Chunk Editor Documentation

This is a live document and will be subject to change as the Chunk Editor develops. Please leave comments for out of date material or sections that you find unclear. Direct all questions to Nathan Williams. ***Last updated 15-12-2013***

Prefix

The endless level in Project Tram will be created out of small predefined sections known as ‘chunks’ that will be tiled repeatedly during the game to create the endless level. These chunks will be made using a custom made Chunk Editor. The Chunk Editor is a collection of scripts that extend the default Unity Editor to allow for quick and easy creation of chunks. This document will focus on using the Chunk Editor for chunk creation.

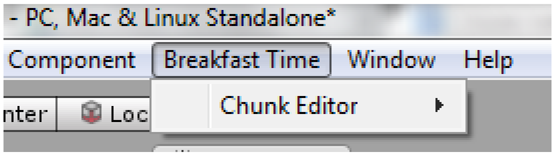
Section 1 - Setup

To use the Chunk Editor you must have the current public version of the Unity project. This can be found in the Breakfast Time Dropbox Folder (Breakfast Time > Programmers > Public Project > ProjectTram) Copy the entire ProjectTram folder onto your PC (please don’t work from the Dropbox folder) and then open the project in Unity.

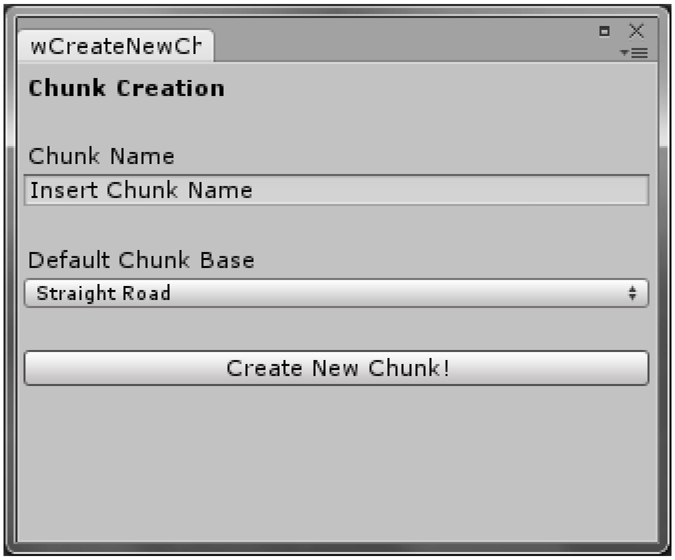
In order to make a chunk you must be in the “Editor” scene (the editor tools will let you know if you are in the wrong scene). The Editor Scene can be found in the Project window in the ChunkEditor folder (Assets > ChunkEditor > Editor.unity). Load the Editor.unity scene.

Section 2 - The Breakfast Time Tool Bar and Chunk Creation

A lot of the Chunk Editor functionality is accessed through the Breakfast Time tool bar. This can be found on Unity’s default tool bar at the top of the Unity Editor:



To begin making a new chunk use the ‘Create New Chunk’ button which can be found in Breakfast Time> Chunk Editor> Create New Chunk. Clicking that button will open the chunk creation window:



