StreamLinked Manual Upgrade Guide

Created with reference to version 2.3.0

Making this guide to document how to update StreamLinked to the latest specification found on their change log page. Mainly to cover the possibility of if I become unavailable or Twitch updates something breaking your application (and StreamLinked). 99% of all changes will be for updating EventSub subscription or API endpoints, I can't anticipate the changes Twitch might make for the 1% but changes like that are very rare and shouldnt impact much in terms of functionality.

Methods have been provided to make requests without needing classes as long as you know the end point and http method, these methods can be returned as JsonValue or string (in json) with all the information provided by twitch, be it successful or error.

EventSub Subscription Updates

C# Class

All EventSub classes are found under /EventSub/Events/…

Implemented subscriptions are created as structs and implement the IEvent interface.

All classes **MUST** contain a constructor that takes a LightJson.JsonValue value, if this is not included the EventSub will not be able to provide the values to the class.

All values listed on the Twitch website for a subscription are listed as properties in the class and populated from the JsonValue value provided. To get the value from the JsonValue, provide the exact name provided for the value on the Twitch website into the values indexer. Checking for the values is done by the JsonValue so don't worry if the value doesn't exist. If a value doesn't exist it will return null for reference types and the minimum value for value types (eg. int.MinValue).

Object type properties are created either inside the used class or in an inheriting interface, they are built in the same way as the subscription so you only need to provide the json for the property to create the object. These are built and maintained under the folder Objects.

Also included is an enum value which is created to represent the subscription and lastly a Scopes array to say which twitch scopes are needed for this event (Relevant for Generic calls to the API).

A good example of an event that makes use of the possible values and built types of objects is ChannelChatMessage.

Enum

All enums for subscriptions are found in /EventSub/ in TwitchEventSubSubscriptionsEnum.

The enum values used for subscriptions are found in /EventSub/ in TwitchEventSubSubscriptions. These are the values sent to Twitch to create the subscriptions.

Enums are created with a custom attribute linking all available information about it together. The attribute EventSubInformation is attached to all Enum values and included inside it is the Twitch value, Twitch version number and lastly the typeof of the create C# class of the subscription. Please ensure any new values increment the largest value in the enum. Code to run in C# Interactive to find the next value to use is in a comment above the enum or you can use the menu option in Unity under StreamLinked/Enums called Check for Duplicates. Any found are outputted to the console.

EventSub Client

Once the C# class has been created, all that's required for the Client is adding the UnityEvent<T> and the switch case to execute it. The UnityEvents are found starting on line 64, the function to execute the UnityEvents is found on line 578 (FireEventDelegate). Ensure the added Event has the name ‘On’ followed by the event name e.g (public UnityEvent<AutomodMessageHoldV2> OnAutomodMessageHoldV2;)
Once added it should be picked up by the function FireEventDelegate which uses reflection to find and execute the action.

API Endpoint Updates

C# Class

All endpoints are found in organised folders under /TwitchAPI/APIEndpoints/Classes…

All endpoints are separated into 3 parts, the retrieved values, the implemented interface properties and the static value names for sending values up to Twitch.

All endpoints implement (at minimum) the interface ITwitchAPIDataObject, this will require the classes to implement, HTTPMethod (GET, PUT, POST ext), Endpoint (Web address to send the request to, All listed in TwitchAPILinks), Initialise(JsonValue body, the function called to build the endpoint properties received from the server) and TypeEnum (TwitchAPIClassEnum, Enum equivalent of the type).

All values listed on the Twitch website for an endpoint are listed as properties in the class and populated from the JsonValue value provided. To get the value from the JsonValue, provide the exact name provided for the value on the Twitch website into the values indexer (case sensitive).

Object type properties are created either inside the used class or in an inheriting interface, they are built in the same way as the class so you only need to provide the json for the property to create the object.

Classes inside folders share a common interface to group the classes together.

Once set up if not already added, make sure a enum value is added to TwitchAPIClassEnum ensuring any new values increment the largest value in the enum. Code to run in C# Interactive to find the next value to use is in a comment above the enum. Or you can perform a manual check with the menu item StreamLinked/Enums called Check for Duplicates. Lastly ensure the attribute TwitchAPIClassInformation is added to the value specifying the type and the resource category/enum.

TwitchAPIDataContainer<T>

Additional data for endpoints is found in the container class TwitchAPIDataContainer<T> which stores any available date provided from the request.
This class is also the main point for building the data provided by Twitch into the class of the endpoints. It will attempt to process all endpoint types provided by Twitch.
If your request errors, all available information about the request will be provided in this class. If you don’t wish to use this class and just want the information as Json then methods will just provide it as such, but if you do wish to convert it to this container, just provide it the JsonValue body in the constructor and it will build it for you. If you do wish to access the request string again as a value was missed, a value called RawResponse is provided.

Scopes

Both API Endpoints and EventSub subscriptions implement the interface IScope. This interface demands the implementation of a getter for a TwitchScopesEnum array. This array will hold any specified scopes twitch has listed on its website saying its required to use the endpoint. Scopes are handled as warnings in StreamLinked as Twitch hasn't dedicated to only having required scopes but some scopes are optional for additional information (For example GetUsers requires no scopes but needs a scope to get a users email). If scopes are not required, just return Array.Empty<TwitchScopesEnum>().

To update the provided scopes, the classes are TwitchScopes for the list of const strings holding the scope itself and TwitchScopesEnum which is an enum connecting the scope to known links between API endpoints and Eventsub subscriptions, extensions in between auto translate between the two so you just need to add them here to be able to apply them.