TwitchPlay Plugin

User Guide and Technical Details

Description

TwitchPlay is a plugin for easy integration with any **Twitch** channel chat. You can **read** or **send** messages from/to the chat effortlessly using **Blueprints** or **C++**. Subscribe and fire events based on **custom chat commands** - with optional parameters -. All of this directly from **inside Unreal Engine!**

Can't wait to see what you will create with this!

Example Project: [Download Now]

Implementation Notes

TwitchPlay uses Unreal Engine's FSockets to communicate with the official Twitch IRC servers. Connection drops are currently not handled automatically.

TMaps are used to save the delegate functions to call when custom chat events are fired. Because of the way the plugin is currently implemented, only one object function per custom command can be bound; if another object or function is associated with a custom chat command the previous bound delegate function will be removed. This might change in future API versions.

Features

- Read messages from Twitch chat
- Send messages to Twitch chat
- Fire events when a message is received
- Dynamically subscribe/unsubscribe to custom chat commands at runtime
- Receive custom chat commands optional parameters

Requirements

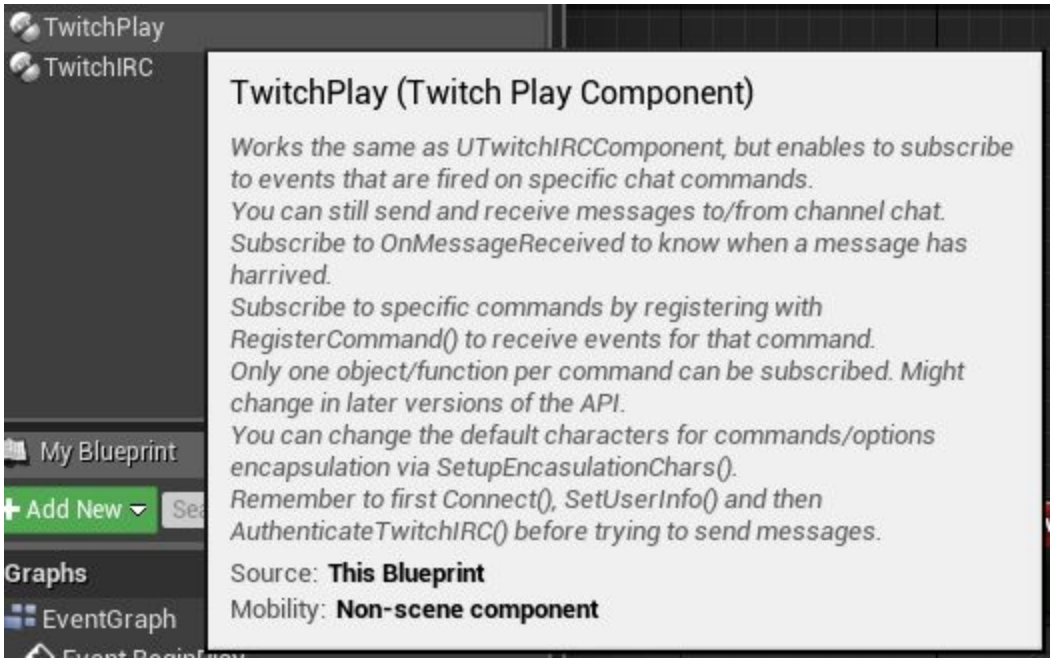
TwitchPlay was tested on Windows 10 running Unreal Engine 4.16.1 but should work on older/different versions of the engine. Please let me know if you encounter any problem.

How To - Overview

Components

The TwitchPlay Plugin enables you to use two components in order to interact with the Twitch chat: **TwitchIRCComponent** and **TwitchPlayComponent**. They can be added to any actor as a component.

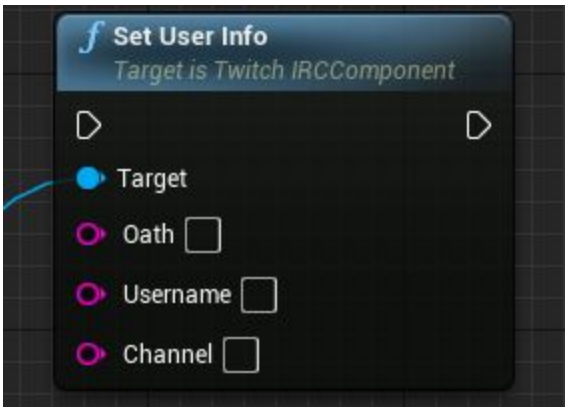
TwitchIRCComponent is the simpler version which only takes care of receiving and sending messages to the Twitch chat. Use this one if you just want to read/send messages and don't care about custom chat events.
TwitchPlayComponent lets you subscribe to custom chat events and fire events based off those at runtime. Use this if you want users to be able to send events to the game.



