


Guilty Gear Strive Live Content Mod FAQ:

1. What is the purpose of this mod?:
 - a. Functionally, the mod exports the gamestate of any battle in GGST. This exported data is then accessible as WebSockets messages to any client that may be listening. The purpose of this is to drive Live Content displays such as OBS effects for Twitch streams, live shows and even interactive side games.
2. How do you get this data?:
 - a. The simple answer is that I'm hooking functions that the game calls regularly and can then scoop out the data that I want, package it up and send it.
3. How are you doing the hooking? I thought Windows had memory protection to prevent this kind of thing!
 - a. You're right in that thought. I'm bypassing this restriction by pretending to be part of the game, so I can access its memory without errors. I'm working within a generic Unreal Engine 4 modding framework called UE4SS. The framework gets itself (and my mod) into the game's address space by replacing "dwmapi.dll" with one that calls our functions in addition to passing other calls to the original file.
4. How did you decide what functions to hook?:
 - a. The arcade version of Strive accidentally shipped with a PDB file next to the executable. Basically, modders like me can see function, data type and other symbol names (and contents) using software like Ghidra. In my case, I had to compare the arcade version with the latest Steam version to find the functions I need. I decided which functions to find based on their names. When they're called things like "UpdateBattleSub", "ExecuteAttackHit" and "RoundReset", it's pretty easy to figure out.
5. How do the function hooks work?:
 - a. Even though I don't fully understand it, I can explain how I use them. Essentially, once I figure out which function in the current version of the game maps to the one I want in the Arcade version, I then copy down the first 16 to 20 bytes of that function, such that the string of bytes is unique in the executable. Then, when the mod is initializing, it uses a UE4SS library (in addition to safetyhook) to scan for those sequences. Then, whenever the game calls those functions, it gets redirected to my code first.
6. How do you know what data you're reading?:
 - a. As mentioned above, the PDB from the Arcade version included the contents of all its data types, so I simply create structs on my end that have the same memory layout. The unfortunate thing is that, with seemingly every update, the offsets within the structures change a little bit. To get around this, I map more functions from Arcade to current and then use them to find the correct offsets.
7. How are you converting the raw data from the game into JSON?:
 - a. I'm using a library called JSON.hpp. It has some macros that allow me to set up a completely abstracted away conversion from types that I create to JSON. Then I simply turn the json object into a string and send it to the WS server as a message to send.
8. Why does the mod break every update!?:
 - a. Because the function signatures and data offsets change every time ArcSys recompiles the game. So every update, I have to basically start from scratch.
9. Why is there a 20 frame delay!?:
 - a. This is the type of mod that can easily enable cheaters and bot-makers, given the level of access the mod allows. For that reason, every message is delayed by the same amount. This would prevent a cheating program from being very responsive, and thus any good.
10. Why is the code private?:
 - a. Basically the same reason as the delay. If the code was public, any bad actor could just comment out the one line that implements the delay and build the mod, circumventing the security entirely. Since Strive still has officially-run online tournaments, I can't in good conscience let that out into the wild.
 - b. Additionally, if you have a real, demonstrable need to see the code, I can allow you access. Please contact me for details.

11. I want to make a mod like this for my favorite game! How do I get started?

a. First, you'd want to learn the basics of reverse engineering. Then read this:

 [How TheLettuceClub goes about modding Fighting Games](#) .