count *optional*	integer(int32)
hasMoreResults *optional*	boolean
items *optional*	< RankResponse > array
leaderboardMode *optional*	enum (GLOBAL, FRIENDS, LOCAL, SEARCH, CUSTOM)
limit *optional*	integer(int64)
rankType *optional*	string
sortField *optional*	string
start *optional*	integer(int64)

Name	Schema
userRank *optional*	SummaryRankResponse

7.224. RankResponse

Name	Schema
appName *optional*	string
applicationId *optional*	integer(int64)
associatedLeaderboard *optional*	LeaderboardResponse
countValue *optional*	integer(int64)
endDate *optional*	integer(int64)
owner *optional*	AccountShortResponse
rank *optional*	string
rankType *optional*	string
scoreValue *optional*	integer(int64)
startDate *optional*	integer(int64)
streakBestCount *optional*	integer(int64)
streakCount *optional*	integer(int64)
timeValue *optional*	integer(int64)
updated *optional*	integer(int64)

7.225. RatingIndexResponse

Name	Schema
category *optional*	CategoryResponse
display *optional*	string
ratable *optional*	object
ratableId *optional*	integer(int64)
ratableType *optional*	enum (LOCATABLE, RESERVABLE, PERMISSIONABLE, NOTABLE, ASSETABLE, LIKABLE, FLAGABLE, FAVORITABLE, RATABLE, ALBUM, COLLECTION, APPLICATION, APPLICATION_SETTING, APPLICATION_CERT, APPLICATION_PLACEMENT, ACCOUNT, ACCOUNT_SETTING, GAME_LEVEL, PACK, MISSION, TOURNAMENT, ASSET, ALBUM_CONTEST, THEME_DESCRIPTOR, OFFER, OFFER_LOCATION, EVENT, RETAILER, RETAILER_LOCATION, NOTE, CREATIVE, FAVORITE, LIKE, RATING, ANALYTIC, THIRD_PARTY_CREDENTIAL, THIRD_PARTY_NETWORK, REGION)
ratingAverage *optional*	number(float)
ratingCount *optional*	integer(int64)
ratingIndexId *optional*	integer(int64)
secondaryType *optional*	string
value *optional*	number(float)

7.226. RatingResponse

Name	Schema
category *optional*	CategoryResponse

Name	Schema
description *optional*	string
display *optional*	string
latitude *optional*	number(double)
locationDescription *optional*	string
longitude *optional*	number(double)
owner *optional*	AccountShortResponse
ratingId *optional*	integer(int64)
ratingValue *optional*	integer(int32)

7.227. Recurrence

Name	Schema
count *optional*	integer(int32)
daysOfWeek *optional*	< integer(int32) > array
frequency *optional*	enum (DAILY, WEEKLY, MONTHLY, ANNUALLY, WEEK_DAY_ONLY)
start *optional*	string(date)
until *optional*	string(date)

7.228. Region

Name	Schema
active *optional*	boolean

Name	Schema
adminActive *optional*	boolean
bounds *optional*	GeoBox
categories *optional*	< Category > array
children *optional*	< Region > array
contentName *optional*	string
created *optional*	string(date-time)
deactivated *optional*	boolean
display *optional*	string
end *optional*	string(date-time)
filters *optional*	< Filter > array
fullName *optional*	string
id *optional*	integer(int64)
latitude *optional*	number(double)
locationDescription *optional*	string
longitude *optional*	number(double)
metaData *optional*	string
owner *optional*	Account
parents *optional*	< Region > array

Name	Schema
polygon *optional*	string
postalCodes *optional*	< PostalCode > array
regionClass *optional*	enum (NATIONAL, STATE, COUNTY, METRO_AREA, CITY, NEIGHBORHOOD, TERRITORY, CUSTOM, ZONE)
root *optional*	boolean
shortName *optional*	string
start *optional*	string(date-time)
truncatedLatitude *optional*	number(double)
truncatedLongitude *optional*	number(double)
updated *optional*	string(date-time)
valid *optional*	boolean
versionCode *optional*	integer(int32)
visibility *optional*	enum (PUBLIC, PRIVATE, FRIENDS)
webActive *optional*	boolean

7.229. RegionResponse

Name	Schema
active *optional*	boolean
categories *optional*	< CategoryResponse > array
children *optional*	< RegionResponse > array

Name	Schema
deleted *optional*	integer(int64)
distance *optional*	number(double)
end *optional*	integer(int64)
fullName *optional*	string
id *optional*	integer(int64)
latitude *optional*	number(double)
longitude *optional*	number(double)
metaData *optional*	string
parent *optional*	RegionResponse
polygon *optional*	string
postalCodes *optional*	< PostalCodeResponse > array
regionClass *optional*	string
root *optional*	boolean
shortName *optional*	string
start *optional*	integer(int64)
updated *optional*	integer(int64)
versionCode *optional*	integer(int32)

7.230. ReportBatchResponse

Name	Schema
batchId *optional*	integer(int64)
created *optional*	integer(int64)
description *optional*	string
endpoint *optional*	string
errorCode *optional*	string
errorMessage *optional*	string
message *optional*	string
name *optional*	string
pageUrl *optional*	string
reportBatchStatus *optional*	enum (NEW, ERROR, COMPLETE, PARSE_ERROR, PROCESSING, DUPLICATE, SAVEONLY)
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
updated *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.231. ReportDefinition

Name	Schema
active *optional*	boolean
columnNames *optional*	< string > array
columnNamesRaw *optional*	string
created *optional*	string(date-time)
dataSource *optional*	enum (WRITE, ANALYTIC, READ1)
deactivated *optional*	boolean
display *optional*	boolean
displayName *optional*	string
displayParams *optional*	string
id *optional*	integer(int64)
maxDateRange *optional*	integer(int32)
maxLimit *optional*	integer(int64)
metaData *optional*	string
nativeSql *optional*	boolean
query *optional*	string
queryName *optional*	string
updated *optional*	string(date-time)

Name	Schema
valid *optional*	boolean

7.232. ReportResponse

Name	Schema
columns *optional*	< ReportTypeResponse > array
count *optional*	integer(int64)
errorCode *optional*	string
hasMore *optional*	boolean
message *optional*	string
queryName *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
rows *optional*	< < string, object > map > array
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
summations *optional*	< string, number > map
valid *optional*	boolean
version *optional*	number(double)

7.233. ReportTypeResponse

Name	Schema
javaType *optional*	string
label *optional*	string
majorAxis *optional*	boolean
number *optional*	boolean
type *optional*	string

7.234. ReservationResponse

Name	Schema
account *optional*	AccountShortResponse
application *optional*	ApplicationResponse
createdDate *optional*	integer(int64)
endDate *optional*	integer(int64)
metaData *optional*	string
reservableId *optional*	integer(int64)
reservableType *optional*	string
reservationId *optional*	integer(int64)
startDate *optional*	integer(int64)
transaction *optional*	OfferShortResponse

Name	Schema
updatedDate *optional*	integer(int64)

7.235. Retailer

Name	Schema
active *optional*	boolean
address *optional*	Address
appKey *optional*	string
billableId *optional*	integer(int64)
businessPhone *optional*	string
businessPhoneExt *optional*	string
categories *optional*	< Category > array
categoryTree *optional*	string
cellCarrier *optional*	CellCarrier
cellPhone *optional*	string
code501c3 *optional*	string
contentName *optional*	string
customType *optional*	string
deactivated *optional*	boolean
detailsBody *optional*	string

Name	Schema
detailsHeader *optional*	string
detailsLineLeft1 *optional*	string
detailsLineLeft2 *optional*	string
detailsLineRight1 *optional*	string
directions *optional*	string
emailAddress *optional*	string
facebookUrl *optional*	string
faxNumber *optional*	string
featured *optional*	boolean
filterTree *optional*	string
homePhone *optional*	string
hours *optional*	string
id *optional*	integer(int64)
latitude *optional*	number(double)
listed *optional*	boolean
locationDescription *optional*	string
locations *optional*	< RetailerLocation > array
longitude *optional*	number(double)

Name	Schema
manager *optional*	Account
metaData *optional*	string
metaDescription *optional*	string
name *optional*	string
noteCount *optional*	integer(int64)
notes *optional*	< Note > array
owner *optional*	Account
responsible *optional*	Account
subType *optional*	string
territory *optional*	Territory
timeZone *optional*	string
twitterUrl *optional*	string
utcOffset *optional*	string
valid *optional*	boolean
visibility *optional*	enum (PUBLIC, PRIVATE, FRIENDS)
web *optional*	string
yipitId *optional*	integer(int64)

7.236. RetailerCountResponse

Name	Schema
locations *optional*	integer(int32)
offers *optional*	integer(int32)
transactions *optional*	integer(int32)

7.237. RetailerFullResponse

Name	Schema
active *optional*	boolean
billableEntity *optional*	BillableEntityResponse
contact *optional*	ContactInfoResponse
counts *optional*	RetailerCountResponse
facebookUrl *optional*	string
latitude *optional*	number(double)
locations *optional*	< RetailerLocationShortResponse > array
logo *optional*	AssetShortResponse
longitude *optional*	number(double)
manager *optional*	ProfileShortResponse
metaData *optional*	string
name *optional*	string

Name	Schema
picture1 *optional*	AssetShortResponse
picture2 *optional*	AssetShortResponse
responsible *optional*	ProfileShortResponse
retailerId *optional*	integer(int64)
twitterUrl *optional*	string
visibility *optional*	string

