

Getting around the system - map making tips for halo ce

By Dustin Vaughan a.k.a. {AHS}SONIC

Hello to all and thank you for reading my article. This will take you through a few steps on making more advanced maps with more advanced content. I will spill most of my knowledge of basic issues into this article, so that others may learn from it. This covers basic issues, but requires you to have some knowledge of map making. If you want to know how to make a map, this article is not for you.

Part 1. Vehicles

Vehicles can be tricky and hard to use in general but here a few ways to make using them in a map less irritating. Having a cool vehicle in your map always spices it up, and they are usually what attracts players. This will go more in-depth of using vehicles, not so much creating them.

→They don't show up in the map.

Incredibly easy to fix. Simply find the source folder of your vehicle (i.e. tags\vehicles\jet) And find the jet.vehicle file or whatever vehicle you are using and rename it to something like mp_warthog.vehicle or banshee_mp.vehicle. Keep in mind that you **MUST** add the .vehicle to the name of the file or else guerrilla, tool, and sapien will not be able to recognize the file, and you must re-add the vehicle with the new file name to sapien palette. After renaming the files, copy all the contents of your vehicles root folder into the vehicles folder you named the original vehicle after. If you named your vehicle mp_warthog.vehicle then you should move all the tags to the warthog folder.

→They show up in the map, but few or no seats in the vehicle work.

Little bit harder to fix. If you extracted the vehicle using hek+ then you should go back and extract the biped used in the map as well. (Usually found in the characters folder.)After extracting your tags open globals.globals in guerrilla and scroll to around the middle and find a section called **multiplayer information**. There should be a box with three dots and the text field will say: characters\cyborg_mp\cyborg_mp. Browse for the biped tags you extracted by clicking the three dots. If you did not extract your tags then check out Tiamat's ncyborg tags posted on this site. They will make it easier to use vehicles and make you get fewer exceptions.

→The first method for placing vehicles works, but I want more than 2 vehicles.

The file names of all vehicles used in halo multiplayer read as follows: mp_warthog, banshee_mp, C_gun_turret_mp, scorpion_mp, ghost_mp, and rwarthog_mp.

→The first method works, but I'm getting tired of constantly reinstalling the hek to get the original tags back.

Open guerrilla and open globals.globals. Scroll down to multiplayer info and right above it you will see a vehicles section. On the little pull down menu, select the vehicle you would like to replace with your own, and click the three dots to browse for it. You don't

have to rename the vehicle to do this, and after replacing (not adding you can only have 6 vehicles at a time using this method) your vehicle will show up in any map you put it in.

→Sapien places the vehicle, tool compiles fine, but halo exceptions when I try to play the map.

This is usually caused by an overabundance of polygons in the map you are using. Try putting the tags in another map and attempt to play it, and if halo still gives you crap, reinstall the hek. (Helpful hint: always keep a backup halo editing kit installer) If you still get exceptions, then try deleting all the vehicles in your map, including the ones in the palette, compiling your map, and play it. After playing re-place all the vehicles in your map and hope it works.

Part 2: **Weapons**

This part of the article will help you use and create weapons, which can and will be difficult but honestly I think it's overall pretty darn cool to see a weapon made by yourself in a map. I won't go very in-depth on making everything for a gun, but I will show you the basics on getting a playable weapon into the game.

→Making your fire stick:

I hope you have 3ds max 6-8 and basic knowledge of modeling or else you have no purpose reading this article. If you do, congratulations! Open it up and make a small sphere named frame gun. Move it anywhere you want, preferably at the coordinates 0, 0, 0 (X, Y, Z). Now make a long rectangular box, tessellate the bottom faces a bit and extrude the faces (or polygons) make a handle for the Master Chief to grab, maybe even two if you think you can handle it. Copy the frame gun sphere and name it #right hand. Then copy that sphere and name it #left hand. Now move the spheres where you want the chief to hold the gun, like on the handle. Right hand to the handle, left hand up a little more to hold the end of the gun. Open the material editor and apply materials as you see fit. Now make another sphere, and name it #primary ejection. Put this sphere where you want the bullets to come out of the gun. Next, make another sphere and name it #ground point. Put it somewhere towards the bottom of the model. Link all the objects besides frame gun to frame gun. (It would be hard to link something to itself in max). For this next part we are going to use Theghost's gbxmodel importing script. You can get it off of halomaps or you can google it (helpful hint: when in doubt, google it!) Before we do anything else hit ctrl+a in 3ds max and right click anywhere. Select hide selected. In 3ds max, look at the top bar and you should see a menu entitled max script. Click it and click run script. Browse for the location you saved the gbxmodel importing script and double click it. Click the load gbxmodel button and browse for shotgun.gbxmodel, or if you are using a less advanced version of the script, open shotgun.weapon in guerrilla, click export, and save the file to the halo ce root folder. Anyways, open the file in the script and click import model. Now delete everything in the scene except the model named _unnamed. Now right click anywhere in the scene and select unhide all. You might notice a bit of a size difference between your model and the shotgun. Right click your model and select scale. Make your model bigger or smaller as you see fit, preferably to match the size of the shotgun. After you are done, delete _unnamed. You might have to setup your materials again. Now we are going to use the blitzkrieg exporter to export the file three times: once for a model, once for collision geometry, and once for a first person

