CITYGEN3D v1.05

Procedural Scene Generation for Unity

Contents

1. Introduction

Thank you for purchasing CityGen3D!
After reading this guide you should have a basic understanding of how to automatically generate scenes from real world map data using CityGen3D within Unity, all without writing any code.

Please read the following instructions carefully in order to get the most out of tool. The manual will be updated periodically with information on new and existing features, so it's also a good idea to revisit it after downloading an update.

CityGen3D had its first public Beta release in November 2018 and has been actively supported since then. Please report any bugs and feature requests so the tool can continue to be improved upon.

For more information and tutorials you can visit the website at www.citygen3d.com and follow the CityGen3D thread on the Unity Forum for the latest information.

A Discord channel has also been created, which is free to join and is a good place to keep up with CityGen3D news and share tips with other CityGen3D users in our growing community.

2. Installation

CityGen3D runs on **Unity 2019.4.15** or later, to optimize some of the processes and take advantage of the very latest Unity features, including Unity Jobs with Burst Compiler for multithreading support, and Terrain Editing on the GPU.

Because it utilises some of the very latest Unity technology, there are a few steps required before you install CityGen3D.

Please read this section of the manual carefully in order to set up your project environment for use with CityGen3D.

2.1 Installation – Setup

It is recommended to install Unity via the Unity Hub. Further information on can be found here: https://docs.unity3d.com/Manual/GettingStartedInstallingHub.html

It is recommended to use the Long Term Support versions of the Unity Editor, which continue to receive critical bug fixes, but offer greater stability. These are 2019.4.x and 2020.3.x.

CityGen3D is a unified release intended to work on both Scriptable Render Pipelines provided by Unity, as well as the older Built-In Pipeline.

The Universal Render Pipeline (URP) provides artist-friendly workflows that let you quickly and easily create optimized graphics across a range of platforms, from mobile to high-end consoles and PCs.

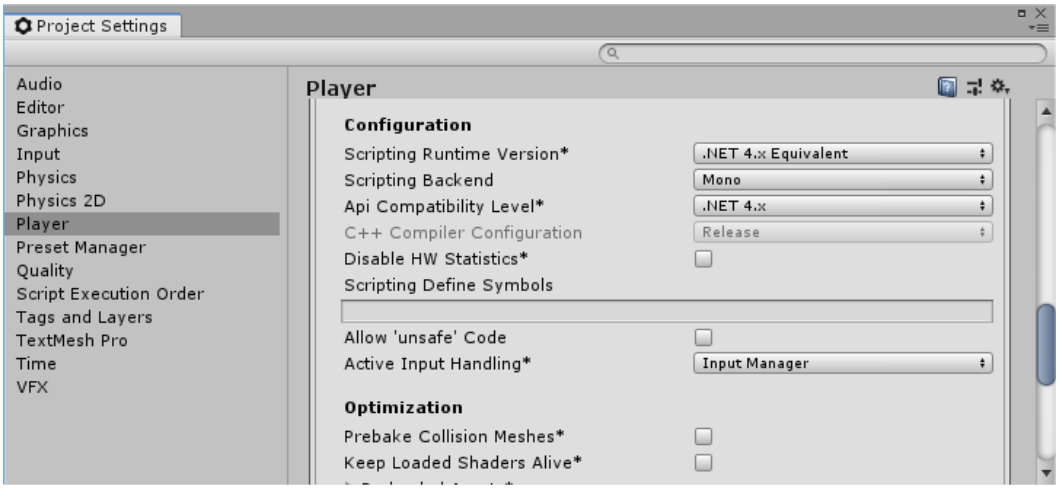
The High Definition Render Pipeline (HDRP) lets you create cutting-edge, high-fidelity graphics for high-end platforms. You should use HDRP for AAA quality games, automotive demos, architectural applications and anything that requires high-fidelity graphics. HDRP uses physically-based lighting and materials, and supports both forward and deferred rendering. HDRP uses compute shader technology and therefore requires compatible GPU hardware.

Note that HDRP terrains do not currently support grass and allow up to eight textures, whereas URP supports up to sixteen. So unless you specifically need features provided by HDRP over URP, then URP will likely give you a bit more flexibility.

Once you have decided on the SRP that is right for your project, please start a new 3D project as normal, in your chosen pipeline.

You then need to set up Unity to be compatible with a few Unity Packages:

From the Player tab on the Project Settings panel in the Unity Edit menu, change the API Compatibility Level to ".NET 4.x". On older versions of Unity there may also be an option for changing the Scripting Runtime Version and you should select ".NET 4.x Equivalent".

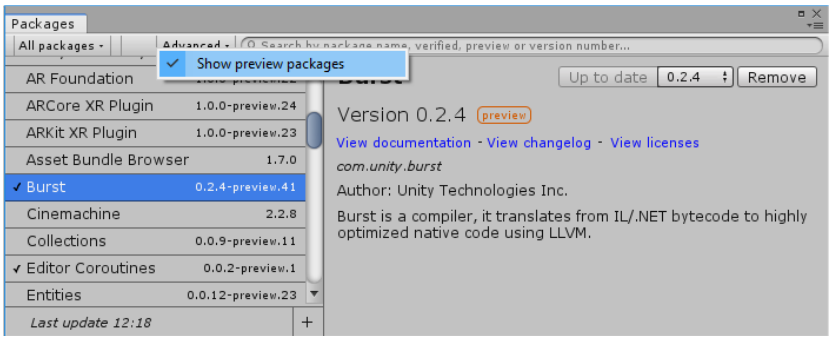


CityGen3D uses several Unity features that are distributed separately to the main Unity installation. These are freely available via the Unity Package Manager and take just a few minutes to install.

Window > Package Manager...

Select each package from this list in turn and click the Install button, noting that Post Processing is only required for Built-In Pipeline projects because URP and HDRP have their own post processing systems, which will be set up for you with appropriate defaults.

- Burst
- Editor Coroutines
- Post Processing



If this is your first time using CityGen3D it is highly recommended that you use it in a brand new project, so you can experiment and learn the basic functionality separate to any existing project you may be working on.

Import the CityGen3D package into your project via the Unity menu:

Assets > Import Package > Custom Package...

Make sure all files are selected and click Import to extract the package into your project.
All shaders will be compiled automatically based on your installed render pipeline.

If you find your material preview images are showing up in magenta after installing the CityGen3D package, but no shader errors are being reported, then the material thumbnail images can be refreshed by right-clicking on the offending materials in Project view and selecting the Reimport menu option.
You can also ReImport entire folders if needed.

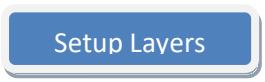
2.2 Installation - Layers

Once CityGen3D is imported you can verify the installation by observing a new menu category called CityGen3D in your Tools menu in Unity.

If you get an error after importing CityGen3D saying that a DLL cannot be loaded, it means you do not have required Packages installed in your project. Please create a new project and follow the instructions in section 2.1 for installing Packages before importing CityGen3D.

CityGen3D is quite unique in that it offers a 2D and 3D workflow within the same project and Unity Layers are used to manage that transition seamlessly within the Editor.

The "Map" layers are specifically used by the camera when in 2D mode and not rendered in 3D mode. They also need to be unique to identify different object types really quickly using a multithreaded job system, where the asset uses batched raycasting.



A LayerManager component handles the assignment of Unity Layers to CityGen3D for you, and you will be prompted to click this button to set them up in any new project before you are able to select a location.

The supplied CityGen3D FreeCam camera prefab is already set up appropriately, but any other game camera added to your project should be set up to ignore the 2D map layers (Map Surface, Map Road, etc) so the 2D map objects are not rendered in 3D.

Please note that you are able to change the layer values to avoid conflicts with other assets, or simply to reorganise them as you see fit for your project. This is best done at the start of the project because it won't retrospectively change the layers assigned to existing object instances in your scenes.

To make changes to the layers that CityGen3D uses you must view the LayerManager component which is on the CityGen3D game object in your scene. Here, you'll be able to assign each CityGen3D layer the value of your choice and click the button to apply the changes.

Please note that in most cases the default LayerManager settings will be fine and you can leave everything by just accepting the default settings when prompted. However, it's worth being familiar with this functionality in case you need it in future.

To prevent 2D map layers having a physical presence in your 3D scenes you can edit the Layer Collision matrix:

2.3 Installation - Updating

CityGen3D will be periodically updated with new features and bug fixes. Please note that while every effort is made to ensure compatibility between CityGen3D versions, this isn't always possible, especially after big upgrades.

So just like the Unity Editor itself, it's only recommended to update CityGen3D to a new version during a project if you really have to.

Please back-up your project before attempting to upgrade Unity or CityGen3D!

If you need to update CityGen3D in an existing project, here are some tips to try and make updating as simple as possible:

- It is recommended to completely remove the CityGen3D folder from the Assets in your project and re-import CityGen3D Unity Package. You may want to ensure you have a backup of any location databases or third-party data (eg. SRTM heightmaps or coastline JSONs) you have since included in the Resources\Data folder in your project so you can copy them back in after updating.
- Before re-importing, delete the Data & Generator gameobjects from the hierarchy.
- After import, you should then add these to your scene as normal. This ensures you are getting up to date versions compatible with latest code.
- You can then load the database you are working on and reprocess the data before making further changes to your scene via the new generator.

3. Quick Start

The easiest way to get started is to load the New City scene from Assets\CityGen3D\Scenes.
This will load a blank scene in Unity comprising only of two GameObjects: FreeCam (camera) and Environment (light, wind & post processing). You now have a suitable setup ready to use CityGen3D in your project.

From here, you are just a few more clicks away from having a 3D scene to explore!

This video shows how easy it is to create a new environment in under five minutes just by using the Location panel on the Data window: https://youtu.be/mNwhDG5L9yE
The steps in the video are also summarized below:

1) *Having loaded the New City scene, click the Tools->CityGen3D->Data (Ctrl+T) menu option.*
2) *In the Settings section of the Location panel, enter a name for the location: "West Hendon".*
3) *Click the Download button, then OK to close dialog once it has downloaded the OSM data.*
4) *In the Processing section click the Process button to generate 2D map objects from this data.*
5) *In the Generator section click the Load Generator button.*
6) *Then click Run Generator to generate your 3D environment, which will appear in Scene view.*

Check out the other videos on the CityGen3D YouTube channel for more information on some of the features and demos showing off some test environments:
https://www.youtube.com/channel/UCF-LSDZRc2wGOOhtH9jCE1w

The following chapters will cover CityGen3D in more detail showing how you can customize each part of the Generator to get the results you want in your scenes.

4. Location

CityGen3D has two main functions. The first is to download and process the map data ("Data"), the second is to generate the 3D scene based on this data ("Generator").
Both Data and Generator have their own Window within Unity for setting options.

So first thing to do is to add a CityGen3D data prefab to the scene via the menu bar so we can interact with CityGen3D via the Data window:

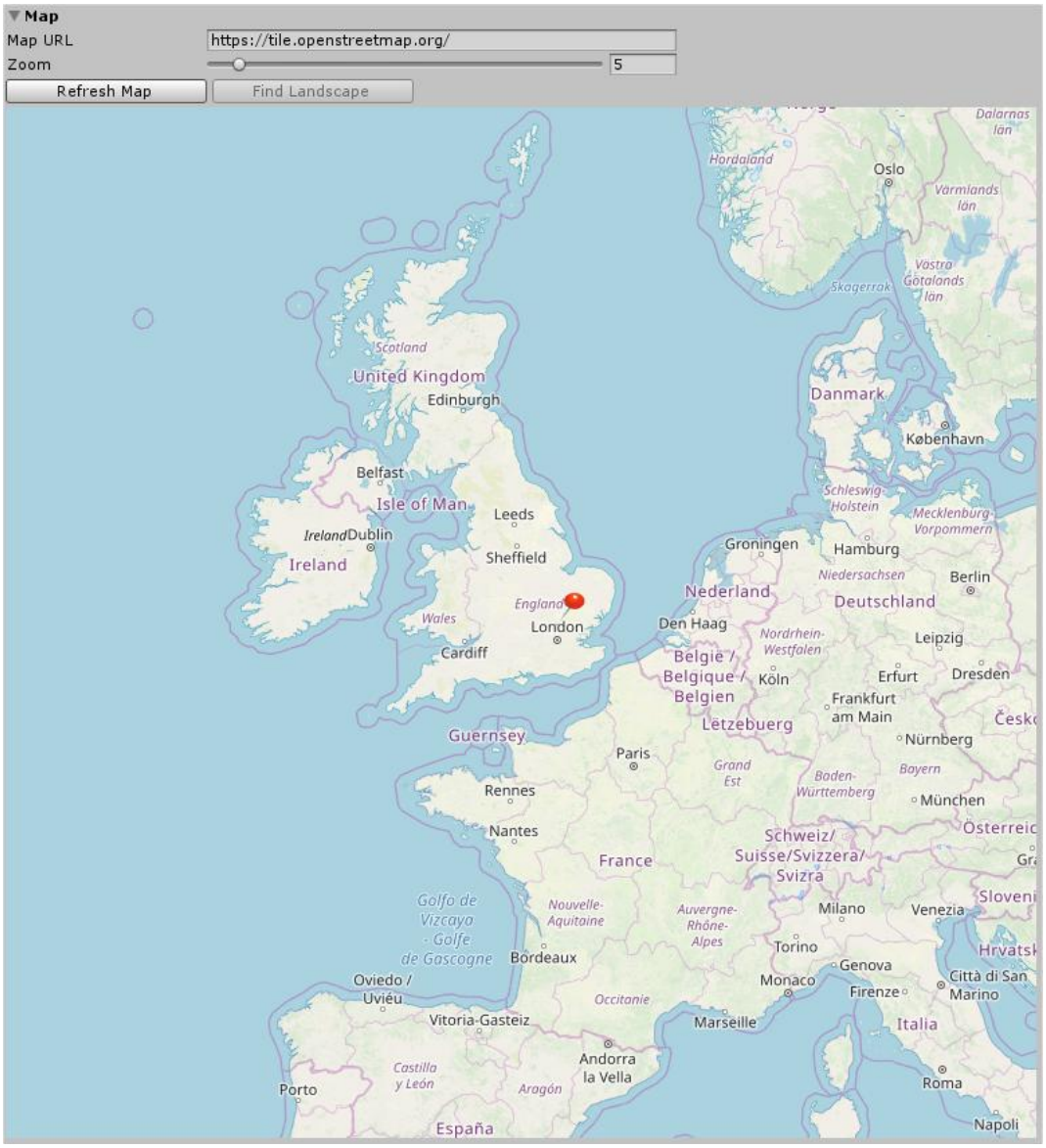
Clicking the Data menu item or pressing Ctrl+T will create a Data window in Unity. You can drag it around the screen and position it just like another other Unity window such as the Inspector, but it's a good idea to make it fairly large so you can see the embedded map.

The Location tab is the primary panel on the Data window and is where you tell CityGen3D what data to download from OpenStreetMap (OSM). More information on OpenStreetMap can be found here: https://wiki.openstreetmap.org/wiki/Main_Page

4.1 Location - Map

The easiest way to select your location is via the map. You can drag the map around using Right-Click and select a new location using Left-Click.

The ***Map URL*** allows you to specify the server address from which the map is downloaded from in case this changes sometime in the future. Otherwise you should just be able to leave this on default setting.



When viewing the map from far away, the location is marked on the map by a red pin. ***Zoom*** in further using the slider, or by using Ctrl + ScrollWheel, to see the map in more detail. At higher zoom levels, a shaded grey area shows the precise extent of your currently selected location and area.

This button helps you to find your currently selected location on the map and forces a redraw.

If you have already generated a landscape, you can select it and click this button to apply origin and tile settings from the selected landscape. This can be very useful if you want to download another data set and have it align correctly with existing terrains.

4.2 Location - Settings

Make sure ***Data Source*** is set to *Download* and give an appropriate ***Name*** to your location, such as a town, village, or city name.

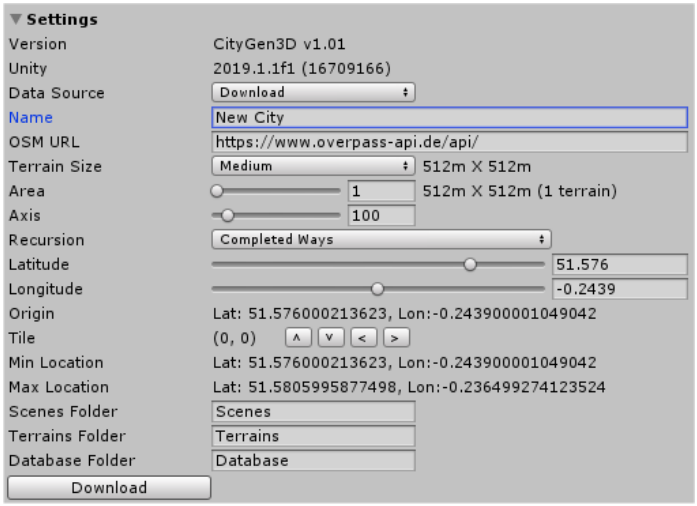
If you already have a database for a specific location, having downloaded the data in CityGen3D previously, set your Data Source to *Database* and load the appropriate asset from Resources\Locations folder.
The *XML* data source shouldn't be needed for most users, but allows you to import raw OSM data from an external source if you are not able to access the OSM servers from within Unity.

It's possible in future that OpenStreetMap may change their API and you may need to update the ***OSM URL*** to point to a new location, or to try an alternative server. Otherwise simply leave it on the default web address.

In addition to the map, you can also select a location using the ***Latitude*** and ***Longitude*** sliders, or enter a specific location in the text boxes. The map will auto update as you move the sliders, although you may need to click the Refresh Map button to find your new location.

Note that when selecting a new location via the map, or using the Latitude/Longitude sliders, you are choosing the South-West extent of your environment and a new reference origin.

Having set your origin you should subsequently move around the map using the ***Tile*** arrow buttons only, instead of clicking on the map or moving the sliders.



Preserving the origin in a project is an important concept in CityGen3D because it allows you to download other data sets and have them align to previously created terrains. You could even have multiple users in your team working on different areas of your environment knowing that the terrain tiles will later align seamlessly with each other because of the shared origin setting.

The tile values represent how many terrains to the East and North of the origin the selected location is, so the origin is always considered to be tile [0,0].

Clicking the chevron icons will shift your location exactly one tile in chosen direction.
The solid arrow buttons will shift your location enough distance to cover the current area size.

So you could start a project by selecting a new location on the map, which would set the origin coordinates and reset the tile to [0,0]. You then download your chosen area size and generate these terrains using the Generator.
You could later expand your environment by using the arrow buttons to shift the location to be adjacent to your existing terrains, rather than attempting to manually select a new location that

aligns with it. Retaining the same origin across different groups of terrains within your project ensures that terrain heightmaps and textures seamlessly align with each other.

If you accidentally select a new origin in your project and want to create a new group of terrains to align with existing terrains, you can use the Find Landscape button to reset the origin based on an existing terrain and use the arrow buttons to select a new location.

Use the ***Area*** drop-down to select the data you wish to download and process. Note that CityGen3D will also download any OSM data outside of this area within a one hundred metre margin. It will also download all nodes and ways outside of this area if they belong to ways or relations that appear partially inside it. This helps to ensure that CityGen3D has all the data required to create an accurate map for your area.

The origin, tile values, and area size all contribute to the currently selected data area, which at higher zoom levels appears shaded on the map. The bounds of this shaded area are also specified with the ***Min Location*** and ***Max Location*** labels. Naturally if the current tile value is [0,0] then the Min Location is identical to the origin.

Each landscape is assigned an index based on their tile value where the terrain at origin has index of 0 and is therefore called “Landscape 0”.
The ***Axis*** integer is used to auto number landscapes and specifies the maximum number of terrains that can be created across the horizontal axis in your current project.
So landscape indices are incremented by one for every tile increment across x-axis and incremented by Axis value for every y-axis increment. This means that the Axis setting must be at least equal to the Area size, and this will be set for you if needed when you increase the Area slider.

Although CityGen3D allows you to download and process areas as big as 8km, it is recommended to create big environments using multiple smaller areas and the tiling feature as detailed above.
In other words, ensure your Axis setting is large enough to accommodate your entire environment, but only work on small areas within that at any given time. This reduces the overhead on the Editor and makes it easier to work with.
Each database you create will save a tile reference as part of the filename so you can easily find different portions of your environment.

Later, your CityGen3D scenes will be saved to the specified ***Scenes Folder***.
Terrain data and terrain layers are saved to the specified ***Terrains Folder***.

CityGen3D downloads the data automatically for your location when you click the Download button, but how do we determine exactly what data is downloaded from the OSM server?
Initially it may seem like there is an obvious answer. Surely we can just download all OSM nodes that fall within the location's bounding box and not download anything else?
In many cases this wouldn't actually be enough data to correctly identify shapes that partially fall outside of the bounds of your location.

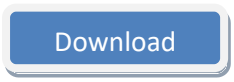
Imagine a lake that is defined by a single OSM way, made up of many nodes to form an enclosed body of water.
Some of the nodes in the shape fall outside of the bounds of your location and some fall inside.

If we were only to retain the nodes inside the bounding box we would not correctly be able to identify the shape of the lake.

This is where the ***Recursion*** setting comes in. By default it is set to *CompletedWays*, which means any way that falls partially inside your location's bounding box, will be downloaded in its entirety. This helps to accurately recreate the map in Unity.

However, some enclosed shapes are not only made up of single ways, but made up of multiple linked ways called a relation.
So another option is provided called *CompletedWaysAndRelations*, which will download all the nodes for a relation if one or more nodes fall within the locations bounding box.

Note that the second option can result in quite a lot of data in some cases and therefore much slower processing. So it can be beneficial to run faster tests using *CompletedWays* only, and reserve *CompletedWaysAndRelations* for only when you really need to identify the extra data.



When you are happy with your location and area size, click the Download button to start the download process from OpenStreetMap. If you can't see the Download button, ensure the Data Source is set to Download.

You must be connected to the internet for this to work. This will auto save the location database as an asset in Resources\Locations, so you can load it again anytime without having to re-download the data. Your Data Source will also change automatically to Database to prevent you accidentally selecting a new location while working on your scene.

Densely mapped locations with lots of data can take a long time to process. It is highly recommended that you start working with small areas while learning the tool so you can experiment with different options and learn the workflow much quicker.

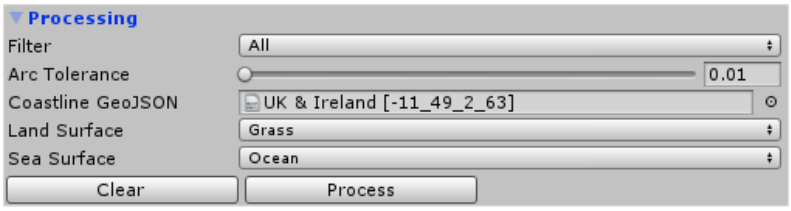
4.3 Location - Processing

Once you have successfully downloaded data from OpenStreetMap, you need to process it to convert it into an editable 2D map within Unity. You have full control over how data is processed via the other tabs in the Data window, but for now we are just going to use the defaults

You can use the different tabs on the Data window to define different key/value combinations that CityGen3D then looks for in the OpenStreetMap data.

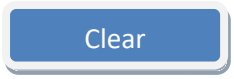
Surfaces and Roads are always processed by CityGen3D as they are fundamental to any scene. Use the ***Filter*** to select which of the other data types are processed.

Roads are defined in OSM data by a polyline known as a "way". During processing, CityGen3D offsets these polylines with rounded edges to describe the extent of each road as a polygon. The ***Arc Tolerance*** setting determines how many points are used to round off the offset at the end of the road, therefore controlling how smooth road corners are.
The default is low for smoother curves in the map data. You can increase this value for a larger tolerance and fewer polygons, which may be preferred in a low-poly rendering style.



A variety of land uses are mapped in OSM, such as farmland, residential, meadow, and many more. But in some places there won't be any such surface in the data. The ***Land Surface*** setting allows you to tell CityGen3D what surface should be used as the default, where this occurs.
The ***Sea Surface*** setting tells CityGen3D which of your defined surfaces should be used where there is no land.

The ***Coastline GeoJSON*** field is used to supply CityGen3D with coastline data.
Unless your location covers a sea/land boundary, this can be left blank.
For more information on coastline processing, please see the Coastline chapter in this manual.



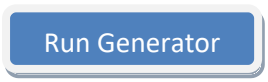
Use this button to delete existing map objects created from any previous Processing of the data.

Click the Process button to start the processing. Once it has finished you should see some new GameObjects in the Unity Hierarchy under CityGen3D.

4.4 Location – Generator

After processing data you'll want to ensure a Generator is loaded into your scene. This manages the transition from 2D map to 3D scene and has its own window with tabs relating to different modules such as buildings, surfaces and highways. It allows you to assign prefabs to different map objects and define rules that determine what your scene will look like.

As you become more experienced with CityGen3D, you may wish to save different Generator prefabs for different projects. For instance you could have a Generator profile setup to utilise 3D graphics from different visual themes (eg Western, Cartoon, Sci-Fi). The ***Generator Profile*** field is where you will tell CityGen3D which profile to use. Then you can click the Load Generator button to activate it in your project.



