

Asset Bundle Loader

By Kyle Gillen

(Last Updated: 6/30/2016)

Configuration

- The Asset Bundle Loader is dependent upon the Asset Bundle Manager, which can be downloaded from the following link:
- <https://www.assetstore.unity3d.com/en/#!/content/45836>
- Note that the AssetBundleSample folder is not needed, though you are free to leave it in your project if you'd like.
- Also note that you should keep this package up to date, as it will be improved by Unity in the future.

Configuration

- You must enable Asset Bundle Integration before being able to utilize the Asset Bundle Loader component.
- There are two ways you can do this:
 - From the top Unity Menu Bar, choose Assets -> Dynamic Loading Kit -> Enable AssetBundle Integration For Current Platform
 - Add an Asset Bundle Loader component to your scene, then click the “Enable AssetBundle Integration For Current Platform” button.

Configuration

- Note that integration is only enabled for the current platform, so if you change the build target, you will need to re-enable integration for the new platform.

Creating Asset Bundles

- To create asset bundles, you must first assign one or more assets to Asset Bundles.
- In the Dynamic Loading Kit, the Asset Bundle Loader is a type of scene loader, which means the assets you will assign to asset bundles will primarily be scenes.
- To aid you in this task, I have added some additional options to the Generate Scenes tool (note, you should no use Scene Generation Files instead of the Generate Scenes Tool).

Creating Asset Bundles

- You can create a Scene Generation File by right clicking a folder in your project hierarchy and choosing Dynamic Loading Kit -> Create Scene Generation File.
- The last set of options on this file are “Asset Bundle Settings”.
- Enable assigning of the scenes to asset bundles by checking the “Assign Scenes To Asset Bundles*” toggle.

Creating Asset Bundles

- Several more options appear when you check this toggle.
- Add Scenes To Build Settings
 - When using Asset Bundle scene loading, it is not necessary to add the scenes to the build settings, however, you may need to for some reason, and enabling this option will force the scenes to be added to the build settings. Bear in mind, this will increase your build size, negating one of the main benefits of using Asset Bundles.

Creating Asset Bundles

- Use Variants
 - If you wish to use Asset Bundle Variants, check this toggle. You can read about variants (and the new Asset Bundle system) [here](#).
 - Checking this option will make visible an additional field, “Variant Name”, which dictates the name of the variant the scene will be assigned to.

Creating Asset Bundles

- When you Generate your scenes, each scene that is created will be assigned to an Asset Bundle of the same name (though as per asset bundle requirements, if the first character in the name is alphabetical, it will be converted to lowercase).
- If you'd rather multiple scenes be assigned to the same asset bundle, you will need to manually assign them or create a tool to automate this, though note that the Asset Bundle Loader expects each scene to be in a separate Asset Bundle, so you will need to write a new loader to load the multi-scene asset bundles correctly.

Creating Asset Bundles

- The Naming Convention referenced in the “Output Naming Convention” must match the Naming Convention referenced on whatever World Grid will be using the Asset Bundles, or if one is null, the other must also be null (so they both use the default naming convention).

Creating Asset Bundles

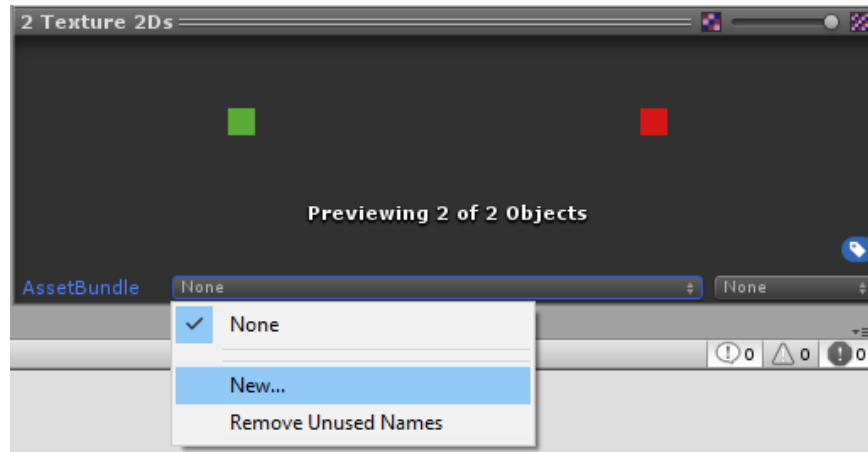
- The Scene Generation File does not generate any Asset Bundles, it merely assigns them so that you don't have to manually do it.
- Asset Bundle creation is left to the Asset Bundle Manager API (i.e., it's out of my hands).
- Before creating your asset bundles, however, there is one more step you should follow.

Creating Asset Bundles

- Identify all assets that will be shared between your scenes. This includes any textures used by your splat maps or custom shaders, meshes/grass textures used as detail prototypes on your terrains, or any other scenery that is parented to the main cell objects in each scene.
- These asset should be placed in a separate Asset Bundle that will be loaded automatically when needed by each individual scene's asset bundle.
- If you don't perform this step, these "shared" assets will be duplicated for each scene that uses them, which not only increases the memory size of each asset bundle, but also reduces the performance of using these assets, as Unity will treat them as separate objects at runtime, negating some built in performance optimizations, such as batching.

Creating Asset Bundles

- You can manually assign an asset (or assets) to an Asset Bundle (and Variant) by using the window at the bottom of the inspector when the asset is selected. Using this window, you can also create a new Asset Bundle name and/or variant.



Simulation Mode

- At this point, it is worth mentioning that building the physical Asset Bundles is not necessary if testing the Dynamic Loading Kit in the editor.
- Instead, you can use the Asset Bundle Manager's Simulation Mode, which will simulate asset bundle loading by retrieving the assets from the bundles directly from the project hierarchy.
- This is very useful and powerful, as it eliminates the need to recreate the physical asset bundles every time you make a change to an asset in the asset bundles.

Simulation Mode

- Simulation Mode is activated by choosing Assets -> AssetBundles -> Simulation Mode from the Unity menu bar.
- Additional details on Simulation Mode can be found at [this link](#).
- Please note, Asset Bundles must be built to be used in your game or any standalone build.
- Also note that Variants will not work in Simulation Mode.

Building Asset Bundles

- To build the physical Asset Bundle objects, select Assets -> AssetBundles -> Build AssetBundles from the Unity menu bar.
- This process may take some time.
- The Asset Bundles are placed in a folder called “AssetBundles” at the root of your project (next to the Assets folder, not inside).
- This placement allows for the use of the local asset server, but in practice you will want to move your asset bundles to another location (more on this later).

Building Asset Bundles

- Within this folder, you will see a folder named after whatever your current build target is (or any previous build targets you've used when building asset bundles).
- Within these build folders will be the asset bundles and manifest files themselves.
- This structure (BuildTargetName/list of bundles and manifest) is very important, as it must be followed regardless of where your asset bundles are placed.

