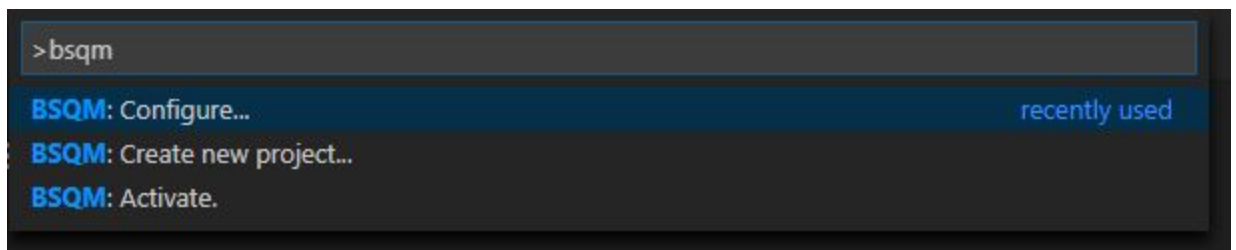
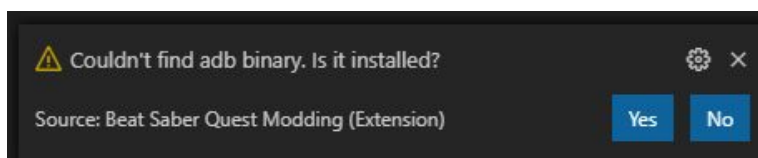


1. Download Unity 2018.3.14f1 <https://store.unity.com/download>
2. Download and install Visual Studio Code  
<https://code.visualstudio.com/download>
3. Open Visual Studio Code and setup
4. Download and install BSQM  
<https://marketplace.visualstudio.com/items?itemName=raftario.bsqm>
5. Configure BSQM

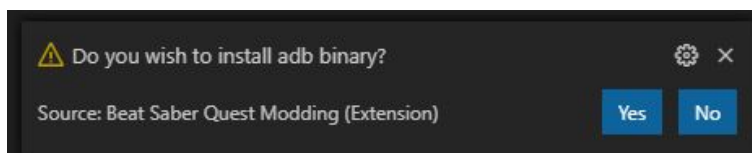


Ctrl+Shift+P  
*bsqm.configure*  
Enter

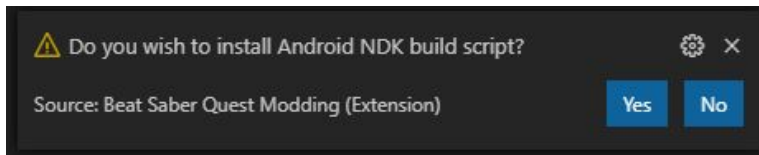
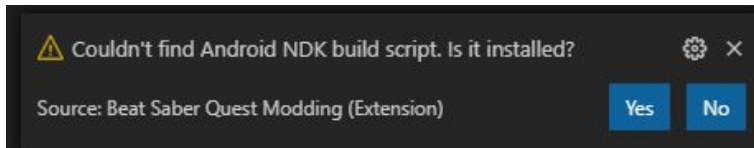
You might see this message...



If you know where it is, click yes to choose the directory. If not, click no to download.



Same again with the NDK build script



## 6. Create a BSQM project

Ctrl+Shift+P

*bsqm.create*

Enter

A dark-themed form titled "Create a new Beat Saber Quest mod". It includes instructions: "These values will be used to fill the manifest and create a basic README. You can check out the manifest format [here](#)." The form has two columns. The left column contains fields for ID (text: "someid"), Name (text: "Some Mod"), Author (text: "Some Person"), Category (dropdown: "Gameplay"), Game Version (dropdown: "1.6.0"), and a "Ready?" checkbox. The right column contains a large text area for Description (text: "Does Something"), and three "Browse..." buttons for Project folder, ndk-bundle folder, and libil2cpp folder. The last button is pre-filled with the path "C:\Program Files\Unity\Hub\Editor\2018.3.14f1\Editor\Data\il2cpp\libil2cpp". A blue "Create" button is at the bottom left.

Fill out the

- ID: Something to identify the mod
- Name: The mod's name
- Author: your name or username
- Category: Gameplay for this tutorial
- Game version: IMPORTANT use that latest version. We will change it to your game version later
- Project folder: Directory to a folder where the files will be generated
- NDK folder: MAKE SURE IT IS FILLED OUT. If it isn't, browse and select the directory.

Click Create.

