Section 2: idAI2s and spawns

Before we edit an encounter, we need to know about the behind-the-scenes elements going on.

Specifically, this section will cover idAI2s and spawns.

idAI2s

Earlier, I said that:

> An .entities file purely consists of defined entities, or entitydefs. Entitydefs can be anything from triggers to encounter managers to pickups.

An idAI2 is an entity for a specific demon.

What do I mean by that? Let's take a look at an idAI2 for an Imp.

```
entity {
	layers {
		"spawn_target_layer"
	}
	entityDef ai_custom_imp {
	inherit = "ai/fodder/imp";
	class = "idAI2";
	expandInheritance = false;
	poolCount = 0;
	poolGranularity = 2;
	networkReplicated = true;
	disableAIPooling = false;
	edit = {
		[ The stuff that usually goes here ]
	}
	}
}
```

You can CTRL + F "ai/fodder/imp" to see what's usually in the edit section. It consists of important but irrelevant settings so I will not be including it here.

Think of idAI2s as the demon itself. These entities, however, will not do anything if you were to copy + paste a new one into your file. They need to be *spawned* in order to exist as you typically see them in-game.

How can we spawn them? We will get to that in this next section:

idTarget_Spawns

idTarget_Spawns are exactly what you think they are: they're spawn points for demons.

The catch is though: they must reference an idAI2 in some way.

> I cannot stress this enough, a target spawn *must* reference an idAI2 in some way, either by itself or by being part of a spawn zone that references a spawn parent that references idAI2s.

Making a spawn point is easy: copy + paste an existing one and edit the four important things:

- entitydefs
- aistateoverride
- spawnOrientation
- spawnPosition

It is these four, *and only these four*, that matter.

Here is an example of a custom spawn point, notice how it references our Imp idAI2 from earlier. I have also marked the other important things to edit with comments.

```
entity {
	entityDef custom_imp_spawn_1 {
	inherit = "target/spawn";
	class = "idTarget_Spawn";
	expandInheritance = false;
	poolCount = 0;
	poolGranularity = 2;
	networkReplicated = false;
	disableAIPooling = false;
	edit = {
		flags = {
			noFlood = true;
		}
```

```
□□spawnConditions = {
□□□maxCount = 0;
□□□reuseDelaySec = 0;
□□□doBoundsTest = false;
□□□boundsTestType = "BOUNDSTEST_NONE";
□□□fovCheck = 0;
□□□minDistance = 0;
□□□maxDistance = 0;
□□□neighborSpawnerDistance = -1;
□□□LOS_Test = "LOS_NONE";
□□□playerToTest = "PLAYER_SP";
□□□conditionProxy = "";
□□}
□□spawnEditableShared = {
□□□groupName = "";
□□□deathTrigger = "";
□□□coverRadius = 0;
□□□maxEnemyCoverDistance = 0;
□□}
□□entityDefs = {
□□□num = 1;
□□□item[0] = {
□□□□name = "ai_custom_imp"; // the idAI2 is referenced in here
□□□}
□□}
□□conductorEntityAIType = "SPAWN_AI_TYPE_ANY";
□□initialEntityDefs = {
□□□num = 0;
□□}
□□spawnEditable = {
□□□spawnAt = "";
□□□copyTargets = false;
□□□additionalTargets = {
□□□□num = 0;
□□□}
□□□overwriteTraversalFlags = true;
□□□traversalClassFlags = "CLASS_A";
□□□combatHintClass = "CLASS_ALL";
□□□spawnAnim = "";
□□□aiStateOverride = "AIOVERRIDE_TELEPORT"; // this specifies that this ai will spawn in
```

```
alongside the red teleport FX you see
□□□initialTargetOverride = "";
□□}
□□portal = "";
□□targetSpawnParent = "";
□□disablePooling = false;
□□spawnOrientation = { // this is where the ai will face when it's spawned. it is ignored if
the ai is teleported in.
□□□mat = {
□□□□mat[0] = {
□□□□□x = -1.000000;
□□□□□y = -0.000000;
□□□□□z = -0.000000;
□□□□}
□□□□mat[1] = {
□□□□□x = 0.000000;
□□□□□y = -1.000000;
□□□□□z=0.000000;
□□□□}
□□□□mat[2] = {
□□□□□x = -0.000000;
□□□□□y = 0.000000;
□□□□□z = 1.000000;
□□□□}
□□□}
□□}
□□spawnPosition = { // this is the spawn position of the ai
□□□x = -30.209999;
□□□y = -11.000000;
□□□z = -10;
□□}
□}
}
}
```

Everything else in the target spawn can, and should, be ignored.

You can use the same idAI2 for as many target spawns as you want. For example I can very much have every single new spawn point I make that I want to support an imp all use "ai_custom_imp".

This is how idAI2s and idTarget_Spawns work with each other. The idAI2 is just an entity for a specific demon that target spawns will take and place wherever you want, however many times you want.