Setting Up Folders

- As stated previously, the easiest way to work with Asset Bundles in the editor is by using Simulation Mode.
- There will come a point, however, when you need to use the physical asset bundles.
- The easiest way to do this is to use the Local Asset Bundle Server.

Setting Up Folders– Local Asset Server

- Like Simulation Mode, the Local Asset Server is there for testing purposes. In an actual game, you will want to use another option (which will be presented shortly).
- The main benefit of the Local Asset Server is that variants will work, and you can accurately test the loading of the physical asset bundles themselves.

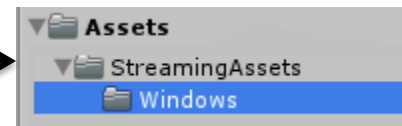
Setting Up Folders– Local Asset Server

- The local asset server can be enabled by selecting Assets -> AssetBundles -> Local AssetBundle Server from the Unity menu bar.
- In order for this to work, the Asset Bundles must be located in a folder called “AssetBundles” at the root of your project.
- Luckily, this is done automatically when using the Build AssetBundles command.
- In a standalone build, I believe you must select “Development Build” in your build settings, but I am not positive.

Setting Up Folders– Streaming Assets

- Another option for using Asset Bundles that can be used in a production game is the streaming assets method.
- With this method, the asset bundles are placed in a special folder called “StreamingAssets”, which must be directly under the Assets folder in your project hierarchy.
- You should think of the StreamingAssets and AssetBundles (used for the local asset server) folders as equivalent folders, just in different locations.
- The hierarchy of folders beneath each folder should match exactly.

Correct →



Setting Up Folders – Remote Server

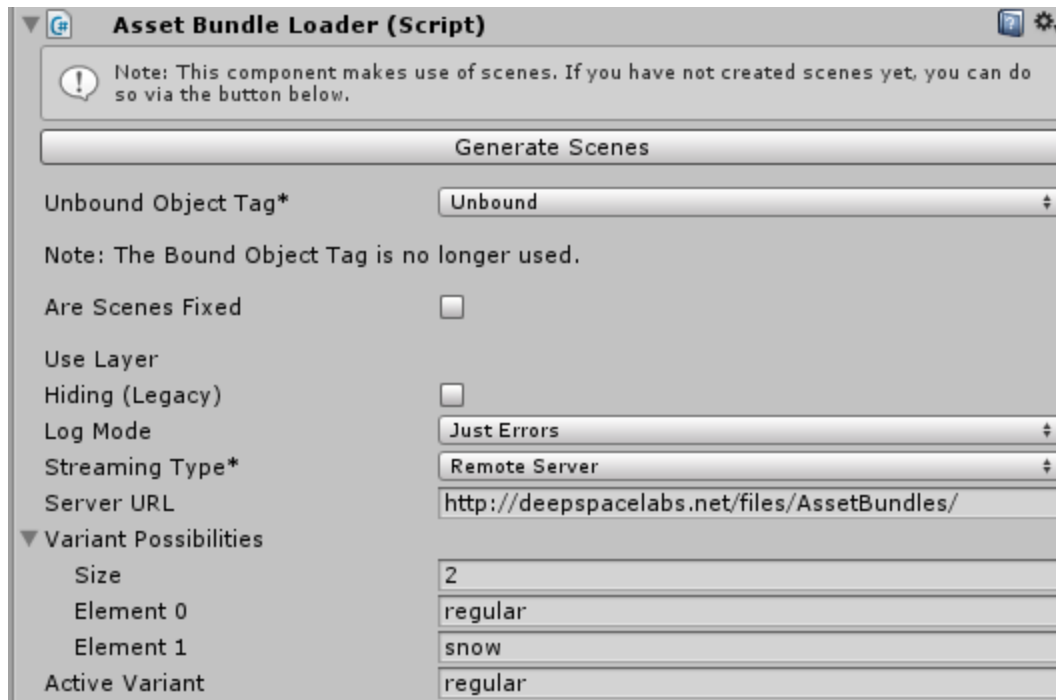
- The final and most likely method for using Asset Bundles is to place them on a remote server.
- This method can be used in the editor or standalone build.
- The folder structure on your server must match the structure found within the AssetBundles folder (i.e., Windows/list of asset bundles and manifest), though note that it's not necessary to have a folder called "Asset Bundles," and the folder structure of the parent directories does not matter.

Asset Bundle Loader

- The Asset Bundle Loader uses the high level Asset Bundle Manager API to load asset bundles, and as such it does not deal with memory management or other low level concepts. If you need a more fine grained loader, you will need to create it yourself.
- You can use the built in Asset Bundle Loader as a template for your custom loader.

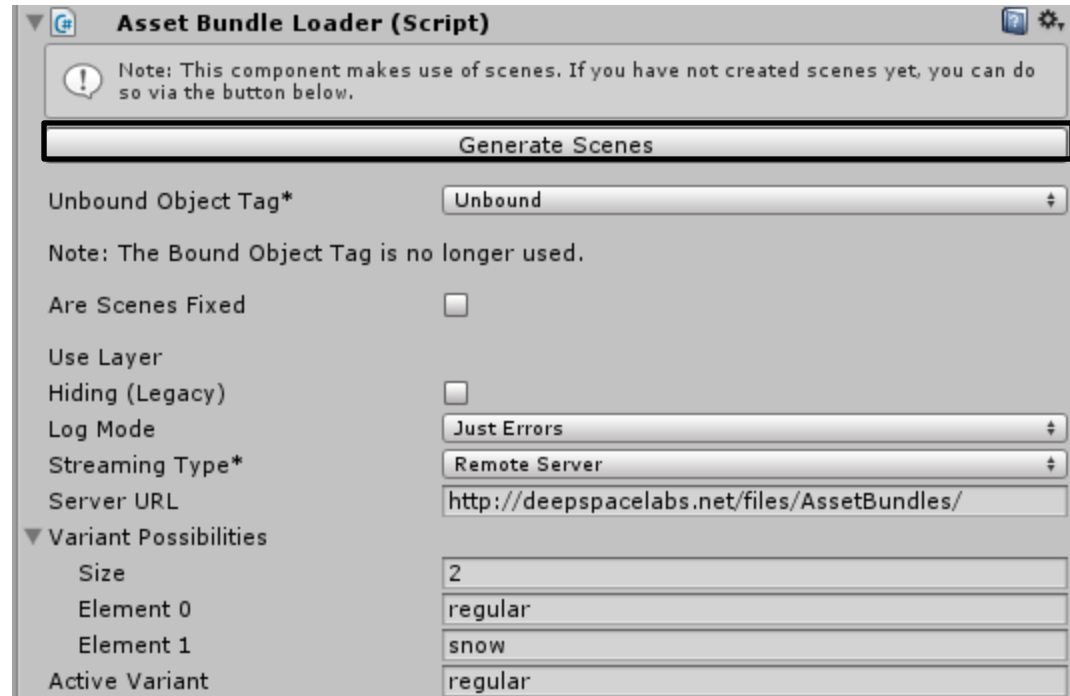
Asset Bundle Loader

- The Asset Bundle Loader will look similar to the below picture in the inspector:



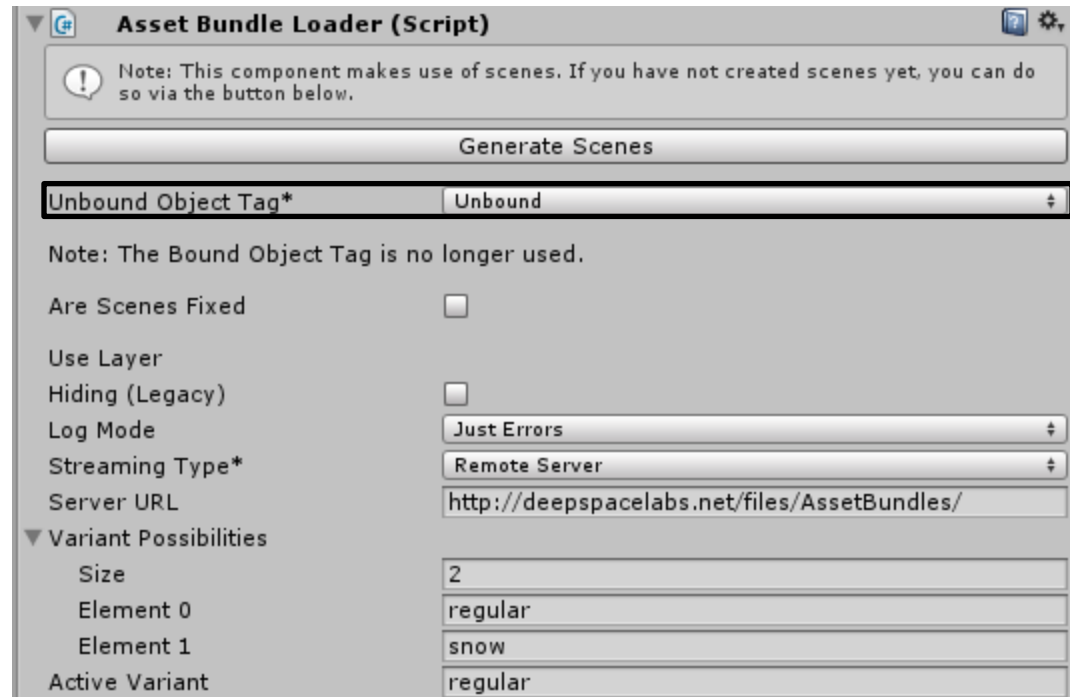
Asset Bundle Loader

Clicking this button will open the Generate Scenes Window (which is now obsolete).



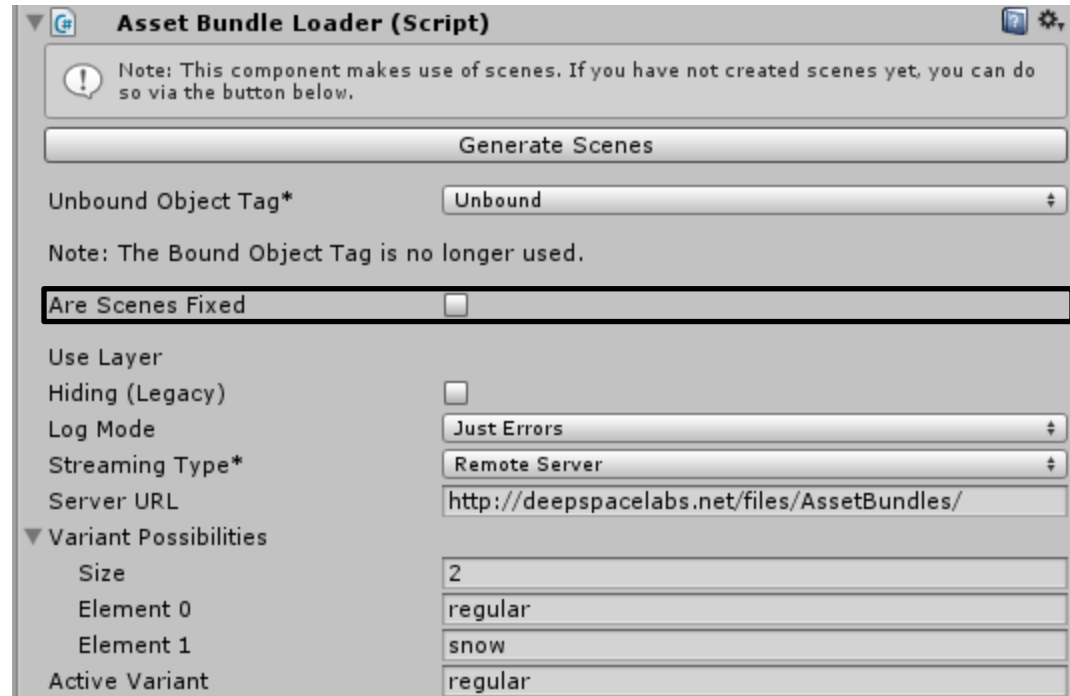
Asset Bundle Loader

This is the tag you should have specified in the Scene Generation File for the generated scenes. Each main cell object in every scene will be assigned this tag.



Asset Bundle Loader

Are your scenes fixed (i.e., static)? More information can be found in the `Dynamic_Loading_Kit_Quick_Guide`.

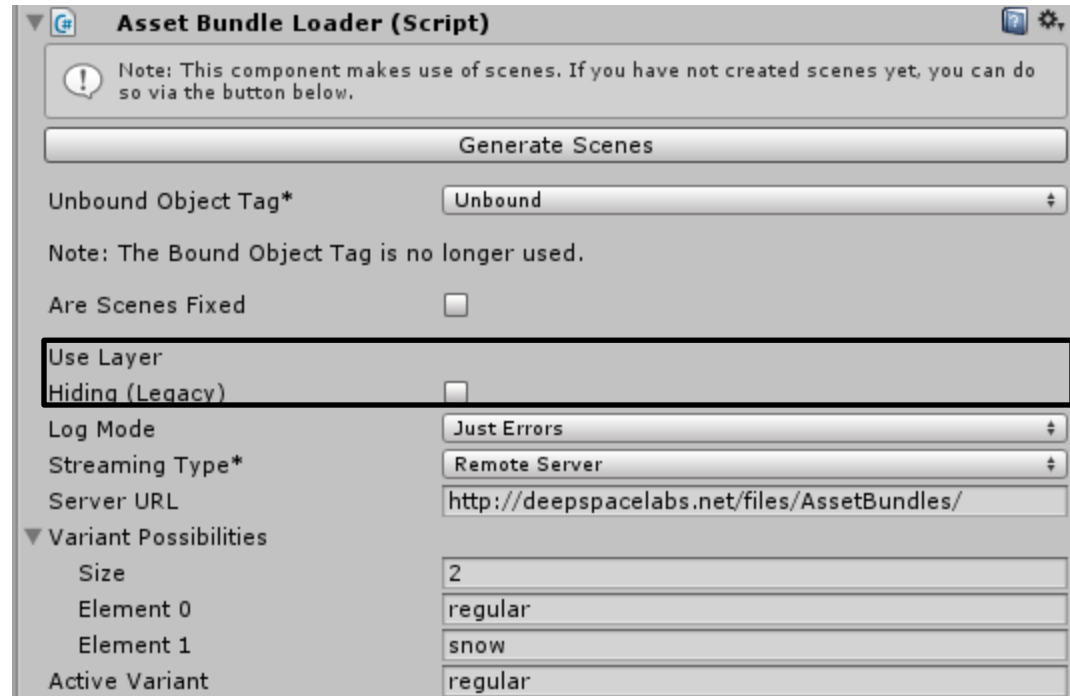


The screenshot shows the Unity Inspector for the **Asset Bundle Loader (Script)** component. At the top, a note states: "Note: This component makes use of scenes. If you have not created scenes yet, you can do so via the button below." Below this is a **Generate Scenes** button. The **Unbound Object Tag*** dropdown is set to **Unbound**. A note below it says: "Note: The Bound Object Tag is no longer used." The **Are Scenes Fixed** checkbox is currently unchecked and is highlighted with a black border. Other settings include: **Use Layer** (unchecked), **Hiding (Legacy)** (unchecked), **Log Mode** (Just Errors), **Streaming Type*** (Remote Server), and **Server URL** (http://deepspacelabs.net/files/AssetBundles/). A collapsed section titled **Variant Possibilities** contains: **Size** (2), **Element 0** (regular), **Element 1** (snow), and **Active Variant** (regular).

Property	Value
Generate Scenes	[Button]
Unbound Object Tag*	Unbound
Note	The Bound Object Tag is no longer used.
Are Scenes Fixed	<input type="checkbox"/>
Use Layer	<input type="checkbox"/>
Hiding (Legacy)	<input type="checkbox"/>
Log Mode	Just Errors
Streaming Type*	Remote Server
Server URL	http://deepspacelabs.net/files/AssetBundles/
Variant Possibilities	
Size	2
Element 0	regular
Element 1	snow
Active Variant	regular

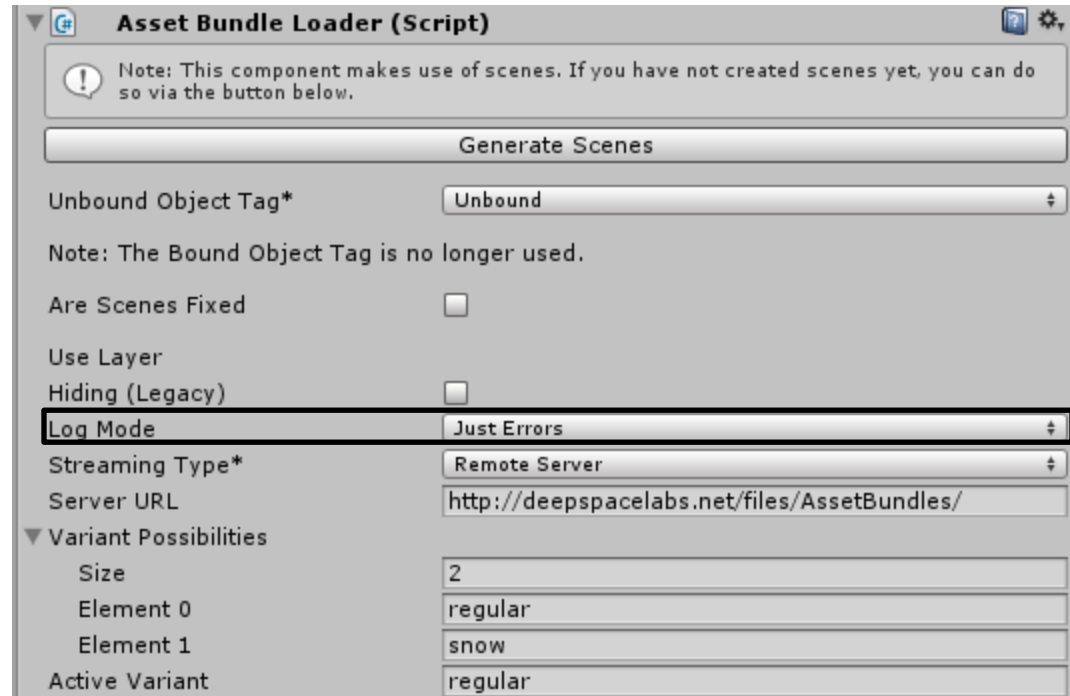
Asset Bundle Loader

If your scenes are not fixed (i.e., static), are you using the legacy Layer Hiding System? More information can be found in the [Dynamic_Loading_Kit_Quick_Guide](#).



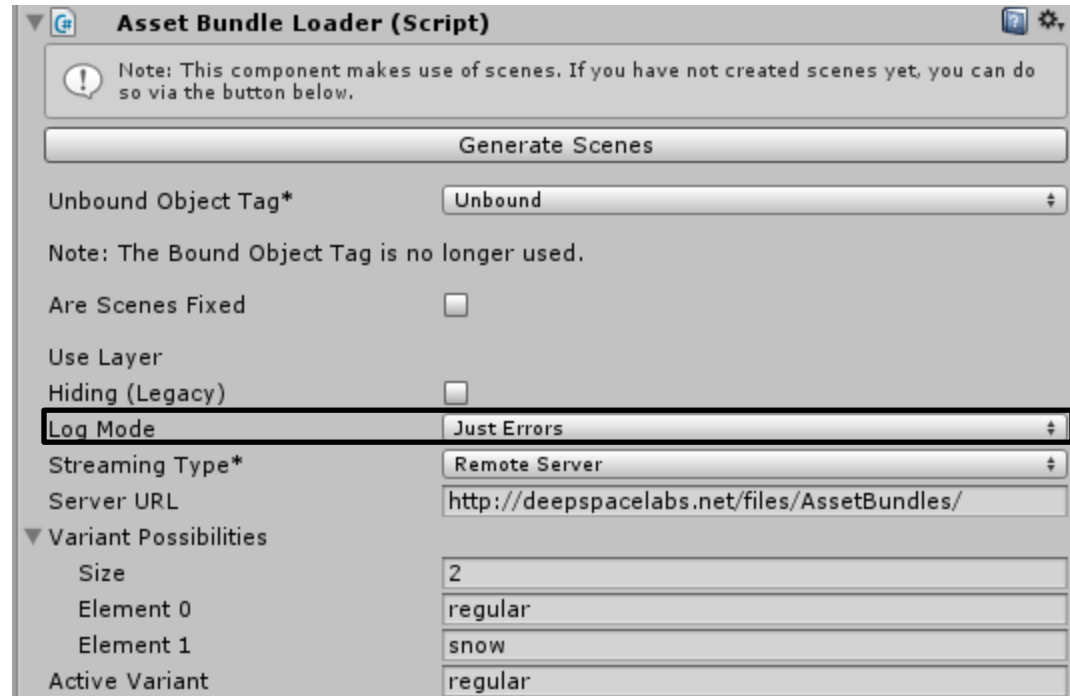
Asset Bundle Loader

Tells the Asset Bundle Manager what messages should be logged. “Just Errors” will only log errors, while “All” will log everything relating to Asset Bundle Loading.



Asset Bundle Loader

Tells the Asset Bundle Manager what messages should be logged. “Just Errors” will only log errors, while “All” will log everything relating to Asset Bundle Loading.



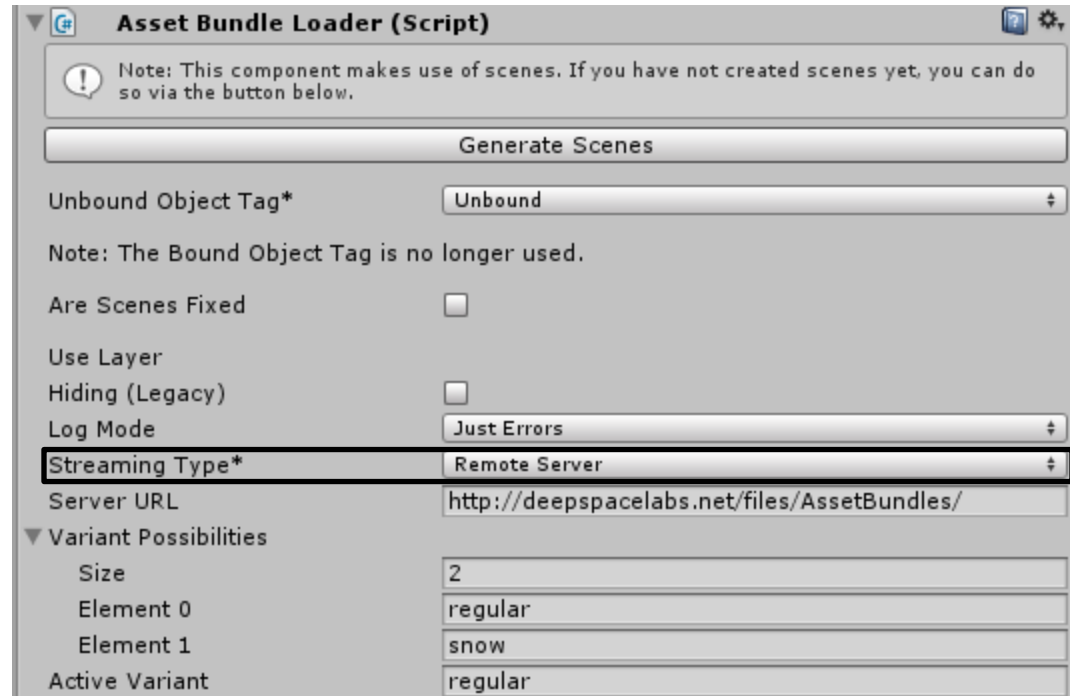
Asset Bundle Loader

Three Options

1) Simulation Mode or
Local Asset Server

2) Streaming Assets
Folder

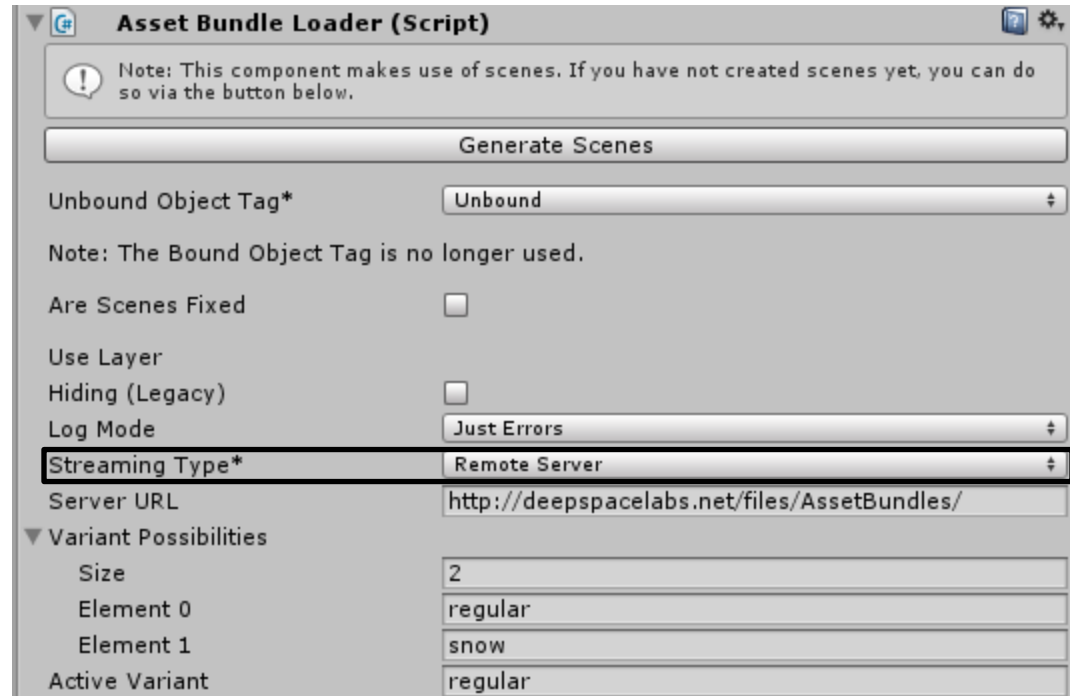
3) Remote Server



Asset Bundle Loader

Simulation Mode or Local Asset Server

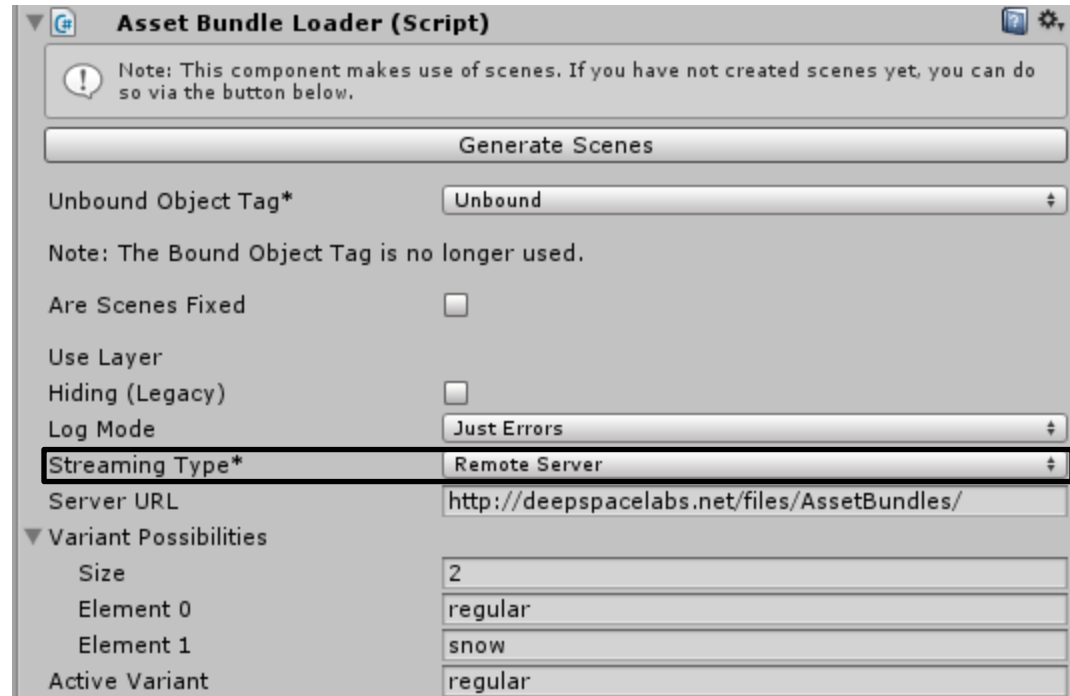
Will use Simulation Mode in the editor if it is enabled, otherwise will attempt to use the Local Asset Server (note, in a standalone build, “Development Build” may need to be enabled in your build settings).



