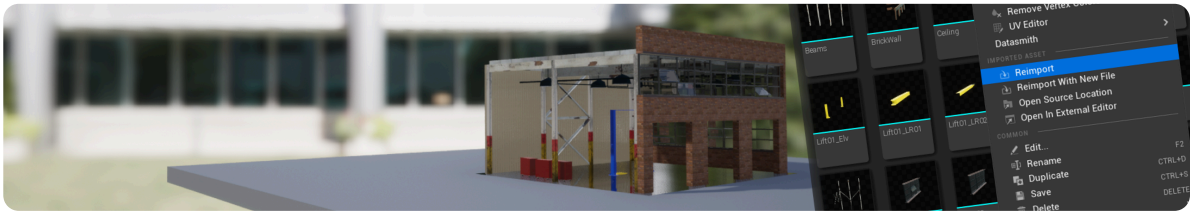
Datasmith Reimport Workflow

Describes what happens when you reimport content that you brought into Unreal using Datasmith, and how you can take advantage of this iterative workflow.



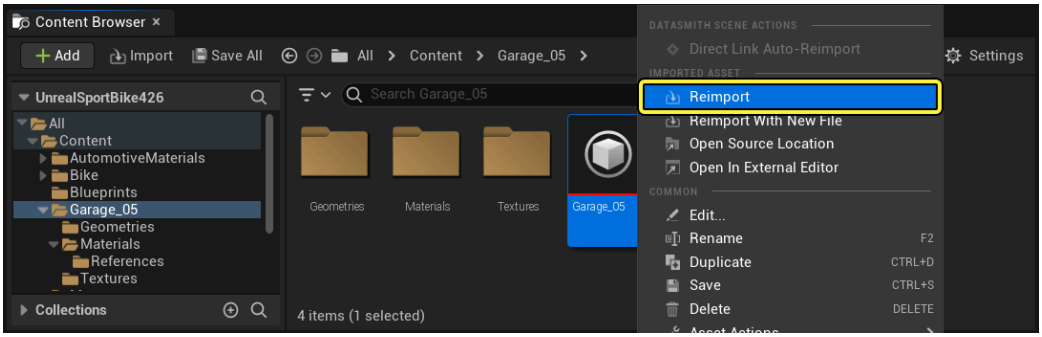
Datasmith brings design data from a variety of source applications into Unreal Engine, typically for the purpose of building real-time visualizations and experiences around that data. But often, while you're working on building those visualizations and experiences in Unreal, the scene or design data that you're basing your work on needs to change in order to meet new requirements or incorporate feedback from stakeholders. To avoid painful and costly re-work, you need to be able to incorporate those upstream changes without losing all the work you've done in the Unreal Editor.

This page describes the different ways you can pull updates to Datasmith content into your Unreal Project, and what happens to your Assets and Actors in the process. You have two options:

1. **Reimport the complete Datasmith Scene -** This updates the Datasmith Scene Asset to contain all the latest changes that have been made in your source scene, and attempts to reconcile those changes with the work you've done in the Unreal Editor. See Full Scene Reimport below.
2. **Reimport individual Assets imported by Datasmith -** such as Static Meshes, Materials, or Textures. This allows you to process changes to selected Assets without affecting the rest of the Datasmith Scene. See Individual Asset Reimport below.

For detailed instructions on how to use these import workflows in the Unreal Editor, see Datasmith Reimport Workflow.

Full Scene Reimport



Click image for full size.

You can reimport an entire Datasmith scene, all at once. This approach updates all of the Assets created by Datasmith in your Content Browser to match the latest changes you've made in your source application or data file. From there, you

can selectively update the Datasmith Scene Actors in your Levels to synchronize them with the new scene hierarchy maintained by the Datasmith Scene Asset.

Changes to the Scene Hierarchy

When you reimport a Datasmith scene, the scene hierarchy contained in the Datasmith Scene **Asset** is immediately updated with all the latest changes that you've made in your source scene. After the import, the scene hierarchy in the Datasmith Scene Asset will have all the same information as if you were importing the scene fresh for the first time.