## 7. Get the APK

Download Sidequest <https://sidequestvr.com/setup-howto>

Connect your Quest



Click the grid



Click the gear next to Beat Saber



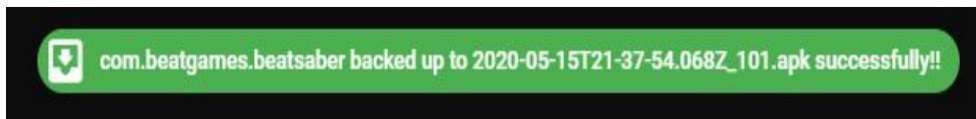
Click Backup Apk File



You can click the number to see the progress



When you see this...



... then click the grid and gear again, and this time click Open Backups



Unzip the file with WinRar or something else

Copy the following into a new folder:

- `/assets/bin/Data/Managed/global-metadata.dat`
- `/lib/arm64-v8a/libil2cpp.so`

## 8. Extract the DLLs

Download il2cppdumper <https://github.com/Perfare/Il2CppDumper/releases>

Make sure it is the non-netcore version



Extract and run the il2cppdumper application

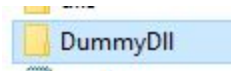


Open the two files from the apk in the FOLLOWING ORDER

1. libil2cpp.so
2. global-metadata.dat

Wait patiently until the program says “Press any key to exit”, then do so.

Your dlls will be in the DummyDLL folder.



Colors.dll	5/15/2020 2:57 PM	Application exten...	6 K
Core.dll	5/15/2020 2:57 PM	Application exten...	4 K
DynamicBone.dll	5/15/2020 2:57 PM	Application exten...	9 K
FinalIK.dll	5/15/2020 2:57 PM	Application exten...	280 K
HMLib.dll	5/15/2020 2:57 PM	Application exten...	94 K
HMRRendering.dll	5/15/2020 2:57 PM	Application exten...	47 K
HMMUI.dll	5/15/2020 2:57 PM	Application exten...	118 K
Il2CppDummyDll.dll	5/15/2020 2:56 PM	Application exten...	2 K
LIV.dll	5/15/2020 2:57 PM	Application exten...	3 K
Main.dll	5/15/2020 2:57 PM	Application exten...	990 K
MediaLoader.dll	5/15/2020 2:57 PM	Application exten...	10 K
Mono.Security.dll	5/15/2020 2:56 PM	Application exten...	5 K
mscorlib.dll	5/15/2020 2:56 PM	Application exten...	1,559 K
netstandard.dll	5/15/2020 2:57 PM	Application exten...	2 K
nunit.framework.dll	5/15/2020 2:57 PM	Application exten...	125 K
Oculus.VR.dll	5/15/2020 2:56 PM	Application exten...	652 K
OculusPlatform.dll	5/15/2020 2:57 PM	Application exten...	227 K
Polyglot.dll	5/15/2020 2:57 PM	Application exten...	21 K
Rendering.dll	5/15/2020 2:57 PM	Application exten...	29 K
SteamVR.dll	5/15/2020 2:56 PM	Application exten...	349 K
Steamworks.NET.dll	5/15/2020 2:57 PM	Application exten...	2 K
System.Configuration.dll	5/15/2020 2:56 PM	Application exten...	5 K
System.Core.dll	5/15/2020 2:56 PM	Application exten...	46 K
System.Diagnostics.StackTrace.dll	5/15/2020 2:57 PM	Application exten...	2 K
System.dll	5/15/2020 2:56 PM	Application exten...	193 K
System.Globalization.Extensions.dll	5/15/2020 2:57 PM	Application exten...	2 K
System.Xml.dll	5/15/2020 2:56 PM	Application exten...	85 K



## 9. Open the DLLs

Download dnSpy <https://github.com/0xd4d/dnSpy/releases>

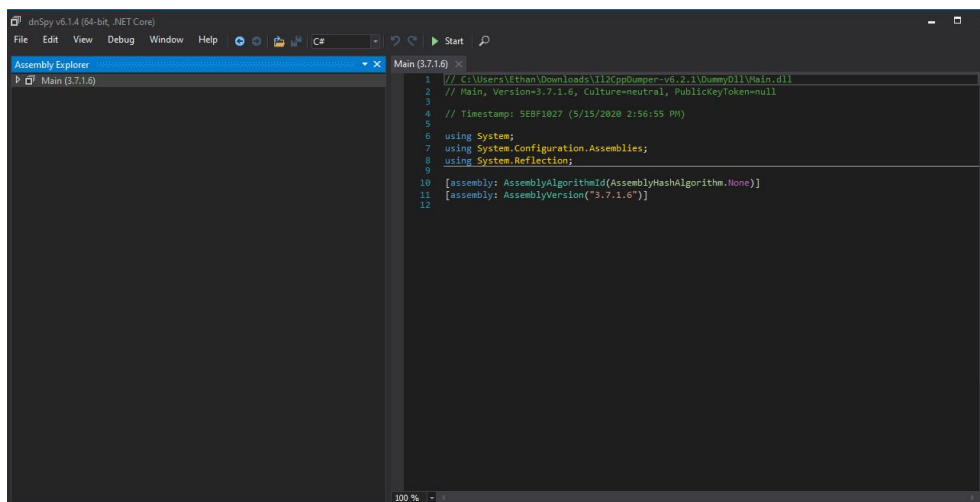
I recommend one of these, as it doesn't require NET core to be installed.

