Tribes:Ascend API Developer Guide

Registration
To register to become developer, go here and register:

If your application is accepted, then you will receive custom credentials to access the API.

Credentials
To access the api you'll need your own set of credentials which consist of a developer id (devId) and an authentication key (authKey). Here are the credentials for a sample account:

- DevId: (eg, 1004)
- AuthKey: (eg, 23DF3C7E9BD14D84BF892AD206B6755C)

You'll use your personal credentials to access the api via a *Representational State Transfer* (RESTful) webservice hosted at api.TribesAscend.com.

Getting Started
To begin using the API, you will first need to establish a valid Session. To do so you will start a session (via **createsession**) and receive a SessionId. Sessions are used for authentication, security, monitoring, and throttling. Once you obtain a SessionId, you will pass it to other methods for authentication. Each session only lasts for 15 minutes and must be recreated afterward.

List of Methods and {Parameters}

1. createsession[ResponseFormat]/{devId}/{signature}/{timestamp}
2. getplayer[ResponseFormat]/{devId}/{signature}/{timestamp}/{playername}
3. getmatchhistory[ResponseFormat]/{devId}/{signature}/{timestamp}/{playername}
4. gettimeplayed[ResponseFormat]/{devId}/{signature}/{timestamp}/{playername}
5. getmatchstats[ResponseFormat]/{devId}/{signature}/{timestamp}/{matchId}
6. getdataused[ResponseFormat]/{devId}/{signature}/{timestamp}
7. ping[ResponseFormat]

The url format for calling a method from the api is http://api.tribesascend.com/tribesapi.svc/ + the pattern for the method above, where [ResponseFormat] is replaced by the formatting that you want returned(either XML or JSON).

So to create a session with a JSON response, we would call the **createsession** method like so:

http://api.tribesascend.com/tribesapi.svc/createsessionJson/1004/8f53249be0922c94720834771ad43f0f/20120927183145

which would return something like this:

```
{
      "ret_msg": "Approved",
      "session_id": "0ECDF26BC1F04EE4BA4AF10EC3604E04",
      "timestamp": "2/14/2013 7:50:20 PM"
}
```

You can see that the sessionId is contained in an element called "session_id". You'll use this parameter to call the other methods.

You'll see that we passed in a few other variables besides our **devId** and **authKey**. Actually the authKey is not passed directly, but instead embedded and hashed in another parm (**signature**).

Creating a Signature

The signature is created by concatenating several fields and then hashing the result with an MD5 algo. The components of this hash are (in order):

1. your devId
2. the method name you're calling (eg, "createsession")
    a. This will not include the ResponseType, just the name of the method.
3. your authKey
4. utc timestamp (formatted yyyyMMddHHmmss)

Sample Code to Create a Signature in c#:

```
var signature = GetMD5Hash("1004" + "createsession"
+ "23DF3C7E9BD14D84BF892AD206B6755C" + "20120927183145");


private static string GetMD5Hash(string input) {
      var md5 = new System.Security.Cryptography.MD5CryptoServiceProvider();
      var bytes = System.Text.Encoding.UTF8.GetBytes(input);
      bytes = md5.ComputeHash(bytes);
      var sb = new System.Text.StringBuilder();
      foreach (byte b in bytes) {
            sb.Append(b.ToString("x2").ToLower());
      }
      return sb.ToString();
}

I've included an example of how to make a call to the getplayer method and
what the response will look like.
```

Calling the "getplayer" Method

To get stats for a given player, call **getplayer** like so:

http://api.tribesascend.com/tribesapi.svc/getplayerjson/1004/0abd990b4ca9f86817e087ad684515db/83B082E576584DA8B1DB073DECA9E819/20120927193800/carnage

Here is the pattern for this call:
getplayer[ResponseFormat]/{devId}/{signature}/{sessionId}/{timestamp}/{playerName}

For this call, I returned the results in JSON. The response was:


```
[
	{
	"Assists": 0,
	"Base_Assets_Destroyed": 29,
	"Base_Repairs": 486,
	"Base_Upgrades_Purchased": 11,
	"Belt_Kills": 204,
	"Callin_Kills": 30,
	"Callins_Made": 221,
	"Created_Datetime": "11/5/2012 8:11:39 PM",
	"Flag_Caps": 242,
	"Flag_Returns": 1582,
	"Full_Regenerations": 30,
	"Generators_Destroyed": 33,
	"Headshots": 45,
	"High_Speed_Flag_Grabs": 521,
	"Kills": 4585,
	"Kills_In_Vehicle": 140,
	"Last_Login_Datetime": "2/28/2013 6:23:56 PM",
	"Level": 50,
	"Matches_Completed": 464,
	"Melee_kills": 47,
	"Midairs": 449,
	"Multikills": 89,
	"Name": "HiRezAPC",
	"Ski_Distance": 253732,
	"Sprees": 124,
	"Tag": "MIA",
	"Vehicle_Roadkills": 63,
	"Vehicles_Destroyed": 27,
	"ret_msg": null
	}
]
```


Tribes Web Service URI
You've probably noticed that the uri that we're using to make our calls is:

http://api.tribesascend.com/tribesapi.svc/

This is the uri that you'll use for all API calls. You'll add a slash + the method + any parameters to complete the RESTful call.

MatchId

The pattern for getmatchstats is:
getmatchstats[ResponseFormat]/{devId}/{signature}/{sessionId}/{timestamp}/{matchId}

The {matchId} parm is a unique id for each map that's created by the server for a set of players. One place you can get this value from will be getmatchhistory.

Graphics

You can find any graphics that we've published for use here.