1.1.0 Change Log

Primary Changes

1. Added a new component called SAMInitializer (now added with Default SAM Setup command, or add manually via Add Component -> Deep Space Labs -> SAM -> Primary Components), which handles initializing SAM either in "immediate" mode or gradually. You can hook the Initializer up to a button or call it from another script. It also has support for activating and/or deactivating game objects automatically before and/or after the initialization starts/ends.

   You can hook up a SAMText and/or a SAMSlider (also new components) to the SAMInitializer to have the Initialization Progress shown to the player as well. They are abstractions that allow the SAMInitializer's progress updating to be used with 3rd party UI assets or your own custom solutions.

   Default SAMText components supporting UnityEngine.UI.Text, TextMeshPro, and TextMeshProUGUI components have also been added, as well as one default SAMSlider component supporting UnityEngine.UI.Slider (all can be added via Add Component -> Deep Space Labs -> SAM -> Secondary Components; just add to the same game object as the component each one supports).

2. When using the "Default SAM Setup" command to add default game objects for SAM to the scene, the Origin Change Strategy is now set to Shift World by default. We also now create a default Standard Hierarchy Organizer, a default Standard Hierarchy World Shifter, and a Frame Rate Dependent Execution Controller on the "Default Components" game object, which are automatically assigned to the World. Finally, the SAMInitializer is added under a new game object of the same name.

3. Improved the performance of the Scene Chunk Streamer's mechanism for unloading Asset Chunks (it now works close to how the Prefab Chunk Streamer works). The method can take longer to complete but should help avoid bottlenecks.

4. Added a LOD Filter to the WorldGroupingListener class, which effectively allows you to stop any World Grouping's that derive from this class from being used with batches of World Cells associated with one or more specific LOD Groups.

5. Moved transitionTime, makeVisibleCurve and makeInvisibleCurve fields from respective Transitioner classes to base CellVisualTransitionerClass, as these properties were used in almost all Transitioners and should be useful when creating your own custom Transitioners (all can be accessed with capitalized versions of the names, e.g. TransitionTime).

6. Added two new Cell Visual Transition Controllers, the PositionTransitioner and the ScaleTransitioner. The former manipulates the Position value of transforms to perform transitions, while the latter manipulates the Scale value.

7. Made several changes to the World Designer Tool that should increase advanced operation speeds, especially when using Build Setting based scene assets.

8. Added a Scene Conversion Tutorial Chapter to the In-Editor Guide.

Fixes

1. Fixed an issue that caused a dark GUI window intended for the Dark Editor Theme to be used with the Light Editor Theme.

2. Fixed a few bugs that caused inconsistencies in the way some World Grouping Listener methods were called.

3. Modified StaticBatcherListener to not continuously generate garbage when its enumerator based methods are executed.

4. Fixed a timing bug with the “Duplicate World” Origin Cell Change Strategy that resulted in the Player being moved after the removal of the pre-duplicated Asset Chunks rather than before.

5. Fixed/changed some text/settings in editor windows/inspectors.

6. Fixed store links in Editor Guide.

7. Fixed some documentation/XML issues. This is mostly beneficial to for the Website API however is also useful for IDE scripting.

8. Fixed a potential issue with the Burst Compiler and BurstEnabledJobImplementation.

9. Fixed some errors related to using Temp allocated NativeArrays inside single frame jobs.

10. Fixed a bug causing some improper Naming Convention Format Strings to not be detected (Core.dll change).

11. Fixed a bug that could cause a NullReferenceException when using the Editor Guide (EditorCore.dll change).

Other

1. Made an attempt at fixing Native Array errors from showing when using the Loading Blueprint Editor and World Designer Tool. It is likely further fixes will be needed to address this completely. Please message us if you see errors related to Native Arrays not being properly disposed!

2. Improved several Chapters/Sections in the Editor Guide to be clearer.

3. When using the World Designer Tool, non-loaded enabled cells will now show the number of chunks used by LOD Group 1.

4. Naming Convention Format strings can no longer have a '/' character in their name, as this will mess up some World Designer Tool operations (Core.dll change).

5. Added a Change Logs Chapter to the In-Editor Guide, with links to all change logs (hosted on deepspacelabs.net).