**CustomSpawner USER GUIDE v0.2**

Due to the split between CMS and MoCreatures in 6.1.0+, the configs have been completely changed and should be completely wiped out to allow it to regenerate it fresh. If you install MoC without CMS, it will use the default Vanilla Spawner and ONLY allow you to alter canspawn, frequency, minspawn, maxspawn settings. You will NOT be able to adjust other spawn settings such as spawn rates since MoCreatures by itself is not a Custom Spawner and will not affect Vanilla Spawner core functionality in any way.

If you install CMS, the MoC GUI option for Spawn settings will disappear since everything will now be controlled by CMS(which has no GUI currently).

*IMPORTANT*: CMS will use the default Vanilla settings for Vanilla entities(Sheep, Cow, Zombie, Creeper, etc.). It is important you adjust the config settings through the /cms commands or configs directly as some of these default vanilla settings are high. One such setting is frequency, be sure to lower all vanilla default frequencies or they will spawn much more than any other mod entity.

**Environments**

The big feature in CMS 3.0.0+ is the ability to configure world environments. Every world in minecraft is created using an environment. Environments get created in

./config/CustomSpawner/.

For example, there are 3 main environments in Vanilla which are : Overworld, Nether, and End.

When you navigate to ./config/CustomSpawner/ you should see the following 4 folders :

overworld, nether, end, and logs. The ‘logs’ folder contains all relevant logs for all environments. Each environment generates its own log file. Turning on debug in EnvironmentSettings.cfg will output much more information to these logs.

All spawning settings in CMS are isolated to the specific environment. So what this means is, if you alter spawn tick rates for overworld, it will not affect any other environment such as Nether and End. If you install dimension mods, you may get extra environments that were made by the author of the mod. One such mod is TwilightForest which adds one dimension using the environment "Twilight". A new folder ./config/CustomSpawner/Twilight will be created for all environment settings.

Now what happens when you install a mod that allows you to add multiple dimensions with the same environment? Well one such mod is Mystcraft and this would use the environment "Myst". Every single dimension that myst creates will be controlled by the same myst environment config. So you could have 100 dimensions in your world based off Myst and all would use the same Myst environment settings. If you run a server using MCPC+, you may be using a multiworld plugin such as MultiVerse, MyWorlds, or MultiWorld. These plugins allow you to create new worlds based on a specific environment. So if you created a second overworld for mining, this new world would use the same overworld environment settings as the main one.

Note: I plan to add dimension overrides soon which will allow you to create sub dimension configs in an environment if needed.

**EntityBiomeGroups**

This config is practically the same as the old BiomeGroups.cfg but has been renamed to EntityBiomeGroups.cfg. As CMS scans vanilla biome spawn lists, it will create a biomegroup for each entity that it finds and add every biome it sees with it. So as an example :

S:MC\_WOLF\_DEFAULT <MC|Forest:MC|Taiga:MC|ForestHills:MC|TaigaHills:BOP|Wasteland>

The format of this line is as follows :

* S is the type of input this line takes, in this case S = String.
* : is the seperator(delimiter).
* <> is an array of type, in this case array of Strings
* | is the seperator between a tag and biome name.

So if we continue with the previous example :

MC\_WOLF\_DEFAULT represents the name of the biome group

MC|Forest represents the Forest biome of Vanilla(Minecraft)

BOP|Wasteland represents the Wasteland biome of Biomes O' Plenty.

and so on...

The entire line shows that Vanilla's Wolf was found in vanilla biome's Forest, Taiga, ForestHills, TaigaHills, and BOP's Wasteland. 'MC' is the default tag used to signify Vanilla while 'BOP' is the default tag used to signify Biomes O' Plenty. For CMS, this line basically tells it that any entity using the group "MC\_WOLF\_DEFAULT" will be added to biomes Forest, ForestHills, Taiga, and TaigaHills. By default, MC's Wolf would already have this as its group in ./config/CustomSpawner/[environment]/Creatures/Vanilla.cfg. The group name would show up as “WOLF\_DEFAULT” as it doesn’t need to include the Mod tag. The reason for using Mod tags before the group name is to prevent duplicate group names from multiple mods. If you change the group name of any entity in Creature configs, be sure to leave out the mod tag and only add it in EntityBiomeGroups.cfg. So for example, if you wanted to change MC’s Wolf default biome group from “WOLF\_DEFAULT” to “MYWOLFGROUP”, you would add “MC\_MYWOLFGROUP” to EntityBiomeGroups.cfg.

**Mod Creature Configs**

Another config that gets generated is each mod's creature config. When an entity is found in a specific biome spawn list, CMS determines what mod it came from and either generates a fresh creatures config for it or adds to an existing one. So as an example, all MoCreature entities would be found in ./config/CustomSpawner/[environment]/Creatures/MoCreatures.cfg.