7.238. RetailerLocation

Name	Schema
active *optional*	boolean
addressDisplay *optional*	string
administrators *optional*	< Account > array
appKey *optional*	string
assignments *optional*	< Assignment > array
billableId *optional*	integer(int64)
building *optional*	Building
categories *optional*	< Category > array
categoryTree *optional*	string
code *optional*	string

Name	Schema
contentName *optional*	string
created *optional*	string(date-time)
currentAssignment *optional*	Assignment
customType *optional*	string
deactivated *optional*	boolean
display *optional*	string
favoriteCount *optional*	integer(int64)
filterTree *optional*	string
filters *optional*	< Filter > array
fullDisplay *optional*	string
googlePlaceId *optional*	string
hasOffers *optional*	boolean
hasRatings *optional*	boolean
id *optional*	integer(int64)
inSearchIndex *optional*	boolean
latitude *optional*	number(double)
locationAddress *optional*	string
locationDescription *optional*	string

Name	Schema
locationId *optional*	string
locationToken *optional*	string
logo *optional*	Asset
longitude *optional*	number(double)
mapQuery *optional*	string
metaData *optional*	string
name *optional*	string
noteCount *optional*	integer(int64)
notes *optional*	< Note > array
offerLocations *optional*	< OfferLocation > array
offerPrintKeyPrefix *optional*	string
owner *optional*	Account
paymentProvider *optional*	enum (AUTHORIZE_NET, AMAZON_FPS, BILL_COM, CLEAR, CLEAR_MERCHANT)
picture1 *optional*	Asset
picture2 *optional*	Asset
profile *optional*	RetailerProfile
publicLocation *optional*	boolean
qrCode *optional*	Asset

Name	Schema
regions *optional*	< Region > array
retailer *optional*	Retailer
searchIndexUpdated *optional*	string(date-time)
secondaryType *optional*	string
sortName *optional*	string
sqootId *optional*	integer(int64)
subType *optional*	string
updated *optional*	string(date-time)
valid *optional*	boolean
validOfferLocationIds *optional*	< integer(int64) > array
yelpId *optional*	string
yipitId *optional*	integer(int64)

7.239. RetailerLocationResponse

Name	Schema
active *optional*	boolean
billableEntity *optional*	BillableEntityResponse
building *optional*	string
categories *optional*	< CategoryResponse > array

Name	Schema
contact *optional*	ContactInfoResponse
counts *optional*	RetailerCountResponse
detailsBody *optional*	string
detailsHeader *optional*	string
distance *optional*	number(double)
favoritableId *optional*	integer(int64)
favoritableType *optional*	string
favorite *optional*	boolean
favoriteId *optional*	integer(int64)
filters *optional*	< FilterResponse > array
googlePlaceId *optional*	string
hours *optional*	string
internalId *optional*	string
latitude *optional*	number(double)
locationToken *optional*	string
logo *optional*	AssetShortResponse
longitude *optional*	number(double)
manager *optional*	ProfileShortResponse

Name	Schema
metaData *optional*	string
name *optional*	string
offers *optional*	ListResponse
paymentProvider *optional*	string
picture1 *optional*	AssetShortResponse
picture2 *optional*	AssetShortResponse
qrCodeUrl *optional*	string
responsible *optional*	ProfileShortResponse
retailer *optional*	RetailerShortResponse
retailerLocationId *optional*	integer(int64)
tags *optional*	string
yelpId *optional*	string

7.240. RetailerLocationShortResponse

Name	Schema
active *optional*	boolean
contact *optional*	ContactInfoResponse
favoritableId *optional*	integer(int64)
favoritableType *optional*	string

Name	Schema
favorite *optional*	boolean
favoriteId *optional*	integer(int64)
latitude *optional*	number(double)
longitude *optional*	number(double)
name *optional*	string
retailerLocationId *optional*	integer(int64)
tags *optional*	string

7.241. RetailerOfferResponse

Name	Schema
activated *optional*	integer(int64)
active *optional*	boolean
addedCount *optional*	integer(int32)
addedLimit *optional*	integer(int32)
address *optional*	string
altitude *optional*	number(double)
audienceCount *optional*	integer(int32)
audiences *optional*	< AudienceResponse > array
availabilitySummary *optional*	string

Name	Schema
barcode *optional*	string
barcodeEntry *optional*	string
barcodeType *optional*	enum (NONE, UPC, CODE_128, QR, CUSTOM_MEDIA)
billableEntityId *optional*	integer(int64)
categories *optional*	< CategoryResponse > array
clickedCount *optional*	integer(int32)
created *optional*	integer(int64)
customValue *optional*	number(double)
deleted *optional*	integer(int64)
details *optional*	string
device *optional*	DeviceResponse
discountPercentage *optional*	number(double)
discountPrice *optional*	number(double)
dislikeCount *optional*	integer(int64)
distance *optional*	number(double)
endDate *optional*	integer(int64)
errorCode *optional*	string
estimatedValue *optional*	number(double)

Name	Schema
event *optional*	EventResponse
expiration *optional*	integer(int64)
expires *optional*	string(date-time)
externalId *optional*	string
externalRedeemAuth *optional*	integer(int32)
externalRedeemOptions *optional*	string
externalUrl *optional*	string
favorite *optional*	boolean
favoriteCount *optional*	integer(int64)
favoriteId *optional*	integer(int64)
filters *optional*	< FilterResponse > array
finePrint *optional*	string
fullPrice *optional*	number(double)
image *optional*	string
image1 *optional*	string
image2 *optional*	string
image3 *optional*	string
image4 *optional*	string

Name	Schema
image5 *optional*	string
images *optional*	< AssetResponse > array
latitude *optional*	number(double)
likeCount *optional*	integer(int64)
location *optional*	RetailerLocationResponse
locationAddedCount *optional*	integer(int32)
locationName *optional*	string
locations *optional*	< < string, object > map > array
logId *optional*	integer(int64)
longitude *optional*	number(double)
media *optional*	MediaResponse
message *optional*	string
metaData *optional*	string
missionListResponse *optional*	MissionListResponse
noteCount *optional*	integer(int64)
offerId *optional*	integer(int64)
offerLocationId *optional*	integer(int64)
offerName *optional*	string

Name	Schema
offerType *optional*	string
parentOffer *optional*	OfferShortResponse
paymentFees *optional*	number(double)
product *optional*	ProductResponse
publisher *optional*	string
purchaseLimit *optional*	integer(int32)
purchaseLimitPerUser *optional*	integer(int32)
purchasesRemaining *optional*	integer(int32)
qrCodeUrl *optional*	string
redeemableEnd *optional*	integer(int64)
redeemableStart *optional*	integer(int64)
redemptionCode *optional*	string
redemptionLimit *optional*	integer(int32)
redemptionStatus *optional*	integer(int32)
redemptionsRemaining *optional*	integer(int32)
removedCount *optional*	integer(int32)
request *optional*	< NameStringValueResponse > array
responsibleAccountId *optional*	integer(int64)

Name	Schema
retailerId *optional*	integer(int64)
retailerPhoneNumber *optional*	string
returning *optional*	string
serializeNulls *optional*	boolean
showRedeemed *optional*	boolean
showRemaining *optional*	boolean
specialOfferType *optional*	enum (ALL, RESERVABLE, REGULAR_OFFER, ACT_NOW, GET_THERE_NOW, SQOOT, TICKETS, YIPIT)
sqootUrl *optional*	string
startDate *optional*	integer(int64)
startTimeLog *optional*	integer(int64)
starts *optional*	string(date-time)
subDetails *optional*	string
subTitle *optional*	string
tags *optional*	string
ticketPrice *optional*	integer(int64)
ticketPriceType *optional*	string
ticketsReward *optional*	integer(int64)
ticketsRewardType *optional*	string

Name	Schema
transactionId *optional*	integer(int64)
updated *optional*	integer(int64)
usedCount *optional*	integer(int32)
userRedemptionLimit *optional*	integer(int32)
valid *optional*	boolean
version *optional*	number(double)
viewedCount *optional*	integer(int32)
voucherDiscountPrice *optional*	number(double)
voucherPrice *optional*	number(double)
walletCount *optional*	integer(int32)

7.242. RetailerProfile

Name	Schema
address *optional*	Address
businessPhone *optional*	string
businessPhoneExt *optional*	string
cellCarrier *optional*	CellCarrier
cellPhone *optional*	string
code501c3 *optional*	string

Name	Schema
detailsBody *optional*	string
detailsHeader *optional*	string
detailsLineLeft1 *optional*	string
detailsLineLeft2 *optional*	string
detailsLineRight1 *optional*	string
directions *optional*	string
emailAddress *optional*	string
faxNumber *optional*	string
featured *optional*	boolean
homePhone *optional*	string
hours *optional*	string
listed *optional*	boolean
metaDescription *optional*	string
timeZone *optional*	string
utcOffset *optional*	string
web *optional*	string

7.243. RetailerResponse

Name	Schema
active *optional*	boolean
billableEntity *optional*	BillableEntityResponse
contact *optional*	ContactInfoResponse
counts *optional*	RetailerCountResponse
facebookUrl *optional*	string
latitude *optional*	number(double)
logo *optional*	AssetShortResponse
longitude *optional*	number(double)
manager *optional*	ProfileShortResponse
metaData *optional*	string
name *optional*	string
picture1 *optional*	AssetShortResponse
picture2 *optional*	AssetShortResponse
responsible *optional*	ProfileShortResponse
retailerId *optional*	integer(int64)
twitterUrl *optional*	string
visibility *optional*	string