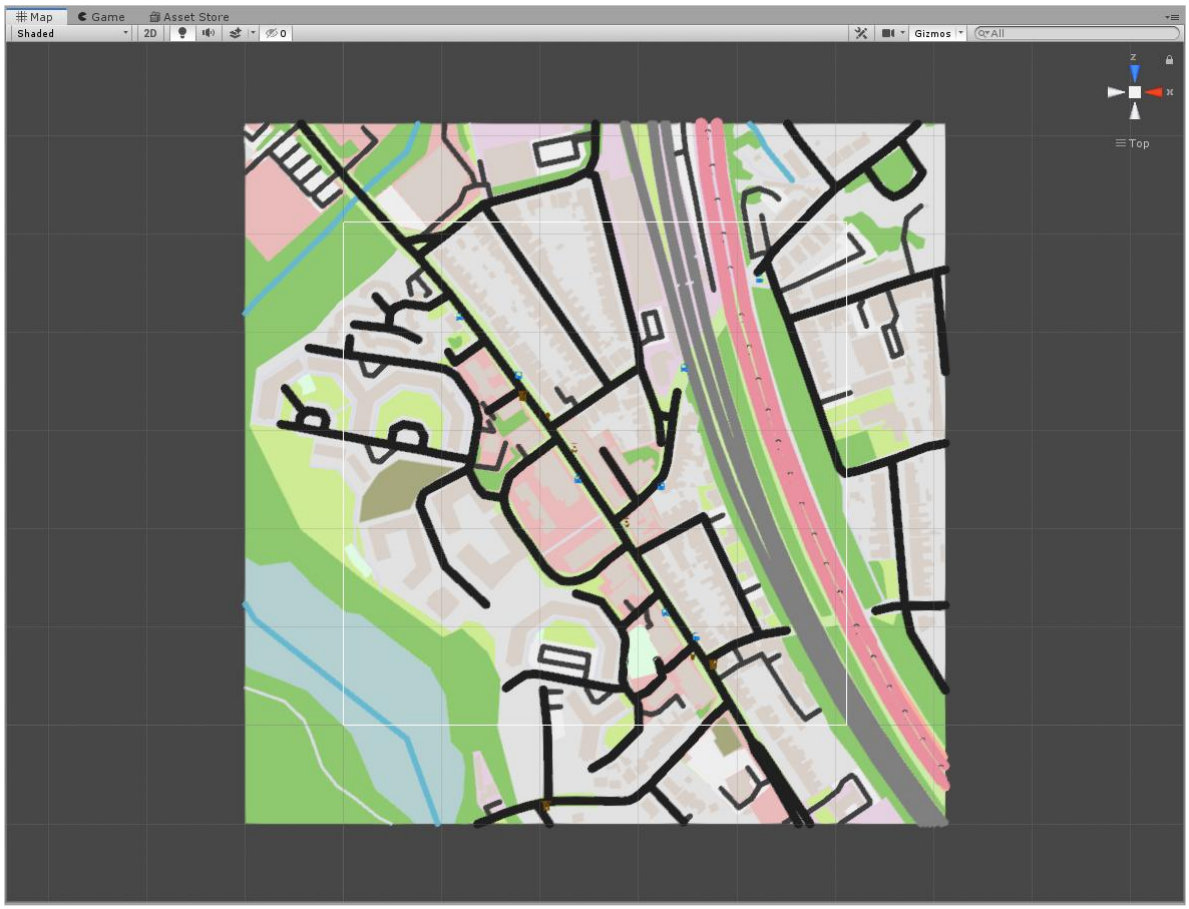
The Generator is highly customizable, but once it is all set up how you want it, the Run Generator command is a useful shortcut for running a sequence of ***Generator Actions*** that you can define above it.

5. Map View

Now that we have processed some map data we can view the data in 2D.
Press Ctrl+M or use the Unity menu:

Your Scene view will be renamed to “Map” and will be fixed to a locked Top-Down perspective.
Select the CityGen3D GameObject in your Hierarchy and with focus on the Map window (the Scene view) press the [F] key.
This will zoom out the scene camera so you can see a rendering of the downloaded data in 2D.
You can also use your mouse wheel to zoom in and out.

Note that a buffer zone is downloaded around your chosen area, making the 2D map slightly larger than your selected area. However, a white rectangle will mark out the extents of your map in relation to what will become visible on your terrains. Anything outside this rectangle will not appear in 3D generation.



6. Heightmap

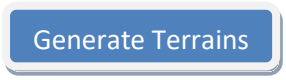
Now you have a Generator prefab in your project you can start creating a 3D scene. The Generator is split into modules, each with their own interface and you select the module you want to work on by clicking the relevant tab in the Generator window.
It's best to work through them from left to right, so we start with the Heightmap panel.

6.1 Heightmap - Terrains

The first thing that needs to be done is to generate the terrains on which everything else is built.
A CityGen3D environment is split into one or more equally sized terrains, the size of which you already selected before downloading the data. So if your downloaded data is larger than one terrain in size, you will have the option of selecting which terrains to generate using the checkboxes.

Each terrain is attached to a Landscape object, which will later act as a parent object for all other prefabs generated above it at that location. This approach results in an environment that is broken down into smaller more manageable chunks.

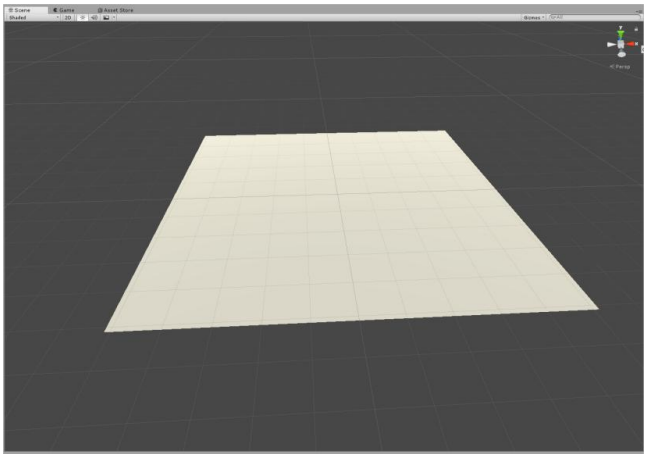
If your environment is particularly large, then you'll want to consider saving each Landscape as a separate scene, which will allow you to load only have the portions of your world that you actually need in memory.
CityGen3D has a Runtime solution built in for managing this process for you, but for now simply ensure the ***Save As Scenes*** checkbox is enabled to have each Landscape saved separately, to make your scene compatible with it.



Click Generate Terrains which will make each of your terrains and they will appear in your Scene view as well as your Hierarchy named Landscape 0, 1, 2, etc

Click this button to convert pre-existing landscapes into scenes for compatibility with scene streaming. This is redundant if your landscapes were saved as scenes on creation using the Save As Scenes option.

By default, your terrains will be flat planes, and currently won't have a texture applied to them.
It should look something like this:



If you can't see your terrain, ensure your Scene view is active and not still in 2D Map view mode. Press Ctrl+M to toggle between 2D and 3D view at any time.

6.2 Heightmap – NASA HGT

OpenStreetMap doesn't have detailed topographical data, but CityGen3D allows you to download heightmap data in HGT file format for easy integration of real world height mapping.

By default the ***NASA URL*** is set up to download NASADEM heightmap data. NASADEM is a global, high-quality digital elevation model, produced by NASA at 1 arc-second (~30m) horizontal resolution, derived primarily from reprocessed data captured via the Shuttle Radar Topography Mission (SRTM).

NASADEM data may also be used without restriction for any purpose whatsoever, commercial or otherwise, free of any royalties or other restrictions, making it the current dataset of choice for digital elevation model (DEM) data.

However, if preferred, the URL can be changed to download other HGT data sets from the NASA server, such as older SRTM releases.

Note that the NASA URL is not location specific and the default setting only needs to be changed if you wish to change the source data.

An example SRTM data file (N51W001.hgt) is included in the CityGen3D package for reference, which covers the default location and surrounding area of England.

In order to access the heightmap data, you will need an Earthdata login, which can be freely obtained here: https://urs.earthdata.nasa.gov/users/new/

Once you have signed up simply enter your ***NASA Username*** and ***NASA Password*** into the fields provided on the Heightmap window. Then when you attempt to download data in CityGen3D, your request can be authorized automatically.

Click the Download Heightmap button to automatically download the HGT files that are needed for your selected Landscapes.

The compressed files are automatically unzipped and placed in your Unity project here:
`Assets\CityGen3D\Resources\Data\HGT`

If you have previously downloaded HGT files in another project, you can manually transfer them between projects and place them in this folder so you don't have to download them again.

Offset allows you to align the downloaded heightmap data to OSM data if you find there's a slight mismatch. The defaults of 0,0 should be fine in most instances.

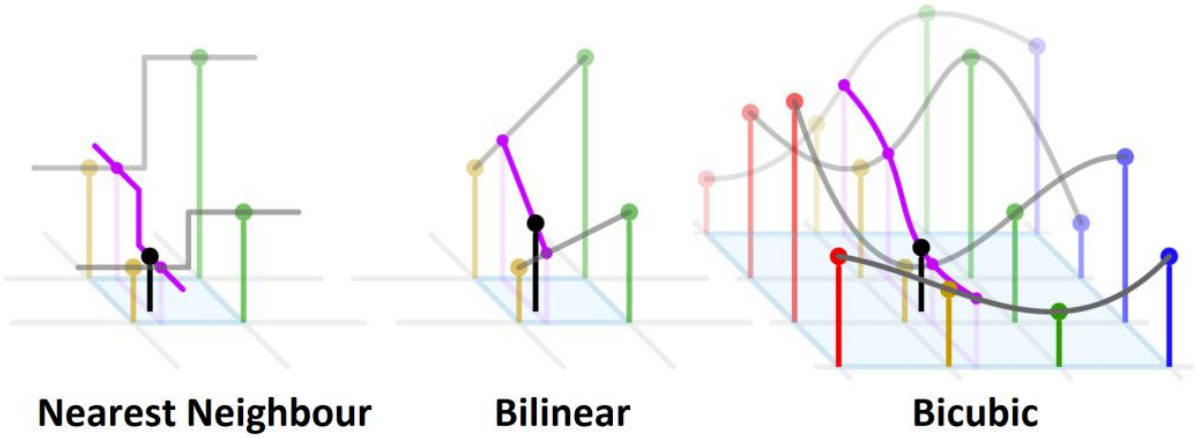
Water Depth adjusts how far below sea level the terrain can be.

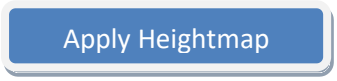
Peak and ***Nadir*** describe the highest and lowest points of your terrain in meters. These are important settings because a Peak that is too low for your environment will artificially cap the heightmap at this height. It is important to consider these values at the start of your project to ensure the heightmap range is as small as possible, which will allow for smaller height increments to be represented on your terrains. So although a peak of 10000 meters would be good enough to cover all regions on Earth, it would result in poorer quality heightmaps due to the reduced granularity of each increment.

Tick the ***Auto Range*** checkbox to set these two values automatically based on the range of your downloaded heightmap data and your chosen Water Depth setting.
You should untick this option and set the Peak and Nadir manually if you intend to build your environment using multiple data sets with a shared origin. This is because all terrains in your project need to have the same height range in order for them to align with each other seamlessly.

Because heightmap data was obtained via radar, it can contain unwanted artefacts such as picking up the height of tall buildings on otherwise flat terrain. This can make it difficult to determine realistic terrain altitudes from the available data, especially in urban areas.
To get around this problem, the *Filtered* ***Sampling*** mode examines all raw samples around a location and takes the lowest available one, effectively filtering out erroneous data. Therefore it is advised to use this sampling mode in built-up areas to smooth out the terrain. The alternative Raw sampling mode can be used away from cities for higher definition where erroneous data from manmade structures is less likely.

Raw heightmap data is sampled at relatively low resolution at about one sample every 30 metres. So in order to produce a high resolution heightmap CityGen3D interpolates the raw data in one of three ways, chosen via the ***Interpolation*** listbox.
Bicubic is the default and results in a smooth heightmap using spline interpolation of the surrounding raw samples.
Bilinear is a bit faster, but results in a more angled low-poly style heightmap due to linear interpolation between raw samples.
Nearest Neighbour just uses the closest sample at each point resulting in a grid like terrain, which is intended for the visualization of raw data rather than realistic terrain generation.





Click the Apply Heightmap button to read downloaded HGT files, interpolate the raw data, and apply it to your terrains. If you are notified about a missing HGT file, click Download Heightmap first.