Asset Bundle Loader

Streaming Assets Folder

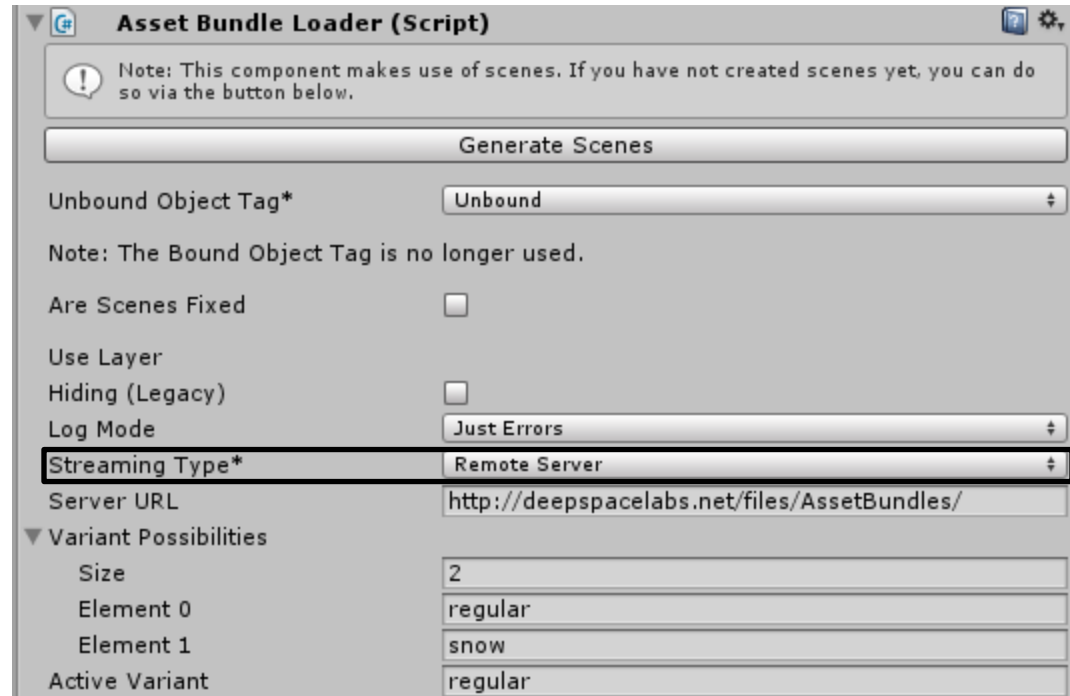
Streams in the assets
from the Streaming
Assets Folder.



Asset Bundle Loader

Remote Server

Streams in assets from a remote server. The DLK does not handle anything needed to connect to the server.

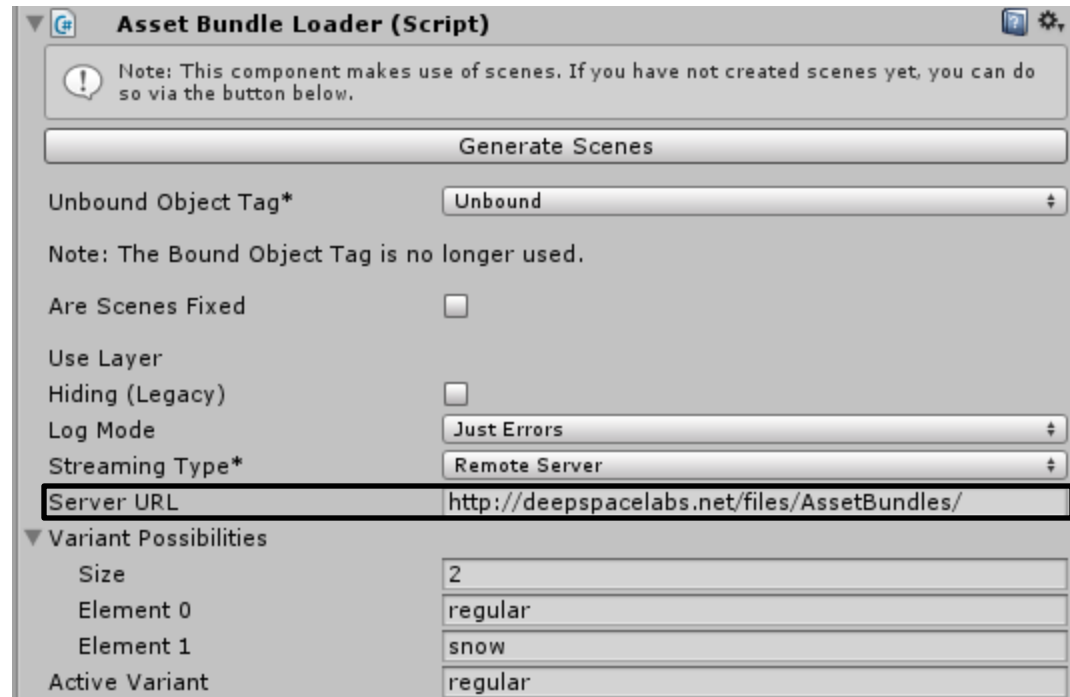


Asset Bundle Loader

The remote server address to stream the assets from (only displayed when the Streaming Type is set to "Remote Server").

This should be a folder and end with a '/' character.

Within this folder should sit your Build Specific Folders (ex: Windows), which will in turn contain the list of bundles and manifests.



