Billboard Generator

Documentation

Table of Contents

1. Introduction

Billboard Generator is an editor and run-time utility that allows you to generate billboard textures for any object in Unity.

What is a billboard?

”In computer graphics, billboards are textured rectangles that are transformed such that they always appear parallel to the view plane. Thus, they are similar to billboards along highways in that they are rotated for best visibility. However, they are different from highway billboards since they are dynamically rotated to always offer best visibility.

The main use of billboards is to replace complex three-dimensional models (For example: grass, bushes, or even trees) by two-dimensional images. In fact, Unity also uses billboards to render grass. Moreover, billboards are often used to render two-dimensional sprites. In both applications, it is crucial that the billboard is always aligned parallel to the view plane in order to keep up the illusion of a three-dimensional shape although only a two-dimensional image is rendered.”

(Source: https://en.wikibooks.org/wiki/Cg_Programming/Unity/Billboards)

2. API

Billboard Generator uses a very simple application programming interface (API) for both the editor and the run-time.

As long as you’re using the plugin inside the editor you’ll mostly not need to write any code unless you want to perform custom operations on the generated billboard textures before they’re written to the disk, in that case you can use the built-in `Generator` static class.

That class has some properties that allows you to customize all aspects of the billboard generation process any they’re:

- **Directions** (Integer): How many directions to capture billboard textures from?

- **TextureWidth** (Integer): The width of the generated billboard texture.

- **TextureHeight** (Integer): The height of the generated billboard texture.

- **TextureBackgroundColor** (Color): The background color of the generated billboard texture (Must always have an alpha value of 0.0f other wise artifacts might appear in the result billboard texture).

- **CameraClearFlags** (CameraClearFlags Enum): The clear flags of the camera used to render the billboard textures.

- **CameraOffset** (Vector3): The distance that the rendering camera should stay away from the model while rendering.

- **BillboardAxis** (BillboardAxis Enum): On what axis should the billboard rendering camera rotate while rendering the billboard textures? (For example: You might want billboards that are from rotation around the Y-axis, X-axis or both of them).

Also, there’s the “Generate” method which is the method that takes the properties above along with some parameters and generates billboard textures, the parameters are:

- **gameObjects** (IEnumerable<GameObject>): Any generic enumerable object that has GameObject as it’s generic parameter.

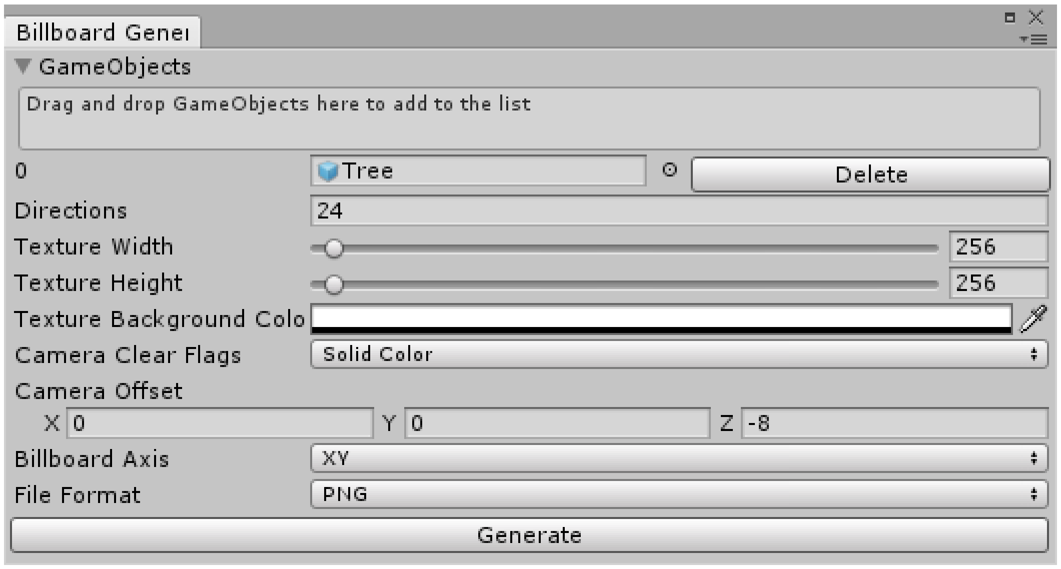
- **begin** (Action) [Optional]: An action that’s invoked when the generation process starts.

- **progress** (Action<Texture2D>) [Optional]: An action that takes 1 parameter of type Texture2D. It’s invoked every time a texture is generated during the generation process.

- **error** (Action) [Optional]: An action that’s invoked when an error occurs during the generation process (The error will be printed to the Unity Console window BEFORE this action is invoked).

- **finish** (Action) [Optional]: An action that’s invoked ONLY when the generation process has been completed successfully.

3. Using Billboard Generator in the Editor



Billboard Gene

GameObjects

Drag and drop GameObjects here to add to the list

0 | Tree | Delete

Directions: 24

Texture Width: 256

Texture Height: 256

Texture Background Colo

Camera Clear Flags: Solid Color

Camera Offset: X 0 Y 0 Z -8

Billboard Axis: XY

File Format: PNG

Generate

(The Generator GUI wrapper)

The previous image shows the Generator class GUI wrapper window. Each of the fields represent a static property inside the Generator class (Already covered in the API section) except the **File Format** field and the **Generate** button.

The **File Format** field allows you to specify the output file format for all the generated billboard textures while the **Generate** button calls the **Generate.Generate(...)** method.

(Covered above in the API section)

Once you click generate, you’ll be asked to choose a directory to save the generated billboard textures in.

The generated texture file will have the following name style:

{GameObject.name}_Billboard_Texture_{ zero-based index }

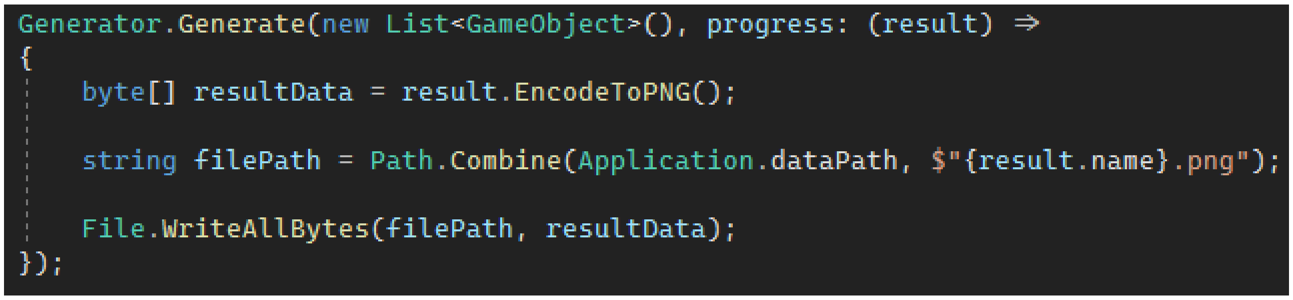
NOTE: When choosing output directory, if you choose a directory that has files with the same name (and extension) as the textures that will be generated THESE FILES ARE GOING TO BE OVERWRITTEN.

4. Using Billboard Generate at Run-time

Since the core Billboard Generator does not rely on any external dependencies or editor functionality, it can be used at run-time by calling the **Generator.Generate(…)** method.

(Covered above in the API section)

The only thing that you must handle yourself in this case is how to save the generated textures. Since you don't have access to file/folder dialog (s).



```
Generator.Generate(new List<GameObject>(), progress: (result) =>
{
    byte[] resultData = result.EncodeToPNG();

    string filePath = Path.Combine(Application.dataPath, $"{result.name}.png");

    File.WriteAllBytes(filePath, resultData);
});
```

(Example texture saving code snippet)