6.3 Heightmap - Perlin Noise

If you are making a completely custom environment, not based on a real world location, or want to use a different heightmap for a real world location, you can use a heightmap generated using Perlin Noise.

Reduce the ***Scale*** setting to create more frequent changes in elevation, or increase it to flatten out your terrain.

Click the Apply Heightmap button to apply your Perlin Noise settings to selected terrains.

You can then proceed to use Unity Terrain Tools as normal to modify your terrain heightmap further, as well as using the additional CityGen3D Heightmapping tools described below.

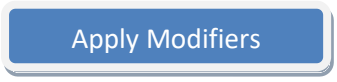
If you don't Apply Heightmap because you want a completely flat terrain, it is advisable to still set up the range of your terrain (peak/nadir) to give you flexibility later on should you wish to apply modifiers. It is also advised to use the Flatten functionality in Unity to ensure your terrain is not all set to the lowest value, as this prevents negative modification, such as digging out terrain for roads and lakes.

6.4 Heightmap - Modifiers

Heightmaps are sampled from real world data at relatively low resolution and interpolated between these samples to create a smooth heightmap. However, CityGen3D is designed for high detail rendering of environments, so you can make up for this lack of precision by adding in extra detail to the heightmap using various options, based on the OpenStreetMap data (or your custom 2D map).

One such option is to apply surface modifiers to the downloaded heightmap. For example, it is recommended to have a negative ***Altitude Offset*** for each water surface type. This will effectively dig out the land for lakes and rivers in your scene.

In order to prevent distortion of roads and pathways caused by modifying the heightmap, offsets are applied only when far enough away from them. Unticking the ╢ checkbox removes this protection and gives priority to the heightmap offset instead of nearby highways.



Click this button to apply the height offsets to the selected terrains.

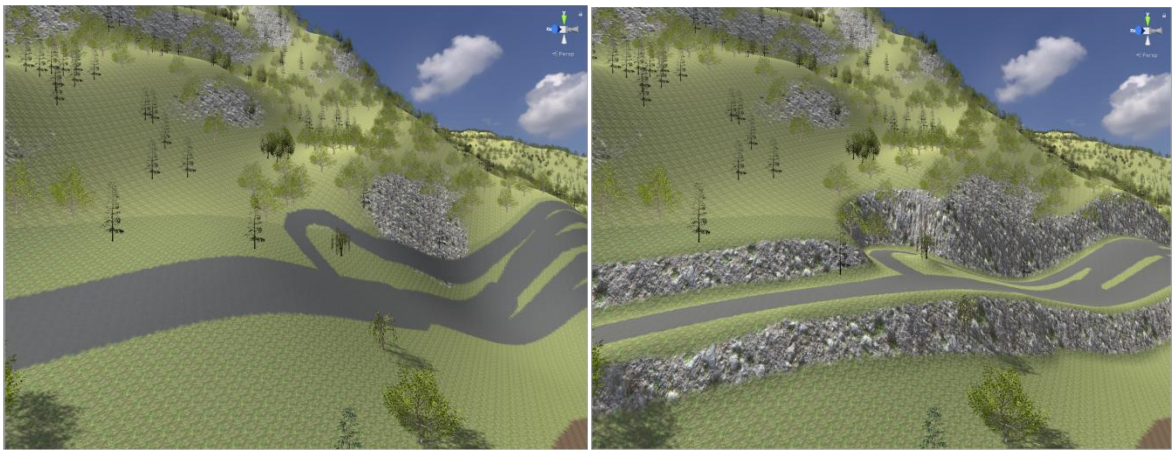
6.5 Heightmap - Levelling

Hilly landscapes can result in roads being at unrealistic angles on the terrain. If you imagine a hill and a road running around that hill, by default, the road will be slanted from left to right at the gradient of the hill.
The levelling can help reduce this making road networks on hilly terrain much more realistic and appear as if built into the landscape, just like real life.

The number of ***Iterations*** determines how many times the levelling routine is run. Every iteration checks each point on the heightmap for road or path. Where it is found, the altitude is changed to the average height of all road within the surrounding ***Sample Size*** radius. Over several iterations this will have the effect of levelling out the terrain. The ***Road Size*** setting determines the maximum distance from a road that is considered still be road for the purposes of levelling. So you can increase this value to level out more of the surrounding terrain, if desired.

Apply Levelling

Click this button to apply road levelling across all selected terrains.



Levelling: Before & After

Check out a video showing road levelling in action at https://youtu.be/ZQx8aFmGtms.

6.6 Heightmap - Smoothing

After applying an offset through the Modifiers interface, or levelling out the heightmap for roads, you will likely want to smooth the heightmap to remove steep vertical drops. This works in the same way as Unity's built in smoothing tool, but instead of having to paint manually on the terrain, the smoothing is applied uniformly across the heightmap.

To apply more smoothing, increase the ***Iterations*** setting, which controls how many times the smoothing is applied.

There are also options for adjusting the ***Brush Size*** and ***Brush Falloff***.
Notice how the brush image changes as you move the Falloff slider, giving you control over the brush intensity.

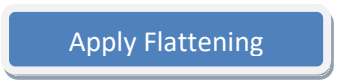
Apply Smoothing

Click this button to apply smoothing evenly across all selected terrains.

6.7 Heightmap - Flattening

Placing prefabs over uneven terrain can result in unwanted gaps appearing between the base of the object and the terrain mesh.
After placing prefabs over your terrain, either manually or automatically using the Roadside panel, you can use the Flattening feature to fix this for you.

For example, to flatten terrain under buildings you would set the ***Apply To*** layer mask to include the Buildings layer.
The ***Radius*** setting controls how far away from the prefab the flattening is applied. The lower the radius, the steeper the incline on the terrain.



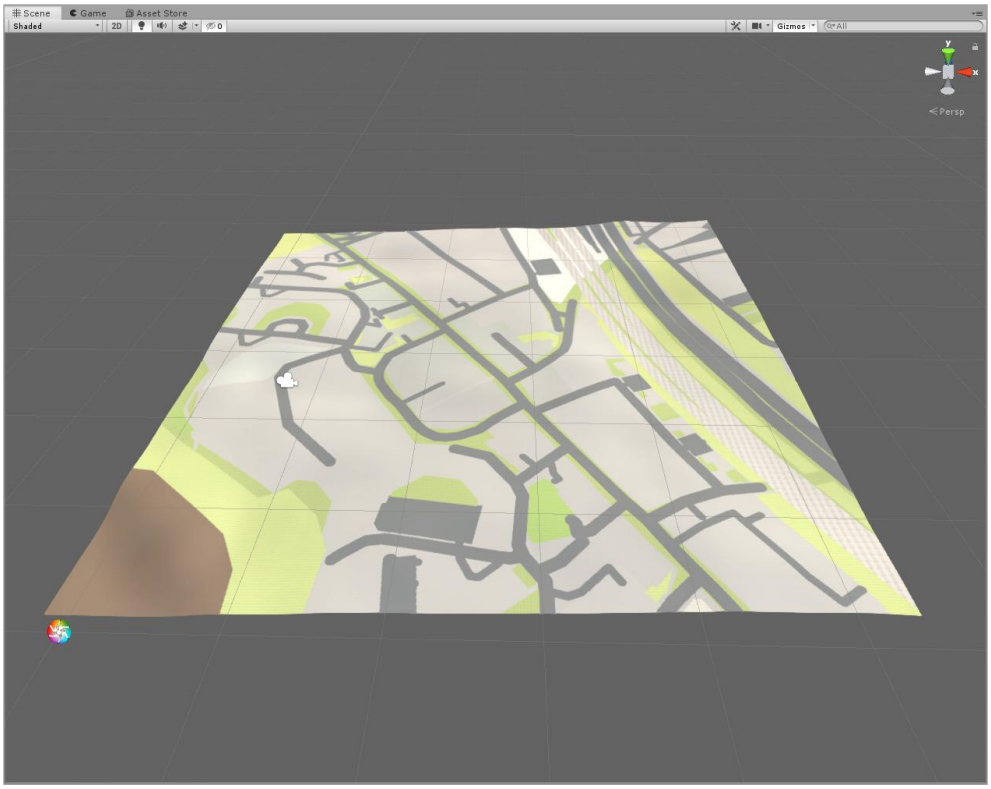
Click this button to apply flattening on selected terrains, under all colliders associated with objects in specified layers.

7. SplatMap

Terrains in CityGen3D can be automatically textured based on your material library and surface rules that you define.

Texture Base Surfaces

To texture your terrains using your chosen materials and rules for each surface type, click this button while one or more Landscapes are selected in the scene. Your terrains will then appear textured appropriately, similar to the image below.



Texture Custom Surfaces

It is also possible to detect surface types from spawned prefabs and texture them appropriately. Take a look at the House 05 prefab in the CityGen3D Prefabs folder. That has a Lawn plane with a Map Surface component attached. The surface type is set to Grass to ensure that when the prefab is spawned you can click this button to texture the area the lawn covers as defined by your texturing rules for Grass. You could have hundreds of these houses and all their lawns would be painted appropriately with one button click. So when designing your prefabs consider using this terrain technique sometimes instead of adding additional geometry.

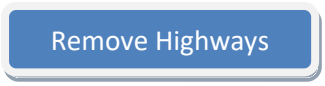
8. Highways

The unique way in which roads are created in CityGen3D allows for complex road networks to be generated automatically. You can make adjustments to the look of your roads by interacting with the Highways panel.

Each road type can be associated with different road markings using the Markings Distribution controls. Allowing you to customise it quite easily for different locales.

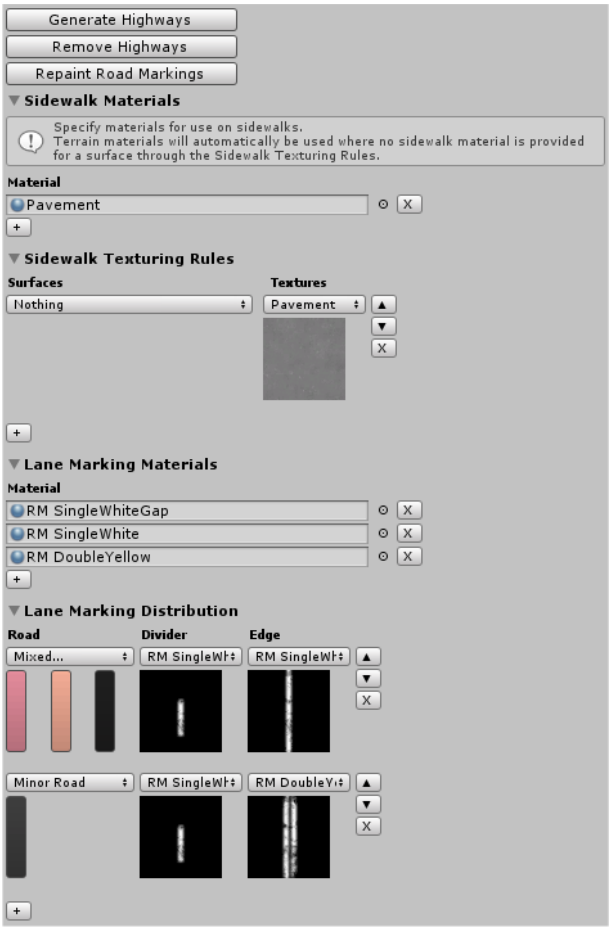
By default, sidewalks will be textured with the same material as specified on the Splatmap panel for the same suface type at that location.
You can override this with specific sidewalk materials if required.

Click this button to create your roads, which will automatically generate suitable batched geometry and offset the terrain heightmap as appropriate.