Chunk Creation

Chunk Name

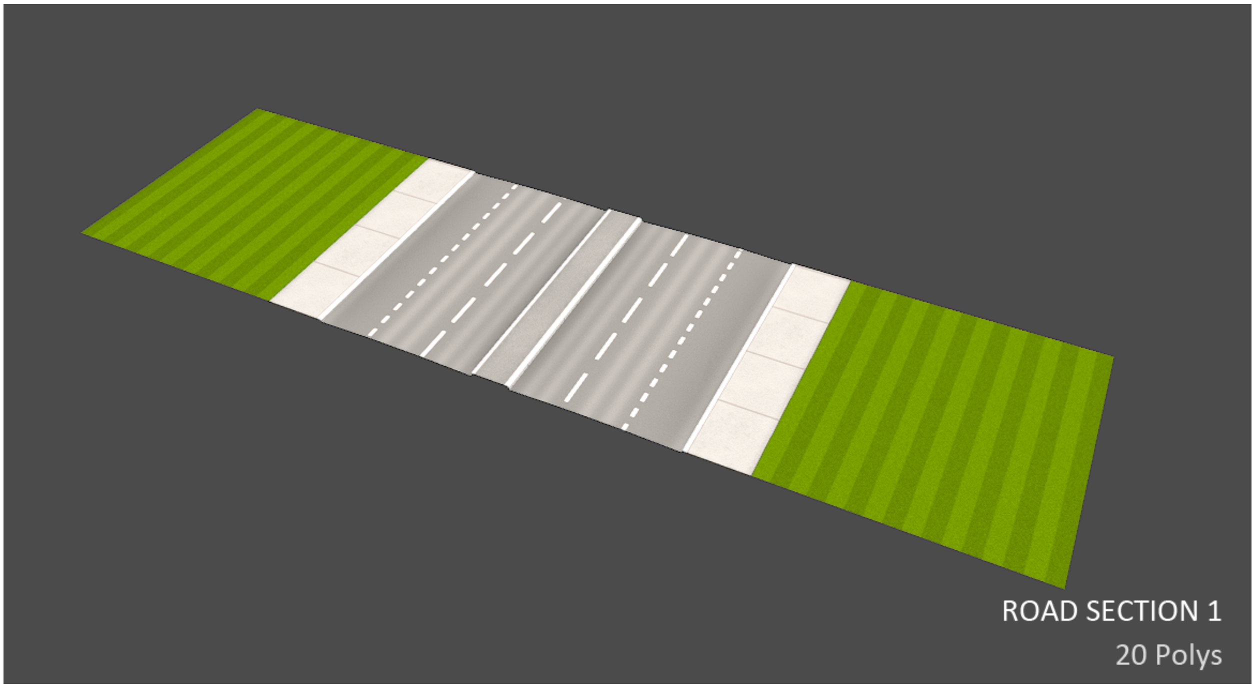
Insert Chunk Name

Default Chunk Base

Straight Road

Create New Chunk!

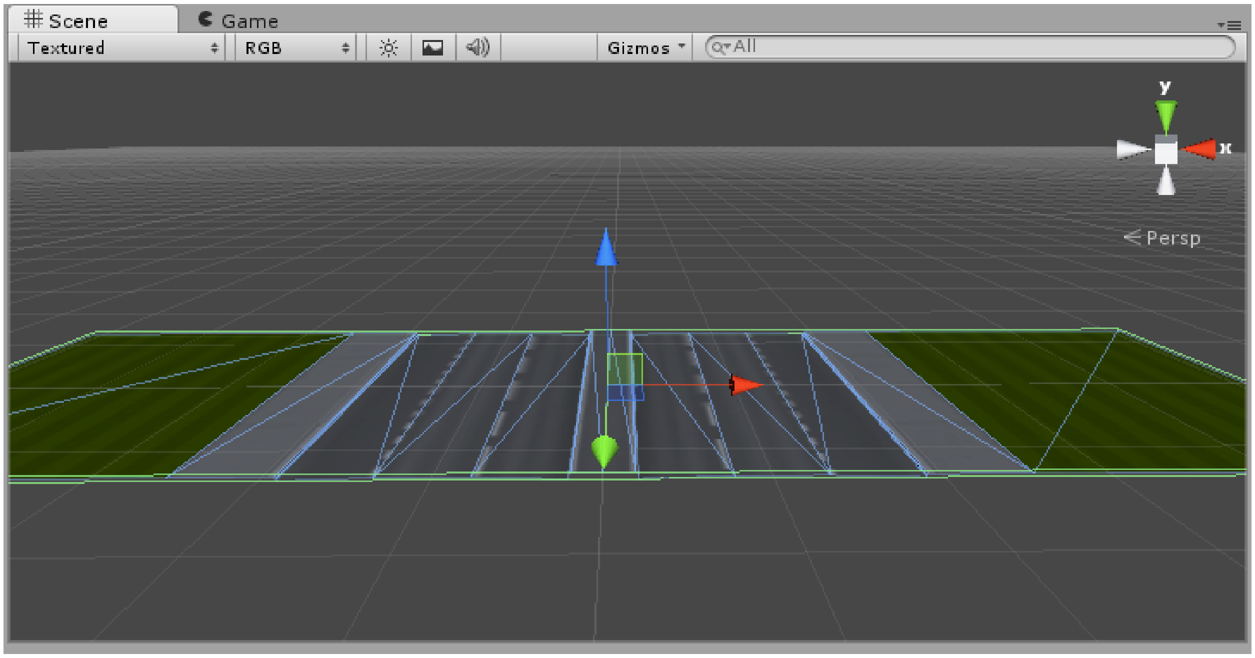
The Chunk Name field is the name the chunk will be saved as. The Default Chunk Base field lets you choose which chunk base will be used for this chunk. A chunk base is a premade asset, often featuring a road, but not necessarily always, that has been created to serve as a chunk base. Below is an example of a chunk base:



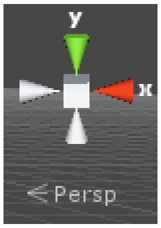
Clicking “Create New Chunk” will place the selected chunk base into the Editor scene. You are now ready to populate this chunk.

Section 3 - Chunk Editor Orientation

As it stands, at present there is no convenient ways to tell which way you are viewing the chunk in relation to how it will look in the game. When you create a new chunk you should see something similar to the below image in the scene:



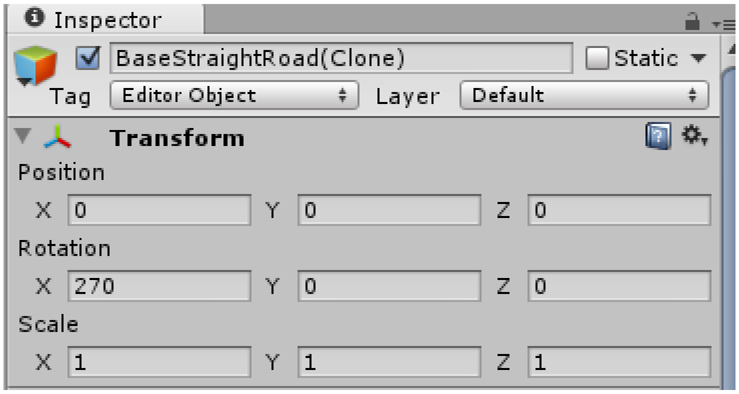
To understand which way you are viewing the chunk you can use the viewport gizmo in the top right of the Scene window:



If you are looking at the chunk, so that your view gizmo matches the one above, with the red x axis on the right hand side then you are looking at this chunk from the same perspective you will see it in game (i.e you are looking down the road from behind the tram)

Section 4 - Chunk Angle

Before you begin editing your chunk you should set the angle you wish for this chunk to be at. Although you can change this at any time, any Prefabs you have placed on the chunk will not necessarily update correctly if rotation is changed after they are placed. You should only change the rotation on the **X Axis**. Although it is possible to change all three rotations, only the x rotation will be saved in the chunk, and changing y or z will make the chunk not work correctly in the game. You can rotate the chunk either using the rotation tool in the top left of the Unity Editor(⟳) or by inserting the rotation amount on the Transform tab in the Inspector for the prefab:



Inspector

☑ BaseStraightRoad(Clone) □ Static

Tag Editor Object Layer Default

Transform

Position

X	0	Y	0	Z	0

Rotation

X	270	Y	0	Z	0

Scale

X	1	Y	1	Z	1

Section 5 - Chunk Editing

Populating chunks with game content is more or less the same process as creating a typical scene in Unity. You use pre made Prefabs to populate the chunk. The Prefabs that are used with the Chunk Editor can be found in the Unity project in Assets > Chunk Editor > Resources > Chunk Assets. **Note: There is a Chunk Bases folder in this folder. You should not be using any Prefabs from that folder.** Inside the Chunk Assets folder you will find a collection of subfolders that contain prefabs you can add into the scene. You can use the normal Unity Editor controls to move and rotate Prefabs within the chunk.