model. Open the halo ce root directory folder and then the data folder. Make a folder called gun. (Subtle, isn't it?) Inside gun, create a folder called models, a folder called physics, a folder called fp, and inside the fp folder, create a folder named models. In max, go to file, then export, and export a model into each of the folders we just made. When you export into the physics folder, name the model gun_collision. I have no idea why, I guess it's just to play it safe or something. Time for the worst part of this process, the dreaded tool. Open cmd and type in: tool model gun. Give it awhile to compile (holy crap that rhymes!); this can take a few minutes depending on your computer. After tool is done making your model, type in: tool collision-geometry gun. If it gives you any errors, try making a simpler weapon in 3ds max and try again. After collision geometry, type in tool: tool model gun\fp. After tool has made all these models (and I hope it did without giving you any problems) open guerrilla and open shotgun.weapon. (If you have another weapon you would like to mod instead then by all means, use it). At the very top of the file there will be a few text boxes entitled: model, animation graph, and collision model. For model, click the three dots and browse for the model we just made in tool. It will be called gun.gbxmodel. Leave the animation graph as it is. For collision model, browse for the model we made in tool. It will be called gun.model_collision_geometry. Scroll down and find interface. Replace first person model with the fp model we made. Click the three dots and browse into your guns folder. If we made the model right there should be a folder called fp inside our guns folder. Go into it and open fp.gbxmodel. Place your tag in the map and enjoy.

Part 3: **Bitmaps and Shaders**

Making your textures appear in a map is of the utmost importance, because a plain, white map is not very interesting. This section of the article will tell you how to make, use, compile, and replace bitmaps and shaders.

→I want to change a texture in my map without recompiling it and losing all my work I used this trick on the Hugsnap map, and learned that you should use textures that are equal in size to each other, or else you will get horrible distortions. Ill be using sidewinder and bloodgulch to show you how to switch ground textures. First you will need hek+. Boot it up and extract all of bloodgulch and all of sidewinder. Open guerrilla, go to file and click open. Browse into the bloodgulch folder (tags\levels\test\bloodgulch) and open the shaders folder within the bloodgulch folder (man I've been saying bloodgulch a lot). In the shaders folder, open bloodgulch_ground.shader_environment. Scroll down a ways until you see base map. Now click the three dots and browse into the sidewinder folder. Open bitmaps, and open sidewinder_ground.bitmap. Scroll down some more and make sure there are no other bitmaps currently loaded in the shader file. Save the tag, compile your map, and have fun with your newly textured map. You might want to change the background sound and add some snow to the weather to make the mood just right.

→I enter the shader number in tool but it says "failed to build model" or gives an error of some sort.

Simply push the up arrow key and hit enter, and it should compile the map this time.

→I make a shader file, but when I play my map, the texture shows up as white. Simply open the shader in guerrilla, and find the base map section of the shader file. Browse for your bitmap.

→I don't know how to make a bitmap or tool gives me errors when I do. All image files MUST be in the tagged image file format (tiff, tif). If they are, and tool still won't compile, open them up in paint shop and click new in the file menu. make it the dimensions you think would fit the texture you are trying to get in the game, and make it 16 million colors (24 bit) Go over to the picture you want to get in the game and right click it. Select copy, and go over to the new image file we just made. Right click it and select paste as new layer. Make a folder in the data folder inside the halo ce root directory named bitmap and save the new image to that folder or whatever folder you want to save it in. open tool and type in: tool bitmaps bitmap or whatever folder you saved it in. you now have a bitmap file.

Part 4: Sound

This section of the article will address many issues related to creating sound in halo.

→I want to make a sound in halo

Good, me too. First you will need a program called goldwave, which you can download a trial of for free. Second, you will need the Xbox sound codec (when in doubt, google it!). After you have obtained these tools, open your sound in goldwave. Save it as a wav file, PCM signed, 16 bit stereo into the data\sound folder of halo ce. If there is no sound folder, it would be a good idea to make one. After saving your sound, open tool and type in: tool sounds sound {your file path here} Xbox 1. Hit enter. Please note that if you edited the sound in anyway using goldwave or some other program, then it will void the sound file and will refuse to compile in tool. If tool says importing sound {your sound name} then it is working, but if it goes on for less then 3 seconds, it most likely failed. Also, if the file size of the .sound file (which can be found in tags\sound) is less than or is 1 kb, then it failed to create the file.

Part 5: Scripts

This part of the article will explain scripting, and help with getting a script into the game.

→What is a script and what does it do?!

A script is basically a small program that can be embedded into any halo map, and will allow certain things to take place in the game, mainly cut scenes and music, but it can also tell certain elements in halo what to do and when to do it, like in the level the Truth and Reconciliation, after you kill all the covenant in the hanger of the ship, the doors will open and hunters come out. This is all operated via script, and if you are a heavy devmode user, most of the cheats and codes you use are actually script expressions and commands.

→How do I make a script?

A script can be made by typing up a text file and putting in the data (not tags folder) of your level, such as data\levels\b40\scripts. All text files must be saved as .hsc before being compiled in sapien. To compile the scripts in sapien (make them work in the map), simply click file\compile scripts. If you get any errors or sapien does not say “scripts successfully compiled”, there isn’t much I can do to help you. But try reading what the top line of the error says; don’t pay attention to the “left parenthesis is unmatched” thing unless the top line says that.

→How big can a single hsc file be before I have to make another one?

I haven’t really made any scripts that big, but the max file size limit to a script before you must make another one is 150 kb.

→What does a script look like?

Here is a basic script that will add a surprise to your map. It does not require you to add anything to the map. Copy and paste this code onto a fresh text document, preferably created using wordpad or notepad:

```
(script startup cordiostart
(begin
  (cinematic_show_letterbox true)
  (show_hud false)
  (fade_in 1 7 200)
  (player_enable_input false)
  (sleep 200)
  (sv_say "Welcome to CordioVexa")
  (sleep 200)
  (sv_say "Fight")
  (show_hud true)
  (cinematic_show_letterbox false)
  (player_enable_input true)
)
```

Thanks to Strider Vega Core for the script.

This concludes the article on Halo CE basics, I hope to release a second, but only if I can find the time for it.