Crux 2.0 Documentation

This is an offline version. It is recommended that you use the Online Version for the most up to date documentation.

Table of Contents

The Basics

Crux is a runtime procedural AI spawner capable of spawning AI dynamically across terrains according to customized biomes and conditions. Individual Scriptable Objects are used for storing all the information that Crux needs to properly spawn your objects. When using a Unity terrain, the texture at the spawning position is used to determine what biome to pick objects from. When this happens, Crux will randomly spawn objects according to that specific biome, as well as other global and individual object conditions. When the spawned objects hit Crux's despawn radius, they will be despawned and put back into

Crux's object pool. The total number of objects are always tracked to ensure a well-balanced population. This is also done for performance reasons which allow games to feel populated with more complex objects, such as AI, without having to have 100's of AI running in the background.

Testing the Demo Scenes

It is recommended that you give the demo scenes a try as these can give you a better idea with how Crux works and functions. When trying the demo scenes, you will see that Crux spawns AI, and objects, procedurally within a certain range of the player.

Creating your Own Crux System

To create your own Crux System, go to Window>Crux>Create>Create New Crux System. This will create a brand new Crux system for you to work with. Once you've done this, follow the video tutorial below. It covers assigning a player to the Crux system, creating multiple biomes, and creating custom Crux Objects that the system will spawn.

The Crux Editor

The Crux Editor is categorized into 3 tabs; the Biome Management, Global Settings, and Terrain Info. Below, each one will be discussed in-depth.

Biome Management

The Biome Management section allows you to manage all of your Biomes, create new Biomes, and assign [Crux Objects](#) to each spawning category. Within the Biome Management, you can create a Biome for each environment type. These Biomes allow you to choose exactly what Wildlife, Creatures, and NPCs will spawn according to the textures for the environment. **Note:** Textures are not used when using Mesh Terrains. This is because Unity doesn't have an efficient method for getting the texture at a position of a mesh. When using Mesh Terrains, only the first biome will be used.

Create Biome

The Create New Biome button allows you to create a new biome that you can customize as need with custom set textures and objects. For a detailed guide on creating a biome, see the Creating a Biome section.

Biome List

A list of all biomes the current Crux system has. The name of each biome can be customized at the top of each biome.

Texture Options

The section that allows you to control what textures determine the biome. Any textures that are added to this section will allow objects from the Biome Objects Options section to be spawned, given that they are present on your Unity Terrain. **Note:** Textures are not used when using Mesh Terrains. This is because Unity doesn't have an efficient method for getting the texture at a position of a mesh. When using Mesh Terrains, only the first biome will be used.

Biome Object Options

The Biome Object section has 3 tabs, one for each category of AI. Any Crux Object can be used for any category. How you do this is entirely up to you. Typically, the Animal and Creature categories are used for AI such as creatures that can attack your player or wildlife that can flee when your player gets near them. The last tab, is the NPC tab. This tab can also be used for anything and is designed for objects or AI. Spawnable objects can be good if you want your player to wander around looking for objects that don't always appear in the same spot or only spawn on in a certain biome. Within these tabs, you may have realized that the objects that are listed are Crux Objects. Crux Objects are an easy and more efficient way for handling spawnable objects. See the [Crux Objects](#) section, for more information on Crux Objects.

Global Settings

The Global Settings has an additional 3 tabs. These tabs will be broken into 3 sections.

Global Settings

Spawning Options	Spawn ID Options	Documentation

Player Settings

Controls the type of method that Curx uses to get your player.

Player Transform Type — Stardard

The Player Transform Type is the transform that Crux will use as a reference point for spawning objects.

Player Transform — Crux Demo Player (Transform)

The Standard Player Transform Type will allow you to manually set the player transform.

Spawning Settings

Controls the type of spawning techniques Crux will use.

Terrain Spawning Layer — Default

The layer mask used for spawning on the terrain. This should be the layer that your terrain is using. This feature is also useful to exclude layers that you do not want to be spawned on such as in water or on rocks. Note: The Terrain Info also uses this layer.

Spawn Type — Stardard

The Standard Spawn Type option will simply spawn objects around the player. They will remain active as long as they don't leave Crux's despawn radius. This option is helpful for those who want objects to continue to be active even though they might not be visible or near the player.