7.244. RetailerShortResponse

Name	Schema
active *optional*	boolean
contact *optional*	ContactInfoResponse
facebookUrl *optional*	string
latitude *optional*	number(double)
logo *optional*	AssetShortResponse
longitude *optional*	number(double)
metaData *optional*	string
name *optional*	string
picture1 *optional*	AssetShortResponse
picture2 *optional*	AssetShortResponse
retailerId *optional*	integer(int64)
twitterUrl *optional*	string
visibility *optional*	string

7.245. RewardResponse

Name	Schema
artwork *optional*	AssetShortResponse
details *optional*	string

Name	Schema
estimatedValue *optional*	number(double)
finePrint *optional*	string
offerId *optional*	integer(int64)
redemptionCode *optional*	string
redemptionStatus *optional*	integer(int32)
subTitle *optional*	string
title *optional*	string
transactionId *optional*	integer(int64)

7.246. Route

Name	Schema
active *optional*	boolean
allShipmentsConfirmed *optional*	boolean
approved *optional*	string(date-time)
completed *optional*	string(date-time)
concierge *optional*	Account
contentName *optional*	string
created *optional*	string(date-time)
customType *optional*	string

Name	Schema
deactivated *optional*	boolean
driver *optional*	Account
enableOptimization *optional*	boolean
endStop *optional*	Stop
estimatedDistance *optional*	number(double)
estimatedTime *optional*	integer(int64)
externalId *optional*	string
featured *optional*	boolean
glympseToken *optional*	string
hub *optional*	ServiceHub
id *optional*	integer(int64)
image *optional*	Asset
name *optional*	string
noteCount *optional*	integer(int64)
notes *optional*	< Note > array
optimizationCompleted *optional*	string(date-time)
optimizationStarted *optional*	string(date-time)
optimizedDate *optional*	string(date-time)

Name	Schema
parent *optional*	Route
polyline *optional*	string
program *optional*	Program
scheduled *optional*	string(date-time)
shipmentOrderCount *optional*	integer(int32)
shipments *optional*	< Shipment > array
startStop *optional*	Stop
started *optional*	string(date-time)
stops *optional*	< Stop > array
totalDistance *optional*	number(float)
totalTime *optional*	integer(int64)
updated *optional*	string(date-time)
valid *optional*	boolean
vehicle *optional*	Vehicle
zone *optional*	Region

7.247. RouteSettings

Name	Schema
active *optional*	boolean

Name	Schema
created *optional*	string(date-time)
deactivated *optional*	boolean
description *optional*	string
hub *optional*	ServiceHub
id *optional*	integer(int64)
name *optional*	string
program *optional*	Program
setting *optional*	string
settingType *optional*	enum (FLOAT, DOUBLE, INTEGER, LONG, BOOLEAN, STRING)
updated *optional*	string(date-time)
valid *optional*	boolean

7.248. RoutingListResponse

Name	Schema
count *optional*	integer(int32)
debug *optional*	< string > array
errorCode *optional*	string
items *optional*	< RoutingResponse > array
message *optional*	string

Name	Schema
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.249. RoutingResponse

Name	Schema
config *optional*	ConfigRoutingRequest
debug *optional*	< string > array
drivers *optional*	< DriverResponse > array
earliestPickupWindow *optional*	integer(int64)
endTime *optional*	string(date-time)
error *optional*	string
errorCode *optional*	string
generated *optional*	boolean
items *optional*	< LoadResponse > array
itineraries *optional*	< ItineraryResponse > array

Name	Schema
latestPickupWindow *optional*	integer(int64)
loadSize *optional*	integer(int32)
message *optional*	string
orignalRequest *optional*	string
request *optional*	< NameStringValueResponse > array
responses *optional*	< RoutingResponse > array
returning *optional*	string
runtime *optional*	integer(int64)
score *optional*	integer(int64)
serializeNulls *optional*	boolean
startTime *optional*	string(date-time)
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
vehicles *optional*	< VehicleResponse > array
version *optional*	number(double)

7.250. RuleListResponse

Name	Schema
count *optional*	integer(int32)

Name	Schema
countTotal *optional*	integer(int64)
hasMoreResults *optional*	boolean
items *optional*	< RuleResponse > array
limit *optional*	integer(int32)
start *optional*	integer(int32)

7.251. RuleResponse

Name	Schema
description *optional*	string
ruleId *optional*	integer(int64)
ruleTypeDescription *optional*	string
ruleTypeId *optional*	integer(int64)
ruleTypeTitle *optional*	string
ruleValue *optional*	string
ruleValueType *optional*	string
title *optional*	string

7.252. ScheduledNotification

Name	Schema
accountIds *optional*	< integer(int64) > array

Name	Schema
active *optional*	boolean
albums *optional*	< Album > array
application *optional*	Application
approvalStatus *optional*	enum (PENDING, REJECTED, APPROVED, FEATURED)
audiences *optional*	< Audience > array
billableEntity *optional*	BillableEntity
conditionalCode *optional*	string
conditionalInput *optional*	string
connectionGroups *optional*	< ConnectionGroup > array
contentAsset *optional*	Asset
contentId *optional*	integer(int64)
contentName *optional*	string
contentType *optional*	string
created *optional*	string(date-time)
cronExpression *optional*	string
cronType *optional*	enum (CRON4J, QUARTZ, UNIX)
customType *optional*	string
deactivated *optional*	boolean

Name	Schema
deepLinkURI *optional*	string
deliverCount *optional*	integer(int64)
deliverEstimate *optional*	integer(int64)
deliverFailed *optional*	integer(int64)
deliverLimit *optional*	integer(int64)
deliverStart *optional*	integer(int64)
deliverTotal *optional*	integer(int64)
endDate *optional*	string(date-time)
endpointURL *optional*	string
errorMessage *optional*	string

Name	Schema
event *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
groupingId *optional*	string
id *optional*	integer(int64)
lastActionDate *optional*	string(date-time)
message *optional*	string
metaData *optional*	string
mobilePush *optional*	boolean
name *optional*	string
owner *optional*	Account
parentId *optional*	integer(int64)
parentType *optional*	string
payload *optional*	string

Name	Schema
permissionRequestEvent *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
permissionSharedEvent *optional*	enum (REASSIGNMENT_REQUEST, FEEDBACK, SUGGESTION_RECEIVED, ASSIGNMENT_CHANGE, WELCOME_PARTNER, WELCOME_BUSINESS, PARTNER_INVITE, PARTNER_EXISTING_INVITE, PARTNER_CODE_INVITE, BATCH_REPORT_FINISHED, WELCOME_BY_APP, VALIDATE_MEMBER, NEW_SIGNUP, VOUCHER_PURCHASED, PENDING_MISSION_REVIEW, SEND_APP_KEYS, RESET_PASSWORD, PASSWORD_CHANGED, OFFER_24HR_EXPIRE_RETAILER, OFFER_24HR_EXPIRE_CONSUMER, DOWNLOADED_APP, LEVEL_COMPLETED, LEVEL_CREATED, ALBUM_LIKE, ALBUM_CONTEST_LIKE, ASSET_LIKE, COLLECTION_LIKE, ACCOUNT_LIKE, NOTE_LIKE, GAME_LEVEL_LIKE, LIKE, ALBUM_DISLIKE, ALBUM_CONTEST_DISLIKE, ASSET_DISLIKE, COLLECTION_DISLIKE, ACCOUNT_DISLIKE, NOTE_DISLIKE, GAME_LEVEL_DISLIKE, ALBUM_NOTE, ALBUM_CONTEST_NOTE, ASSET_NOTE, COLLECTION_NOTE, GAME_LEVEL_NOTE, COMMENT, ALBUM_SHARED, ALBUM_CONTEST_SHARED, COLLECTION_SHARED, GAME_LEVEL_SHARED, ASSIGNMENT_SHARED, ANONYMOUS_ALBUM_SHARED, EVENT_SHARED, RETAILER_LOCATION_SHARED, ALBUM_SHARE_REQUEST, ALBUM_CONTEST_SHARE_REQUEST, COLLECTION_SHARE_REQUEST, GAME_LEVEL_SHARE_REQUEST, ASSIGNMENT_SHARE_REQUEST, OFFER_FAVORITE, OFFER_LOCATION_FAVORITE, RETAILER_LOCATION_FAVORITE, ALBUM_FAVORITE, CATEGORY_FAVORITE, ALBUM_PENDING, ALBUM_CONTEST_PENDING, ALBUM_APPROVED, ALBUM_CONTEST_APPROVED, ALBUM_FEATURED, ALBUM_CONTEST_FEATURED, ALBUM_NEW, ALBUM_CONTEST_NEW, ALBUM_ASSETS_ADDED, COLLECTION_ASSETS_ADDED, ALBUM_PHOTOS_ADDED, ALBUM_CONTEST_ALBUM_ADD, CHALLENGE, EARN_COMMISSION_TICKETS, FRIEND_REQUESTED, FRIEND_REQUEST_ACCEPTED, FRIEND_REQUEST_REJECTED, FRIEND_REMOVED, FRIEND_ADDED, FOLLOW, PURCHASE_ITEM_GIFTED, PROFILE_PROXIMITY, CUSTOM, ADMIN, INVOICE, SUGGESTIONS_SUMMARY, SUGGESTIONS_LINK, EVENT_ATTENDANCE, EVENTS_TODAY, OFFER_CREATE, EVENT_CREATE, MATCH_WINNER, MATCH_LOSER, MATCH_TIE, MATCH_TIE_WINNER, MATCH_TIE_LOSER, TOURNAMENT_REFUND, TOURNAMENT_WON, TOURNAMENT_WINNER, ROUND_START_SOON, ROUND_END_SOON, ROUND_TIE_BREAKER,