Section 6 - Chunk Saving

To save your changes to your chunk use the “Save Chunk” button in the Breakfast Time tool bar (Breakfast Time > Chunk Editor > Save Chunk). You can save a chunk multiple times (and you should save often to avoid losing work).

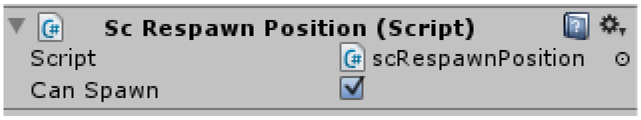
Section 7 - Chunk Loading

If you would like to make changes to a chunk that has already been previously made, you can load it into the chunk editor by using the “Load Chunk” button in the Breakfast Time tool bar (Breakfast Time > Chunk Editor > Load Chunk). This will open the load chunk window which simply has a drop down box that lets you select the chunk to load.

It is also possible to load preview chunks in this window. There are two types of preview chunks, “Front Preview” and “Back Preview”. A Front Preview chunk will connect to the chunk you are editing as if it was the previous chunk generated in the game. A Back Preview chunk connects to the chunk you are editing as if its the next chunk that will generate in game. You cannot make changes to Front Preview or Back Preview chunks, you must load them as the chunk you intend to edit if you wish to change them. They are simply there to let you easily align obstacles across chunks when making Super Chunks.

Section 8 - Respawn Positions

Automatically placed inside chunks when they are created is a red sphere named ‘RespawnPosition’. This is simply the position the player will respawn at if they respawn inside this chunk. Moving this sphere will change the position the player respawns at. If for some reason you do not want it to be possible to respawn inside this chunk then select the ‘RespawnPosition’ prefab in the editor. In the Inspector there is a script attached to it called ‘Sc Respawn Position’.



Sc Respawn Position (Script)

Script: scRespawnPosition

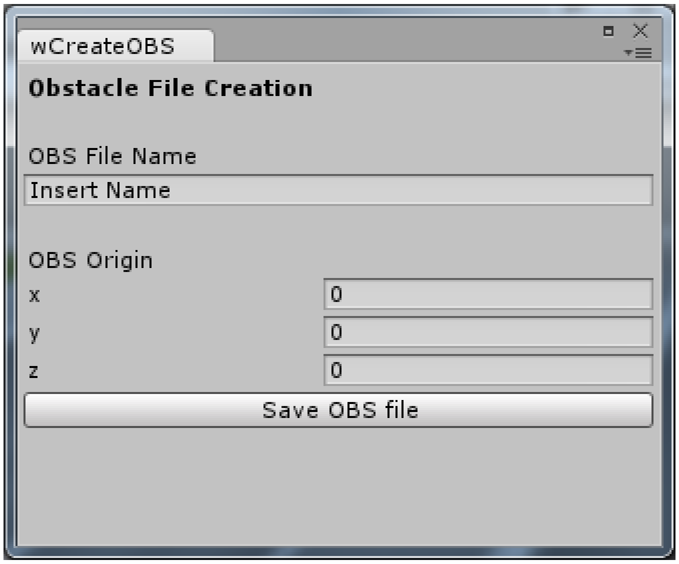
Can Spawn ☑

On this script there is a tick box labeled “Can Spawn”. Unticking this box will make it so the player can not respawn in this chunk. In game what will happen if this box is ticked is the game will search chunks in front of the current chunk until one is found that allows respawning. The player will then respawn at that position.

Section 9 - Random Obstacle Spawners

Inside the Chunk Assets folder where chunk prefabs are located, there is a prefab called ‘randomSpawner’. This prefab allows you to set obstacles files to it, and during the game, one of the obstacle files set to the spawner will be selected and the obstacles in that file will be created. This allows the possibility to create random variations to chunks.

In order to use this you first need to make an Obstacle file that can be used by the spawner. An Obstacle file is simply a collection of one or more obstacles. To create an Obstacle file select the obstacle(s) you wish to use in this file in the Scene. While they are selected click the ‘Create OBS file of selected’ button in the breakfast time tool bar (Breakfast Time > Chunk Editor > Create OBS file of selected). This will open the Obstacle File Creation window.



wCreateOBS

Obstacle File Creation

OBS File Name

Insert Name

OBS Origin

x: 0

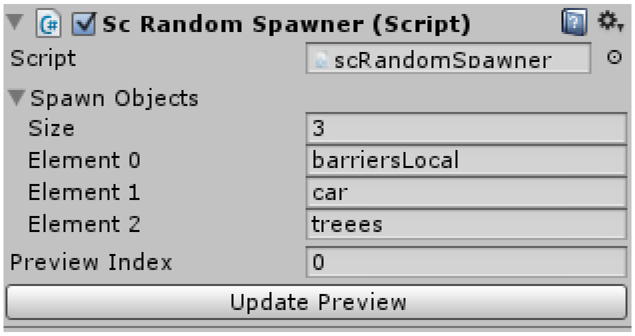
y: 0

z: 0

The name field is simply the name the Obstacle file will be saved as. The OBS Origin however is quite important. The obstacles in the file will be saved in local space from the origin you set. They will then be created the same distance from the randomSpawner as they are from the origin you set. If the Obstacle file you are creating only contains one object, and you want it to be created in the exact same position as the random spawner (which in most cases you will) then you simply set the OBS Origin to the position of the obstacle you are saving. If you are saving multiple objects, then depending on their layout depends on where you will want to set the origin.

Before clicking the Save OBS File button, make sure all the obstacles you want saved in this file are still selected in the Scene/Hierarchy.

To set an OBS File to a randomSpawner, select the randomSpawner prefab in the Scene/Hierarchy. In the Inspector there is a script attached named ‘Sc Random Spawner’.



Script	scRandomSpawner
▼Spawn Objects	
Size	3
Element 0	barriersLocal
Element 1	car
Element 2	treees
Preview Index	0

First, set the Size field to the number of OBS files you are planning to assign to this spawner. In the above picture there are three assigned so Size is set to 3. Once Size is set more fields should appear named Element 0 - Element [size you entered-1]. Type the name of the OBS files you have created inside these fields.

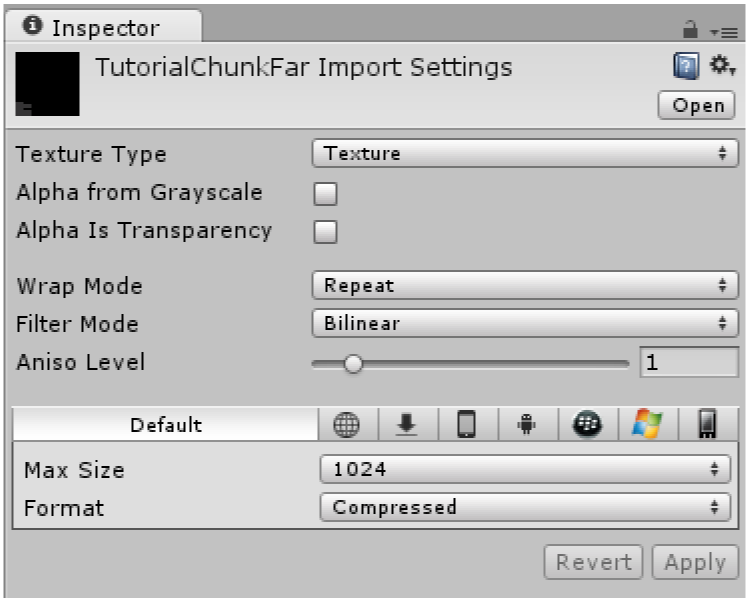
It is good practice to use the OBS preview functionality on each OBS file you assign to the spawner, as this will help find any errors or problems with your OBS files or the way you have set up the spawner. To do this set the Preview Index to one of the Elements numbers in your spawner and click “Update Preview”. This will create the objects in the OBS file on top of the spawner as they would appear in game in the chunk editor, as well as show a warning if the file you are trying to load doesn’t exist (if for example you spelt it wrong). You should do this for each element.