 <a href="#">dnSpy-netcore-win32.zip</a>	74.9 MB
 <a href="#">dnSpy-netcore-win64.zip</a>	81.2 MB

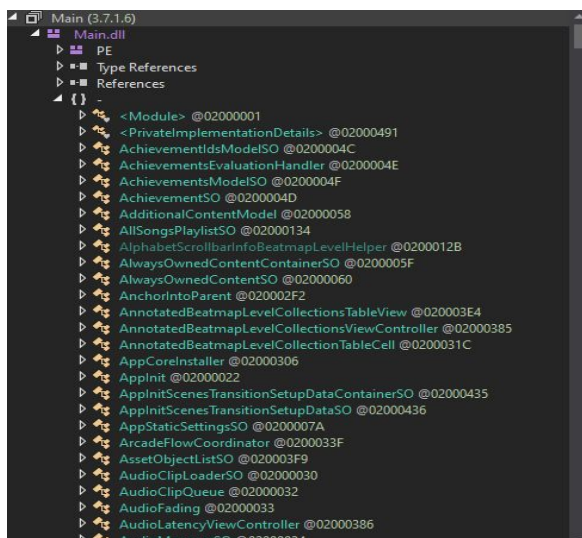
Open dnSpy (not dnSpy.console)

 bin	5/15/2020 8:25 AM	File folder	
 dnSpy.Console	3/17/2020 11:28 AM	Application	166 KB
 dnSpy	3/17/2020 11:28 AM	Application	234 KB

Drag Main.dll (located in DummyDLL) onto the left of dnSpy



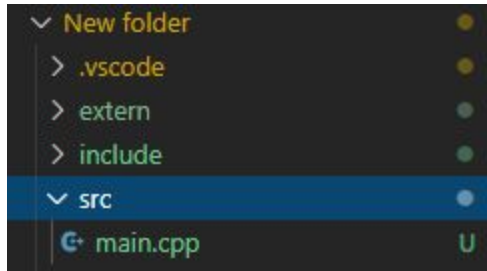
That's it! All the Hook methods are in the tree!



## 10. Let's Make a better Sign

Now we will make an example mod.

Open Visual Studio Code and navigate to *(your mod folder)/src/main.cpp*



Replace the document with this:

```
#include "../include/mod_interface.hpp"
#include <unordered_set>
#include "../extern/beatsaber-hook/shared/utils/il2cpp-utils.hpp"
#include "../extern/beatsaber-hook/shared/utils/il2cpp-functions.hpp"
#include "../extern/beatsaber-hook/shared/utils/utils.h"

extern "C" void load() {
    log(INFO, "Hello from il2cpp_init!");
    log(INFO, "Installing hooks...");

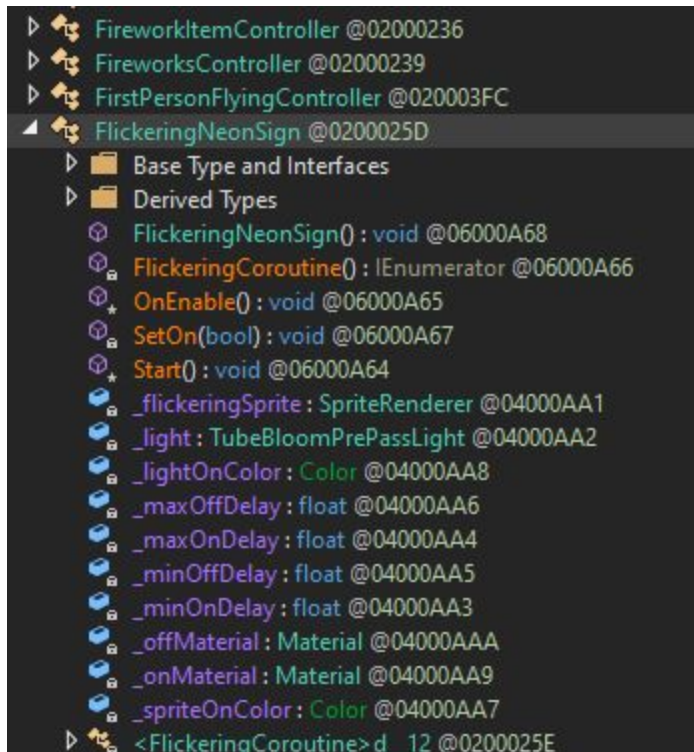
    log(INFO, "Installed all hooks!");
}

enum class Space {
    World,
    Self
};
DEFINE_IL2CPP_ARG_TYPE(Space, "UnityEngine", "Space");
```

Now, this is the BASE of any mod. The *#includes* at the top basically tell the file where other code to run is. *extern "C" void load()* is a function that runs the code in it when the mod is loaded. *enum class Space* defines the world area.

Do you know the flickering Beat Saber logo? Let's make it better!

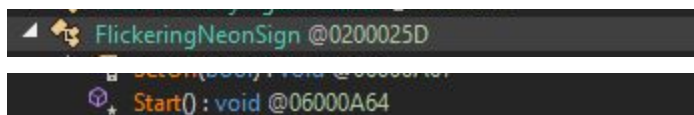
In dnSpy, there is a FlickeringNeonSign class. Within that, there is are some floats (numbers) that define the random range of the flicker. We are going to use that.



First, we need to make a hook - to “hook” on to that function when it is called in the game. Under the *#includes*, type *MAKE\_HOOK\_OFFSETLESS()* to make a new hook. *MAKE\_HOOK\_OFFSETLESS* takes three parameters:

- Function name
- Function type
- Object

In this example, we will hook *FlickeringNeonSign* at its *Start*, so the first parameter is *FlickeringNeonSign\_Start*



As you can see in the picture, *Start* is a *void*, so the second parameter is *void*.

The object will be *Il2CppObject\* self*. So the third parameter will be that, too.

Altogether, the comma-separated parameters will be *FlickeringNeonSign\_Start, void, Il2CppObject\* self*, So put that in the parentheses

```
MAKE_HOOK_OFFSETLESS(FlickeringNeonSign_Start, void, Il2CppObject* self) {
}
```

Because we made a hook, we have to install it. We will install it at the start of the game - in *extern "C" void load() {}*.

The function for installing a hook is *INSTALL\_HOOK\_OFFSETLESS()*. The first parameter is the same, *FlickeringNeonSign\_Start*, and the other is almost the same: *il2cpp\_utils::FindMethod("", "FlickeringNeonSign", "Start")*.

Put it together with comma separation, and you have *FlickeringNeonSign\_Start, il2cpp\_utils::FindMethod("", "FlickeringNeonSign", "Start")*. Put that in parentheses to get *INSTALL\_HOOK\_OFFSETLESS(FlickeringNeonSign\_Start, il2cpp\_utils::FindMethod("", "FlickeringNeonSign", "Start"))*;

Put it in *extern "C" void load() {}* to get

```
extern "C" void load() {  
    log(INFO, "Hello from il2cpp_init!");  
    log(INFO, "Installing hooks...");  
  
    INSTALL_HOOK_OFFSETLESS(FlickeringNeonSign_Start,  
        il2cpp_utils::FindMethod("", "FlickeringNeonSign", "Start"));  
  
    log(INFO, "Installed all hooks!");  
}
```

Your current code should look like this:

```
#include "../include/mod_interface.hpp"  
#include <unordered_set>  
#include "../extern/beatsaber-hook/shared/utils/il2cpp-utils.hpp"  
#include "../extern/beatsaber-hook/shared/utils/il2cpp-functions.hpp"  
#include "../extern/beatsaber-hook/shared/utils/utils.h"  
  
MAKE_HOOK_OFFSETLESS(FlickeringNeonSign_Start, void, Il2CppObject* self) {  
  
}  
  
extern "C" void load() {  
    log(INFO, "Hello from il2cpp_init!");  
    log(INFO, "Installing hooks...");  
    INSTALL_HOOK_OFFSETLESS(FlickeringNeonSign_Start,  
        il2cpp_utils::FindMethod("", "FlickeringNeonSign", "Start"));  
    log(INFO, "Installed all hooks!");  
}
```

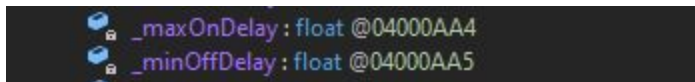


```
enum class Space {
    World,
    Self
};
DEFINE_IL2CPP_ARG_TYPE(Space, "UnityEngine", "Space");
```

To finally change the sign flicker randomness, we need to hook the variables. Put the following in the `MAKE_HOOK_OFFSETLESS` function.

```
il2cpp_utils::SetFieldValue(self, "_minOffDelay", 999999999999.0f);
il2cpp_utils::SetFieldValue(self, "_maxOffDelay", 999999999999.0f);
il2cpp_utils::SetFieldValue(self, "_minOnDelay", 0.0f);
il2cpp_utils::SetFieldValue(self, "_maxOnDelay", 0.0f);
```

`SetFieldValue` sets the values of the delays. They are in `dnSpy` if you remember:



Now, we need to start `Start`

```
FlickeringNeonSign_Start(self);
```

The sign will stay on for at least (a lot of) seconds, and stay off for a maximum of 0 seconds. Basically the sign will always stay off.

`MAKE_HOOK_OFFSETLESS` should look like

```
MAKE_HOOK_OFFSETLESS(FlickeringNeonSign_Start, void, Il2CppObject* self) {

    il2cpp_utils::SetFieldValue(self, "_minOffDelay", 999999999999.0f);
    il2cpp_utils::SetFieldValue(self, "_maxOffDelay", 999999999999.0f);
    il2cpp_utils::SetFieldValue(self, "_minOnDelay", 0.0f);
    il2cpp_utils::SetFieldValue(self, "_maxOnDelay", 0.0f);

    FlickeringNeonSign_Start(self);

}
```

Your entire `main.cpp` file should be:

```

#include "../include/mod_interface.hpp"

#include <unordered_set>

#include "../extern/beatsaber-hook/shared/utils/il2cpp-utils.hpp"
#include "../extern/beatsaber-hook/shared/utils/il2cpp-functions.hpp"
#include "../extern/beatsaber-hook/shared/utils/utils.h"
MAKE_HOOK_OFFSETLESS(FlickeringNeonSign_Start, void, Il2CppObject* self) {

    il2cpp_utils::SetFieldValue(self, "_minOffDelay", 999999999999.0f);
    il2cpp_utils::SetFieldValue(self, "_maxOffDelay", 999999999999.0f);
    il2cpp_utils::SetFieldValue(self, "_minOnDelay", 0.0f);
    il2cpp_utils::SetFieldValue(self, "_maxOnDelay", 0.0f);

    FlickeringNeonSign_Start(self);

}

__attribute__((constructor)) void lib_main() {
    log(INFO, "Hello from the first time this mod is loaded!");
}

extern "C" void load() {
    log(INFO, "Hello from il2cpp_init!");
    log(INFO, "Installing hooks...");
    INSTALL_HOOK_OFFSETLESS(FlickeringNeonSign_Start,
il2cpp_utils::FindMethod("", "FlickeringNeonSign", "Start"));
    log(INFO, "Installed all hooks!");
}

enum class Space {
    World,
    Self
};

DEFINE_IL2CPP_ARG_TYPE(Space, "UnityEngine", "Space");

```

Your code is ready! Make sure to save it

## 11. Change game version

In *bmbfmed.json*, there is a line:

`"gameVersion": "1.6.0",`

Change it to

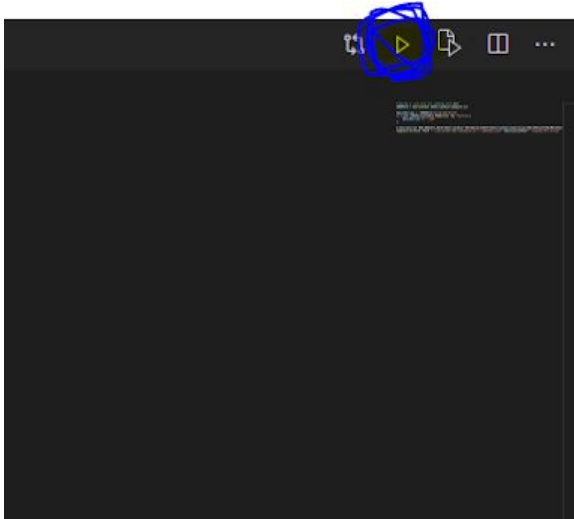
`"gameVersion": "1.9.0",`

Or your latest version, and save.

## 12. Build

Open *buildbmbf.ps1*

Click play in the top right



Open File Explorer and open your mod folder. If there is a zip file, you are good to go!  
Upload that to BMBF.

## 13. Test

You are good if you see this!

