Section 1: Required tools part 1

Now that we know that the file we actually edit to create a level mod is the .entities file, we will need a way to edit it.

And not just edit it, but find a way to edit it on the fly so we don't need to re-compress and re-inject the file after each and every change.

It is important to note a few, crucial details, each of which will have a corresponding tool or helpful asset to help with that.

1. As you have seen in the last page, the .resources files aren't exactly openable. We will need to extract the .entities file from them to edit it.

2. If you were to open the .resources files of a map, including its _patch versions, you may notice multiple ones have their own .entities file. What gives?!

3. This .entities file is actually just text and can be opened with any text editor. Unlike id Software who uses their own software called idStudio to edit levels in a 3D environment, we are left with 2d text. An .entities file purely consists of defined entities, or entitydefs. Entitydefs can be anything from triggers to encounter managers to pickups.

Example 1: An entity that is defined to be a spawn point for a marauder.

```
entity {
	entityDef encounter_5_marauder1 {
	inherit = "target/spawn";
	class = "idTarget_Spawn";
	expandInheritance = false;
	poolCount = 0;
	poolGranularity = 2;
	networkReplicated = false;
	disableAIPooling = false;
	edit = {
		{  The stuff that usually goes here }
	}
}
}
```

Example 2: An entity that is defined to be shotgun ammo.

```
entity {
	entityDef pickups_pickup_ammo_shells_26 {
```

```
	inherit = "pickup/ammo/shells";
	class = "idProp2";
	expandInheritance = false;
	poolCount = 0;
	poolGranularity = 2;
	networkReplicated = false;
	disableAIPooling = false;
	edit = {
		[ The stuff that usually goes here]
	}
}
}
```

4. If you ever tried to open the .entities file within an existing level mod, you may notice that it's made up of nonsense weird characters. This is because the file is compressed. Decompressed files are the plainly text versions you see, but are too large to replace an existing .entities file, which is why they are compressed before being put into a level mod.

5. Assuming you want to spawn in an enemy on a brand new spawn point, you do not have any readily available way to have your current coordinate match the file's formatting.

Example 1: A basic spawnposition and spawnorientation. Does this look like you can manually type it out by hand, every single time?

```
spawnOrientation = {
	mat = {
		mat[0] = {
			x = 0.707107;
			y = -0.707107;
			z = 0.000000;
		}
		mat[1] = {
			x = 0.707107;
			y = 0.707107;
			z=0.000000;
		}
		mat[2] = {
			x = -0.000000;
			y = 0.000000;
			z = 1.000000;
		}
	}
```

```
}
spawnPosition = {
	x = -99.080002;
	y = -278.399994;
	z = 22.00265;
}
```

If these details made no sense to you, don't worry. I will provide each tool and let you know how they work in setting up your foundation to edit a level with.

They are:

1. SAMUEL to extract the .entities file from the .resources files.

Download link for SAMUEL: GitHub

2. A list of a map's current highest priority so you know *which* .resources file to extract from and subsequently edit. The game uses the .entities file from the map's highest priority patch. If you choose the wrong one and have your mod replace the .entities file in the wrong patch, it won't work.

Link to the list: Google Docs

3. NotePad++ to edit the .entities file with. You may also use EntitySlayer, a program that is specifically designed to work with these files, but this guide will not use it as I have no experience with it.

Download link for NotePad++: Official website

Download link for EntitySlayer: GitHub

4. idFileDeCompressor. You will not need to use this tool to decompress extracted .entities files as SAMUEL neatly gives them to you already readable and editable, but you will need it to re-compress them when packing them in the final .zip file in order to let level mods use it without giving you a memory error. EntitySlayer also has an option to compress files without the need of a dedicated compressor tool.

Download link for idFileDeCompressor: GitHub

5. meathook.dll. This tool is so useful for level modding that without it, I would say at least 99% of all master level mods would not exist today because they would've taken too long to make, ludicrously long. Meathook allows for on the fly editing and without it level modders would've been stuck re-compressing and re-injecting their edited .entities file over and over.

Download link for v7.2: Direct download

Next up, we will go over how to use these tools to set up your foundation for making a level mod.

Revision #26
Created 6 February 2022 21:07:14
Updated 25 July 2024 17:39:29 by PowerBall253