Name	Schema
processingTime *optional*	integer(int64)
publicPermissions *optional*	Permissions
report *optional*	ReportDefinition
reportParams *optional*	string
scheduledDate *optional*	string(date-time)
sendingTime *optional*	integer(int64)
startDate *optional*	string(date-time)
status *optional*	enum (NEW, ERROR, COMPLETE, PROCESSING, TEMPLATE)
templateType *optional*	enum (REPORTING, ACHIEVEMENT_OWNER, ACHIEVEMENT_TRIGGER, GEOFENCE_OWNER, GEOFENCE_TARGET, GEOFENCE_TRIGGER, GEOFENCE_HTTP_CALL, SIGNUP)
type *optional*	enum (HTTP, SAVE, EMAIL, SMS, APNS, GCM, PUSH, MOBILE_NOTIFICATION)
updated *optional*	string(date-time)
userPermissions *optional*	< UserPermissions > array
valid *optional*	boolean
visibility *optional*	enum (PUBLIC, PRIVATE, FRIENDS)

7.253. ScheduledNotificationFullResponse

Name	Schema
active *optional*	boolean

Name	Schema
application *optional*	ApplicationMiniResponse
billableEntity *optional*	BillableEntityShortResponse
conditionalInput *optional*	string
connectionAccountIds *optional*	< integer(int64) > array
contentId *optional*	integer(int64)
contentName *optional*	string
contentType *optional*	string
created *optional*	integer(int64)
cronExpression *optional*	string
deepLinkURI *optional*	string
deliverCount *optional*	integer(int64)
deliverEstimate *optional*	integer(int64)
deliverFailed *optional*	integer(int64)
deliverTotal *optional*	integer(int64)
endDate *optional*	integer(int64)
endpointURL *optional*	string
groupedTypes *optional*	< enum (HTTP, SAVE, EMAIL, SMS, APNS, GCM, PUSH, MOBILE_NOTIFICATION) > array
groupingId *optional*	string

Name	Schema
message *optional*	string
metaData *optional*	string
name *optional*	string
owner *optional*	AccountShortResponse
parentId *optional*	integer(int64)
parentType *optional*	string
payload *optional*	string
processingTime *optional*	integer(int64)
scheduledDate *optional*	integer(int64)
scheduledNotificationId *optional*	integer(int64)
sendingTime *optional*	integer(int64)
startDate *optional*	integer(int64)
status *optional*	enum (NEW, ERROR, COMPLETE, PROCESSING, TEMPLATE)
targetId *optional*	integer(int64)
targetName *optional*	string
targetParams *optional*	string
targetType *optional*	string

Name	Schema
templateType *optional*	enum (REPORTING, ACHIEVEMENT_OWNER, ACHIEVEMENT_TRIGGER, GEOFENCE_OWNER, GEOFENCE_TARGET, GEOFENCE_TRIGGER, GEOFENCE_HTTP_CALL, SIGNUP)
type *optional*	enum (HTTP, SAVE, EMAIL, SMS, APNS, GCM, PUSH, MOBILE_NOTIFICATION)
updated *optional*	integer(int64)

7.254. ScheduledNotificationShortResponse

Name	Schema
active *optional*	boolean
endDate *optional*	integer(int64)
groupingId *optional*	string
message *optional*	string
name *optional*	string
scheduledDate *optional*	integer(int64)
scheduledNotificationId *optional*	integer(int64)
startDate *optional*	integer(int64)
status *optional*	enum (NEW, ERROR, COMPLETE, PROCESSING, TEMPLATE)
type *optional*	enum (HTTP, SAVE, EMAIL, SMS, APNS, GCM, PUSH, MOBILE_NOTIFICATION)

7.255. ScoreListResponse

Name	Schema
count *optional*	integer(int32)
countTotal *optional*	integer(int64)
hasMoreResults *optional*	boolean
items *optional*	< ScoreResponse > array
limit *optional*	integer(int32)
start *optional*	integer(int32)

7.256. ScoreResponse

Name	Schema
accountDisplay *optional*	string
accountId *optional*	integer(int64)
action *optional*	enum (DATA, SAVE, DELETE, GET)
gameObjectId *optional*	integer(int64)
gameType *optional*	string
points *optional*	integer(int32)
scoreId *optional*	integer(int64)
status *optional*	enum (COMPLETED, OWNER_CREDIT, SUMMATION, TOURNAMENT)
ticketsEarned *optional*	integer(int64)
timeTaken *optional*	integer(int32)

7.257. SearchResponse

Name	Schema
count *optional*	integer(int32)
countTotal *optional*	integer(int64)
errorCode *optional*	string
hasMoreResults *optional*	boolean
items *optional*	< object > array
limit *optional*	integer(int64)
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
start *optional*	integer(int64)
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.258. ServiceHub

Name	Schema
active *optional*	boolean

Name	Schema
business *optional*	Retailer
created *optional*	string(date-time)
deactivated *optional*	boolean
externalId *optional*	string
id *optional*	integer(int64)
latitude *optional*	number(double)
locationDescription *optional*	string
longitude *optional*	number(double)
name *optional*	string
timeZone *optional*	string
updated *optional*	string(date-time)
valid *optional*	boolean

7.259. ServiceType

Name	Schema
active *optional*	boolean
additionalLoadTime *optional*	integer(int64)
business *optional*	Retailer
created *optional*	string(date-time)

Name	Schema
dateOverride *optional*	enum (NONE, PICKUP_START, PICKUP_END, DROPOFF_START, DROPOFF_END)
deactivated *optional*	boolean
dropoffAddressOverride *optional*	Address
dropoffEndTimeOverride *optional*	LocalTime
dropoffStartTimeOverride *optional*	LocalTime
hub *optional*	ServiceHub
id *optional*	integer(int64)
name *optional*	string
pickupAddressOverride *optional*	Address
pickupEndTimeOverride *optional*	LocalTime
pickupStartTimeOverride *optional*	LocalTime
updated *optional*	string(date-time)
valid *optional*	boolean

7.260. Shipment

Name	Schema
active *optional*	boolean
batchId *optional*	string
canceled *optional*	string(date-time)

Name	Schema
cargoType *optional*	CargoType
complete *optional*	boolean
confirmed *optional*	string(date-time)
contentName *optional*	string
created *optional*	string(date-time)
customType *optional*	string
deactivated *optional*	boolean
dropOffStop *optional*	Stop
dropOffTimeWindow *optional*	DateTimeRange
id *optional*	integer(int64)
name *optional*	string
noteCount *optional*	integer(int64)
notes *optional*	< Note > array
owner *optional*	Account
pickupStop *optional*	Stop
pickupTimeWindow *optional*	DateTimeRange
polyline *optional*	string
promoCode *optional*	PromoCode

Name	Schema
quantity *optional*	integer(int32)
refunded *optional*	boolean
replacePassenger *optional*	Shipment
reserved *optional*	string(date-time)
rider *optional*	Account
riderName *optional*	string
route *optional*	Route
routeId *optional*	integer(int64)
serviceType *optional*	ServiceType
totalVolume *optional*	number(double)
updated *optional*	string(date-time)
valid *optional*	boolean

7.261. ShipmentBatch

Name	Schema
active *optional*	boolean
averageTravelSpeed *optional*	number(double)
completed *optional*	string(date-time)
created *optional*	string(date-time)

Name	Schema
deactivated *optional*	boolean
externalId *optional*	string
failedCount *optional*	integer(int32)
hardScore *optional*	integer(int64)
hub *optional*	ServiceHub
id *optional*	integer(int64)
importStatus *optional*	enum (NEW, STARTED, FETCHING_DISTANCE_DATA, OPTIMIZING, CREATING_ROUTES, COMPLETED, INCOMPLETE)
initScore *optional*	integer(int32)
invalidCount *optional*	integer(int32)
legsRequested *optional*	integer(int32)
maxSeconds *optional*	integer(int64)
mediumScore *optional*	integer(int64)
optimizationCompleted *optional*	string(date-time)
optimizationStarted *optional*	string(date-time)
processedCount *optional*	integer(int32)
program *optional*	Program
retailer *optional*	Retailer

Name	Schema
routesCreated *optional*	integer(int32)
shipmentImportStatuses *optional*	< ShipmentImportStatus > array
shipments *optional*	< Shipment > array
softScore *optional*	integer(int64)
source *optional*	string
started *optional*	string(date-time)
successCount *optional*	integer(int32)
total *optional*	integer(int32)
totalLegsRequests *optional*	integer(int32)
totalRoutes *optional*	integer(int32)
updated *optional*	string(date-time)
uploaded *optional*	string(date-time)
valid *optional*	boolean
validCount *optional*	integer(int32)

7.262. ShipmentImportStatus

Name	Schema
active *optional*	boolean
batchId *optional*	string

Name	Schema
completed *optional*	string(date-time)
created *optional*	string(date-time)
deactivated *optional*	boolean
id *optional*	integer(int64)
message *optional*	string
name *optional*	string
shipment *optional*	Shipment
started *optional*	string(date-time)
updated *optional*	string(date-time)
valid *optional*	boolean

7.263. ShipmentOrder

Name	Schema
AccountNo *optional*	string
BatchID *optional*	integer(int32)
BatchSize *optional*	integer(int32)
ClientRefNo *optional*	string
ClientRefNo2 *optional*	string
CompanyName *optional*	string

