How to RME, for idiots $made by Dev___1_$

A detailed guide on how to EXACTLY do custom RME options with the sunset source

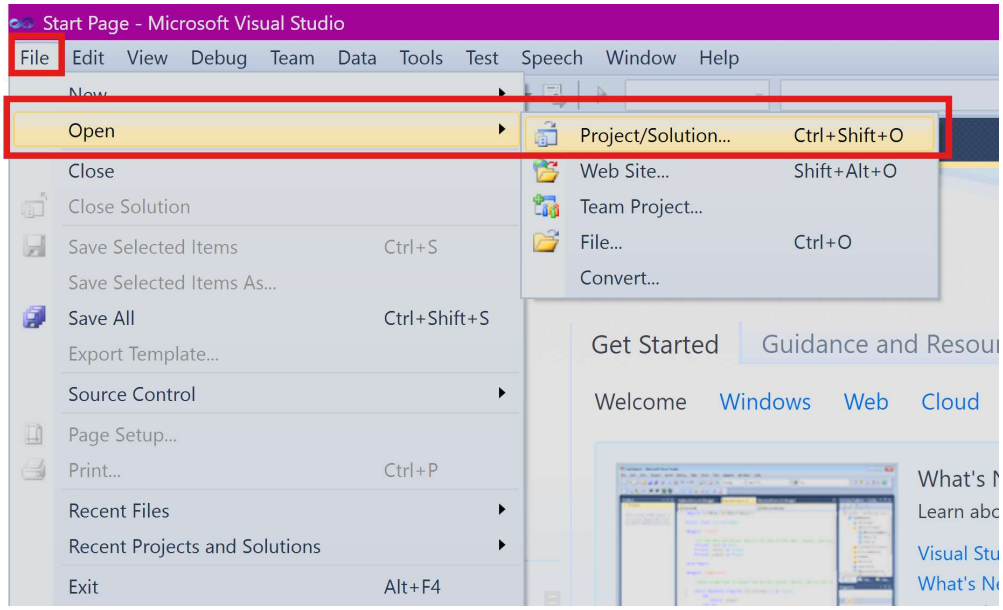
For which people is this guide aimed at: Complete beginners that have no clue at all what the hell they are doing and are allways telling themselves that its too hard to do this sort of stuff. Grow up and start screwing around with your consoles, instead of just playing games!

What you can do after going through the guide: You will be able to do very own CUSTOM RME options. You can do RME DVAR edits and exact memory edits. You will be able to get on your own memory adresses to use RME on AND you will be able to convert those values into the correct format for RME use.

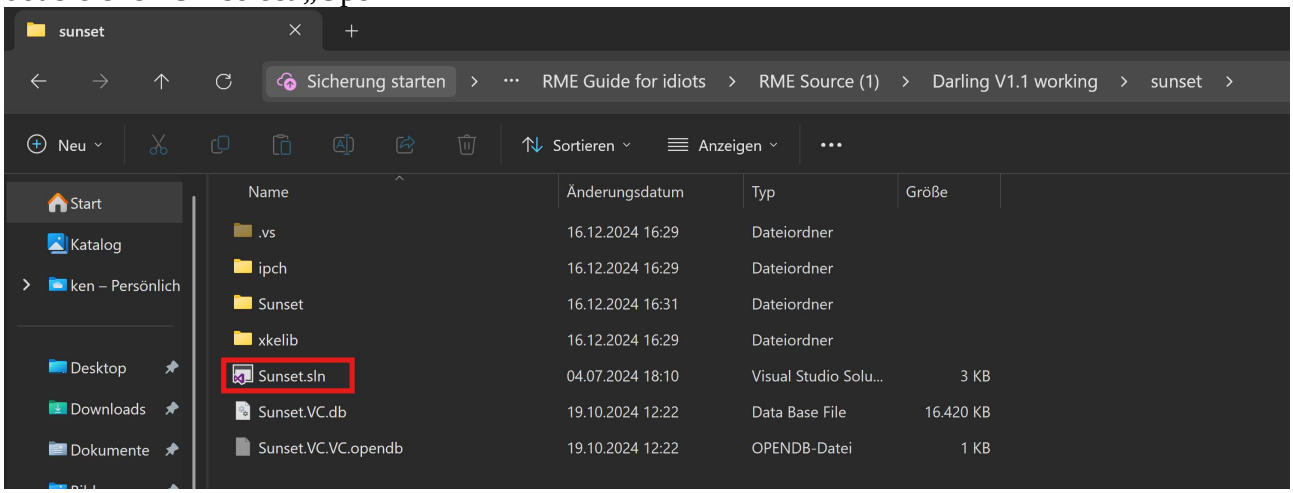
What you need to have before you start:
1. Have Visual Studio 2010 (vs 2010) installed [https://www.mediafire.com/file/ijyfbxk8r05nuwf/Visual-Studio-2010-Professional.iso/file] = the direct link to the download, but this > [https://consolecrunch.com/threads/ps3-sdk-4-75-and-prodg-v4-70-and-visual-studio-2010-and-2013-professional.30195/] is where it is from, if you dont trust my direct link then use the second one, if you dont trust either then go fuck yourself. Get also a vs 2010 key from the second link, it needs to be redeemed after the download
2. Have the sunset source, its inside this repo under the folder name (Sunset Source), this source WASNT modified by me except that BO2.pdb was added, dont worry about it, its just shit important for building the xex later and was before missing
3. Be really motivated to go through this guide