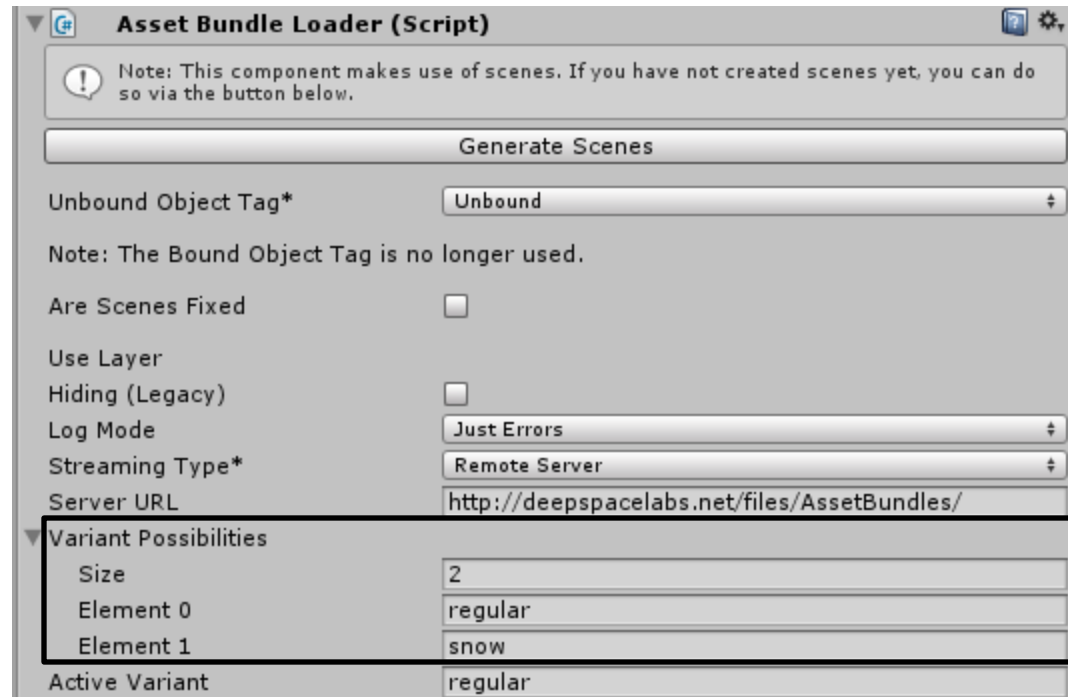
Asset Bundle Loader

The number of asset bundle variants you wish to use.

You can have more variants than those listed here; these are simply the ones related to the asset bundles containing your DLK scenes.

If you don't wish to use variants, set the Size field to 0.

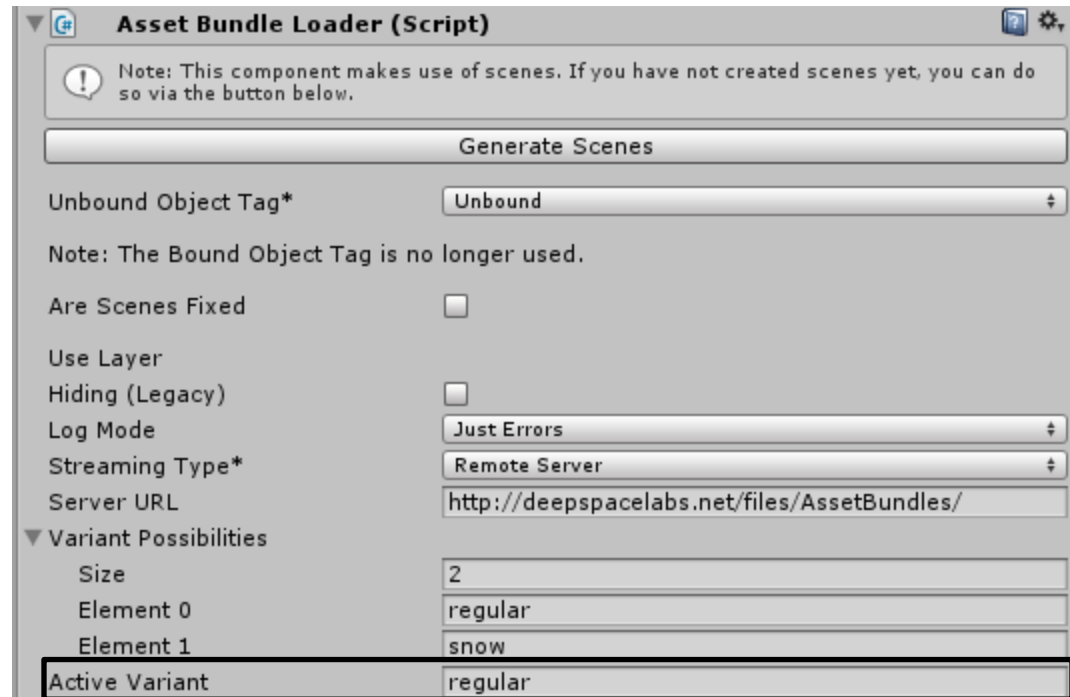
More information on variants can be found [here](#).



Asset Bundle Loader

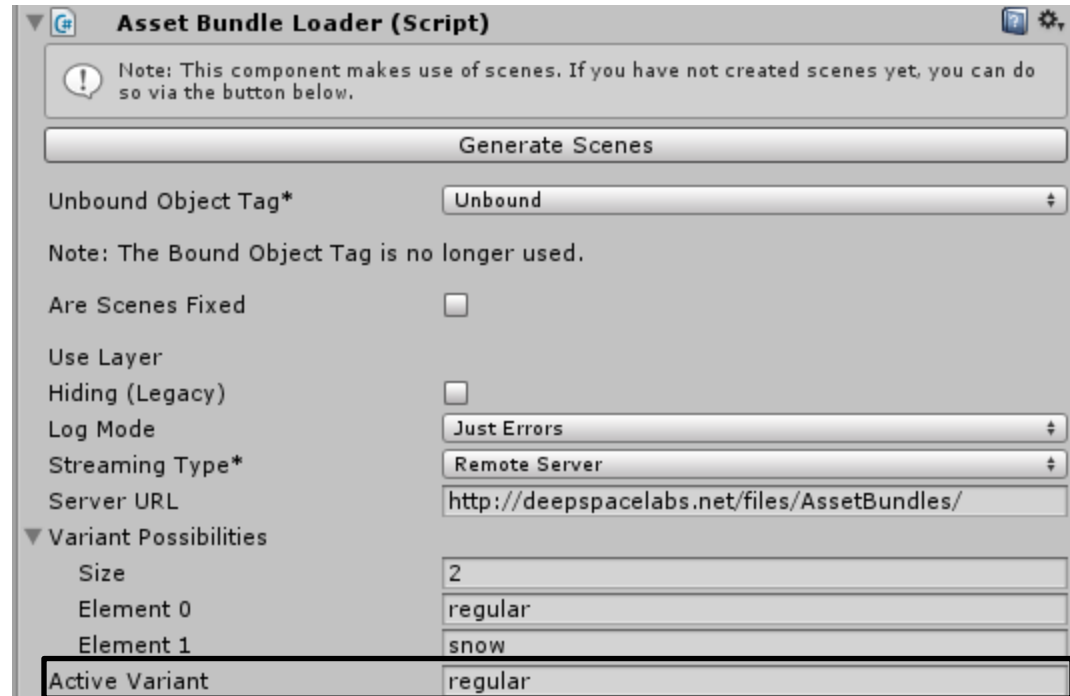
Only displayed when
Variant Possibilities
Size is greater than 0.