Name	Schema
CubicFeet *optional*	number(double)
DeliveryCity *optional*	string
DeliveryCompany *optional*	string
DeliveryContact *optional*	string
DeliveryETA *optional*	string
DeliveryPhone *optional*	string
DeliverySequenceNo *optional*	integer(int32)
DeliveryState *optional*	string
DeliveryStreet *optional*	string
DeliveryStreet2 *optional*	string
DeliveryTargetFrom *optional*	string(date-time)
DeliveryTargetTo *optional*	string(date-time)
DeliveryZip *optional*	string
DriverID *optional*	integer(int32)
HubCode *optional*	string
OrderPackages *optional*	< OrderPackage > array
OrderTrackingID *optional*	number(double)
PickupCity *optional*	string

Name	Schema
PickupCompany *optional*	string
PickupContact *optional*	string
PickupETA *optional*	string
PickupPhone *optional*	string
PickupSequenceNo *optional*	integer(int32)
PickupState *optional*	string
PickupStreet *optional*	string
PickupStreet2 *optional*	string
PickupTargetFrom *optional*	string(date-time)
PickupTargetTo *optional*	string(date-time)
PickupZip *optional*	string
Pieces *optional*	integer(int32)
RouteID *optional*	integer(int32)
ServiceCode *optional*	string
ServiceTypeName *optional*	string
VehicleTypeName *optional*	string
Weight *optional*	number(double)
deactivated *optional*	boolean

Name	Schema
valid *optional*	boolean

7.264. SirqulResponse

Name	Schema
errorCode *optional*	string
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.265. SizeGroup

Name	Schema
active *optional*	boolean
created *optional*	string(date-time)
deactivated *optional*	boolean
description *optional*	string
groupingId *optional*	integer(int64)

Name	Schema
id *optional*	integer(int64)
name *optional*	string
updated *optional*	string(date-time)
valid *optional*	boolean

7.266. SizeGroupResponse

Name	Schema
description *optional*	string
id *optional*	integer(int64)
name *optional*	string

7.267. Step

Name	Schema
active *optional*	boolean
created *optional*	string(date-time)
deactivated *optional*	boolean
deviceId *optional*	string
distance *optional*	number(double)
duration *optional*	integer(int64)
endAltitude *optional*	number(double)

Name	Schema
endDate *optional*	string(date-time)
endLatitude *optional*	number(double)
endLongitude *optional*	number(double)
id *optional*	integer(int64)
leg *optional*	Leg
owner *optional*	Account
startAltitude *optional*	number(double)
startDate *optional*	string(date-time)
startLatitude *optional*	number(double)
startLongitude *optional*	number(double)
updated *optional*	string(date-time)
valid *optional*	boolean

7.268. StepResponse

Name	Schema
accuracy *optional*	integer(int32)
distance *optional*	number(double)
duration *optional*	integer(int64)
endAlt *optional*	number(double)

Name	Schema
endDate *optional*	integer(int64)
endLat *optional*	number(double)
endLng *optional*	number(double)
leg *optional*	LegResponse
missingStartLocation *optional*	boolean
startAlt *optional*	number(double)
startDate *optional*	integer(int64)
startLat *optional*	number(double)
startLng *optional*	number(double)

7.269. Stop

Name	Schema
active *optional*	boolean
actualTime *optional*	string(date-time)
compareEstimatedArrivalTime *optional*	string(date-time)
compareEstimatedTime *optional*	string(date-time)
compareEstimiatedDepartureTime *optional*	string(date-time)
complete *optional*	boolean
contentName *optional*	string

Name	Schema
created *optional*	string(date-time)
customType *optional*	string
deactivated *optional*	boolean
desiredTime *optional*	string(date-time)
endTime *optional*	string(date-time)
estimatedArrivalTime *optional*	string(date-time)
estimatedDepartureTime *optional*	string(date-time)
estimatedTime *optional*	string(date-time)
id *optional*	integer(int64)
latitude *optional*	number(double)
location *optional*	Location
locationAddress *optional*	string
locationDescription *optional*	string
locationDisplay *optional*	string
longitude *optional*	number(double)
name *optional*	string
noteCount *optional*	integer(int64)
notes *optional*	< Note > array

Name	Schema
orderIndex *optional*	integer(int64)
owner *optional*	Account
parent *optional*	Stop
polyline *optional*	string
progressStatus *optional*	enum (PENDING, COMPLETE, INVALID)
route *optional*	Route
shipment *optional*	Shipment
startTime *optional*	string(date-time)
stopType *optional*	enum (PICKUP, DROPOFF, START, END, ROUTING)
timePadding *optional*	integer(int32)
updated *optional*	string(date-time)
valid *optional*	boolean

7.270. StopResponse

Name	Schema
endWindow *optional*	integer(int64)
error *optional*	string
errorCode *optional*	string
eta *optional*	integer(int64)

Name	Schema
etd *optional*	integer(int64)
floorPlanPath *optional*	< NodeRequest > array
geoId *optional*	string
itemId *optional*	string
itemSize *optional*	integer(int64)
latitude *optional*	number(double)
load *optional*	LoadShortResponse
longitude *optional*	number(double)
score *optional*	integer(int64)
startWindow *optional*	integer(int64)
type *optional*	enum (PICKUP, DROPOFF, START, END, ROUTING)
valid *optional*	boolean

7.271. Subscription

Name	Schema
active *optional*	boolean
created *optional*	string(date-time)
deactivated *optional*	boolean
failed *optional*	boolean

Name	Schema
id *optional*	integer(int64)

Name	Schema
message *optional*	enum (_400, _403, _500, _600, _1000, _1001, _1002, _1003, _1004, _1005, _1006, _1007, _1008, _1009, _1010, _1011, _1012, _1013, _1014, _1015, _1016, _1017, _1018, _1019, _1020, _1021, _1022, _1023, _1024, _1025, _1026, _1027, _1028, _1029, _1030, _1031, _1032, _1033, _1034, _1035, _1036, _1037, _1038, _1039, _1040, _1041, _1042, _1043, _1044, _1045, _1046, _1047, _1048, _1049, _1050, _1051, _1052, _1053, _1054, _1055, _1056, _1057, _1058, _1059, _1060, _1061, _1062, _1063, _1064, _1065, _1066, _1100, _1101, _1102, _1103, _1104, _1105, _1106, _1107, _1108, _1109, _1110, _1111, _1112, _1113, _1114, _1115, _2000, _2001, _2002, _2003, _2004, _2005, _2006, _2007, _2008, _2009, _2010, _2011, _2012, _2013, _2014, _2100, _2101, _2102, _2103, _2200, _2201, _2202, _2203, _2204, _2205, _2206, _2207, _2208, _2209, _2210, _2211, _2212, _2213, _2214, _2215, _2216, _2217, _2218, _2219, _2220, _2300, _2301, _2302, _2303, _2400, _2401, _2402, _2403, _2404, _2405, _2451, _2452, _2453, _2454, _2455, _2456, _2457, _2458, _2500, _2501, _2502, _2503, _2504, _2505, _2506, _2507, _2508, _2509, _2510, _2511, _2512, _2513, _2514, _2515, _2600, _2601, _2602, _2603, _2604, _2605, _2700, _2701, _2702, _2703, _2704, _2705, _2800, _2801, _2900, _2901, _2902, _2903, _2904, _2905, _2906, _2907, _3000, _3001, _3002, _3100, _3101, _3102, _3103, _3104, _3200, _3201, _3300, _3301, _3302, _3303, _3304, _3305, _3306, _3307, _3308, _3309, _3310, _3311, _3312, _3313, _3314, _3315, _3316, _3400, _3401, _3500, _3501, _3502, _3503, _3504, _3505, _3506, _3507, _3508, _3600, _3601, _3602, _3603, _3700, _3701, _3702, _3703, _3704, _3705, _3800, _3801, _3802, _3803, _3804, _3805, _3806, _3807, _3808, _3809, _3810, _3811, _3900, _3901, _3902, _3903, _3904, _3905, _4000, _4001, _4002, _4100, _4101, _4102, _4103, _4200, _4201, _4202, _4203, _4204, _4205, _4300, _4301, _4302, _4303, _4304, _4305, _4400, _4401, _4500, _4501, _4502, _4503, _4504, _4505, _4506, _4600, _4700, _4701, _4702, _4800, _4801, _4900, _4901, _4902, _4903, _4904, _4905, _4906, _5000, _5001, _5002, _5003, _5100, _5101, _5102, _5103, _5104, _5200, _5300, _5301, _5302, _5303, _5400, _5500, _5501, _5600, _5601, _5602, _5603, _5604, _5606, _5607, _5608, _5609, _5700, _5701, _5702, _5703, _5704, _5705, _5706, _5707, _5708, _5800, _5801, _5802, _5803, _5804, _5805, _5806, _5807, _5850, _5851, _5852, _5900, _5901, _5902, _5903, _5904, _5905, _5906, _5907, _5908, _5909, _5910, _5911, _5912, _5913, _5914, _5915, _5916, _5917, _5918, _5919, _5920, _5921, _5922, _6000, _6001, _6100, _6101, _6102, _6103, _6200, _6201, _6202, _6300, _6301, _6302, _6303, _6304, _6305, _6306, _6307, _6308, _6309, _6310, _6311, _6312, _6400, _6401, _6402, _6403, _6404, _6500, _6501, _6502, _6600, _6601, _6602,

Name	Schema
metaData *optional*	string
promoCode *optional*	string
remoteSubscriptionId *optional*	string
subscriptionPlan *optional*	SubscriptionPlan
updated *optional*	string(date-time)
valid *optional*	boolean