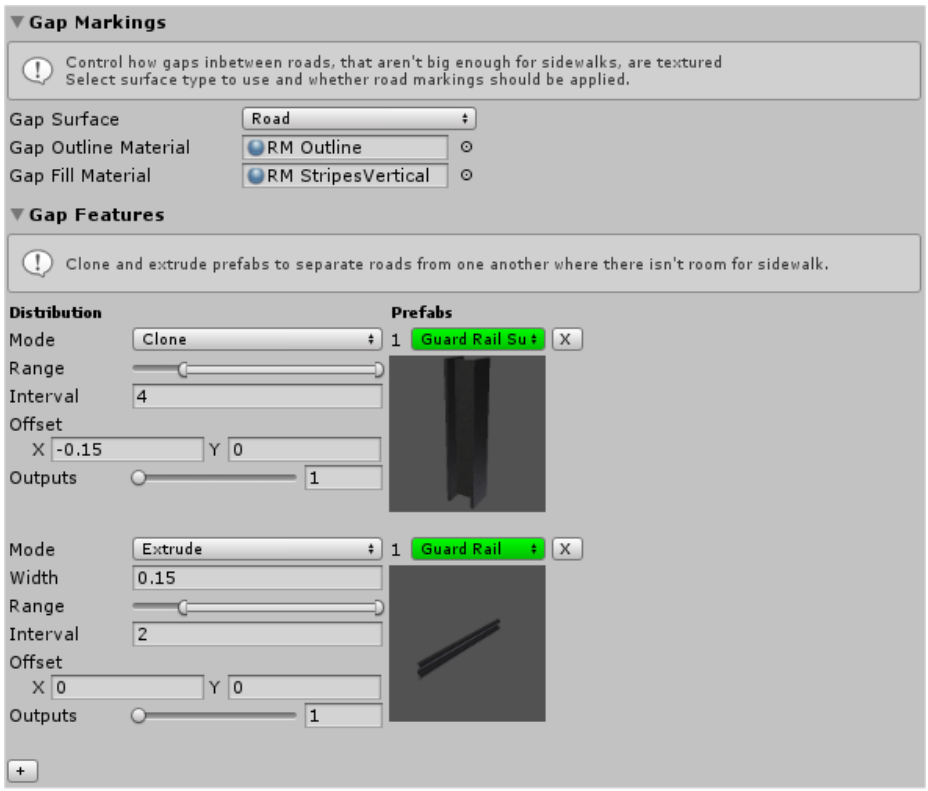
Remove existing roads from selected landscapes.

If you make changes to a road's marking settings via the Inspector then you can ensure these get repainted by clicking this button.



In CityGen3D, roads are generated by creating geometry to form blocks where no road exists. The road itself is simply the terrain mesh with road marking decals on it.

These blocks are created by extruding sidewalks, but in some cases it's possible to get gaps in between roads that are not large enough for a block, because it's not big enough to extrude a sidewalk all the way around it.
These are referred to as Gaps and you can control what happens to these separately for some really nice effects.



Gaps can have their own markings and you can even extrude meshes or clone prefabs around them, such as guard rails or traffic cones.

They are textured according to the ***Gap Surface*** type you assign to them and an outline is drawn around them using the ***Gap Outline Material***, effectively indicating the edge of the road.
In addition the ***Gap Fill Material*** allows you to put hatched markings within the gap itself to better indicate a "no go area".

Note that CityGen3D is not currently able to determine a suitable orientation for the Gap Fill texturing automatically based on road direction. So you may want to substitute RM StripesVertical with RM StripesHorizontal on a case by case basis by dragging it onto the mesh manually for now, where needed.

Gap Features are one of two types.
A ***Clone*** is just an instantiated prefab spawned at specified ***Interval*** around the gap outline.
Extrude can be used to repeat a mesh around the gap outline.
Both are used in the demo setup to produce a guard rail.

The ***Range*** setting provides a simple way of including or excluding gaps from additional geometry based on their size.
The default settings are setup so that smaller gaps do not have guard rails, but longer ones will.

In the above screenshot we can see how the gap between the slip road and the freeway is not deemed to be long enough to extrude a guard rail, so only the Gap Markings are applied to it.

But the long central reservation between the two motorways in the following screenshot falls within the range to trigger the extrusion of the guard rail mesh and cloning of the support beams.

9. Buildings

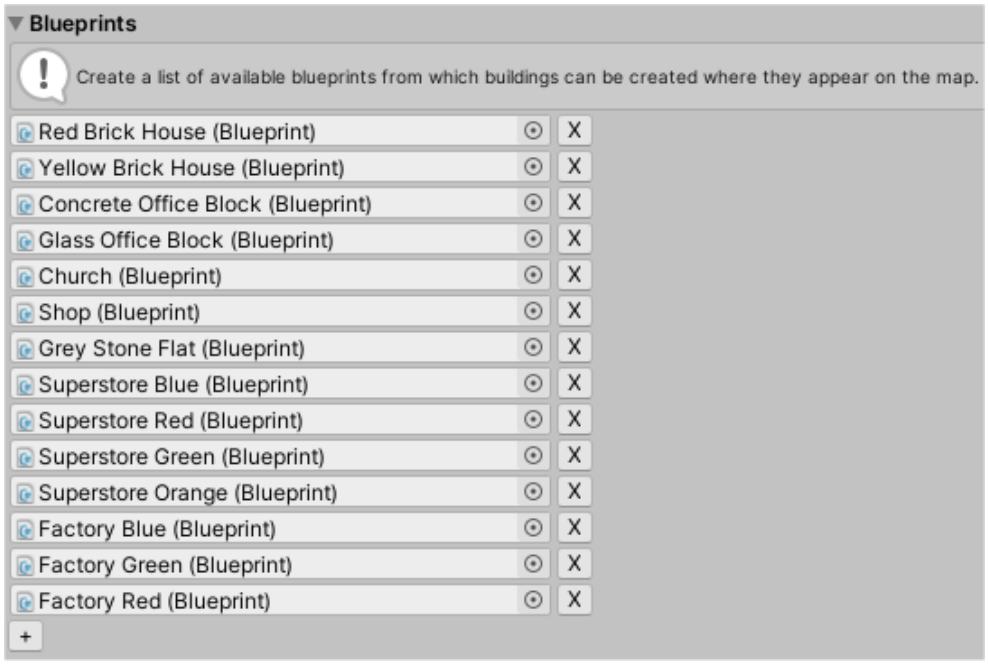
Buildings created using the Buildings module of the Generator are extruded from the 2D map shape that defines them. All their geometric detail comes from a heightmap texture, which gives you a lot of freedom to change a buildings appearance simply by changing the material applied to it, and without having to remodel a mesh.

Procedural buildings have no interior and are primarily intended to be background models, such as incidental buildings in a racing game or a flight sim. For more detailed buildings that the end user will interact with, you can spawn detailed prefabs using the Roadside module.

9.1 Buildings - Construction Rules

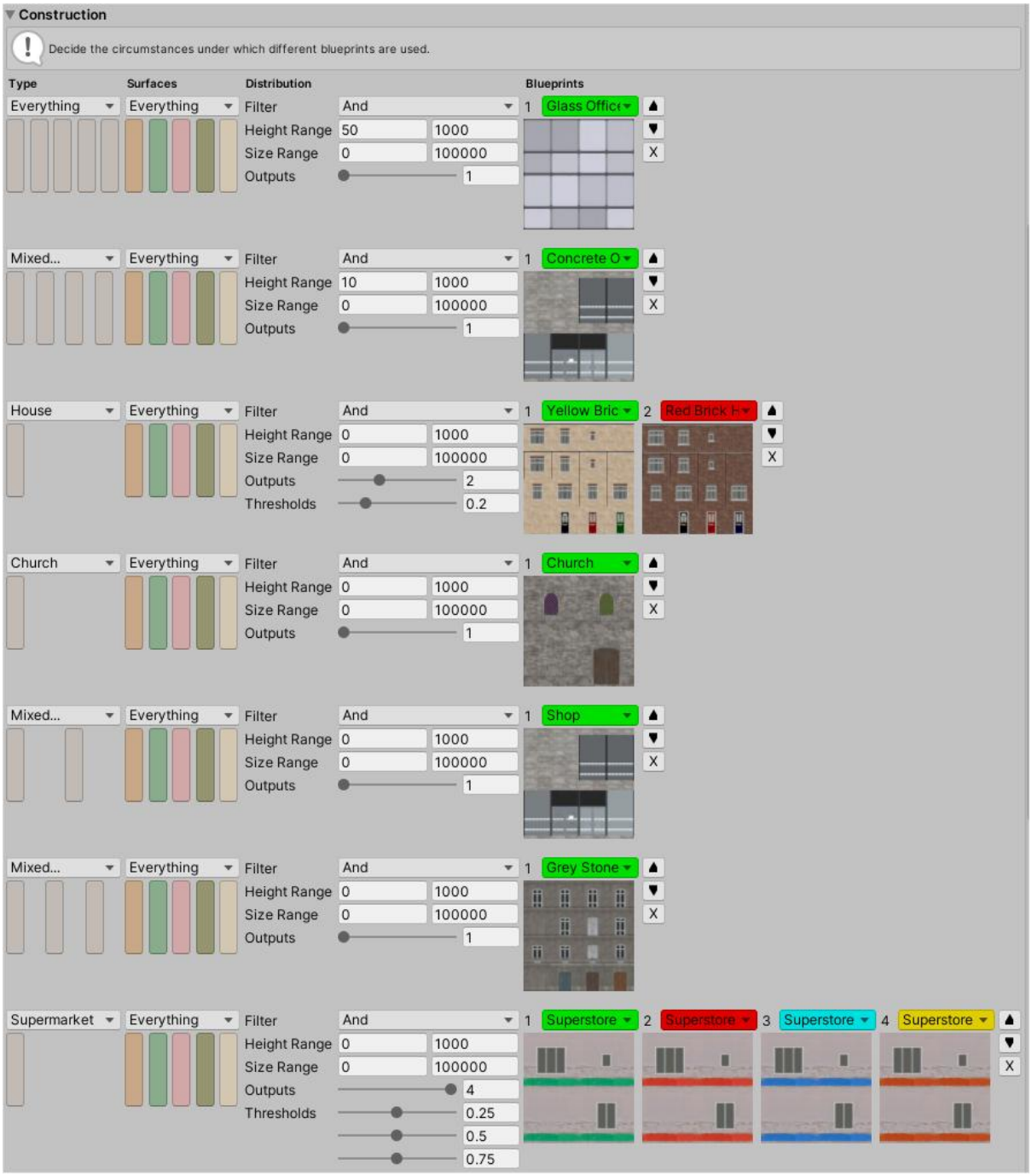
While the shape of any particular building will be unique and come from the map data, the purpose of a Blueprint is to tell CityGen3D how to make up a building's physical appearance when it is generated in 3D.

Each type of building that is available is listed on the ***Blueprints*** section of the Buildings panel. You can add and remove Blueprints using the [+] and [x] buttons, because it is possible to define your own Blueprints, but for now we'll just use the ones supplied with CityGen3D.



The ***Construction*** section of the window provides an interface for matching up these Blueprints to different types of building using construction rules. Each row in the list represents one rule.

When buildings are generated, each of these construction rules will be checked to see if a match is found, so the order that the construction rules are listed in is important. Rules listed at the top are tested before those at the bottom. You can move a rule higher or lower in the list using the arrows to the right of each one and you can delete a rule with the [x] button.



In some cases, a building will already have been identified via OSM tagging and will be assigned an appropriate building type such as Supermarket or Church.

In this situation you can match up a 2D map building with a suitable Blueprint just by selecting the building type from the ***Type*** list for the given rule and selecting an appropriate Blueprint from the list.

However, you will find that not all buildings are individually tagged in OSM with a descriptive type, but we can often infer a suitable 3D appearance from identifying what surface they are above.

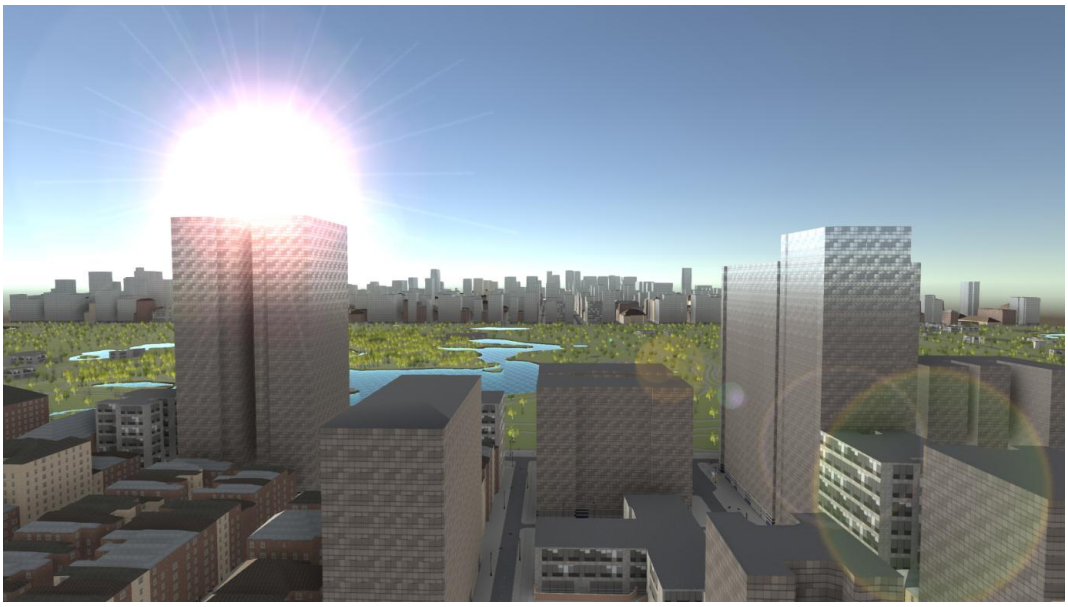
This is where the ***Surfaces*** selection for the rule is useful. If we know nothing about a building other than its location over a mapped Industrial surface, we can assume it's more likely to be a factory than a house and select an appropriate Blueprint on this basis.