Before going into encounter editing, we ought to go over two more entities that use idTarget_Spawns, idTargetSpawnGroups and idTarget_Spawn_Parent, and how they are used in an encounter as well.

After that, we will go over incorporating all of this into an encounter in the next page, including how to obtain usable positions for our custom spawn points.

idTargetSpawnGroups

Earlier, you may have noticed that I said target spawns must reference an idAI2 in one of two ways, either by itself (which you have been given an example of above) or by being part of a spawn group that references a spawn parent that references idAI2s.

Spawn groups are special entities that are a set of multiple target spawns. On their own, they may sound useless (why bother when I want to spawn individual demons in?) but they are crucial for things like maintained AI (which you must have for respawning fodder) or if you want to just spawn many demons at once without having a whole lot of spawnSingleAIs.

Here is an example of an idTargetSpawnGroup below:

```
entity {
	entityDef custom_spawn_group_1 {
	inherit = "encounter/spawn_group/zone";
	class = "idTargetSpawnGroup";
	expandInheritance = false;
	poolCount = 0;
	poolGranularity = 2;
	networkReplicated = false;
	disableAIPooling = false;
	edit = {
		spawnPosition = {
			x = 0;
			y = 0;
			z = -1000;
		}
		renderModelInfo = {
			model = NULL;
		}
		clipModelInfo = {
			clipModelName = NULL;
		}
```

```
□□spawners = {
□□□num = 4;
□□□item[0] = "custom_ai_spawn_1";
□□□item[1] = "custom_ai_spawn_2";
□□□item[2] = "custom_ai_spawn_3";
□□□item[3] = "custom_ai_spawn_4";
□□}
□□targetSpawnParent = "custom_spawn_parent";
□}
}
}
```

Individual target spawns in these groups may have zero entitydefs defined, for so long as the spawn parent contains the idAI2 you wish to use.

What did I just mean by *that?* Let's see an example of a spawn parent:

idTarget_Spawn_Parents

Much like how a spawn group is a collection of target spawns, a spawn parent is a collection of idAI2s. Groups and parents go hand in hand to allow for more flexibility when placing down demons in an encounter.

Here is an example, pretend these idAI2s are already present in your file.

```
entity {
□entityDef custom_spawn_parent_1 {
□inherit = "encounter/spawn_group/parent";
□class = "idTarget_Spawn_Parent";
□expandInheritance = false;
□poolCount = 0;
□poolGranularity = 2;
□networkReplicated = false;
□disableAIPooling = false;
□edit = {
□□flags = {
□□□noFlood = true;
□□}
□□spawnConditions = {
□□□maxCount = 0;
□□□reuseDelaySec = 0;
```

```
□□□doBoundsTest = false;
□□□boundsTestType = "BOUNDSTEST_NONE";
□□□fovCheck = 0;
□□□minDistance = 0;
□□□maxDistance = 0;
□□□neighborSpawnerDistance = -1;
□□□LOS_Test = "LOS_NONE";
□□□playerToTest = "PLAYER_SP";
□□□conditionProxy = "";
□□}
□□spawnEditableShared = {
□□□groupName = "";
□□□deathTrigger = "";
□□□coverRadius = 0;
□□□maxEnemyCoverDistance = 0;
□□}
□□entityDefs = {
□□□num = 2;
□□□item[0] = {
□□□□name = "ai_custom_imp";
□□□}
□□□item[1] = {
□□□□name = "ai_custom_pinky";
□□□}
□□}
□□conductorEntityAIType = "SPAWN_AI_TYPE_ANY";
□□initialEntityDefs = {
□□□num = 0;
□□}
□□spawnPosition = {
□□□x = 17.0000057;
□□□y = 1209.99939;
□□□z = 185.500961;
□□}
□□targets = {
□□□num = 2;
□□□item[0] = "ai_custom_imp";
□□□item[1] = "ai_custom_pinky";
□□}
□}
```

```
}
}
```

When I say flexibility, I mean that when I use these two, I can change the ai spawned in an encounter on the fly without having to go into their individual spawn points and change the linked idAI2.

Instead, hypothetically, every spawn point can be part of a spawn group that is linked to a spawn parent that contains the idAI2s of every ai supported in the level (if DLC ai is not supported in the level, adding a DLC idAI2 into the file will not make it work on its own.)

You can reference the same spawn parent multiple times, so I use a "master parent" that does indeed contain every supported idAI2 in the specific level I'm working on at the time.

When you're spawning in individual demons in an encounter, you may choose to forgo putting in specific idAI2s in your spawn points in favor of this system if you so choose, but it is mandatory when having respawning enemies or spawning multiple enemies at once in its own specialized function rather than having multiple spawnSingleAIs.

Next step, we will bring our knowledge of idAI2s, idTargetSpawns, idTargetSpawnGroups, and idTarget_Spawn_Parents to finally be able to edit an encounter in a way you want to.

Created 2022-02-08 15:11:50 PST
Updated 2022-04-18 19:12:21 PDT