7.272. SubscriptionOption

Name	Schema
active *optional*	boolean
created *optional*	string(date-time)
deactivated *optional*	boolean
id *optional*	integer(int64)
name *optional*	string
offerType *optional*	enum (VOUCHER, COUPON, PRODUCT, MEDIA, EVENT, DEVICE)
optionType *optional*	enum (MISSION, LOCATION, OFFER, NOTIFICATION, REQUEST, STORAGE_S3, STORAGE_DB)
price *optional*	number(double)
quantity *optional*	integer(int32)
threshold *optional*	integer(int32)

Name	Schema
updated *optional*	string(date-time)
valid *optional*	boolean
visible *optional*	boolean

7.273. SubscriptionOptionResponse

Name	Schema
id *optional*	integer(int64)
name *optional*	string
optionType *optional*	string
price *optional*	number(double)
quantity *optional*	integer(int32)
threshold *optional*	integer(int32)
visible *optional*	boolean

7.274. SubscriptionPlan

Name	Schema
active *optional*	boolean
created *optional*	string(date-time)
deactivated *optional*	boolean
defaultPlan *optional*	boolean

Name	Schema
description *optional*	string
id *optional*	integer(int64)
image *optional*	Asset
metaData *optional*	string
name *optional*	string
options *optional*	< SubscriptionOption > array
price *optional*	number(double)
promo *optional*	number(double)
role *optional*	string
transactionFee *optional*	number(double)
updated *optional*	string(date-time)
valid *optional*	boolean
visible *optional*	boolean

7.275. SubscriptionPlanResponse

Name	Schema
defaultPlan *optional*	boolean
description *optional*	string
id *optional*	integer(int64)

Name	Schema
image *optional*	AssetShortResponse
name *optional*	string
options *optional*	< SubscriptionOptionResponse > array
price *optional*	number(double)
promo *optional*	number(double)
role *optional*	string
transactionFee *optional*	number(double)
visible *optional*	boolean

7.276. SubscriptionResponse

Name	Schema
active *optional*	boolean
id *optional*	integer(int64)
metaData *optional*	string
promoCode *optional*	string
subscriptionPlan *optional*	SubscriptionPlanResponse

7.277. SummaryRankResponse

Name	Schema
appName *optional*	string

Name	Schema
applicationId *optional*	integer(int64)
associatedLeaderboard *optional*	LeaderboardResponse
countValue *optional*	integer(int64)
endDate *optional*	integer(int64)
owner *optional*	AccountShortResponse
rank *optional*	string
rankType *optional*	string
scoreValue *optional*	integer(int64)
startDate *optional*	integer(int64)
streakBestCount *optional*	integer(int64)
streakCount *optional*	integer(int64)
sumCounts *optional*	integer(int64)
sumScores *optional*	integer(int64)
sumUsers *optional*	integer(int64)
timeValue *optional*	integer(int64)
updated *optional*	integer(int64)

7.278. TaskResponse

Name	Schema
active *optional*	boolean
application *optional*	ApplicationMiniResponse
billableEntity *optional*	BillableEntityShortResponse
conditionalInput *optional*	string
connectionAccountIds *optional*	< integer(int64) > array
contentId *optional*	integer(int64)
contentName *optional*	string
contentType *optional*	string
created *optional*	integer(int64)
cronExpression *optional*	string
deepLinkURI *optional*	string
deliverCount *optional*	integer(int64)
deliverEstimate *optional*	integer(int64)
deliverFailed *optional*	integer(int64)
deliverTotal *optional*	integer(int64)
endDate *optional*	integer(int64)
endpointURL *optional*	string
groupedTypes *optional*	< enum (HTTP, SAVE, EMAIL, SMS, APNS, GCM, PUSH, MOBILE_NOTIFICATION) > array

Name	Schema
groupingId *optional*	string
message *optional*	string
metaData *optional*	string
name *optional*	string
owner *optional*	AccountShortResponse
parentId *optional*	integer(int64)
parentType *optional*	string
payload *optional*	string
processingTime *optional*	integer(int64)
scheduledDate *optional*	integer(int64)
scheduledNotificationId *optional*	integer(int64)
sendingTime *optional*	integer(int64)
startDate *optional*	integer(int64)
status *optional*	enum (NEW, ERROR, COMPLETE, PROCESSING, TEMPLATE)
targetId *optional*	integer(int64)
targetName *optional*	string
targetParams *optional*	string
targetType *optional*	string

Name	Schema
taskId *optional*	integer(int64)
templateType *optional*	enum (REPORTING, ACHIEVEMENT_OWNER, ACHIEVEMENT_TRIGGER, GEOFENCE_OWNER, GEOFENCE_TARGET, GEOFENCE_TRIGGER, GEOFENCE_HTTP_CALL, SIGNUP)
type *optional*	enum (HTTP, SAVE, EMAIL, SMS, APNS, GCM, PUSH, MOBILE_NOTIFICATION)
updated *optional*	integer(int64)

7.279. Territory

Name	Schema
active *optional*	boolean
created *optional*	string(date-time)
deactivated *optional*	boolean
id *optional*	integer(int64)
name *optional*	string
updated *optional*	string(date-time)
valid *optional*	boolean

7.280. TerritoryResponse

Name	Schema
id *optional*	integer(int64)
name *optional*	string

7.281. ThemeDescriptorResponse

Name	Schema
errorCode *optional*	string
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
themeDescriptorId *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.282. ThirdPartyCredentialResponse

Name	Schema
account *optional*	AccountShortResponse
created *optional*	integer(int64)
errorCode *optional*	string
errorMessage *optional*	string
metaData *optional*	string
network *optional*	ThirdPartyNetworkShortResponse

Name	Schema
secretExpires *optional*	integer(int64)
sessionId *optional*	string
thirdPartyCredentialId *optional*	integer(int64)
thirdPartyId *optional*	string
thirdPartyName *optional*	string
thirdPartyRefreshToken *optional*	string
thirdPartyToken *optional*	string
tokenExpires *optional*	integer(int64)
updated *optional*	integer(int64)
validated *optional*	boolean

7.283. ThirdPartyCredentials

Name	Schema
account *optional*	Account
active *optional*	boolean
created *optional*	string(date-time)
deactivated *optional*	boolean
errorCode *optional*	string
errorMessage *optional*	string

Name	Schema
id *optional*	integer(int64)
metaData *optional*	string
salt *optional*	string
secretExpired *optional*	boolean
secretExpires *optional*	string(date-time)
sessionId *optional*	string
thirdPartyId *optional*	string
thirdPartyName *optional*	string
thirdPartyNetwork *optional*	ThirdPartyNetwork
thirdPartyRefreshToken *optional*	string
thirdPartySecret *optional*	string
thirdPartyUnencryptedToken *optional*	string
tokenExpires *optional*	string(date-time)
updated *optional*	string(date-time)
valid *optional*	boolean
validated *optional*	boolean

7.284. ThirdPartyNetwork

Name	Schema
active *optional*	boolean
billableEntity *optional*	BillableEntity
created *optional*	string(date-time)
deactivated *optional*	boolean
description *optional*	string
enableIntrospection *optional*	boolean
enableMFA *optional*	boolean
id *optional*	integer(int64)
introspectionMethod *optional*	enum (GET, HEAD, POST, PUT, PATCH, DELETE, OPTIONS, TRACE)
introspectionParams *optional*	string
introspectionURL *optional*	string
name *optional*	string
networkUID *optional*	string
requiredRootField *optional*	string
shelfLifeMFA *optional*	integer(int32)
sizeMFA *optional*	integer(int32)
supportedNetwork *optional*	enum (FACEBOOK, TWITTER, CLEAR, USERNAME, PHONE, PHONE_V2, EMAIL, UNKNOWN, CUSTOM, CUSTOM_SECURE, APPLE)

Name	Schema
updated *optional*	string(date-time)
valid *optional*	boolean

7.285. ThirdPartyNetworkResponse

Name	Schema
active *optional*	boolean
created *optional*	integer(int64)
description *optional*	string
enableIntrospection *optional*	boolean
introspectionMethod *optional*	enum (GET, HEAD, POST, PUT, PATCH, DELETE, OPTIONS, TRACE)
introspectionParams *optional*	string
introspectionURL *optional*	string
name *optional*	string
network *optional*	enum (FACEBOOK, TWITTER, CLEAR, USERNAME, PHONE, PHONE_V2, EMAIL, UNKNOWN, CUSTOM, CUSTOM_SECURE, APPLE)
networkUID *optional*	string
requiredRootField *optional*	string
updated *optional*	integer(int64)

7.286. ThirdPartyNetworkShortResponse

Name	Schema
active *optional*	boolean
created *optional*	integer(int64)
description *optional*	string
name *optional*	string
network *optional*	enum (FACEBOOK, TWITTER, CLEAR, USERNAME, PHONE, PHONE_V2, EMAIL, UNKNOWN, CUSTOM, CUSTOM_SECURE, APPLE)
networkUID *optional*	string
updated *optional*	integer(int64)

7.287. TicketCountResponse

Name	Schema
count *optional*	integer(int64)
type *optional*	string

7.288. TicketListResponse

Name	Schema
count *optional*	integer(int32)
countTotal *optional*	integer(int64)
errorCode *optional*	string
hasMoreResults *optional*	boolean

Name	Schema
items *optional*	< TicketResponse > array
limit *optional*	integer(int32)
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
start *optional*	integer(int32)
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.289. TicketOfferResponse

Name	Schema
description *optional*	string
id *optional*	integer(int64)
imageURL *optional*	string
retailer *optional*	string
title *optional*	string