To find even more different building types in OpenStreetMap data, take a look at the Buildings tab of the Data window, which allows you to match up OSM tags with new or existing CityGen3D building types. (Changes on the Data tabs require you to Process your OSM data again to refresh the 2D map).

It is possible to narrow down your match making even further by specifying a valid ***Height Range***. For instance, the default CityGen3D generator profile is setup with a rule to identify any building over fifty metres tall so it is matched against the Glass Office Blueprint for a classic skyscraper style appearance.
Likewise the ***Size Range*** setting can be used to identify buildings that cover a large area, irrespective of their height.
Use the ***Filter*** setting to control the logical relationship between the height and size ranges, which will determine whether both height and size must be in range for a match, or just one.

When a match is found a building will be generated using the matching rule. But this doesn't have to mean that the Blueprint used is always the same.
Notice how in the default setup, Supermarkets and Residential buildings make use of the ***Outputs*** setting to select slightly different blueprints based where a random number falls within the specified ***Thresholds***.

Using a combination of these filters and rules gives you a lot of flexibility to create different types of buildings in different situations, even when the raw OSM data itself may be lacking some detail.



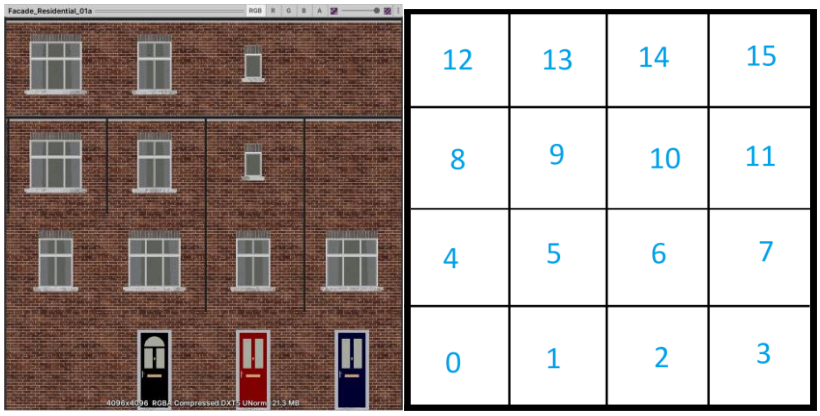
Generate Buildings — Click this button to create 3D buildings on selected landscapes.

Remove Buildings — Remove existing 3D buildings from selected landscapes.

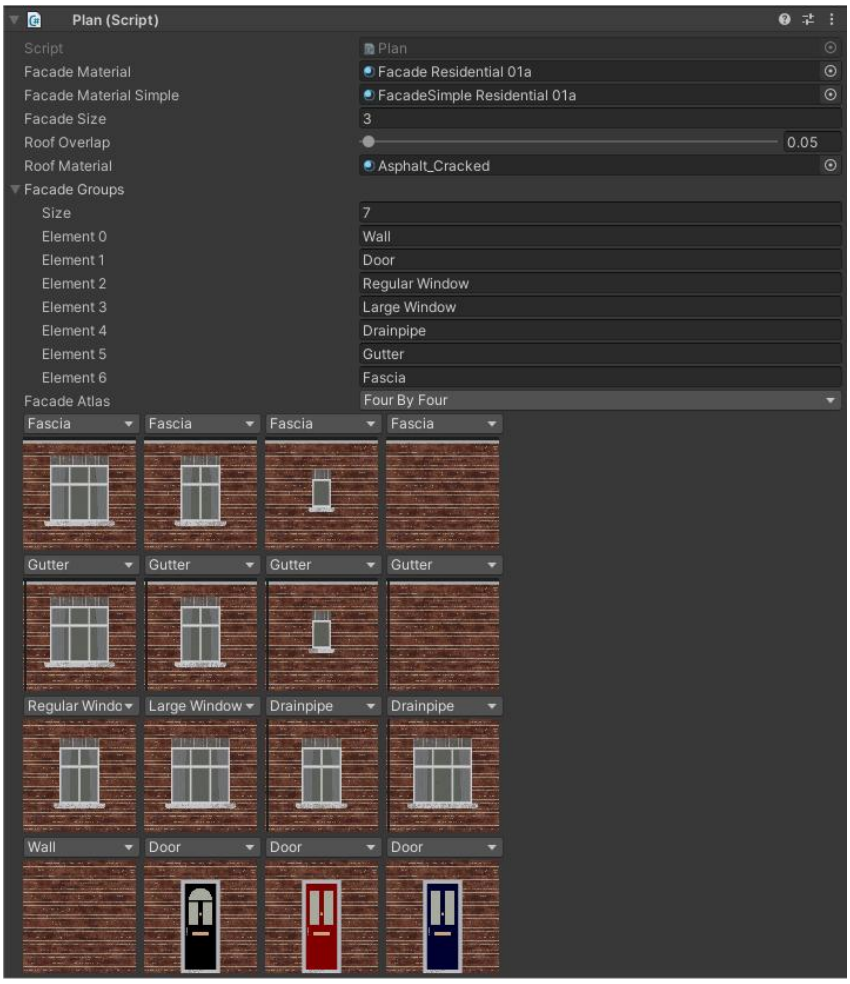
9.2 Buildings - Facades & Plans

A texture atlas called a ***Facade Material*** is applied to all procedural buildings and is paired with a Plan prefab which helps to give meaning to the different parts of the texture.

Textures in a facade material must be of equal size and are numbered starting with zero in the bottom-left. So for a Four By Four atlas, we have sixteen textures like this:

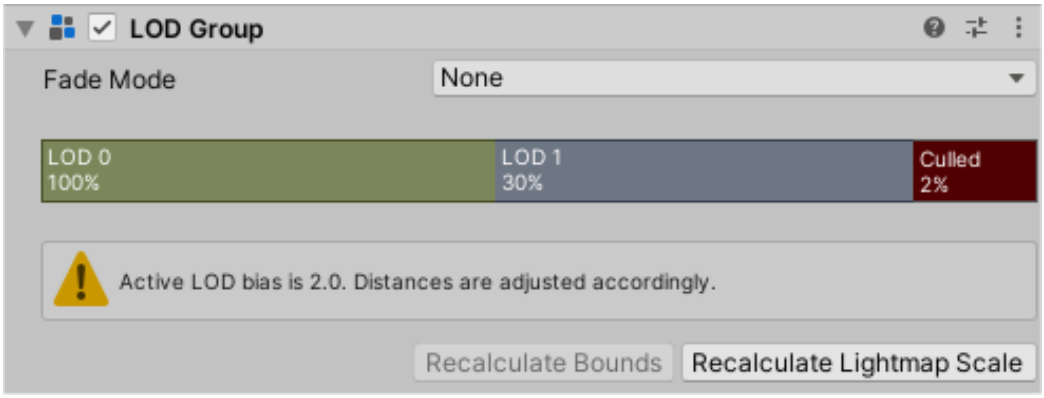


A Plan references two materials that use the same images, but a different shader. The ***Facade Material Simple*** version doesn't use tessellation and is applied to the LOD1 mesh of the building, whereas the tessellation shader is applied to the LOD0 mesh for more detail close up.



Both LOD0 and LOD1 meshes are automatically created for each building and appropriately referenced in the building's LOD Group when you click the Generate Buildings button.

The default Level Of Detail settings for buildings is controlled globally by editing the LOD Group component on the Building prefab found in the Resources\Prefabs folder.



Use the LOD sliders as normal to change the thresholds at which buildings will be rendered using different meshes, or culled completely.

If you are targeting mobile devices or just want to optimize the rendering even further, you can delete one of the LOD Groups. If CityGen3D only finds LOD0 listed in this LOD Group when it generates buildings, it will create only the simple version of the building which has a lower rendering overhead and a simpler shader.

After buildings have been generated, the LOD Group for a building instance can be edited as normal via the Inspector, so this prefab just provides the initial settings that are applied to all.

Facade Atlas is reference by a Blueprint so it knows how many unique textures can be found on the material. You'll notice that the residential material has sixteen arranged in a 4x4 grid, whereas the office material has only four in a 2x2 grid.

When creating your own materials you should balance the detail required against the variation needed for different building types. The more images in the atlas, the more diverse the buildings can be from each other, albeit at a lower resolution.

Each texture in the atlas is assigned a named ***Facade Group***, which allows similar textures to be grouped together. You can have as many or as few groups as you like, and these descriptions are specific to an individual plan.

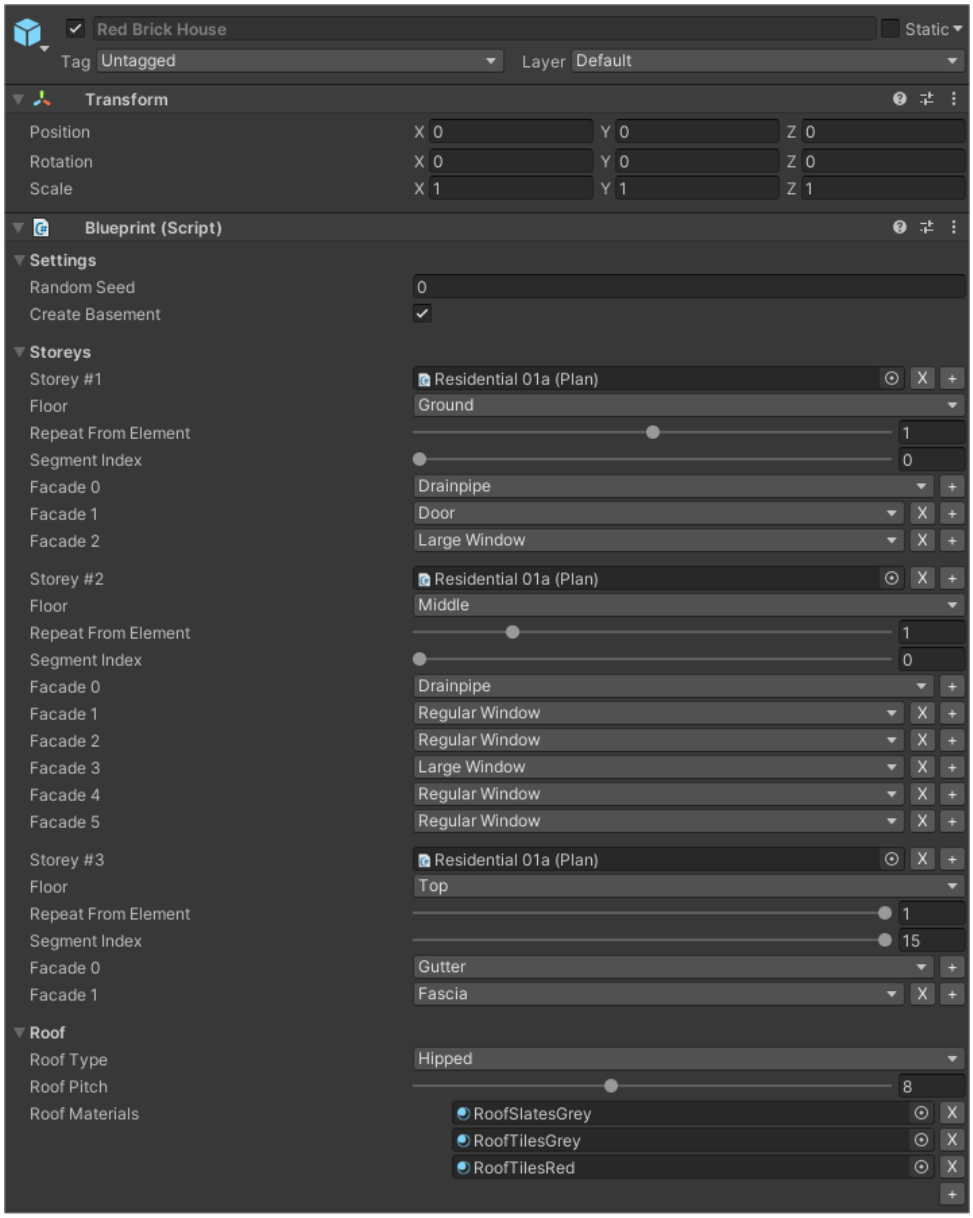
Because each facade on the extruded building mesh is assigned a random texture from within a specified group, these groupings should be used to identify textures that have a similar function. For instance in the sample Residential facades, all three door textures are assigned the "Door" Facade Group.