The beginning

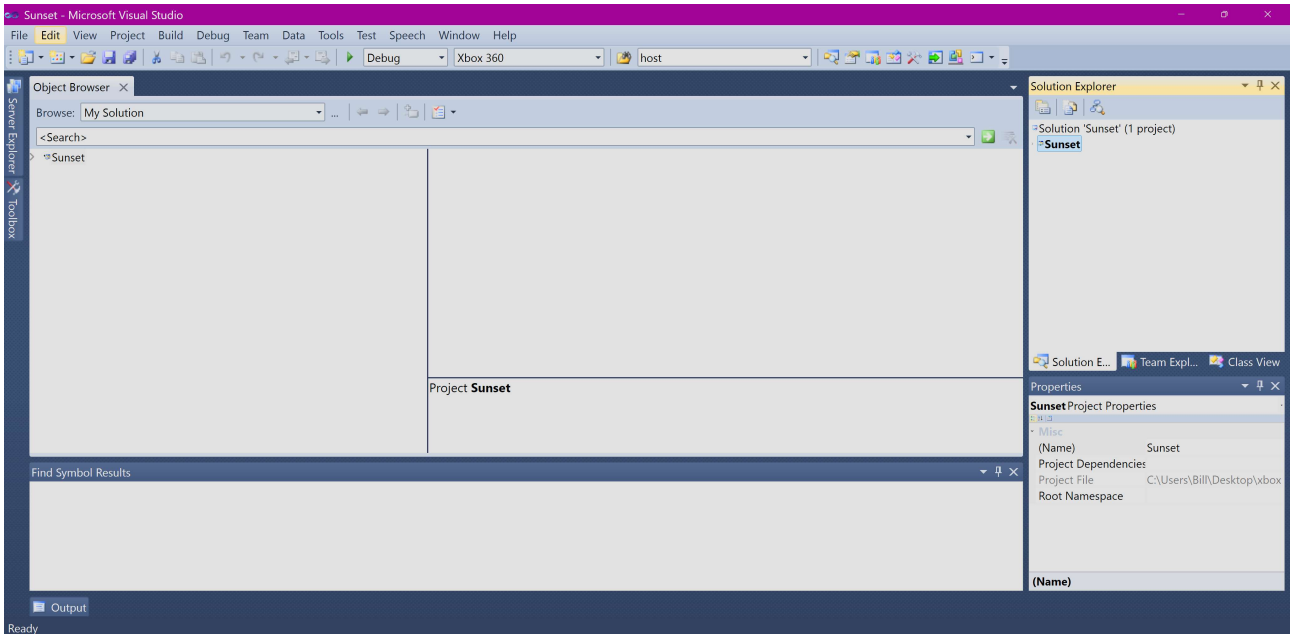
First we want to open the source in vs 2010