As for the config use itself, before an environment is loaded, all biomes that are part of the environment get populated with spawns. The way this is done is first CMS reads each mod's creature config in the environment. So if overworld is being populated, CMS would be reading from the folder ./config/CustomSpawner/overworld/Creatures/\*.cfg. Here is how the layout would be for each entity :

Lets go over each property, the first

* S:biomegroups - Contains a list seperated by ':' of all biome groups that this entity should spawn in.
* B:canSpawn - Toggles whether or not this entity should be allowed to spawn
* I:frequency - Sets the frequency for the spawn list entry. The higher this is, the greater chance it has to be picked during spawning.
* I:maxChunk - Sets the max pack size for this entity in a specific chunk.
* I:maxLightLevel - Sets the max light level this entity should spawn in.
* I:minSpawn - Used during world gen spawning to determine the minimum amount of spawns for this entity in a chunk.
* I:maxSpawn - Used during world gen spawning to determine the maximum amount of spawns for this entity in a chunk.
* I:minSpawnHeight - determines how low this entity can spawn. Height = Y
* I:maxSpawnHeight - determins how high this entity can spawn.
* S:opaqueBlock - \*Not currently working\* Basically determines if this entity can spawn on an opaque block or not.
* S:spawnBlockBlacklist <> - \*Not currently working\* Basically determines what blocks this entity should NOT be allowed to spawn on. The format is ID-meta, if only ID is used the it assumes all meta.
* S:type - the Entity Spawn Type used for the entity. Vanilla default types are AMBIENT, WATERCREATURE, CREATURE, MONSTER. CMS allows you to create your own types in EntitySpawnTypes.cfg. Types have their own spawn settings such as tick rates etc.

**EntitySpawnTypes**

Another config generated in each environment is the EntitySpawnTypes config. This config by default will contain 4 vanilla types, and 1 custom added type. The types are as follows : AMBIENT, CREATURE, WATERCREATURE, MONSTER, UNDERGROUND.

The default spawncaps are as follows :

Creature - 35

WaterCreature - 5

Ambient - 15

Monster - 70

Underground - 15

The default spawn tickrates are as follows:

Creature - 400

WaterCreature - 100

Ambient - 100

Monster - 1

Underground - 400

Lets look over the config for one

* D:chunkgenspawnchance - The chance that a worldgen spawn can occur in a biome. Default is 0.1 which is 10%.
* I:maxspawnheight - The max spawn height allowed for ALL entities tied to this spawn type. Default is 256.
* I:minspawnheight - The mininum spawn height allowed for ALL entities tied to this spawn type. Default is 0.
* S:shouldseesky - Toggles whether or not this entities of this spawn type require sky to be seen in order to spawn. Default true for all types except undeground.
* I:spawncap - The spawn cap for the type. The higher the spawn cap, the more spawns.
* I:spawntickrate - The spawn tick rate for type. The higher the tick rate, the less spawning. Lower tick rate if you want more frequent spawns.

**Commands**

CMS adds a new command '/cms' which allows you to control many spawner configuration options in game. An important thing to know about this command is, it will only affect the environment that you are in. So if you enter commands in overworld to alter zombie settings, these settings will not change in any other environment such as nether.

Here are some examples of command usage :

* /cms frequency TAG|EntityName intvalue

This would alter the frequency of a specific entity, you need to know the mod tag it belongs to and the name. So if you want to adjust the frequency of Vanilla Zombies, the spawn tag is ‘MC’ and the name is ‘Zombie’, the command would look like this

/cms frequency MC|Zombie 8

This adjusts the zombie frequency to 8 which takes effect immediately.

* /cms canspawn TAG|EntityName [true/false]

This allows you to disable a specific entity. To disable spawning of zombies, you would enter

/cms canspawn MC|Zombie false

* /cms killall TAG|EntityName

The killall command allows you to wipe out a specific entity in all dimensions. This command includes logic to prevent killing tamed entities but it is not guaranteed that it will prevent all custom tamed mod entities. To wipe out all Ogres in the mod MoCreatures, you would type the following

/cms killall MOC|Ogre

If you want to wipe out ALL entities including tamed from all dimensions, you would enter the following

/cms killall TAG|EntityName force

* /cms countentities

The countentities command counts all entities in current dimension and displays them in

a paged list. In order to access a specific page you would type

/cms countentities page#

* /cms spawncap EntitySpawnType intvalue

The spawncap command allows you to adjusts the total weight amount of a given entity type.

If you are experiencing low amount of creatures, you could raise the cap by entering the following

/cms spawncap CREATURE 150

* /cms spawntickrate EntitySpawnType intvalue

The spawntickrate command allows you to adjust the tick rate of a specific spawn type.

For example, if you are experiencing a slow amount of creature spawning and wanted to increase the frequency of their spawns, you could raise the tick rate by entering the

following

/cms spawntickrate CREATURE 100

The lower the amount, the faster they will spawn.

* /cms forcedespawns [true/false]

To turn on despawning for all passives, you would set the value to true. While this can provide a more dynamic environment, it can also lead to issues with certain mods that don’t expect their entities to be despawned.

* /cms min TAG|EntityName intvalue

The min command is used during world gen spawning to determine the minimum amount of spawns for this entity in a chunk.

* /cms max TAG|EntityName intvalue

The max command is used during world gen spawning to determine the maximum amount of spawns for this entity in a chunk.

* /cms frequency TAG|EntityName intvalue

The frequency command sets the frequency for the spawn list entry. The higher this is, the greater chance it has to be picked during spawning.

* To check the current value of a specific command, you would leave out the value. For example, to check the current value of canspawn for Zombies it would be

/cms canspawn MC|Zombie