The ***Facade Size*** indicates how many metres tall and wide a facade texture from the material is in 3D space when a texture from the atlas is applied to the building mesh. So this is the setting where you can effectively control the storey height for a building based on its facade material as well as how often it repeats around a wall.

9.3 Buildings – Blueprints

In addition to the facade materials, the physical appearance of a procedural building is defined by a Blueprint prefab. CityGen3D is packaged with a selection of Blueprints, but you can easily create your own, either by copying an existing one or adding a Blueprint component to an empty game object.

To make changes to a Blueprint, click on the prefab and view the Inspector, just like any other Unity prefab. Shown below is the blueprint for "Red Brick House".



When a 3D building is created the bottom of the ground floor is approximately aligned to the highest point of terrain that the building touches. You can use the ***Create Basement*** option to create a floor below which will fill in the gap between uneven terrain and the bottom of the ground floor. It's advised to leave this checked unless you are generating your environment on a perfectly flat terrain.

Every building instance is initialized with a ***Random Seed***, which gives predictable results for random selections during its creation. Therefore each building has a deterministic appearance based on this value in combination with the other Blueprint settings.

The use of ***Storeys*** is where everything is brought together to define a specific layout for a level of a particular building type. Each storey is assigned a Plan and given an order of facades to use to make up that level of a building. We'll examine the first storey of the Red Brick House blueprint in more detail to demonstrate how you have control over the makeup of the ground floor.

The ***Plan*** is just a reference to the one you wish to use for this storey of the building. We are calling this blueprint Red Brick House so the Residential 01a plan is used, which consists of suitable facades. We also wish to define the ground ***Floor***, so this has been selected from the list box.

We can now indicate which facades to use in what order when making up the ground floor for this building type. These facade selections refer to the Facade Groups as defined on the chosen Plan, and you can add and remove facades with the [+] and [x] buttons.

For this ground floor three facades have been selected: Drainpipe, Door, and Large Window. CityGen3D will add a texture from each selected Facade Group in turn, but note that the ***Repeat From Element*** slider is set to 1 and not 0. This ensures that there will only be one Drainpipe texture selected and the rest of the ground floor facades will be made up of Doors and Large Window textures, so we end up with something like this:

It is likely a wall of a building will not be perfectly divisible by the facade size. To ensure windows and doors are not unrealistically clipped at the end of a wall, every Storey has a Segment Index which refers to a texture number from the Plan used to fill in any gaps at the start and end of a wall.

In this case texture 0 is used from the Residential 01a plan, which is a plain wall material suitable for clipping when a shorter segment is required.

Currently only flat and hipped roof types are supported but additional roof types are planned for the future. If Hipped ***Roof Type*** is selected you will have the option to adjust the ***Roof Pitch*** represented as a ratio of rise to run, with the run said to be 12. Therefore a pitch setting of 12 would create a 45 degree slope.

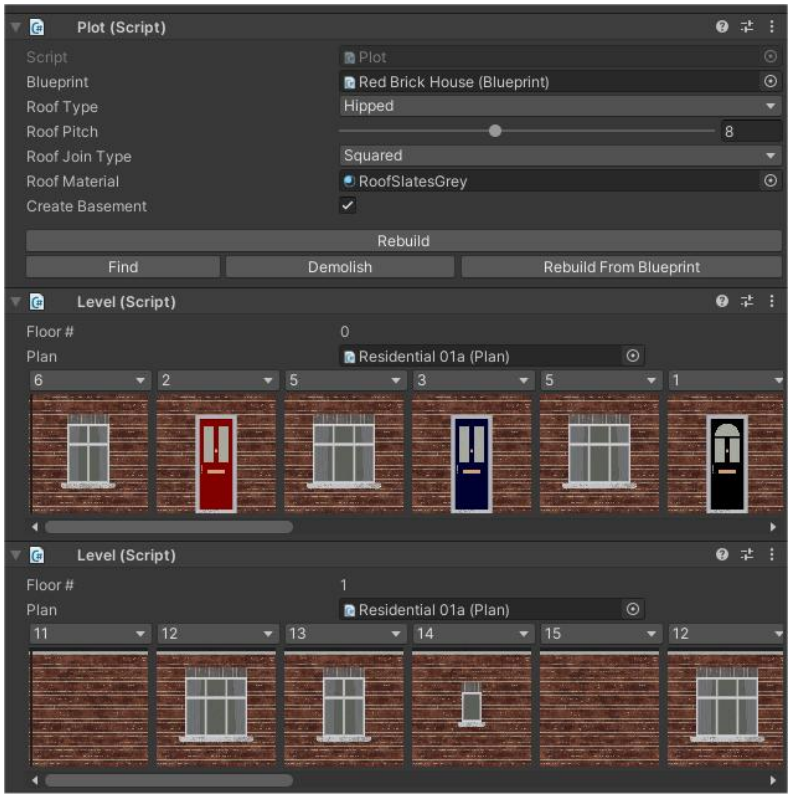
The ***Roof Overlap*** setting can be used to extend the roof over the top of the building.

A roof material is selected at random from the list of ***Roof Materials***. You can use any seamless material for these.

9.4 Buildings – Plot & Level Editing

It is possible to make changes to a 3D building after creation to customise it further. For instance you may wish to move a door so it appears on a different facade, or change a window to different style.

Click on a 3D building in the scene making sure you have selected its root game object and not one of its child meshes. Attached to this object will be a Plot component and changes to a building can be made by editing this component via the Inspector.



The Plot settings allow customization of the selected building, so unlike Blueprint settings, any changes to a Plot only refer to a specific building. Hence only one Roof Material field is required instead of a list, before you are being explicit about the chosen material on a particular building.

Rebuild	Click this button to rebuild the building based on changes you have made to the attached Plot and Level components.
Rebuild From Blueprint	This regenerates the building based on the assigned ***Blueprint*** and ignores any other changes made to the building instance.
Find	Jumps to Map view and focuses on the MapBuilding linked to this building game object.
Demolish	Removes the building from the scene but retains the associated map building.

You can quickly find a 3D building using the 2D map by clicking the Find button on the MapBuilding component of a building while in 2D map view, in the same way the Find button on the Plot component provides a shortcut the other way.

In addition to the Plot component, a building will also have one or more ***Level*** components attached that show the currently applied textures for the specified floor number. You can change these selections to a different texture on the facade material and click the Rebuild button to apply the changes, perhaps to remove a window or move a door somewhere else.

So while a Storey definition on a Blueprint acts as a template defining how a floor of a building should be created, with random textures selected from the specified Facade Groups, a Level component provides an interface for editing the specifics of a particular floor. Any changes to a Level are therefore only for the attached building, just like the Plot settings.

This video also demonstrates the editing of a building: https://youtu.be/LjkRiUncoGM

10. Runtime

In a multi-terrain environment you'll want to consider scene streaming so that nearby terrains are loaded at run-time when they are close to the camera, and unloaded when the camera moves far away from them.

If you have saved your landscapes as individual scenes (see Heightmap - Terrains) then CityGen3D is able to manage this process for you using the Landscape Manager at runtime.

The performance of asynchronous loading and unloading of scenes by Unity is vastly improved in final builds when compared to running in Play mode in the Unity Editor.

This tutorial video walks through a lot of the topics covered in this section regarding large environments in CityGen3D: https://youtu.be/hWtdqj5TicY

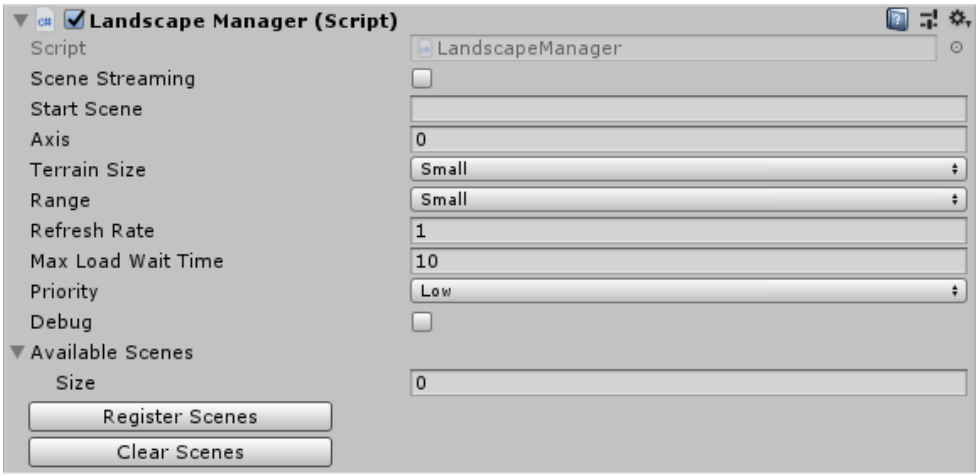
10.1 Runtime – Landscape Manager

When you create a new environment in CityGen3D you have a main scene that has your primary objects in such as the camera, the map data, the Generator, and the Landscape Manager attached to the Environment prefab.

If you started your project by loading the New City scene, an Environment object with Landscape Manager component will already be included in your scene. Otherwise you can manually add the Environment object by dragging it onto the scene from Assets\CityGen3D\Resources\Prefabs.

Note that the Environment prefab also contains a directional light (called Sun), a wind object, a global post processing volume, and a reflection probe. So you'll want to ensure you don't have unnecessary duplicates of these in your scene.

Let's take a look at the Landscape Manager in more detail by clicking the Environment prefab in your main scene and looking at the Inspector:

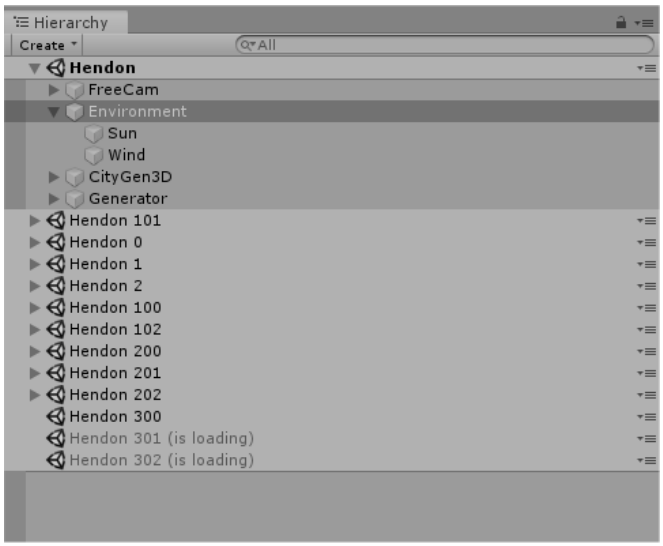


You can think of a Landscape as a terrain tile. Each one has a terrain component and acts as the parent object of all other objects above it. This makes it very easy for Unity to enable and disable different tiles as the user moves around the environment.

To tell the Landscape Manager to manage the loading and unloading of these terrain tiles at runtime, tick the ***Scene Streaming*** checkbox.

Enter the name of your main scene in the ***Start Scene*** field. CityGen3D will use this to identify which scenes to load.

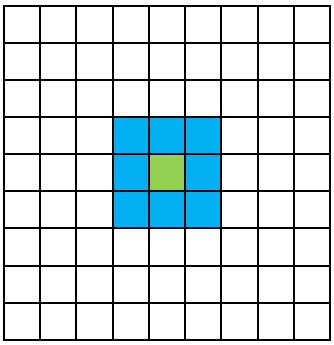
In this example you can see that we downloaded data for Hendon, which became the name of our start scene. All Landscapes created from this data were saved as separate scenes and share this name, but also have a numeric identifier to indicate which terrain tile the Landscape is for.

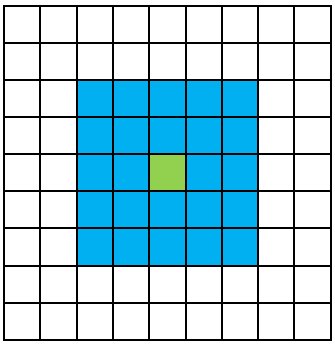


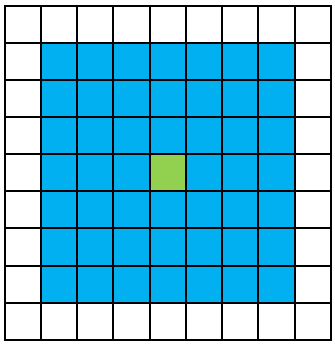
The ***Axis*** setting needs to be the same value as specified when you downloaded the data. The default is 100. Likewise, you should set your ***Terrain Size*** to match the size of your terrains. It is important these two values are set according to the specifics in your environment and are the same as what you chose when you downloaded your data.