7.290. TicketResponse

Name	Schema
accountId *optional*	integer(int64)
action *optional*	enum (DATA, SAVE, DELETE, GET)
actionType *optional*	enum (COMPLETED, REDEEMED, USERS_PLAYED, TOURNAMENT_OWNER, PURCHASED, SUMATION, GIFTED, REFUNDED)
application *optional*	ApplicationShortResponse
count *optional*	integer(int64)
customAsset *optional*	AssetShortResponse
customMessage *optional*	string
id *optional*	integer(int64)
objectId *optional*	integer(int64)
purchaseItem *optional*	PurchaseItemShortResponse
receiptToken *optional*	string
receiver *optional*	< AccountShortResponse > array
sender *optional*	AccountShortResponse
ticketType *optional*	string
type *optional*	enum (GAME_OBJECT, GAME_LEVEL, PACK, GAME, MISSION, PROFILE, APPLICATION, TICKETS, ASSET, CUSTOM)
used *optional*	integer(int64)

7.291. TimeSlotResponse

Name	Schema
dayOfWeek *optional*	integer(int32)
endTime *optional*	integer(int64)
reservationCount *optional*	integer(int32)
reservations *optional*	< AccountShortResponse > array
scheduledDate *optional*	integer(int64)
startTime *optional*	integer(int64)
timeZone *optional*	string

7.292. TokenResponse

Name	Schema
errorCode *optional*	string
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
token *optional*	string
valid *optional*	boolean

Name	Schema
version *optional*	number(double)

7.293. TournamentResponse

Name	Schema
acceptedCount *optional*	integer(int32)
action *optional*	enum (DATA, SAVE, DELETE, GET)
active *optional*	boolean
activePackId *optional*	integer(int64)
advancedReporting *optional*	boolean
allocateTickets *optional*	boolean
audiences *optional*	< AudienceResponse > array
authorOverride *optional*	string
availableLimit *optional*	integer(int32)
balance *optional*	number(double)
childCount *optional*	integer(int64)
costToPlay *optional*	integer(int32)
costToPlayType *optional*	string
creatives *optional*	< CreativeResponse > array
currentRoundType *optional*	string

Name	Schema
description *optional*	string
enableBuyBack *optional*	boolean
enableMultipleEntries *optional*	boolean
enableMultipleVotes *optional*	boolean
endDate *optional*	integer(int64)
fixedReward *optional*	boolean
formatType *optional*	string
games *optional*	GameListResponse
highestAppVersion *optional*	string
icon *optional*	AssetShortResponse
image *optional*	AssetShortResponse
inviteCount *optional*	integer(int32)
inviteStatus *optional*	enum (AVAILABLE, PENDING, UPDATED, ACCEPTED, ACTIVE, BUY_BACK, FAILED, QUIT, COMPLETED, PENDING_REVIEW, REJECTED)
invitee *optional*	AccountShortResponse
joinCode *optional*	string
lastRound *optional*	integer(int32)
metaData *optional*	string

Name	Schema
minimumToPlay *optional*	integer(int32)
missionId *optional*	integer(int64)
missionInvite *optional*	MissionInviteResponse
missionInviteId *optional*	integer(int64)
missionStatus *optional*	string
missionType *optional*	string
overallRankType *optional*	string
owner *optional*	AccountShortResponse
points *optional*	integer(int64)
preliminaryGroupAdvancements *optional*	string
preliminaryGroups *optional*	integer(int32)
refunded *optional*	boolean
results *optional*	< NameStringValueResponse > array
rewarded *optional*	boolean
rewards *optional*	< RewardResponse > array
roundCount *optional*	integer(int32)
scores *optional*	ScoreListResponse
secondsBetweenLevels *optional*	integer(int32)

Name	Schema
secondsBetweenPacks *optional*	integer(int32)
secondsForTieBreaker *optional*	integer(int32)
seedAlbumId *optional*	integer(int64)
sequenceType *optional*	string
splitReward *optional*	string
startDate *optional*	integer(int64)
subType *optional*	string
ticketCount *optional*	integer(int64)
ticketType *optional*	string
ticketsEarned *optional*	integer(int64)
tieTag *optional*	string
title *optional*	string
tournamentData *optional*	string
updatedDate *optional*	integer(int64)
userPermissionsList *optional*	UserPermissionsListResponse
winnerGameObject *optional*	GameObjectResponse
winnerTag *optional*	string

7.294. TriggerResponse

Name	Schema
active *optional*	boolean
application *optional*	ApplicationMiniResponse
billableEntity *optional*	BillableEntityShortResponse
conditionalInput *optional*	string
connectionAccountIds *optional*	< integer(int64) > array
contentId *optional*	integer(int64)
contentName *optional*	string
contentType *optional*	string
created *optional*	integer(int64)
cronExpression *optional*	string
deepLinkURI *optional*	string
deliverCount *optional*	integer(int64)
deliverEstimate *optional*	integer(int64)
deliverFailed *optional*	integer(int64)
deliverTotal *optional*	integer(int64)
endDate *optional*	integer(int64)
endpointURL *optional*	string

Name	Schema
groupedTypes *optional*	< enum (HTTP, SAVE, EMAIL, SMS, APNS, GCM, PUSH, MOBILE_NOTIFICATION) > array
groupingId *optional*	string
message *optional*	string
metaData *optional*	string
name *optional*	string
owner *optional*	AccountShortResponse
parentId *optional*	integer(int64)
parentType *optional*	string
payload *optional*	string
processingTime *optional*	integer(int64)
scheduledDate *optional*	integer(int64)
scheduledNotificationId *optional*	integer(int64)
sendingTime *optional*	integer(int64)
startDate *optional*	integer(int64)
status *optional*	enum (NEW, ERROR, COMPLETE, PROCESSING, TEMPLATE)
targetId *optional*	integer(int64)
targetName *optional*	string
targetParams *optional*	string

Name	Schema
targetType *optional*	string
templateType *optional*	enum (REPORTING, ACHIEVEMENT_OWNER, ACHIEVEMENT_TRIGGER, GEOFENCE_OWNER, GEOFENCE_TARGET, GEOFENCE_TRIGGER, GEOFENCE_HTTP_CALL, SIGNUP)
triggerId *optional*	integer(int64)
type *optional*	enum (HTTP, SAVE, EMAIL, SMS, APNS, GCM, PUSH, MOBILE_NOTIFICATION)
updated *optional*	integer(int64)

7.295. TrilatAppSettings

Name	Schema
enableCustomRegionMatching *optional*	boolean
enableOutsideAdjustments *optional*	boolean
enableQuadrantDetection *optional*	boolean
maxSampleSize *optional*	integer(int32)
minRSSI *optional*	number(double)
minimumRollingAverageEnabled *optional*	boolean
processingType *optional*	enum (DEFAULT, FINGERPRINT, FINGERPRINT_V2)
rollingAverageEnabled *optional*	boolean
rollingAverageInterval *optional*	integer(int32)
rollingAverageMinimumLocationOccurences *optional*	integer(int32)

7.296. TrilatCacheRequest

Name	Schema
minimumSampleSize *optional*	integer(int32)
samples *optional*	< TrilatCacheSample > array
sourceTime *optional*	integer(int64)
udid *optional*	string

7.297. TrilatCacheSample

Name	Schema
alternateId *optional*	string
avatarId *optional*	string
avgRssi *optional*	number(double)
deviceId *optional*	string
deviceSignature *optional*	string
filteredRssi *optional*	number(double)
networkName *optional*	string
probeType *optional*	enum (WIFI_PROBE_REQUEST, IBEACON, BLE, ESTIMOTE, EDDYSTONE_UID, EDDYSTONE_URL, EDDYSTONE_TLM, EDDYSTONE_EID, UNKNOWN)
randomizedId *optional*	boolean
rssi *optional*	< integer(int32) > array

Name	Schema
time *optional*	integer(int64)
type *optional*	string

7.298. Trip

Name	Schema
active *optional*	boolean
created *optional*	string(date-time)
deactivated *optional*	boolean
driver *optional*	boolean
dropOffLocation *optional*	Location
dropOffTime *optional*	string(date-time)
id *optional*	integer(int64)
lastSeen *optional*	string(date-time)
matches *optional*	< Trip > array
notifications *optional*	string
originalId *optional*	integer(int64)
owner *optional*	Account
ownerConnection *optional*	Connection
pickupLocation *optional*	Location

Name	Schema
pickupTime *optional*	string(date-time)
preference *optional*	enum (DRIVE, FLEXIBLE, RIDE)
recurrence *optional*	Recurrence
route *optional*	Route
shipments *optional*	< Shipment > array
timeZone *optional*	string
updated *optional*	string(date-time)
valid *optional*	boolean

7.299. Tutorial

Name	Schema
active *optional*	boolean
alignment *optional*	enum (NONE, IMAGE_ABOVE, IMAGE_BELOW, IMAGE_LEFT, IMAGE_RIGHT, IMAGE_ONLY, TEXT_ONLY)
created *optional*	string(date-time)
deactivated *optional*	boolean
description *optional*	string
id *optional*	integer(int64)
image *optional*	Asset
name *optional*	string

Name	Schema
orderIndex *optional*	integer(int64)
tutorialObjectType *optional*	enum (GAME_OBJECT, GAME_LEVEL, PACK, GAME, MISSION, PROFILE, APPLICATION, TICKETS, ASSET, CUSTOM)
updated *optional*	string(date-time)
valid *optional*	boolean

7.300. TutorialResponse

Name	Schema
alignment *optional*	string
description *optional*	string
errorCode *optional*	string
id *optional*	integer(int64)
image *optional*	AssetShortResponse
message *optional*	string
name *optional*	string
objectType *optional*	string
orderIndex *optional*	integer(int64)
request *optional*	< NameStringValueResponse > array
returning *optional*	string

Name	Schema
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.301. TwiMLResponse