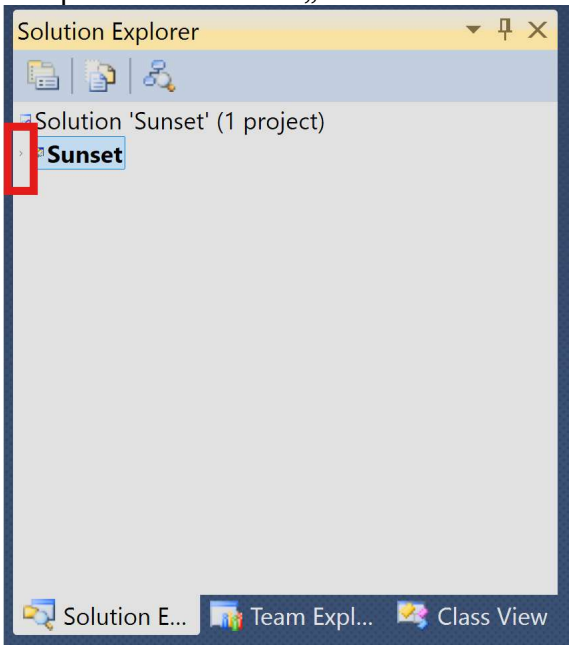
Now find the „ .sln “ file inside the source, its inside the „sunset“ folder. Open this file with a double click OR select „Open“



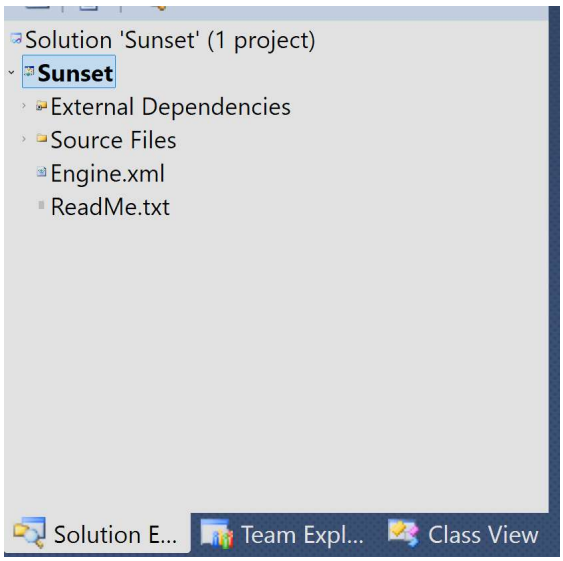
What we now will see:



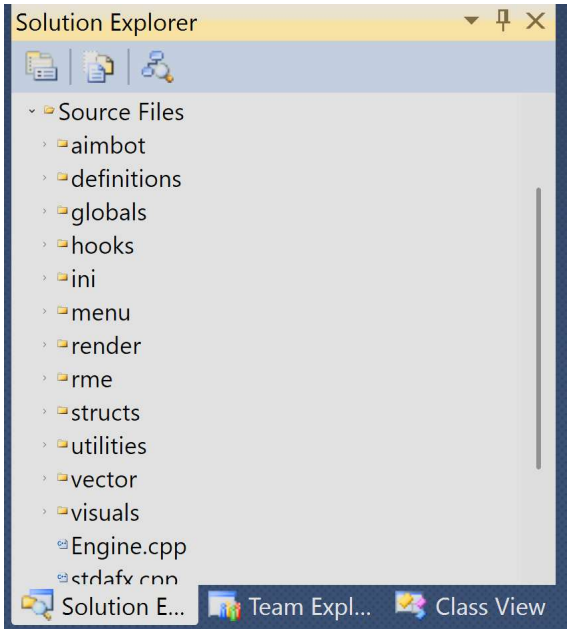
Can you see that tiny arrow that points down at the „Sunset“ name on the left?



Can you see it now?
Good, click that!