Use Object Detection — Yes

Controls spawning objects by checking the area for previously spawned objects of the appropriate layer. This is to avoiding spawning new objects too close to previously spawn objects and keeps objects evenly distributed. This layer can be any layer that is desired other than the Default layer. If the Default layer is used, this feature will be disabled.

Layer Detection Layers — Water

The layers that Crux will use when using Layer Detection Spawning.

Spawning Options

The Spawning Options control all of the Global Spawning settings such as spawning distances, which layers can be spawned on, what object is used for the player, and more.

Spawn ID Options

The Spawn ID Options controls whether or not Crux's Spawn By ID feature

is enabled. This feature allows users to spawn any object according to its Spawn ID. A Spawn ID is created when you create a [Crux Object](#).

Documentation

This tab simply gives you easy access to Crux's Documentation and Tutorial Videos from within the Crux editor.

Terrain Info

When on the Terrain Info tab, you can press and hold the Left Control key while using your mouse to determine the point you'd like detected. The terrain information will be updated within the Crux editor within said tab. If it's not updating, you may need to right click to have the scene view become active. The Terrain Info allows you to easily detect the slope and height of your terrain.
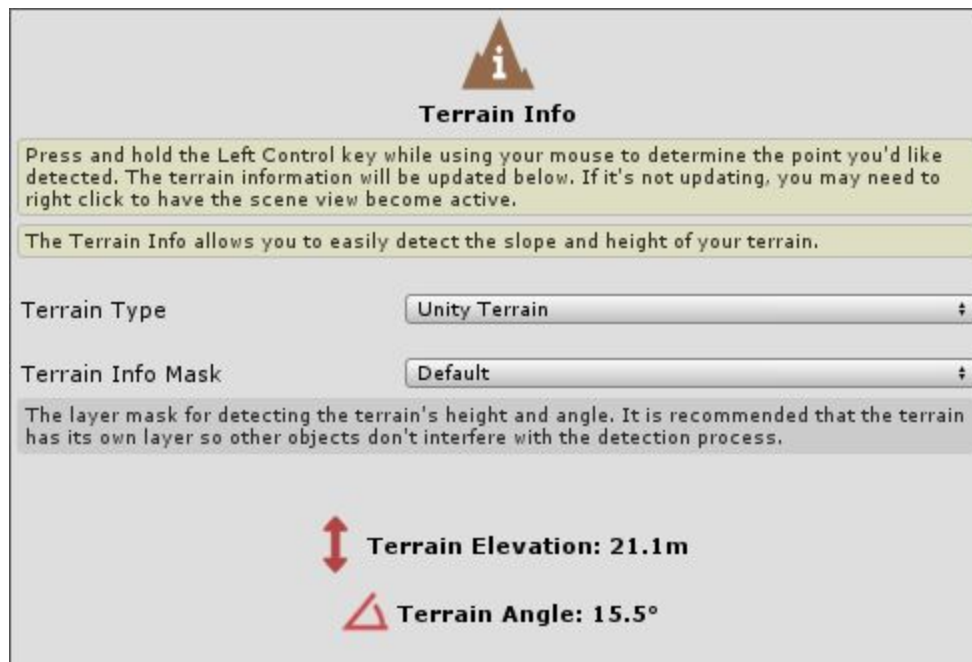
Crux Objects

Crux Objects are Scriptable Objects. Think of these as miniature editors that allow you to customize your spawnable object and store the information in an external file. Crux Objects allow you to individually customize the spawnable object's conditions, rarity tier, information, population amounts, and more.

Creating a Custom Crux Object

To create a Crux Object, see this section of the Crux Getting Started Tutorial.

Biomes

Biomes are a customizable section within the Crux editor that allow you use textures to define the biome. When picking a spawning position, Crux looks up the texture from that specific location to determine what biome it should use to spawn objects. Once it has found the proper biome, it will then spawn objects from said biome using certain conditions from the Crux Object.

Creating a Custom Biome

To create custom biome, see this section of the Crux Getting Started Tutorial.

Crux API

Crux handles most needed functionality, however, there are a couple public functions that may need to be accessed. Crux has a global reference so all that's need to access the Crux System, is using Crux.CruxSystem.Instance.

```
//Removes and despawns the gameobject from Crux.
Crux.CruxSystem.Instance.RemoveObject(GameObject ObjectToRemove)

//Spawns an object using the Crux Object's Spawn ID.
Crux.CruxSystem.Instance.SpawnByID (string SpawnID)
```