Name	Schema
attributes *optional*	< string, string > map
body *optional*	string
children *optional*	< Verb > array
tag *optional*	string

7.302. UserActivityResponse

Name	Schema
appDescription *optional*	string
appId *optional*	integer(int64)
appTitle *optional*	string
appUrl *optional*	string
created *optional*	string(date-time)
customId *optional*	integer(int64)

Name	Schema
customMessage *optional*	string
customText *optional*	string
customType *optional*	string
customValue *optional*	number(double)
errorCode *optional*	string
fields *optional*	string
id *optional*	integer(int64)
imageURL *optional*	string
latitude *optional*	number(double)
longitude *optional*	number(double)
message *optional*	string
profileImage *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
tag *optional*	string
userId *optional*	integer(int64)

Name	Schema
username *optional*	string
valid *optional*	boolean
version *optional*	number(double)

7.303. UserLocationListResponse

Name	Schema
count *optional*	integer(int32)
errorCode *optional*	string
items *optional*	< UserLocationResponse > array
latitude *optional*	number(double)
longitude *optional*	number(double)
message *optional*	string
postalCode *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
userLocationType *optional*	enum (POSTALCODE, EXACT)
valid *optional*	boolean

Name	Schema
version *optional*	number(double)

7.304. UserLocationResponse

Name	Schema
aboutUs *optional*	string
accountId *optional*	integer(int64)
accountType *optional*	string
age *optional*	string
categories *optional*	< CategoryResponse > array
connectionToMe *optional*	ConnectionResponse
contact *optional*	ContactResponse
currentPostalCode *optional*	PostalCodeResponse
display *optional*	string
distanceFromMe *optional*	string
errorCode *optional*	string
flagCount *optional*	integer(int64)
gameExperience *optional*	string
gender *optional*	enum (MALE, FEMALE, ANY)
latitude *optional*	number(double)

Name	Schema
locale *optional*	string
locationDisplay *optional*	string
locationLastUpdated *optional*	integer(int64)
longitude *optional*	number(double)
matchIndex *optional*	integer(int32)
message *optional*	string
personalProfile *optional*	PersonalProfileResponse
premiumExpiration *optional*	integer(int64)
profileAvatar *optional*	AssetShortResponse
profileImage *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
summaryTitle *optional*	string
tags *optional*	string
userSettings *optional*	UserSettingsResponse
username *optional*	string

Name	Schema
valid *optional*	boolean
version *optional*	number(double)

7.305. UserLocationSearchResponse

Name	Schema
count *optional*	integer(int32)
errorCode *optional*	string
hasMoreResults *optional*	boolean
message *optional*	string
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
userLocations *optional*	< UserLocationListResponse > array
valid *optional*	boolean
version *optional*	number(double)

7.306. UserPermissions

Name	Schema
active *optional*	boolean

Name	Schema
admin *optional*	boolean
blocked *optional*	boolean
connection *optional*	Connection
connectionGroup *optional*	ConnectionGroup
created *optional*	string(date-time)
deactivated *optional*	boolean
exclude *optional*	boolean
friendGroup *optional*	boolean
id *optional*	integer(int64)
owner *optional*	Account
pending *optional*	boolean
permissionableId *optional*	integer(int64)
permissionableType *optional*	string
permissions *optional*	Permissions
updated *optional*	string(date-time)
valid *optional*	boolean

7.307. UserPermissionsListResponse

Name	Schema
action *optional*	enum (DATA, SAVE, DELETE, GET)
publicPermissions *optional*	PermissionResponse
userPermissions *optional*	< UserPermissionsResponse > array
visibility *optional*	enum (PUBLIC, PRIVATE, FRIENDS)

7.308. UserPermissionsResponse

Name	Schema
admin *optional*	boolean
blocked *optional*	boolean
connection *optional*	ConnectionResponse
connectionGroup *optional*	ConnectionGroupResponse
errorCode *optional*	string
exclude *optional*	boolean
friendGroup *optional*	boolean
message *optional*	string
pending *optional*	boolean
permissionableId *optional*	integer(int64)
permissionableType *optional*	string
permissions *optional*	PermissionResponse

Name	Schema
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
valid *optional*	boolean
version *optional*	number(double)

7.309. UserSettingsResponse

Name	Schema
favoriteVisibility *optional*	enum (PUBLIC, PRIVATE, FRIENDS)
notifications *optional*	NotificationSettingsResponse
publicShowFriendInfo *optional*	boolean
publicShowGameInfo *optional*	boolean
publicShowProfileInfo *optional*	boolean
showAsZipcode *optional*	boolean
showExactLocation *optional*	boolean
showOthersExactLocation *optional*	boolean
suggestionCount *optional*	integer(int32)
suggestionMethod *optional*	string

Name	Schema
suggestionTimeFrame *optional*	integer(int32)

7.310. Vehicle

Name	Schema
active *optional*	boolean
belongsTo *optional*	Account
contentName *optional*	string
created *optional*	string(date-time)
customType *optional*	string
deactivated *optional*	boolean
description *optional*	string
externalId *optional*	string
hub *optional*	ServiceHub
id *optional*	integer(int64)
inUse *optional*	boolean
licensePlateNumber *optional*	string
markerIcon *optional*	Asset
name *optional*	string
noteCount *optional*	integer(int64)

Name	Schema
notes *optional*	< Note > array
picture *optional*	Asset
programs *optional*	< Program > array
seatQuantity *optional*	integer(int64)
trackingColor *optional*	string
updated *optional*	string(date-time)
valid *optional*	boolean
vehicleType *optional*	VehicleType
vin *optional*	string
zone *optional*	Region

7.311. VehicleCargoSetting

Name	Schema
cargoType *optional*	CargoType
deactivated *optional*	boolean
maxUnit *optional*	integer(int32)
valid *optional*	boolean

7.312. VehicleResponse

Name	Schema
capacity *optional*	integer(int64)
current *optional*	GeoResponse
depot *optional*	GeoResponse
depotAsStop *optional*	StopResponse
endWindow *optional*	integer(int64)
id *optional*	string
name *optional*	string
startWindow *optional*	integer(int64)

7.313. VehicleType

Name	Schema
active *optional*	boolean
business *optional*	Retailer
created *optional*	string(date-time)
deactivated *optional*	boolean
depth *optional*	number(double)
height *optional*	number(double)
hub *optional*	ServiceHub
id *optional*	integer(int64)

Name	Schema
maxWeight *optional*	number(double)
name *optional*	string
updated *optional*	string(date-time)
valid *optional*	boolean
vehicleCargoSettings *optional*	< VehicleCargoSetting > array
volume *optional*	number(double)
width *optional*	number(double)

7.314. Verb

Name	Schema
attributes *optional*	< string, string > map
body *optional*	string
children *optional*	< Verb > array
tag *optional*	string

7.315. WeatherForecastResponse

Name	Schema
forecastDate *optional*	integer(int64)
id *optional*	integer(int64)
precipMM *optional*	number(double)

Name	Schema
tempMaxC *optional*	integer(int32)
tempMaxF *optional*	integer(int32)
tempMinC *optional*	integer(int32)
tempMinF *optional*	integer(int32)
weatherCode *optional*	integer(int32)
weatherDesc *optional*	string
weatherIconUrl *optional*	string
winddir16Point *optional*	string
winddirDegree *optional*	integer(int32)
winddirection *optional*	string
windspeedKmph *optional*	integer(int32)
windspeedMiles *optional*	integer(int32)

7.316. WeatherResponse

Name	Schema
cloudcover *optional*	integer(int32)
forecasts *optional*	< WeatherForecastResponse > array
humidity *optional*	integer(int32)
id *optional*	integer(int64)

Name	Schema
observationTime *optional*	integer(int64)
precipMM *optional*	number(double)
pressure *optional*	integer(int32)
temp_C *optional*	integer(int32)
temp_F *optional*	integer(int32)
visibility *optional*	integer(int32)
weatherCode *optional*	integer(int32)
weatherDesc *optional*	string
weatherIconUrl *optional*	string
winddir16Point *optional*	string
winddirDegree *optional*	integer(int32)
windspeedKmph *optional*	integer(int32)
windspeedMiles *optional*	integer(int32)

7.317. WordzWordResponse

Name	Schema
active *optional*	boolean
allocateTickets *optional*	boolean
authorOverride *optional*	string

Name	Schema
createdDate *optional*	integer(int64)
definition *optional*	string
endDate *optional*	integer(int64)
icon *optional*	AssetShortResponse
id *optional*	integer(int64)
image *optional*	AssetShortResponse
metaData *optional*	string
points *optional*	integer(int64)
scores *optional*	ScoreListResponse
startDate *optional*	integer(int64)
ticketCount *optional*	integer(int64)
ticketType *optional*	string
updatedDate *optional*	integer(int64)
word *optional*	string

7.318. WrappedResponse

Name	Schema
errorCode *optional*	string
item *optional*	object

Name	Schema
message *optional*	string
messages *optional*	MessageListResponse
request *optional*	< NameStringValueResponse > array
returning *optional*	string
serializeNulls *optional*	boolean
startTimeLog *optional*	integer(int64)
type *optional*	string
valid *optional*	boolean
version *optional*	number(double)

7.319. YayOrNay

Name	Schema
active *optional*	boolean
created *optional*	string(date-time)
deactivated *optional*	boolean
id *optional*	integer(int64)
likableObjectId *optional*	integer(int64)
likableObjectType *optional*	string
owner *optional*	Account

Name	Schema
updated *optional*	string(date-time)
valid *optional*	boolean
yayOrNay *optional*	boolean