Section 10 - Baking Lightmaps (this feature has been deprecated)

N.B Lighting still needs to be set up consistently in the chunk editor. If you are planning to bake lighting into a chunk speak to Liam Bower. He is in charge of setting up the lighting which hasn't happen yet, for now there is probably no point messing with light baking except for experimenting.

To bake lighting into a chunk such as A.O you simply bake the lighting using the regular unity light baking tools. Once you bake the lighting you must save the chunk again if you wish for the currently baked light maps to be assigned to the current chunk. Saving the chunk will overwrite any previous lightmaps assigned to this chunk if you baked the lights during editing.

Once you have saved the chunk there is one more additional step you must take. After saving your chunk go to Assets > Chunk Editor > Resources > LightMaps. Inside this folder you should find two new lightmaps, <YourChunkName>Near.exr and <YourChunkName>Far.exr. You need to change the import texture type setting for these two files to lightmap. To do this, select the files in the Project window. You should see the following in the Inspector window:

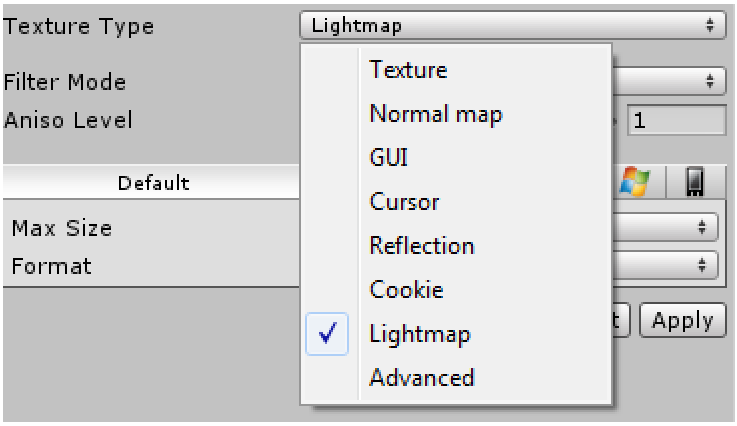


TutorialChunkFar Import Settings

Texture Type: Texture
Alpha from Grayscale □
Alpha Is Transparency □
Wrap Mode: Repeat
Filter Mode: Bilinear
Aniso Level: 1

Max Size: 1024
Format: Compressed

On the Texture Type drop down box you should have the option for Lightmap:



Texture Type Lightmap

Filter Mode

Aniso Level 1

Change the Texture Type to Lightmap then click apply at the bottom of the Inspector window. Your lightmap should now be working correctly during the game.

Section 11 - Testing your chunks in game

Once you have created your chunk's you will probably want to test them in the actual game. To do this you need to add your chunk to a ChunkData file, or create a new one for your chunk. A ChunkData file is a simple text document that holds one or more chunks in it. It may look something like this:

```
sChunk0
sChunk1
sChunk2
sChunk3
sChunk4
```

You place one chunk per row, and as many chunks as you would like to load in it. The chunks will be loaded in the order from top to bottom. This allows you to make Super Chunks that are generated from many chunks in a predefined order, to make larger sections of levels. The ChunkData files are all stored in GamePlay > Resources > ChunkData. To create a new ChunkData file simply using a text editing program and drag that file into that directory. Loading the gameplay scene and running the game (shortcut to running the game is ctrl+p) should show your chunk generating eventually.

A recent change means that chunks are now stored in biomes. Biomes are easy to create/manage and update. All biomes are stored in GamePlay > Resources > ChunkData inside subfolders. To add a ChunkData file to an existing biome (for testing) you just have to put the .txt file inside one of the folders for the existing biome. To create a new biome you simply create a new folder and put the ChunkData files you wish to be part of that biome in that folder.