The Landscape Manager uses asynchronous additive scene loading and unloading to keep the landscapes around the camera loaded, and to unload those that are far away.

The bigger the selected ***Range***, the more landscapes are retained in memory, in a grid layout around the landscape below the camera.

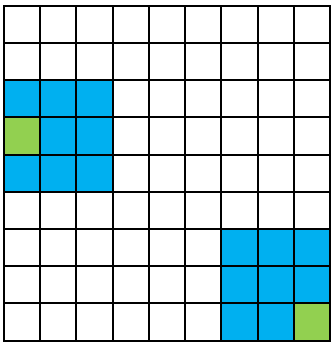






The range respects world boundaries, which are determined by the Axis setting. This keeps the same amount of terrains loaded even when on the edge of the environment.

So with an Axis of 9 (81 terrains) and a small range of 1, this image shows how terrains would be loaded if the camera was above the terrain marked in green in these two examples on the edge of the available environment:



Because the Landscape Manager can identify which scenes to load based on location, it doesn't need to use colliders to trigger scene loading and unloading. Instead it will periodically check to see if the landscape below the camera has changed. How often this is done is controlled by the ***Refresh Rate*** setting, which is specified in seconds and defaults to one.

The ***Max Load Wait Time*** setting provides a failsafe, also specified in seconds, for the maximum length of time Unity should wait for an asynchronous scene load to finish.

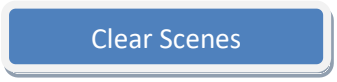
By default, the Landscape Manager will attempt to load scenes in the background with as little impact on performance as possible at the expense of speed. The ***Priority*** setting allows you to have some control over this by allocating more time per frame to asynchronous tasks, which can result in faster loading at the expense of frame rate.

As is the case in any Unity project, you will need to add all your scenes to the Build Settings, which tells Unity which scenes to include in your build. This should include all Landscape scenes you create from your downloaded data that you want CityGen3D to be able to load at run-time.

You also need to tell the Landscape Manager which scenes are available to load by adding the name of each one to its ***Available Scenes*** list. Note that you don't need to add all your project's scenes to this list, just those created when you Generate Terrains. The order they are added is unimportant.

There are two buttons provided to make this process really easy, so you don't have to write the name of each one yourself:

Click to add all available Landscape scenes from the Build Settings into the Available Scenes list.



Removes all references in the Available Scenes list, making it empty and resetting it to zero.

10.2 Runtime – Origin Manager

Even with the aid of asynchronous loading of the environment, implementing very large 3D worlds poses certain challenges.
One of these challenges is to find a way of preventing spatial jittering due to limited floating-point precision.
This is where graphics appear to jump around slightly the further away from the world space origin (0,0,0) their transforms are.

This occurs because a computer can't represent numbers with infinite precision. There is a distinct set of numbers, which are not distributed evenly, and the gap between each number increases with its size and distance from the origin.
This lack of precision as an object moves away from the origin manifests itself in this jittering effect.

CityGen3D attempts to solve this using the Origin Manager, which periodically moves everything in the active seen so the camera is back at the origin.
This happens seamlessly because everything moves by the same amount and all objects retain their positions relative to each other, but it keeps your active environment close to the origin where there is high precision.

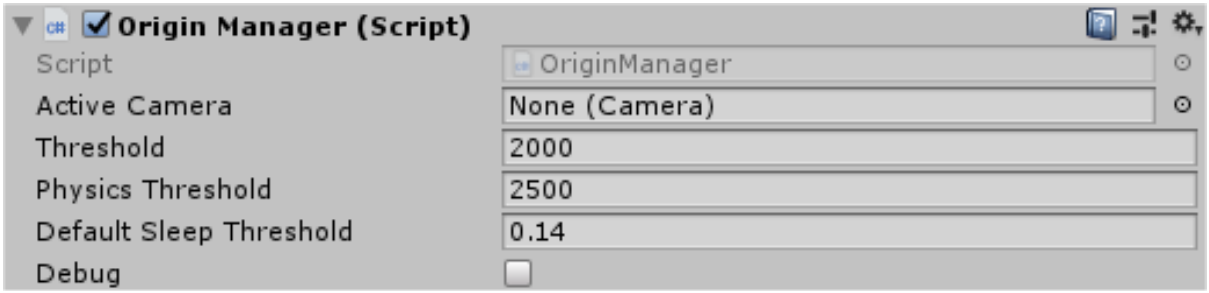
To setup the Origin Manager, simply assign your camera to the ***Active Camera*** field in the Inspector. If you don't do this, CityGen3D will attempt to find it for you at run-time.

The maximum distance the camera is allowed to move relative to the origin before this reset occurs is determined by the ***Threshold*** field. As soon as this is exceeded, the mass translation occurs.

The ***Physics Threshold*** setting specifies a distance at which physics is disabled. A value of zero prevents this and allows physics to be enabled for all objects.

Remember that the Origin Manager needs to be able to move all your objects around in the scene, so you need to ensure your objects are not marked as Static in the Inspector.

If you do not want CityGen3D to manage the origin in your project simply remove the Origin Manager component from your Environment object.



11. Coastline

In order for CityGen3D to know where land and sea is in your location, a separate coastline data file should be referenced using the ***Coastline GeoJSON*** field on the Processing section of the Location panel before you click the Process button to parse the OSM data.

An example coastline data file is provided called "UK & Ireland [-11_49_2_63].json".
As the filename suggests, this is suitable for locations between Longitude -11 to 2 and Latitude 49 to 63, which is large enough to accurately process anywhere in the United Kingdom or Republic of Ireland.

Processing coastline data can take a while, but if your location is not coastal, data processing will run faster without a Coastline GeoJSON and you can leave this blank.
In this case, CityGen3D will automatically create a rectangular landmass for you.

Note that rivers and lakes can be processed without a JSON file, because all the information for those will be included in your downloaded location data from earlier. This extra coastline data is only needed for land/sea boundaries, where your downloaded OSM data sample is not able to resolve large landmasses by itself.

You can also bypass the Coastline processing by unticking the appropriate option in the Processing Filter.
For example, when generating an environment in England it's good to have the aforementioned UK coastline file assigned to Coastline GeoJSON so it can be referenced as needed. You can then toggle the processing off using the Coastline checkbox if you are processing inland terrains.

If your desired location covers a land/sea boundary outside of UK and Ireland, then you'll need to download an appropriate coastline GeoJSON file using the following steps.

For this example, we'll create a much smaller coastline file to cover the Scilly Isles. Regardless of your location, all steps remain the same, aside from step 9 where you would specify an appropriate Latitude/Longitude bounding box to cover your chosen area.

1) Go to https://osmdata.openstreetmap.de/data/land-polygons.html
2) Download the Large Polygons Are Split version of the WGS84 data from that website. Link at time of writing is: https://osmdata.openstreetmap.de/download/land-polygons-split-4326.zip
3) Extract the downloaded zip to a suitable location. This data won't be used by Unity so it shouldn't be in a Unity project folder.
4) Go to https://mapshaper.org/
5) In MapShaper, Import the extracted data by clicking the Select button and selecting the files within the extracted folder.
6) In the Import options, ensure both "detect intersections" and "snap vertices" are unchecked. These aren't needed and may slow things down a bit.
7) Click Import, wait a minute or two, then you should see a map of the world appear in MapShaper.
8) We now want to reduce the scope of our map to only cover the area of interest for your CityGen3D project. Click the Console button in MapShaper to show the command line interface.

9) Type the following command followed by Return, noting that the only space is between "clip" and "bbox" keywords. The bounding box is defined in the format MinLon,MinLat,MaxLon,MaxLat

```
-clip bbox=-6.5,49.8,-6.2,50.0
```

10) After a while, the map should refresh to only show your clipped region. You can click the Home button (house icon on right) to zoom in and focus on your left over region. Then we want to export it in GeoJSON format ready for CityGen3D using this command:

```
-o format=geojson
```

11) Give the file an appropriate name such as "ScillyIsles.json" and copy it to your Unity Project. You can then assign this json file to the Coastline GeoJSON variable in the Processing section of the Location panel in CityGen3D.

Try to keep your coastline file as small as possible by choosing a clipping bounding box that just covers what you need for your project. The provided UK & Ireland file is very large just for the convenience of having it cover the whole region for demonstration, but you can use much smaller areas for faster processing.

The following graphic demonstrates why coastline data is required. The image on the left is what the generated 2D map for the island of St Mary's looks like having processed the island with no coastline data. It results in a solid rectangular landmass, albeit with recognizable features overlaid.

The image on the right is the resulting map when it is processed with the ScillyIsles.json, which identifies the shape of the island correctly due to the island being within the bounds of the coastline data.

12. Standalone Builds

Eventually you'll want to test a build of your project instead of running it in the Editor.

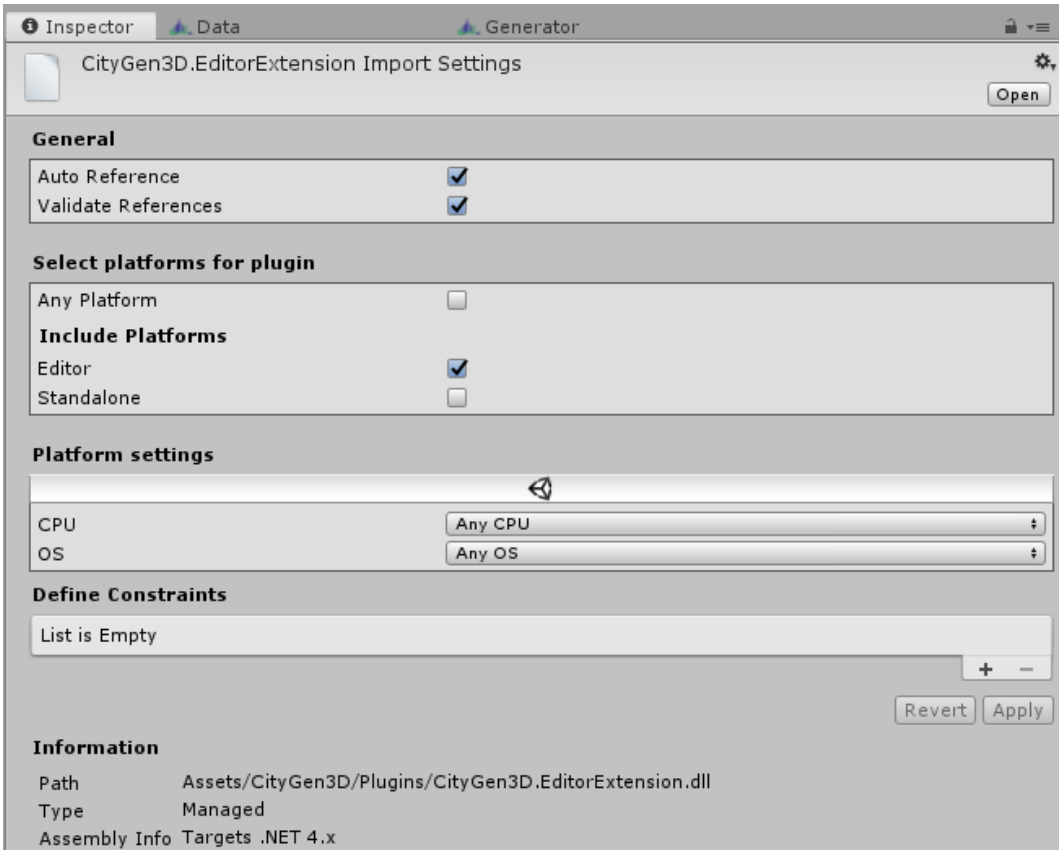
This is particularly necessary when making very large projects as you'll see much improved performance from Unity's asynchronous scene loading in standalone executables, compared with running inside the Unity Editor in Play mode.

Don't forget to build 64-bit executables when building larger projects. For Windows builds you simply select x86_64 Architecture from the Build Settings.

All the code that is dependent on the Unity Editor libraries, and therefore not needed in release builds, is compiled into a separate dynamic-link library (DLL).

You should find that this DLL is excluded from Standalone builds automatically, but if required, you can do this by locating the CityGen3D.EditorExtension.dll file in the Assets/CityGen3D/Plugins folder from within your project, using the Project view inside the Unity Editor.

Click on the CityGen3D.EditorExtension.dll file to view its import settings and ensure it is excluded from standalone builds with an appropriate checkbox, as shown here:



If this is not setup correctly, you will see a build error reporting that CityGen3D.EditorExtension is referencing UnityEditor.dll and can't be included in the build.

If you receive an error saying “Burst compiler failed running”, this means you need to modify your Visual Studio installation to include the Windows 10 SDK and VC++ Toolkit.

You can do this via the Visual Studio Installer as shown here: