TOURNAMENT INTEGRATION API DOCUMENTATION

API Documentation for Tournament Integration

TOURNAMENT INTEGRATION API DOCUMENTATION

RAZER, INC. & GAMINGGRIDS, INC.

Documentation Updated on 10/22/2014 10:11:30 PM

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

INVOCATION WORKFLOW OVERVIEW

Workflow Pathing

The below chart shows the standardized path of method invocations based on event triggers under a standard first-person-shooter integration.

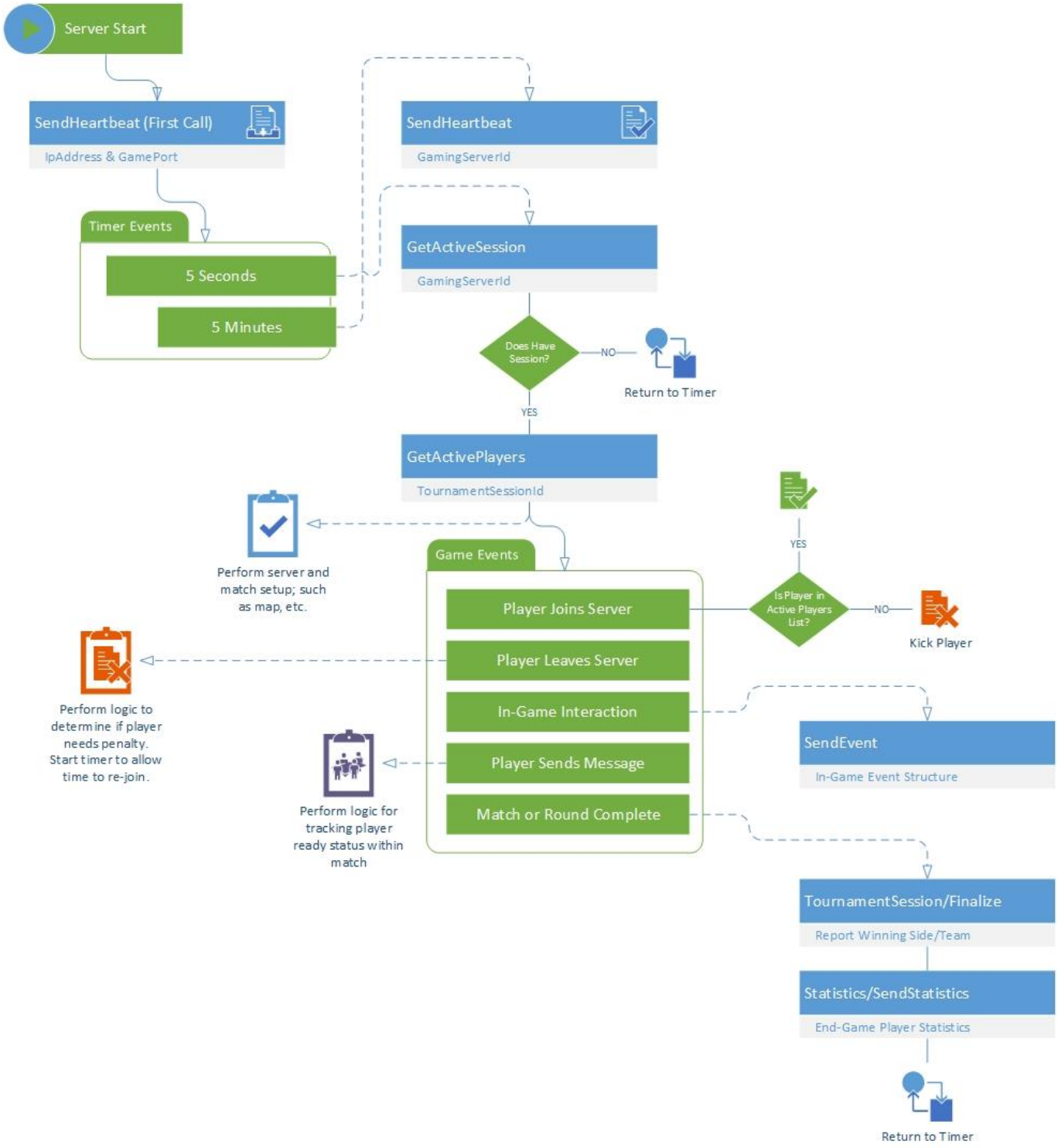
EVENT TRIGGERS:

Server or Integration Startup
Timed Events
Player Join
Player Leave
In-Game Interaction
Player Message Sent
Round/Game Ending

TRIGGER PATHING EXAMPLE:

See following page for larger diagram.

Figure 1a


Server Start
SendHeartbeat (First Call)
IpAddress & GamePort
SendHeartbeat
GamingServerId
Timer Events
5 Seconds
5 Minutes
GetActiveSession
GamingServerId
Does Have Session?
NO
Return to Timer
YES
GetActivePlayers
TournamentSessionId
Perform server and match setup; such as map, etc.
Game Events
Player Joins Server
Player Leaves Server
In-Game Interaction
Player Sends Message
Match or Round Complete
YES
Is Player in Active Players List?
NO
Kick Player
Perform logic to determine if player needs penalty. Start timer to allow time to re-join.
Perform logic for tracking player ready status within match
SendEvent
In-Game Event Structure
TournamentSession/Finalize
Report Winning Side/Team
Statistics/SendStatistics
End-Game Player Statistics
Return to Timer

Workflow Narrative

The below workflow narrative uses Battlefield 4 as an example integration for form a standardized flow of method invocations within the GamingGrids Game Server API. The methods can be manipulated and articulated as needed for form around any type of game; specialized events, items, and structures need be created per game by contacting GamingGrids.

SERVER START/MOD START/PROCON CONNECT TO SERVER (ENTRY)

- Send Server Heartbeat using API/Game/Server/SendHeartbeat and cache GamingServerId
- Set a timer so that every 5 minutes you send a heartbeat so we know it's still online
    - You can omit the IP and port on future calls, and just send the GamingServerId. Don't cache the GamingServerId in a file; just cache it in the context of the application; once servers restarts it needs to send IP/Port again to get ID.
- Set a timer so that every 5 seconds you check for an active session based on this GamingServerId using API/TournamentSession/GetActiveSession/{GamingServerId}
    - If a session is found, certain information regarding the session will be given to help configure certain functions and gameplay of the server
        - Set the map of the server for this match based on the name of the map of the session (stored as MapFileName)
        - Use the "ParticipantA.TeamSideId" to determine what TeamA is within Battlefield…
            - 1 = US Forces
            - 2 = Opposition
            - 3 = Random Sides
        - Set Team names based on ParticpantA.Nickname and ParticipantB.Nickname
        - Invoke the method API/TournamentSession/GetActivePlayers/{TournamentSessionId} to get players in this session; store locally so you can refer to the GamePlayerIdentifier when players join.

PLAYER JOINS SERVER

- If there is an active session for this server, check the Punkbuster.GUID of the player on ProCon vs. the GamePlayerIdentifier of players in the GamingGrids array of allowed players for the session. If it's valid, allow them to join – if not, kick then with a message saying they aren't allowed to play in this match.
- If there is not an active session for this server, kick the player and say there is no match for this server.

- On the method invoked previously to get a list of active players, there is a flag for "IsOnTeamA" that denotes whether this player is on TeamA -- if they are, automatically put them on that team, and prevent them from changing teams.
- You need to keep track of a player's ready status, and keep any scores from counting on the server until they all are ready. You need to fire events on messages the users' type and track for ".ready" and ".notready" and change the user accordingly. The logic is that, once you see all players ".ready" (based on the number of players in the session, not just currently in the server – since you need to make sure all players join before going live) then you can reload the map and let scores count towards the match results.
- If the player is one who has left the server prior, but has now rejoined (see Player Leave event below), cancel the 3-minute timer, and restore their Kills, Death, Score, etc to the scoreboard in-game (if possible).

EVENT: KILL

- Send the realtime event to the API with player data for who killed, who died, what weapon id it was, where it was hit at, and if we can get info on the XYZ for each player; send using API/Game/Event/SendEvent

EVENT: BULLET HITS A PLAYER (DAMAGE)

- Send the realtime event to the API with player data for who shot, who was damaged, what weapon id it was, where it was hit at, what the damage was for and if we can get info on the XYZ for each player; send using API/Game/Event/SendEvent

EVENT: FLAG CAPTURE

- Send the realtime event to the API with player data for who captured the point and what team they were on; send using API/Game/Event/SendEvent

EVENT: TICKET TICK-DOWN DECREASE

- Send the realtime event to the API with what the current ticket counts are for TeamA and TeamB; send using API/Game/Event/SendEvent

PLAYER SENDS CHAT MESSAGE

- .ready – set user to ready to start
  - Perform logic to check whether all players who are supposed to be in the session are ready (based on array in GetActivePlayers); if so, reload the map and start tracking the stats and scores.
- .unready – user is not ready
- .pause – pause the match if possible

- .unpause – resume gameplay

PLAYER LEAVES SERVER

- When a player leaves the server, send a message to chat saying "{0} has disconnected. Waiting 3 minutes before removing from session."
- Capture the date/time the player left, and after 3 minutes if they don't re-join, invoke API/TournamentSession/Players/RemovePlayer to remove them from the session.

MAP ROUND IS COMPLETE

- This happens when a team loses all of their tickets.
- The plugin will then continue the map or reload map, and rest tickets to default per team and let them play again; switching sides (so if TeamA was Side1, after the round switch TeamA to Side2)
- There is a maximum of 3 rounds (full ticket bleed scenario); logic dictates that the first team to win 3 rounds will win the match. A team can win as early as the 2nd round.

IN-GAME MATCH IS COMPLETE (TEAM WIN)

- This is determined by the above map round completion; once a team wins two rounds, they win.
- Finalize the match by invoking the API/TournamentSession/Finalize method in the API and sending TeamA score and Team B score (which is NOT the tickets, it will be the rounds won; 2-0).
- For each player in the server, invoke API/Players/Statistics/SendStatistics to send the overview statistics for the player.
- Reset the TournamentSessionId in the server so it's ready to start checking for the next match
- Kick all players from the server with a message saying "The match is complete. Visit the website to view statistics".

SEND EVENT DATA

Send Single Event

API/Game/Event/SendEvent

ACTION:

POST

HOST HEADER:

gg-client-api-key – value is client api key

POST BODY DATA PARAMETERS:

Name	Type	Valid Values	Can Null	Can Omit
GameTitleId	Int32			
GamingServerId	Int32			
TournamentSessionId	Int32			
RoundNumber	Int32			
EventTypeId	Int32			
EventData	Object	See Event Definitions		

DATA STRUCTURE:

```
{
GameTitleId: ,
GamingServerId: ,
TournamentSessionId: ,
RoundNumber: ,
EventTypeId: ,
EventData: {Object}
}
```

RESPONSE:

```
Success: Boolean, true if a valid response
Message: Error message if Success is false
```

Send Multiple Events

API/Game/Events/SendEvents

ACTION:

POST

HOST HEADER:

gg-client-api-key – value is client api key

POST BODY DATA PARAMETERS:

Name	Type	Valid Values	Can Null	Can Omit
GameTitleId	Int32			
GamingServerId	Int32			
TournamentSessionId	Int32			
RoundNumber	Int32			
Events	Array	See Event Definitions		

DATA:

```
{
GameTitleId: ,
GamingServerId: ,
TournamentSessionId: ,
RoundNumber: ,
Events: [ { EventTypeId:, Event: {}}, …]
}
```

RESPONSE:

Success: Boolean, true if a valid response

Message: Error message if Success is false

SEND EVENT DEFINITIONS

Data Definitions

- GameTitleId
  - is the integer value from the database that identifies which game title is being referenced
- GamingServerId
  - is the integer value from the database that represents the server being monitored
- TournamentSessionId
  - is the integer value from the database that corresponds to the tournament session being played
- RoundNumber
  - is the integer value of the round in the match that this event took place in
- EventTypeId
  - is the type of event. Valid values are:

Game	EventTypeId	Proper Name / Detail
Counter-Strike: Global Offensive	1	RoundSnapshot
	2	Kill
	3	Damage
	4	Assist
	5	Suicide
	6	Spawn
	7	BombExplode
	8	BombPlant
	9	BombDefuse
	10	TeamScore
	11	ItemDroppedOrPickedUp
Battlefield 4	1	Kill
	2	Damage
	3	PointCapture
	4	TicketBleed

SEND EVENT VALUE DEFINITIONS

Event Breakdown by Game

Below is a listing of specialized events per game title; color coding on the left side relates directly to the game title listed in the table below.

Refer to the table above in [Send Event Definitions] for EventTypeId's

GAME LISTING COLORIZATION

Color	Game Name	GameTitleId
Purple	Counter-Strike: Global Offensive	10
Green	Battlefield 4	3

COUNTER STRIKE: GO

Type: Assist

Valid Fields:

SecondsInToGame:

AttackerPlayerId:

VictimPlayerId:

Type: Bomb Detonate

Valid Fields:

SecondsInToGame:

PlayerId:

Type: Bomb Explode

Valid Fields:

SecondsInToGame:

PlayerId:

Type: Bomb Plant

Valid Fields:

SecondsInToGame:

PlayerId:

Type: Damage

Valid Fields:

SecondsInToGame:

AttackerPlayerId:

AttackerPositionX:

AttackerPositionY:

AttackerPositionZ:

VictimPlayerId:

VictimPositionX:

VictimPositionY:

VictimPositionZ:

WeaponItemCode:

HitBox:

Damage:

Type: Kill

Valid Fields:

SecondsInToGame:

AttackerPlayerId:

AttackerPositionX:

AttackerPositionY:

AttackerPositionZ:

VictimPlayerId:

VictimPositionX:

VictimPositionY:

VictimPositionZ:

WeaponItemCode:

TeamKill:

HeadShot:

Damage:

KillFactor:

Type: Round Snapshot

Valid Fields:

SecondsInToGame:

Snapshot:

Type: Spawn

Valid Fields:

SecondsInToGame:

PlayerId:

Money:

PrimaryWeaponItemCode:

SecondaryWeaponItemCode:

Grenade1ItemCode:

Grenade2ItemCode:

Grenade3ItemCode:

Grenade4ItemCode:

Helmet:

Armor:

Type: Suicide

Valid Fields:

SecondsInToGame:

PlayerId:

Type: Team Score

Valid Fields:

SecondsInToGame:

TeamId:

Reason:

Type: Item Dropped Or Picked Up

Valid Fields:

SecondsInToGame:

PlayerId:

ItemCode: Id of item

ItemDropped: true/false

PlayerDied: true/false

ItemPickedUp: true/false

BATTLEFIELD 4

Type: Kill

Valid Fields:

SecondsInToGame:
AttackerPlayerId:
AttackerPositionX:
AttackerPositionY:
AttackerPositionZ:
VictimPlayerId:
VictimPositionX:
VictimPositionY:
VictimPositionZ:
WeaponItemCode:
TeamKill:
HeadShot:
Damage:
KillFactor:

Type: Damage

Valid Fields:

SecondsInToGame:
AttackerPlayerId:
AttackerPositionX:
AttackerPositionY:
AttackerPositionZ:
VictimPlayerId:
VictimPositionX:
VictimPositionY:
VictimPositionZ:
WeaponItemCode:
HitBox:

Damage:

Type: PointCapture

Valid Fields:

SecondsInToGame:

AttackerPlayerId:

IsTeamA:

Type: TicketBleed

Valid Fields:

SecondsInToGame:

TeamATickets:

TeamBTickets:

PointsHeldByTeamA:

PointsHeldByTeamB:

PointsNeutral:

GAME TITLE ITEM LOOKUP

Counter-Strike: Global Offensive

ItemCode	Item Name
1	Unknown
2	P228
3	Glock
4	Scout
5	Explosive Grenade
6	XM1014
7	C4
8	MAC10
9	Aug
10	Smoke
11	Elite
12	Five Seven
13	UMP45
14	SG550
15	Galil
16	Famas
17	Usp
18	Awp
19	MP5Navy
20	M249
21	M3
22	M4A1
23	TMP
24	G3SG1
25	Flashbang
26	Deagle
27	AK47
28	Knife

29	P90
30	Shield
31	Kevlar
32	Assaultsuit
33	Nightvision
34	Galilar
35	Bizon
36	MAG7
37	Negev
38	Sawed Off
39	TEC9
40	Taser
41	HKP2000
42	MP7
43	MP9
44	Nova
45	P250
46	SCAR17
47	SCAR20
48	SG556
49	SSG08
50	Molotov
51	Decoy
52	Incendiary Grenade
53	Defuser

Battlefield 4

ItemCode	Item Name
U_AEK971_M320_HE	AEK971 M320 HE
U_AEK971_M320_FLASH	AEK971 M320 Flashbang
U_AEK971_M320_LVG	AEK971 M320 LVG
U_AEK971_M320_SHG	AEK971 M320 Shotgun
U_AEK971_M320_SMK	AEK971 M320 Smoke
U_AEK971_M320_3GL	AEK971 M320 3GL
U_AK12_M320_HE	AK12 M320 HE
U_AK12_M320_FLASH	AK12 M320 Flashbang
U_AK12_M320_LVG	AK12 M320 LVG
U_AK12_M320_SHG	AK12 M320 Shotgun
U_AK12_M320_SMK	AK12 M320 Smoke
U_AK12_M320_3GL	AK12 M320 3GL
U_CZ805_M320_HE	CZ805 M320 HE
U_CZ805_M320_FLASH	CZ805 M320 Flashbang
U_CZ805_M320_LVG	CZ805 M320 LVG
U_CZ805_M320_SHG	CZ805 M320 Shotgun
U_CZ805_M320_SMK	CZ805 M320 Smoke
U_CZ805_M320_3GL	CZ805 M320 3GL
U_M16A4_M320_HE	M16A4 M320 HE
U_M16A4_M320_FLASH	M16A4 M320 Flashbang
U_M16A4_M320_LVG	M16A4 M320 LVG
U_M16A4_M320_SHG	M16A4 M320 Shotgun
U_M16A4_M320_SMK	M16A4 M320 Smoke
U_M16A4_M320_3GL	M16A4 M320 3GL
U_M416_M320_HE	M416 M320 HE
U_M416_M320_FLASH	M416 M320 Flashbang
U_M416_M320_LVG	M416 M320 LVG
U_M416_M320_SHG	M416 M320 Shotgun
U_M416_M320_SMK	M416 M320 Smoke

U_M416_M320_3GL	M416 M320 3GL
U_QBZ951_M320_HE	QBZ951 M320 HE
U_QBZ951_M320_FLASH	QBZ951 M320 Flashbang
U_QBZ951_M320_LVG	QBZ951 M320 LVG
U_QBZ951_M320_SHG	QBZ951 M320 Shotgun
U_QBZ951_M320_SMK	QBZ951 M320 Smoke
U_QBZ951_M320_3GL	QBZ951 M320 3GL
U_SAR21_M320_HE	SAR21 M320 HE
U_SAR21_M320_FLASH	SAR21 M320 Flashbang
U_SAR21_M320_LVG	SAR21 M320 LVG
U_SAR21_M320_SHG	SAR21 M320 Shotgun
U_SAR21_M320_SMK	SAR21 M320 Smoke
U_SAR21_M320_3GL	SAR21 M320 3GL
U_SCAR-H_M320_HE	SCAR-H M320 HE
U_SCAR-H_M320_FLASH	SCAR-H M320 Flashbang
U_SCAR-H_M320_LVG	SCAR-H M320 LVG
U_SCAR-H_M320_SHG	SCAR-H M320 Shotgun
U_SCAR-H_M320_SMK	SCAR-H M320 Smoke
U_SCAR-H_M320_3GL	SCAR-H M320 3GL
U_SteyrAug_M320_HE	SteyrAug M320 HE
U_SteyrAug_M320_FLASH	SteyrAug M320 Flashbang
U_SteyrAug_M320_LVG	SteyrAug M320 LVG
U_SteyrAug_M320_SHG	SteyrAug M320 Shotgun
U_SteyrAug_M320_SMK	SteyrAug M320 Smoke
U_SteyrAug_M320_3GL	SteyrAug M320 3GL
U_L85a2_M320_HE_V2	L85a2 M320 HE
U_L85a2_M320_FLASH_V2	L85a2 M320 Flashbang
U_L85a2_M320_LVG_V2	L85a2 M320 LVG
U_L85a2_M320_SHG_V2	L85a2 M320 Shotgun
U_L85a2_M320_SMK_V2	L85a2 M320 Smoke
U_L85a2_M320_3GL_V2	L85a2 M320 3GL
U_AR160_M320_FLASH	AR160 M320 Flashbang

U_AR160_M320_HE	AR160 M320 HE
U_AR160_M320_LVG	AR160 M320 LVG
U_AR160_M320_SHG	AR160 M320 Shotgun
U_AR160_M320_SMK	AR160 M320 Smoke
U_AR160_M320_3GL	AR160 M320 3GL
U_M416_M26_Buck	M416 M26 Buckshot
U_M416_M26_Flechette	M416 M26 Flechette
U_M416_M26_Frag	M416 M26 Fragment
U_M416_M26_Slug	M416 M26 Slug
U_SCAR-H_M26_Buck	SCAR-H M26 Buckshot
U_SCAR-H_M26_Flechette	SCAR-H M26 Flechette
U_SCAR-H_M26_Frag	SCAR-H M26 Fragment
U_SCAR-H_M26_Slug	SCAR-H M26 Slug
U_CZ805_M26_Buck	CZ805 M26 Buckshot
U_CZ805_M26_Flechette	CZ805 M26 Flechette
U_CZ805_M26_Frag	CZ805 M26 Fragment
U_CZ805_M26_Slug	CZ805 M26 Slug
U_M16A4_M26_Buck	M16A4 M26 Buckshot
U_M16A4_M26_Flechette	M16A4 M26 Flechette
U_M16A4_M26_Frag	M16A4 M26 Fragment
U_M16A4_M26_Slug	M16A4 M26 Slug
U_AR160_M26_Buck	AR160 M26 Buckshot
U_AR160_M26_Flechette	AR160 M26 Flechette
U_AR160_M26_Frag	AR160 M26 Fragment
U_AR160_M26_Slug	AR160 M26 Slug
U_AEK971	aek-971
U_AK12	ak-12
U_CZ805	cz-805
U_FAMAS	famas
U_GalilACE23	ace-23
U_M16A4	m16a4
U_M416	m416

U_QBZ951	qbz-95-1
U_SAR21	sar-21
U_SCAR-H	scar-h
U_SteyrAug	aug-a3
U_L85A2	L85a2
U_F2000	f2000
U_AR160	ar160
U_A91	a-91
U_ACR	acw-r
U_AK5C	ak-5c
U_AKU12	aku-12
U_G36C	g36c
U_GalilACE	ace-21-cqb
U_GalilACE52	ace-52-cqb
U_M4A1	m4
U_SG553LB	sg553
U_Type95B	type-95b-1
U_MTAR21	mtar-21
U_GalilACE53	ace-53-sv
U_M39EBR	m39-emr
U_MK11	mk11-mod-0
U_QBU88	qbu-88
U_RFB	rfb
U_SCAR-HSV	scar-h-sv
U_SKS	sks
U_SVD12	svd-12
U_C4	c4-explosive
U_C4_Support	c4-explosive
U_Claymore	m18-claymore
U_Claymore_Recon	m18-claymore
U_M15	m15-at-mine
U_SLAM	m2-slam

U_MGL	m32-mgl
M224	m224-mortar
U_XM25	xm25-airburst
U_XM25_Flechette	xm25-dart
U_XM25_Smoke	xm25-smoke
U_AAMine	aa-mine
U_Handflare	hand-flare
U_Flashbang	m84-flashbang
U_Grenade_RGO	rgo-impact
U_M18	m18-smoke
U_M34	m34-incendiary
U_M67	m67-frag
U_V40	v40-mini
U_M93R	93r
U_CZ75	cz-75
U_FN57	fn57
U_Glock18	g18
U_HK45C	compact-45
U_M1911	m1911
U_MP412Rex	m412-rex
U_M9	m9
U_MP443	mp443
U_P226	p226
U_QSZ92	qsz-92
U_Taurus44	44-magnum
U_SW40	sw40
Melee	bayonet
Melee	acb-90
Melee	bowie
Melee	shank
Melee	boot
Melee	scout