Setup

The standard process to setup and authenticate to the Twitch chat is the following:

Set User Info



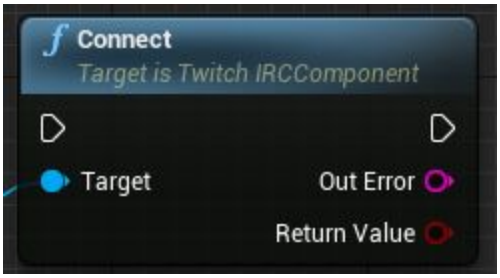
Here you will **setup the user info** in order to later connect and join a channel.

Oath is the authentication token you should get from Twitch (**KEEP THIS SECRET!**).

Username is the username you use to login on Twitch. All lowercase.

Channel [OPTIONAL] is the channel you want to join upon authentication.

Connect



You first need to **connect** to the server in order to authenticate later. This just creates a socket to communicate with the Twitch server. Does **NOT** authenticate the user.
You will receive a **boolean** value which indicates whether the connection was successful or not, along with an error string indicating what happened.

Authenticate

Authenticate the user to the Twitch IRC server and, if a specific channel was set, joins that channel afterwards.
Returns a **boolean** value indicating whether the authentication was successful.
Note, however, that in the current API version authentication will report **true** even when **password and/or username are wrong**. Authentication will fail only because of **connection issues**. I will address this in later versions of the API.

Sending and Receiving Messages

Sending and receiving messages is easy!

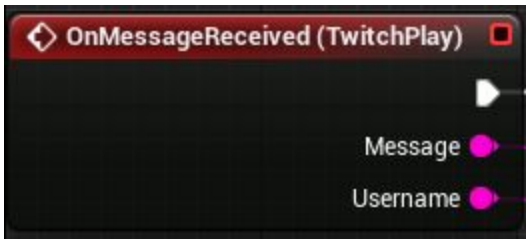
Sending

Message is the string you want to send to the server.

B Send To and **Channel** work together in order to let you send your messages inside a specific channel. Most of the time you'll want to set this to **true** and pass in the same **channel** you used during the **setup**.

Returns whether the message was sent correctly.

Receiving



Each time a new message is received the **OnMessageReceived** event is fired.

Message is the content of the message received.

Username is the string representing the username of the sender.

Custom Chat Commands

TwitchPlayComponent lets you subscribe to any chat command sent to the Twitch chat.

Command Delimiters

Setup command and command options delimiters for messages coming from the Twitch chat.

Default characters are **“!”** (**commands**) and **“#”** (**command options**).

Commands will be read in the form **CHAR_Command_CHAR** (no spaces or underscores are actually present!)

Command options will be read in the form **CHAR_Option1[,Option2,..]_CHAR** (no spaces or underscores are actually present!). Multiple options can be specified and will be split into a FString array upon parsing.

Example:
!SPAWN!#Big,red,33#
This would translate to: **Command = SPAWN - Options = [Big, red, 33]**

Register Command

Registers a **custom command** and binds its execution to an **event**.

Command Name is the name of the command as it’s coming from the chat. Spaces and upper/lower case **DO MATTER!**

Callback Function is the event you want to be called whenever the specified chat command is received.

Returns the result of the operation or why it might have failed.

NOTE: only one object function per custom command can be bound; if another object or function is associated with a custom chat command the previous bound delegate function will be removed. This might change in future API versions.

Unregister Command

Unregisters the specified command so that it cannot be called anymore.

Command Name is the name of the command you want to unregister.

Returns the result of the operation or why it might have failed.

Command Events

The **signature** for a function/event bindable to a command must be structured as follows.

Command Name - FString: Name of the command that was received. Ideally you will create a **1 to 1 relationship** between available commands and command events, however you could try and handle **multiple commands with a single function** based on this string.

Command Options - FString Array: Command options, if any, that were received together with the command.

Sender Username: Username of who sent the command.

Support / Bugs / Suggestions:

- altairjp@gmail.com
- https://twitter.com/TheDiG3
- https://www.facebook.com/TheDiG3