However, those changes to the scene hierarchy won't necessarily be reflected in Unreal Engine Levels that contain your Datasmith Scene. Instances of the Datasmith Scene that you've dropped into a Level in your Project become out of date with the scene hierarchy in the Level until you **synchronize** them with the updated Asset.

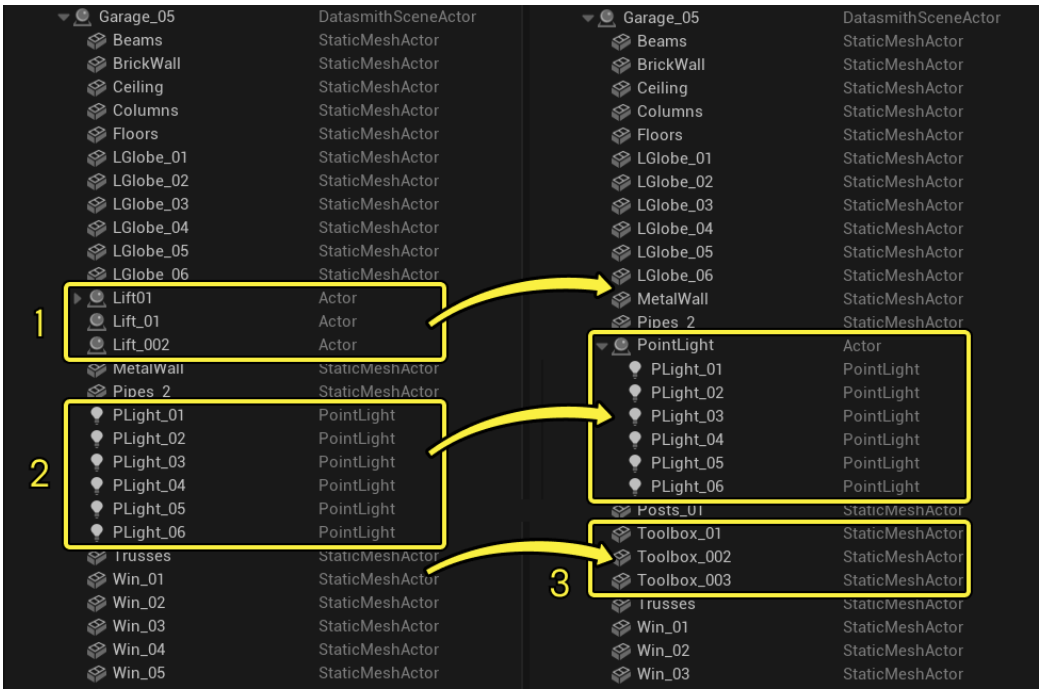
Synchronizing updates the information in the Datasmith Scene Actor, and its children in the World Outliner, to match the updated Asset as follows:

- **Modified Actors.** Any Actors that you *haven't* modified in the Unreal Editor are immediately updated to match the latest information about them in the reimported Datasmith Scene. Any Actors that you *have* modified in the Unreal Editor are updated to match the latest information in the reimported Datasmith Scene, *except* for changes that Datasmith has tracked as overrides.
- **New Actors.** Any new objects that you've added to your source file since the last time you imported have new Actors added into the Level to represent them.
- **Deleted Actors.** Any objects that you've deleted in your source file have their corresponding Actors *removed* from the Unreal Level — unless those Actors contain other overrides. If the Unreal Actor that corresponds to a deleted object contains overrides, it is not deleted from the Level until you specifically delete it. Any Actors that you've deleted from the Level in Unreal but that still exist in the source file are *not* added back to your Level by default. However, you can enable this option when you synchronize the Datasmith Scene Actor in your Level with the reimported Datasmith Scene Asset.

- **Parenting relationships.** The parenting hierarchy of the Actors in the Level is updated to match changes in the source file.

For example, the following before-and-after image illustrates:

1. Objects that have been deleted in the source scene — in this case, the Lift objects — get removed from the Datasmith Scene.
2. Changes that have been made to the object parenting structure in the source scene — in this case, the lights being grouped together under a new parent — are reflected in the Datasmith Scene.
3. Objects that have been added to the source scene — in this case, the toolboxes — get added to the Datasmith Scene.



Click image for full size.

There are two ways you can synchronize your Datasmith Scene Actor with its Asset:

- When you reimport a Datasmith Scene Asset, you can automatically update any Datasmith Scene Actors in the currently opened Level that were created from that Asset. This is a simple approach that works well if you've only added your Datasmith Scene to a single Level.
- You can synchronize the Datasmith Scene Actor in a Level with its Datasmith Scene Asset on demand at any time. You'll need to use this approach if you have multiple different Levels in your Project that contain instances of your Datasmith Scene, and you need them all to reflect the changes in your source scene.

For details, see Reimporting Datasmith Content.

Changes to Geometry, Materials, and Textures

When you reimport a Datasmith scene:

- Datasmith recreates Static Mesh, Texture, and Material Instance Assets *only* if you have modified them in your source file since the last time you imported the scene.
- Datasmith **always** recreates all Parent Material Assets, regardless of whether or not they have been modified in your source file.

> When Datasmith recreates an Asset, it may overwrite changes that you've made to that Asset in Unreal!
>
> Typically, you should not lose any information that Datasmith tracks as overrides, or any information that is specific to the Unreal representation of your scene's objects. However, you might lose changes that you've made in Unreal to information that is brought in by Datasmith, but that Datasmith does not track as an override. For example, overwriting an Asset will overwrite changes that you've made in Unreal to the internals of a Parent Material graph or to the geometry of a Static Mesh Asset.

To avoid losing changes that you make to your Assets in Unreal, you have two options:

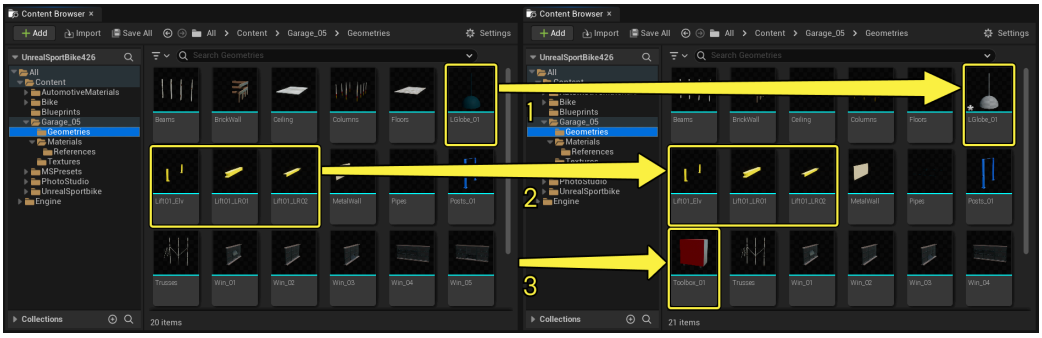
- When you re-import, deselect the **Process** checkbox for the corresponding asset type in the **Reimport Options** dialog box, to avoid reimporting that type of Asset. For example, if you've modified an imported Material graph in Unreal, and you don't want to lose those changes, you could deselect the **Materials & Textures** option. However, if there are other Material Assets that you *do* want to update from the source, this approach won't work. In this case, you could selectively reimport only the Assets of that type that you want to update; see Option 2. Individual Asset Reimport below.

- Whenever you need to modify an Asset that you've imported with Datasmith, duplicate the Asset you want to modify into a different folder and modify the duplicate instead. Then, update any uses of the original Asset to use your duplicate instead. When you re-import, the original Asset will be overwritten, but your duplicate will remain untouched.

When you reimport the scene after deleting objects or materials in your source file, Datasmith does not delete the Assets that it had previously created for those objects. The Assets remain in your Content Browser, so that you can continue using them in the Unreal Editor as standalone Assets, outside the context of the Datasmith Scene. However, when you synchronize a Datasmith Scene Actor in one of your Levels to match the reimported Datasmith Scene Asset, instances of those deleted Assets will be removed (unless they contain any overrides).

The following before-and-after image illustrates what happens to Static Mesh Assets when you reimport a scene:

1. Objects that you've modified in the source scene — the light globe in this example — are re-created in Unreal, and the old versions of the Assets are overwritten.
2. Objects that you've deleted in the source scene — the yellow lift elements in this example — are not removed from your Project's Content Browser.
3. Objects that you've added to the source scene — the toolbox in this example — are added to the Content Browser as new Assets.



Click image for full size.

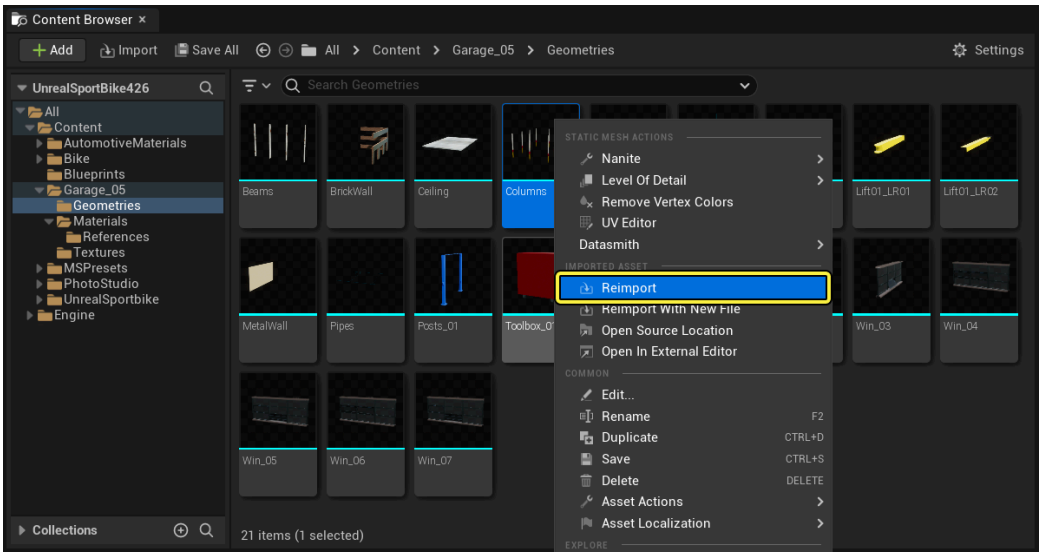
Remember that when you make a change to an Asset in the Unreal Engine, that change is immediately reflected everywhere that the Asset is used. The same applies when Datasmith recreates or modifies an Asset during a reimport. For example, if you change the geometry of an object in your source application, you'll see that after reimporting the corresponding Static Mesh Asset, all Actors in all Levels that instantiate that Static Mesh Asset are updated automatically to show the new geometry.

When to Use Full Scene Reimport

We recommend using full scene reimport in most cases. In particular, you need to reimport the full scene if:

- You need to bring in new objects or materials from your source scene that didn't exist previously in Unreal.
- The layout of the scene objects in 3D space has changed, or objects have new parenting relationships in the scene hierarchy that you want to be reflected in Unreal.
- You aren't able to reimport your Assets individually. This may be either because too many objects have changed in your source file to work with them one-by-one, or because you don't know which Assets have changed.

Individual Asset Reimport



Click image for full size.

Instead of reimporting the entire Datasmith Scene, you can reimport selected Unreal Engine Assets that were created by the Datasmith import process — for example, individual Static Mesh Assets, Materials or Textures.

When you use this option, almost exactly the same thing happens as described under Changes to Geometry, Materials, and Textures above, but limited to the single Asset that you select. The only differences are:

- The Asset is always recreated from scratch, even if it has not been modified in the source scene. For example, in the case of a Static Mesh Asset, the geometry of the Static Mesh is always rebuilt, so any changes that you've made to the geometry in Unreal are overwritten.
- The Datasmith Scene Asset does not change, so no changes are made to the scene hierarchy in any Level that you've placed your scene into, regardless of whether you re-synchronize the Datasmith Scene Actors in those Levels.
- Because the Datasmith Scene Asset does not change, it continues to track overrides to your Asset relative to the last time you imported the scene as a whole. If you reset the overrides on your reimported Asset, it will revert to its state the last time you imported the entire Datasmith Scene.

When to Use Individual Asset Reimport

Individual Asset reimport can be a good choice when:

- You want to bring in changes to a small number of Assets, but you know that the overall Datasmith Scene contains changes to other Assets or to the scene hierarchy that you *don't* want to bring in.
- You want to apply different Datasmith import settings to specific Assets than you applied when you imported the whole Datasmith Scene.

This workflow can be particularly useful for CAD models that need to be tessellated on import, because you can apply different tessellation settings to different parts of the Datasmith Scene. This may help you optimize the performance of your scene by controlling the number of triangles in each Static Mesh Asset. For example, you could import your CAD file with low-resolution settings, then selectively re-import a smaller number of important Static Mesh Assets at higher resolutions.

You can achieve the same purpose by retessellating selected Static Mesh Assets instead of reimporting them. Unlike reimporting, retessellating does not reimport the geometry of the Asset from the CAD scene file. It regenerates the triangular mesh for the Static Mesh Asset based on the last imported geometry, using the new settings you provide. For details, see Retessellating CAD Geometry.

Data Preservation

Whether you choose to reimport the complete Datasmith scene or to reimport individual Assets, much of the work you've done in Unreal is preserved in the reimport process. This includes:

- Properties that are tracked by Datasmith as overrides. See Datasmith Overrides below.
- Properties that are exclusive to Unreal Engine. See Unreal Engine Properties below.

Unreal Engine Properties

You should never lose any changes you make to properties on your Assets and Actors that are specific to Unreal — that is, properties that Datasmith does not import from your source file.

For example, suppose that you use Datasmith to import a scene, then you disable the **Cast Shadows** property of one of your imported Static Mesh Actors in the Unreal Editor. When you reimport the Datasmith Scene Asset and synchronize the Datasmith Scene Actor in your Level, that Cast Shadows property remains disabled.

Datasmith Overrides

When you reimport a Datasmith Scene or Asset, you should never lose any change you've made to an Unreal Engine Asset or Actor that Datasmith recognizes as an *override*. Overrides include things like which Materials are assigned to your Static Mesh Assets and Actors, and the 3D transforms of Actors in your Levels that were created as part of a Datasmith Scene.

For details on what changes are considered overrides, see Datasmith Import Process.

The next time you reset overrides to an Asset or an Actor after you reimport a Datasmith Scene, the Unreal Asset or Actor will be reset to its *post*-reimport values, not its *original* values. For example:

- Suppose that your scene contains a Static Mesh Asset named "Chair", and after your initial Datasmith import it is assigned a Material named "Color_00000000".
- In the Unreal Editor, you modify this Asset to use a new, physically based Material named "Black Leather". Datasmith tracks this change as an override.
- Now, in your source scene, you change the color properties of the chair, and reimport the Asset or the Datasmith Scene that contains it. At first, your reimported Asset is still be assigned the "Black Leather" Material that you assigned in Unreal.
- If you reset the overrides on the Asset, it does *not* revert to using the original "Color_00000000" Material, it reverts to using the new Material Asset that Datasmith generated to match the new color properties of the chair.

Auto Reimport

You can configure the Unreal Editor to monitor selected folders for Assets that have changed, and reimport those Assets automatically.

We don't recommend using this system for your Datasmith Scene Assets, or for the individual Assets created by Datasmith. Use auto-reimport only for other kinds of models or Assets that you add to your Project, like FBX files that you import separately as additional set dressing around your Datasmith models.

For details on setting up this system for your non-Datasmith content, see Auto Reimport.