Melee	seal
Melee	carbon-fiber
Melee	trench
Melee	machete
Melee	survival
Melee	improvised
U_LSAT	lsat
U_M240	m240b
U_M249	m249
U_MG4	mg4
U_Pecheneg	pkp-pecheneg
U_QBB95	qbb-95-1
U_RPK12	rpk-12
U_Type88	type-88-lmg
U_Ultimax	u-100-mk5
U_rpk-74	rpk-74m
U_M60E4	M60-E4
U_AWS	aws
U_CBJ-MS	cbj-ms
U_JS2	js2
U_MX4	mx4
U_P90	p90
U_MagpulPDR	pdw-r
U_PP2000	pp-2000
U_Scorpion	cz-3a1
U_UMP45	ump-45
U_UMP9	ump-9
U_mp7	mp7
U_ASVAL	AS VAL
U_SR2	sr-2
U_FGM148	fgm-148-javelin
U_FIM92	fim-92-stinger

U_Sa18IGLA	sa-18-igla
U_AT4	m136-cs
U_NLAW	mbt-law
U_RPG7	rpg-7v2
U_SMAW	mk153-smaw
U_SRAW	fgm-172-sraw
U_Starstreak	hvm-ii
U_M26Mass_Flechette	m26-dart
U_M26Mass_Frag	m26-frag
U_M26Mass	m26-mass
U_M26Mass_Slug	m26-slug
U_USAS-12_Nightvision	usas-12-flir
U_870	870-mcs
U_DBV12	dbv-12
U_HAWK	hawk-12g
U_M1014	m1014
U_QBS09	qbs-09
U_SAIGA_20K	saiga-12k
U_SerbuShorty	shorty-12g
U_SPAS12	spas-12
U_USAS-12	usas-12
U_UTAS	uts-15
U_DAO12	dao-12
U_AMR2	amr-2
U_AMR2_CQB	amr-2-cqb
U_AMR2_MED	amr-2-mid
U_CS-LR4	cs-lr4
U_FY-JS	fy-js
U_JNG90	jng-90
U_M200	srr-61
U_M40A5	m40a5
U_M82A3	m82a3

U_M82A3_CQB	m82a3-cqb
U_M82A3_MED	m82a3-mid
U_M98B	m98b
U_Scout	scout-elite
U_SRS	338-recon
U_SV98	sv-98
U_L96A1	l96a1
U_GOL	gol-magnum
U_SR338	sr338
U_Defib	defibrillator
U_Repairtool	repair-tool
DamageArea	
Death	
RoadKill	
SoldierCollision	
Suicide	Suicide
U_Medkit	
U_PortableMedicpack	
U_Tomahawk	
U_M320_HE	m320-he
U_M320_FLASH	m320-fb
U_M320_LVG	m320-lvg
U_M320_SHG	m320-dart
U_M320_SMK	m320-smk
U_M320_3GL	m320-3gl
U_Bulldog	
U_Unica6	
U_DesertEagle	
U_CS5	cs5
U_BallisticShield	
U_MPX	

PLAYER ENTITY VALIDATION

Validate & Verify Player Entity

API/Player/Game/ConfirmPlayer

ACTION:

POST

HOST HEADER:

`gg-client-api-key – value is client api key`

POST BODY DATA PARAMETERS:

Name	Type	Valid Values	Can Null	Can Omit
EntityPlayerId	Int32		Yes	Yes
EntityPlayerEmailAddress	String		Yes	Yes
EntityPlayerName	String		Yes	Yes
GameTitleId	Int32			
PlayerIdentifier	String	GUID of Player in-game		

SPECIAL NOTES:

One of the entity identifiers; EntityPlayerId, EntityPlayerEmailAddress, or EntityPlayerName must be specified to locate the player being validated.

RESPONSE:

```
Success: Boolean, true if a valid response
Message: Error message if Success is false
```

GAME SERVER MANAGEMENT

Server Endpoint Configuration

API/Game/Server/EndpointConfiguration

ACTION:

GET

HOST HEADER:

`gg-client-api-key – value is client api key`

POST BODY DATA PARAMETERS:

Name	Type	Valid Values	Can Null	Can Omit
None (GET)				

SPECIAL NOTE:

If there is no data or the key is expired, the "success" will return "false" with an error message. If the key is valid and a record is found, it will be in the response.

RESPONSE:

```
Success: Boolean, true if a valid response

Message: Error message if Success is false

Response:

        String DeveloperName

        Guid ApiKey

        Bool DoesExpire

        DateTime ExpirationDateTime

        String ApiEndPoint

        String ClientEndPoint

        String FtpHost

        String FtpPath

        String FtpUserName

        String FtpPassword

        Bool SslAvailable
```

Server Registration/Heartbeat

API/Game/Server/SendHeartbeat

ACTION:

POST

HOST HEADER:

gg-client-api-key – value is client api key

POST BODY DATA PARAMETERS:

Name	Type	Valid Values	Can Null	Can Omit
GamingServerId	Int32		Yes	Yes
IpAddress	String			
GamePort	Int32			
GameTitleId	Int32			
ServerTypeCode	String	GGS = GamingGrids Server MPS = Multiplay Dynamic EXS = External Server		
GamingModeCode	String	NA = N/A ONP = OpenPlay MKG = Matchmaking CHG = Challenge		
HostingProviderCode	String	GG = GamingGrids RZR = RAZER		

SPECIAL NOTE:

Must pass parameter

RESPONSE:

Success: Boolean, true if a valid response

Message: Error message if Success is false

Response: Long int that represents the server id

Get Active Tournament Session

API/TournamentSession/GetActiveSession/{GamingServerId}

ACTION:

GET

HOST HEADER:

gg-client-api-key – value is client api key

POST BODY DATA PARAMETERS:

Name	Type	Valid Values	Can Null	Can Omit
None (GET)				

RESPONSE:

Success: Boolean, true if a valid response

Message: Error message if Success is false

Response:

```
long TournamentSessionId
long TournamentId
long TournamentMatchId
Int32 GamingServerId
long EntityPlayerControllerId
DateTime SessionLiveDateTime
bool MatchIsLive
DateTime MatchLiveDateTime
bool MatchIsComplete
DateTime MatchCompleteDateTime
string PlaybackVideoUrl
Int32 MinimumNumberOfEntities
Int32 MaximumNumberOfEntities
Int32 TeamSize
Int32 GameTitleId
string SteamApplicationId
string GameName
bool RequiresManualVoting
```

```
Int32 TournamentTypeId
string TournamentTypeName
bool IsTeamTournament
TournamentSessionParticipantModel ParticipantA
        long EntityId
        Int32 TeamSideId
        String TeamSideName
        String Nickname
        Int32 CountryCode
TournamentSessionParticipantModel ParticipantB
        long EntityId
        Int32 TeamSideId
        String TeamSideName
        String Nickname
        Int32 CountryCode
bool GridKillEnabled
bool GridKillTeamShareEnable
Int32 GridKillStakesId
string TournamentGridKillStakeName
decimal AmountPerKill
decimal AmountPerAssist
decimal AmountPerDeath
Int32 MapNameId
string MapFileName
string MapProperName
bool ServerSendPlayerSummaryStatistics
```

TOURNAMENTTYPE DATA LOOKUP

TournamentTypeID	TournamentTypeName
1	OpenPlay
2	Round Robin
3	Bracket
4	PvP Challenge
5	TvT Challenge

6	Pickup Game	

TOURNAMENT SESSION (LIVE PLAY)

Get Associated Players for SessionId

API/TournamentSession/GetActivePlayers/{TournamentSessionId}

ACTION:

GET

HOST HEADER:

gg-client-api-key – value is client api key

POST BODY DATA PARAMETERS:

Name	Type	Valid Values	Can Null	Can Omit
TournamentSessionId	Int32			

RESPONSE:

Success: Boolean, true if a valid response

Message: Error message if Success is false

Response: Array of players containing the following fields:

- TournamentSessionId,
- EntityPlayerId,
- Nickname,
- IsReady,
- IsOnTeamA,
- GamePlayerIdentifier,
- SteamId32,
- SteamId64

Remove Player from Matchmaking Session

API/TournamentSession/Players/RemovePlayer

ACTION:

POST

HOST HEADER:

gg-client-api-key – value is client api key

POST BODY DATA PARAMETERS:

Name	Type	Valid Values	Can Null	Can Omit
TournamentSessionId	Int32			
EntityPlayerId	Int64			
HasPenalty	Bool	true, false		

DATA:

```
{
TournamentSessionId: ,
EntityPlayerId: ,
HasPenalty:
}
```

RESPONSE:

Success: Boolean, true if a valid response

Message: Error message if Success is false

Send Player Feedback & Report Player

API/Player/Feedback/SendFeedback

ACTION:

POST

HOST HEADER:

`gg-client-api-key – value is client api key`

POST BODY DATA PARAMETERS:

Name	Type	Valid Values	Can Null	Can Omit
EntityId	Int32			
TournamentSessionId	Int64		Yes	Yes
GameTitleId	Int32		Yes	Yes
FeedbackReportedById	Int32	EntityId of reporting player		
FeedbackRating	Int32	Rating of -5 to 5		
FeedbackMessage	String		Yes	Yes
InappropriateFlags	bool		Yes	Yes
ReportToAdministrator	bool		Yes	Yes

SPECIAL NOTE:

This method is used for players in-game to either rate players (-5 to 5), or for players & automated detection to report others as suspected of cheating. If reporting for cheating, must have "true" for InappropriateFlags and ReportToAdministrator

RESPONSE:

```
Success: Boolean, true if a valid response

Message: Error message if Success is false
```

TOURNAMENT SESSION (END GAME)

Insert End of Game Single Player Statistics

API/Player/Statistics/SendStatistics

ACTION:

POST

HOST HEADER:

gg-client-api-key – value is client api key

POST BODY DATA PARAMETERS:

Name	Type	Valid Values	Can Null	Can Omit
GamingServerId	Int32			
TournamentSessionId	Int64			
GameTitleId	Int32			
GameTitleMapId	Int32	ID of Map from Session		
StatisticsData	Array	See definition below		

DATA:

```
{
GamingServerId: ,
TournamentSessionId: ,
GameTitleId: ,
GameTitleMapId: ,
StatisticsData: [{}]
}
```

RESPONSE:

```
Success: Boolean, true if a valid response
Message: Error message if Success is false
```

Insert End of Game Multiple Player Statistics

API/Players/Statistics/SendStatistics

ACTION:

POST

HOST HEADER:

`gg-client-api-key – value is client api key`

POST BODY DATA PARAMETERS:

Name	Type	Valid Values	Can Null	Can Omit
GamingServerId	Int32			
TournamentSessionId	Int64			
GameTitleId	Int32			
GameTitleMapId	Int32	ID of Map from Session		
StatisticsData	Object	See definition below		

DATA:

```
{
GamingServerId: ,
TournamentSessionId: ,
GameTitleId: ,
GameTitleMapId: ,
StatisticsData: [{}]
}
```

RESPONSE:

```
Success: Boolean, true if a valid response
Message: Error message if Success is false
```

SPECIAL NOTES:

Refer to [End of Game Player Statistics Data] for more information on data types within `StatisticsData` object.

Finalize Matchmaking Session

API/TournamentSession/Finalize

ACTION:

POST

HOST HEADER:

gg-client-api-key – value is client api key

POST BODY DATA PARAMETERS:

Name	Type	Valid Values	Can Null	Can Omit
TournamentSessionId	Int64			
ParticipantAResults	Object	See results data notes		
ParticipantBResults	Object	See results data notes		
ServerSentPlayerSummary Statistics	Bool	true, false		

DATA:

```
{
TournamentSessionId: ,
ParticipantAResults: [{FinalScore: , IsNoShow: , DidForefit:}],
ParticipantBResults: [{FinalScore: , IsNoShow: , DidForefit:}],
ServerSentPlayerSummaryStatistics:
}
```

RESPONSE:

Success: Boolean, true if a valid response

Message: Error message if Success is false

SPECIAL NOTES:

Participant Results Data

FinalScore

IsNoShow

DidForefit

Add Playback URL

API/TournamentSession/Playback/SendUrl

ACTION:

POST

HOST HEADER:

gg-client-api-key – value is client api key

POST BODY DATA PARAMETERS:

Name	Type	Valid Values	Can Null	Can Omit
TournamentSessionId	Int64			
PlaybackUrl	String			

DATA:

```
{
TournamentSessionId: ,
PlaybackUrl:
}
```

RESPONSE:

```
Success: Boolean, true if a valid response
Message: Error message if Success is false
```

END OF GAME PLAYER STATISTICS DATA

Game Title: Counter-Strike: Global Offensive

Valid Fields:

Name	Type	Valid Values	Can Null	Can Omit
EntityPlayerId	Int64			
WasOnTeamA	Bool			
KillsCount	Int32			
DeathsCount	Int32			
AssistsCount	Int32			
HeadshotsCount	Int32			
ObjectivesCount	Int32			
Kills2InRowCount	Int32			
Kills3InRowCount	Int32			
Kills4InRowCount	Int32			
Kills5InRowCount	Int32			
Clutch1V1Count	Int32			
Clutch1V2Count	Int32			
Clutch1V3Count	Int32			
Clutch1V4Count	Int32			
Clutch1V5Count	Int32			
ClutchTryCount	Int32			
GamingTitleMapId	Int32	ID of Map from Session		
HasPenalty	Bool			

Game Title: Battlefield 4

Valid Fields:

Name	Type	Valid Values	Can Null	Can Omit
EntityPlayerId	Int64			
WasOnTeamA	Bool			
KillsCount	Int32			
DeathsCount	Int32			
HeadshotsCount	Int32			
Kills2InRowCount	Int32			
Kills3InRowCount	Int32			
Kills4InRowCount	Int32			
Kills5InRowCount	Int32			
GamingTitleMapId	Int32	ID of Map from Session		
HasPenalty	Bool			

GAME SERVER GAMEPLAY RECOVERY

Get Snapshots

API/TournamentSession/GetSnapshots/{TournamentSessionId}

ACTION:

GET

HOST HEADER:

gg-client-api-key – value is client api key

POST BODY DATA PARAMETERS:

Name	Type	Valid Values	Can Null	Can Omit
TournamentSessionId	Int64			

DATA:

```
{
TournamentSessionId:
}
```

RESPONSE:

```
Success
Message
Response: (array of the data below)
        EventId,
        EventDateTime,
        Snapshot
```