Determines the variant
to use when the
Component Manager
is initialized and asset
bundles are first
loaded.



Asset Bundle Loader

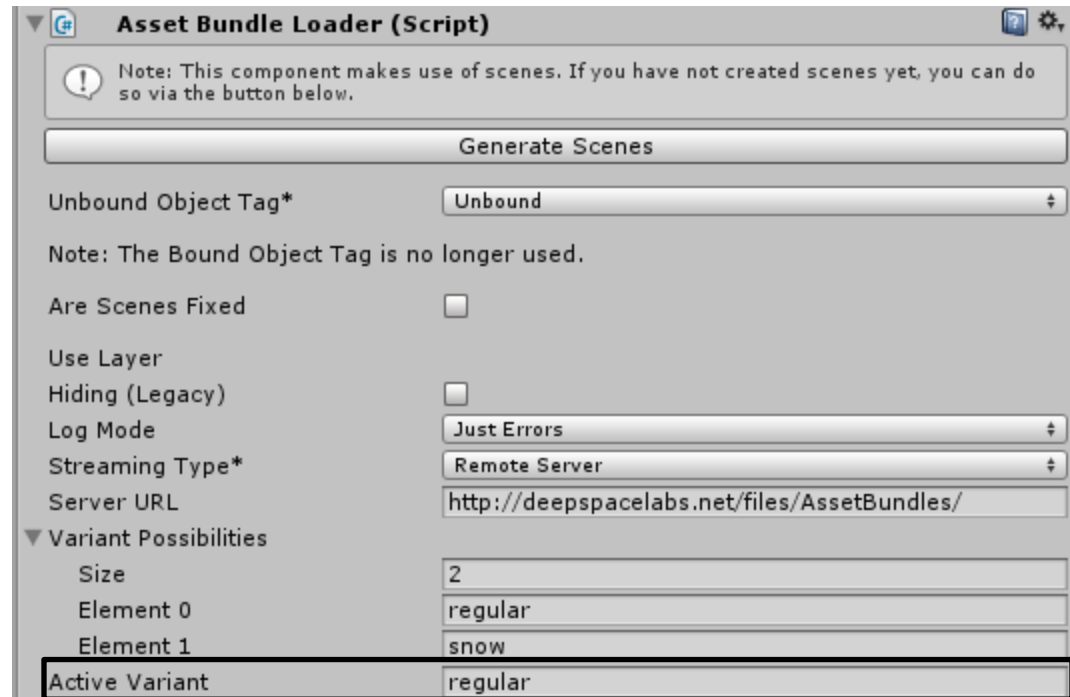
You can change the active variant at runtime by using the Asset Bundle Loader “SetVariant” method. You can pass in the name itself, or for a slightly more performant operation, the index of the variant within the Variant Possibilities array.



Asset Bundle Loader

Important

The active variant is not saved with the Component Managers data, so if you change this value and want this change to be preserved between game sessions, you will need to manually do so.



Asset Bundle Variants

- From the Unity docs, “AssetBundle Variants are designed to support one specific use case - remapping a choice of different Assets to individual objects in a project.”
- In the context of the DLK, this allows you to easily variations of a single world.
- You could create a normal world and then a snow world, or an HD and SD world.
- Note that each asset can only be assigned to one variant, so if you wanted to have an HD normal world, HD snow world, SD normal world, and HD snow world, you would need to create 4 different variants (eg `hd_normal`, `sd_normal`, `hd_snow`, `sd_snow`).

Asset Bundle Variants

- From the Unity docs, “AssetBundle Variants are designed to support one specific use case - remapping a choice of different Assets to individual objects in a project.”
- In the context of the DLK, this allows you to easily create variations of a single world.
- For example, you could create a normal world and then a snow world, or an HD world and then an SD world.
- Note that each asset can only be assigned to one variant, so if you wanted to have an HD normal world, HD snow world, SD normal world, and HD snow world, you would need to create 4 different variants (eg `hd_normal`, `sd_normal`, `hd_snow`, `sd_snow`), which you’d then assign to the Asset Bundle Loader’s “Variant Possibilities” array.

Asset Bundle Versioning

- The Asset Bundle Manager should handle all versioning automatically.
- It does this by creating a unique manifest each time you build, and the hash generated for this manifest is used as the version number.
- If you need more fine grained version control, you will need to implement your own solution and custom Asset Bundle Loader.

Common Errors

- Win32Exception when trying to enable the Local Asset Bundle Server via the Assets -> Asset Bundles -> Local AssetBundle Server option.
 - To avoid this error, build your asset bundles via the Assets -> Asset Bundles -> Build AssetBundles option.

Common Errors

- “Failed downloading bundle . . .” message when entering Play Mode or running your game as a standalone application.
 - In the Editor, this occurs when Simulation mode is disabled and the Local AssetBundle Server option is not enabled. Choose Assets -> Asset Bundles -> Local AssetBundle Server to enable it (or enable simulation mode instead).
 - In a standalone application, this error occurs whenever the program cannot access the server or local folder where the asset bundles are stored.
 - If streaming the bundles from the local StreamingAssets folder, ensure the bundles have been correctly placed (as described here).
 - If streaming bundles from a remote server, ensure the URL is correct and that you have permission to access the server.

Common Errors

- Failed to get pixels of splat texture:
 - This can occur because the textures assigned to the terrain are not read/write enabled. To fix, each texture's Texture Type should be set to "Advanced," and then the "Read/Write Enabled" option should be checked. You must rebuild your Asset Bundles after all textures have been changed.
 - Note that a similar error may occur if the meshes/textures that make up your detail meshes are not marked as read/write enabled.

Common Errors

- There is no scene with name “GroupName_1_1” in “groupName_1_1”.
 - This error occurs when the Asset Bundle Manager cannot find the scene within the asset bundle.
 - If you’ve verified that your scenes and asset bundles are named correctly, then it is likely that you are trying to use variants in Simulation Mode.
 - Remember, variants will not work in simulation mode. You should build your asset bundles and use the Local Asset Server, Streaming Assets folder, or a Remote Server if you wish to use them.

Additional Reading

- If you need additional help with Asset Bundles, you can look [here](#). If you find any additional help material published by Unity, please let me know.