The following opened now: 

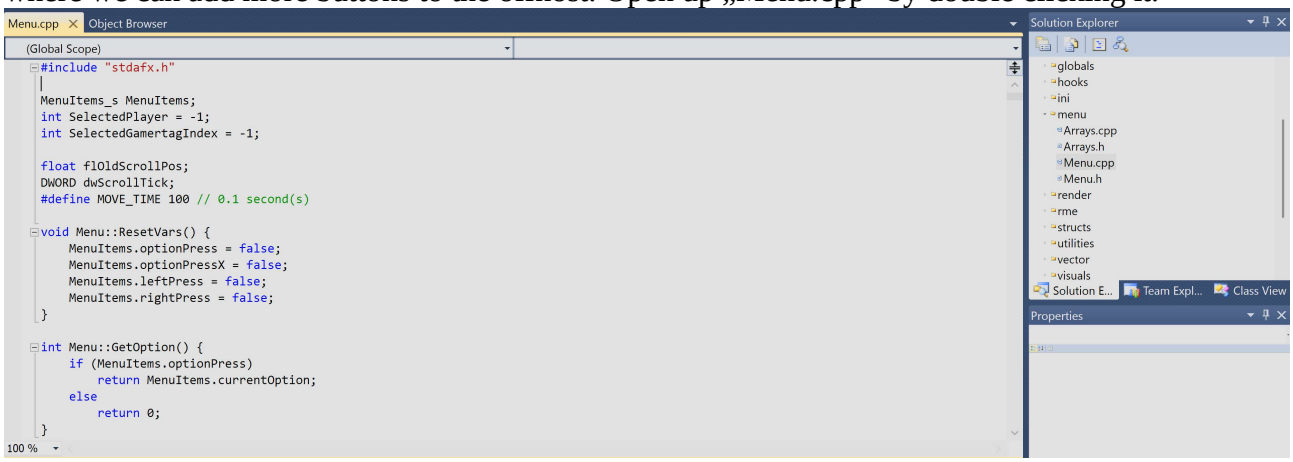
What is this? Its where all the imortant files are for coding. „External Dependencies“ isnt important for you, your a idiot afterall.
What is important is „Source Files“ tho, so open it up. You see the following now:



Is this black magic? No, its the god damn folders for all the damn offhost functionality basically, you idiot…
In these folders you can find the code for how the offhost looks like, how ESP is drawn/looks, how exactly aimbot works and so much more. What we only care about really is the folder for „menu“, „rme“ and „globals“.
But please now first go and do the propertie changes like it was shown in my other guide called „how to build a xex.pdf“. This guide is also inside this repo in the folder „How to do propertie changes feat Tupac“. You DONT need to add anything new to the xml file! Just let it be as is! And instead of using „Release“ for what its supposed to build, choose „BO2“. And DONT add the xkelib folder to the source folder, its already inside the source folder > it would be retarded to do again. Dont be a retard. These three things are the only things that need to be done differently from the guide.

If you did all the propertie changes you SHOULD be able to build the offhost, this step was also shown inside the guide mentioned before, try it out NOW!

Are you done? Cool, didnt ask. Lets start to understand what exactly we need to edit to make own options, shall we.
Inside the „menu“ folder we got the code that is responsible for drawing the offhost. So this is where we can add more buttons to the offhost. Open up „Menu.cpp“ by double clicking it.



This is code for the offhost look. Scroll through it, read a bit through the code. Concentrate on the code that is in between „“. At some point you will see normal readable text. The very first example for this is the following:

```
//Current Menu
PCHAR MenuTitles[15] = { "Main", "Main Menu", "Aimbot Menu", "Visual Menu", "Settings", "Client List", "RME Menu", "RME Extra Option
```

the green shit is a comment, and from that we can already assume for what the hell this code is used. From that and also the variable name „MenuTitles“. Its the titles for the different sub menus. You see each of these titles when you go into a sub menu like the aimbot menu. Try changing a name and build the source again. Then load the .xex that you now build and throw it onto your console. You will see the new name of the sub menu now. Congrats, your a 1337 coder now!
Lets look at a actual button from the offhost, let us take the button that would give god mode to a certain user. This option would be under the Client List, select a client and then choose RME Options. Now you would see in the offhost that option. Here is the option inside the code:

```
case CLIENT_RME_OPTIONS:
    if (Utilities::ReadBoolFromDWORD(Global.sunsetInfo.dwHasExtras)) {
        Menu("Extra Options").Submenu(CLIENT_RME_EXTRAS);
    }

    Menu("God Mode").ArrayEditor(RME_Toggle[MenuOptions::dwRME_Client_GodMode], ARRAYSIZE(RME_Toggle), (int*)&MenuOptions::dwRME_Client_GodMode).Ca
    Menu("Super Speed").ArrayEditor(RME_Toggle[MenuOptions::dwRME_Client_SuperSpeed], ARRAYSIZE(RME_Toggle), (int*)&MenuOptions::dwRME_Client_Super
    Menu("Invisibility").ArrayEditor(RME_Toggle[MenuOptions::dwRME_Client_Invisibility], ARRAYSIZE(RME_Toggle), (int*)&MenuOptions::dwRME_Client_Ir
```

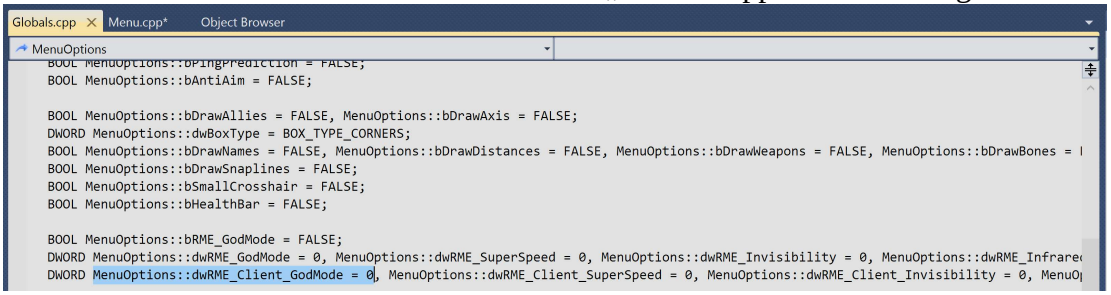
As we are idiots who just want to have some fun doing custom shit we gonna ignore ALL shit we dont need. Lets copy/paste that exact code right under it:

```
case CLIENT_RME_OPTIONS:
    if (Utilities::ReadBoolFromDWORD(Global.sunsetInfo.dwHasExtras)) {
        Menu("Extra Options").Submenu(CLIENT_RME_EXTRAS);
    }

    Menu("God Mode").ArrayEditor(RME_Toggle[MenuOptions::dwRME_Client_GodMode], ARRAYSIZE(RME_Toggle), (int*)&MenuOptions::dwRME_Client_GodMode).Ca
    Menu("God Mode").ArrayEditor(RME_Toggle[MenuOptions::dwRME_Client_GodMode], ARRAYSIZE(RME_Toggle), (int*)&MenuOptions::dwRME_Client_GodMode).Ca

    Menu("Super Speed").ArrayEditor(RME_Toggle[MenuOptions::dwRME_Client_SuperSpeed], ARRAYSIZE(RME_Toggle), (int*)&MenuOptions::dwRME_Client_Super
    Menu("Invisibility").ArrayEditor(RME_Toggle[MenuOptions::dwRME_Client_Invisibility], ARRAYSIZE(RME_Toggle), (int*)&MenuOptions::dwRME_Client_Ir
```

If you wanna understand what exactly happens then do that yourself by screwing around. We arent here to be smart, we are here to do custom shit! No nerds allowed!
Double click onto „`dwRME_Client_GodMode`“ and do a right click and select „Go to definition“
Now we are all of the sudden inside the folder „Globals.cpp“ but we dont give a shit.



Select all the shit I did and copy it(yes baby, we be skidding). Now go all the way to the right, make a comma after the last 0, and paste that exact shit in right before the „:“.
Rename the shit you pasted. So from „Client_GodMode“ make „dwRME_NewRMEOption“. Important: DONT MAKE THE SAME FUCKING NAME TWICE.

```
MenuOptions::dwRME_Client_SetMode = 0, MenuOptions::dwRME_NewRMEOption = 0;
```

After that we go back to „Menu.cpp“ and go back to our two god mode options. We now double click again onto „dwRME_Client_GodMode“, right click and select „Go to Declaration“.

```
extern DWORD dwRME_Client_GodMode, dwRME_Client_SuperSpeed, dwRME_Client_Invisibility, dwRME_Client_InfraredVision, dwRME_Client_Set
```

New place, old me. Now we are inside „Globals.h“. Should we care? We are idiots, so no. We basically do the exact same as before BUT this time we paste in only „dwRME_NewRMEOption“ at the end, before the „:“ and before that new pasted shit set a comma

```
dwRME_AllClients_TakeWeaponsAway, dwRME_AllClients_TakeAmmoAway, dwRME_NewRMEOption;
```

„Are we almost done yet, im tired…“ shut the fuck up, go back to „Menu.cpp“ and now edit in our second new god mode option our two new names in the code. While at it, rename the text that the option is displayed as inside the offhost to something different. The new code looks like this:

```
Menu("God Mode").ArrayEditor(RME_Toggle[MenuOptions::dwRME_Client_GodMode], ARRAYSIZE(RME_Toggle), (int*)&MenuOptions::dwRME_Client_GodMode).Ca
Menu("New RME Option").ArrayEditor(RME_Toggle[MenuOptions::dwRME_NewRMEOption], ARRAYSIZE(RME_Toggle), (int*)&MenuOptions::dwRME_NewRMEOption).
```

The full code is: `Menu("New RME Option").ArrayEditor(RME_Toggle[MenuOptions::dwRME_NewRMEOption], ARRAYSIZE(RME_Toggle), (int*)&MenuOptions::dwRME_NewRMEOption).Callback(ChangeGodMode, SelectedPlayer, (MenuOptions::dwRME_NewRMEOption == 0));`

If you now would already build the source again, throw the new .xex onto you console and load it up, then you already would see our new option under „Client List“ under „RME Options“
It will not work yet, we will now actually code some Custom RME function. Dont be scared, we are idiots afterall and just gonna copy/paste from existing code. So lets get to it!
In our newly made button we can see the following name „ChangeGodMode“ its the text right after „.Callback“:

```
ARRAYSIZE(RME_Toggle), (int*)&MenuOptions::dwRME_NewRMEOption).Callback(ChangeGodMode, SelectedPlayer, (MenuOptions::dwRME_NewRMEOption == 0));
```

Double click it, and do the same as we did before with the other text shit. Dont even think much about it. Just do. First go to the declaration:

```
extern void ChangeGodMode(int clientNum, bool toggle);
```

We do exactly what we done before, we ‚borrow‘ that code and paste it in right under it and just rename „ChangeGodMode“

```
extern void ChangeGodMode(int clientNum, bool toggle);
extern void NewRMEOption(int clientNum, bool toggle);
```

Now we will go to the „ChangeGodMode“ definition. Dont be scared, you will see a load of ‚scary‘ code, but we dont care, we are idiots. Just copy/paste the whole void bullshit. You know where exactly the code ends by looking at the line at the left

```
void ChangeGodMode(int clientNum, bool toggle) {
    if (!Game::CheckInGame() || !CG::bLobbyInitialized)
        return;

    g_Netchan.setup_buffer_rme();

    int iClientNum = clientNum;

    if (iClientNum == -1)
        iClientNum = Structs::get_cg()->clientNumber;

    if (toggle) {
#if defined (BO2)
        g_Netchan.write_byte((gClient(iClientNum) + 0x1B), 0x05);
#elif defined (BO1) || defined (BO1_ZM)
        g_Netchan.write_byte((gClient(iClientNum) + 0x17), 0x05);
#endif
    }
    else {
#if defined (BO2)
        g_Netchan.write_byte((gClient(iClientNum) + 0x1B), 0x04);
#elif defined (BO1) || defined (BO1_ZM)
        g_Netchan.write_byte((gClient(iClientNum) + 0x17), 0x04);
#endif
    }

    //CG_GameMessage(0, va("%s God Mode %s %s", toggle ? "^2Set^7" : "^1Taken^7", toggle ? "for" : "from", partyMember[iClientNum].gamertag));
```

Copy all that shit untill the line ends and paste it under that code. When you paste it make sure you paste it under that fucking line. That line shows you all the code that is used for that function, if you would just paste it right into the already existing funciton youd be getting a error. I hope nobody gets this shit wrong, that would be sad. We gonna paste that whole shit in right where the arrow is, make a few extra spaces by hitting enter too while your at it!

```
#endif
    }
    else {
#if defined (BO2)
        g_Netchan.write_byte((gClient(iClientNum) + 0x1B), 0x04);
#elif defined (BO1) || defined (BO1_ZM)
        g_Netchan.write_byte((gClient(iClientNum) + 0x17), 0x04);
#endif
    }

    //CG_GameMessage(0, va("%s God Mode %s %s", toggle ? "^2Set^7" : "^1Taken^7", toggle ? "for" : "from", partyMember[iClie
}
```

After we pasted it in it looks like so:

```
(Global Scope)                                   ChangeGodMode(int clientNum, b
        g_Netchan.write_byte((gClient(iClientNum) + 0x17), 0x04);
#endif
    }

    //CG_GameMessage(0, va("%s God Mode %s %s", toggle ? "^2Set^7" : "^1Taken^7", toggle ? "for" : "from", partyMember[iClientNum].gamertag));
}

void ChangeGodMode(int clientNum, bool toggle) {
    if (!Game::CheckInGame() || !CG::bLobbyInitialized)
        return;

    g_Netchan.setup_buffer_rme();

    int iClientNum = clientNum;

    if (iClientNum == -1)
        iClientNum = Structs::get_cg()->clientNumber;

    if (toggle) {
#if defined (BO2)
        g_Netchan.write_byte((gClient(iClientNum) + 0x1B), 0x05);
#elif defined (BO1) || defined (BO1_ZM)
        g_Netchan.write_byte((gClient(iClientNum) + 0x17), 0x05);
#endif
    }
    else {
#if defined (BO2)
        g_Netchan.write_byte((gClient(iClientNum) + 0x1B), 0x04);
#elif defined (BO1) || defined (BO1_ZM)
        g_Netchan.write_byte((gClient(iClientNum) + 0x17), 0x04);
#endif
    }

    //CG_GameMessage(0, va("%s God Mode %s %s", toggle ? "^2Set^7" : "^1Taken^7", toggle ? "for" : "from", partyMember[iClientNum].gamertag));
}

void GodModeEveryone(bool toggle, bool bEnemy) {
```

As you can see, above would be the old code, and under our new code we also got some other old code.that yellow line on the left a signal that we did some editing in that line. If its green then it means we did some editing and after that saved/build the project. Keep that in mind for future use.
Now rename our new function. So we will rename it from „ChangeGodMode“ to „NewRMEOption“

```
void NewRMEOption(int clientNum, bool toggle) {
    if (!Game::CheckInGame() || !CG::bLobbyInitialized)
        return;

    g_Netchan.setup_buffer_rme();

    int iClientNum = clientNum;

    if (iClientNum == -1)
        iClientNum = Structs::get_cg()->clientNumber;
```

Now go back to „Menu.cpp“ and change in our new button, the button we renamed already, the name from „ChangeGodMode“ to „NewRMEOption“. It needs to look like this:

```
on], ARRAYSIZE(RME_Toggle), (int*)&MenuOptions::dwRME_NewRMEOption).Callback(NewRMEOption, SelectedPlayer, (MenuOptions::dwRME_NewRMEOption == 0));
```

After that is done we ALMOST have a very own custom option. Now go to the definition from „NewRMEOption“ again. We will now change what memory adress we will edit after we use our new option.

```
    if (toggle) {
#if defined (BO2)
        g_Netchan.write_byte((gClient(iClientNum) + 0x1B), 0x05);
#elif defined (BO1) || defined (BO1_ZM)
        g_Netchan.write_byte((gClient(iClientNum) + 0x17), 0x05);
#endif
    }
    else {
#if defined (BO2)
        g_Netchan.write_byte((gClient(iClientNum) + 0x1B), 0x04);
#elif defined (BO1) || defined (BO1_ZM)
        g_Netchan.write_byte((gClient(iClientNum) + 0x17), 0x04);
```

The marked code is THE ONLY code that is important for us. „0x1B“ is the memory address we edit and „0x05“ and „0x04“ are the values that we set to that address. It is also important to know what type of value we edit, the code „`g_Netchan.write_byte`“ is important for that. To be exact, we only care about what comes after the „_“. We can edit bytes, int, float and so on. Most of the time it will be enough to use „_byte“ or „_int“. Figure it out yourself what you should use by playing around with it.
As we wanna do a custom option we need to first find some address that would do something cool. How? Read my fucking other guide for it. In short: use peak poker and go to some address in rme range, it starts at „0x83551A10“, then just blindly poke values to some other value and see what happens, if something cool changes then write that address down and to what you set it because you just found something CUSTOM. Then either calculate by hand the „restToOffset“ or use my calculator which is in the same repo as the guide for it was.

Lets say we found something cool at the follwoing address: „0x83551ED2“
Turns out we can disable sprint by setting the value that is in that address to „0xFF“
Cool, isnt it? Try it out in peek poker by poking that address to „FF“ and you wont be able to sprint untill you die! If we now use either basic math, or the simple calculator I did, we get the following value for restToOffset → „0x4C2“
This is the address we need to replace „0x1B“ for inside our code.

So after replacing the old value to our new one the code would look like this:

```
void NewRMEOption(int clientNum, bool toggle) {
    if (!Game::CheckInGame() || !CG::bLobbyInitialized)
        return;

    g_Netchan.setup_buffer_rme();

    int iClientNum = clientNum;

    if (iClientNum == -1)
        iClientNum = Structs::get_cg()->clientNumber;

    if (toggle) {
#if defined (BO2)
        g_Netchan.write_byte((gClient(iClientNum) + 0x4C2), 0x05);
#elif defined (BO1) || defined (BO1_ZM)
        g_Netchan.write_byte((gClient(iClientNum) + 0x17), 0x05);
#endif
    }
    else {
#if defined (BO2)
        g_Netchan.write_byte((gClient(iClientNum) + 0x4C2), 0x04);
#elif defined (BO1) || defined (BO1_ZM)
        g_Netchan.write_byte((gClient(iClientNum) + 0x17), 0x04);
#endif
    }

    //CG_GameMessage(0, va("%s God Mode %s %s", toggle ? "^2Set^7" : "^1Taken^7
```

Now we need to edit the value after that, the „0x05“ and „0x04“ to „0xFF“ like so:

```
    if (toggle) {
#if defined (BO2)
        g_Netchan.write_byte((gClient(iClientNum) + 0x4C2), 0xFF);
#elif defined (BO1) || defined (BO1_ZM)
        g_Netchan.write_byte((gClient(iClientNum) + 0x17), 0x05);
#endif
    }
    else {
#if defined (BO2)
        g_Netchan.write_byte((gClient(iClientNum) + 0x4C2), 0xFF);
#elif defined (BO1) || defined (BO1_ZM)
        g_Netchan.write_byte((gClient(iClientNum) + 0x17), 0x04);
#endif
    }
```

And thats it. Your done. What this code now does is it will set from a certain player, we select from the client list, that certain memory address to our new value. In our case we will disable by that the sprint for that player we use the option on. Wont work on yourself if you host.
So congrats, you now did your first ever custom option! Build it and try it out.
Now you can start to experiment yourself with how stuff works. I wont be feeding you even more shit, find it out yourself if you care. And please, dont be a faggot by doing malicious/annoying shit...

Me knowing that it could lead to malicious RME options after teaching everyone how to do it themselves:

FAQ

How to load my offhost after im done?
Find the .xex from the offhost your done and put the .xex file onto your console. Then use „Module Loader“ to inject the .xex into your console.

I cant inject my .xex
Be sure to set your console as Default in „Xbox 360 Neighborhood“.

Where do I get that „Module Loader“ thing?
Did you ever hear about that new thing called „Google“? Stop beeing a lazy retard.

Where the heck is that calculator you talk about?
In my github repo: https://github.com/XxKarl666xX/XBOX-360-BO2-RME-Guide

The Menu looks like shit, change please
Do it yourself, lazy prick.

Isnt this beeing a skid?
Your the one who came to this guide, so shut the fuck up and have some fun while experimenting with the game.

Why do you give people the ability to be total skids and still ruin the game further?
My hope is that most wont use it to do annoying shit, but rather to create cool/fun shit.

Can we have another meme?