User Manual for

Introduction

Merlin's Prefab fixes some of the problems you might face when working with Unity Assets. This document describes the Unity way of working, as well as the new way using *Prefab Lab*.

Working with Assets: the Unity way

When you import 3D models into Unity, they are called **Assets**. When you add an Asset to your scene, a corresponding **GameObject** is created.

A smart thing to do is to add Assets to your scene using **Prefabs**. This has several advantages:

- Prefabs can easily be instanced in your scene as often as you want
- Any changes made to a Prefab will be instantly applied to all instances
- Any Components (scripts, colliders etc) added to a Prefab will be applied to all instances
- You can put multiple GameObjects into one Prefab, including hierarchy

You can easily create Prefabs by selecting 'Assets -> Create -> Prefab' from the menu bar. All you need to do now is drag the Asset you want onto the Prefab, and presto!

Updating Assets

In theory, whenever you update your source 3D model Asset, Unity should automatically synchronize all related GameObjects and Prefabs. In reality, certain actions break the link between Asset and GameObject, or Asset and Prefab.

Consider the following situations:

1. GameObjects based on Assets

When you add a 3D model Asset to your scene directly, Unity creates a GameObject for you. Now when you add Components to this GameObject, the link to the 3D model Asset is broken.

You can restore the link by clicking ‘Reconnect’ on the GameObject, but this will clear all Components you painstakingly added!

2. GameObjects based on Prefabs

When you add Components to a 3D model in a Prefab, the link to the source Asset is broken. You can still add the Prefab to your scene, and such instances will still update whenever you make changes to the Prefab.

However, if you update the source Asset, neither the Prefab nor its instances in your scene will synchronize automatically. The only way to re-establish the link is to drag the source Asset onto the Prefab again. Having to do this each time you update the source file quickly becomes a chore.

Working with Assets: the *Prefab Lab* way

At Paladin Studios we ran into all of the problems described above. After a bit of thinking we came up with a good solution and decided to make it available to others in similar situations. That means you!

Merlin’s Prefab Lab does the following great things:

1. When you add a new model Asset to *Prefab Lab’s* Source folder, a Prefab will automatically be generated for it. This will save you time.
2. When a model Asset is updated and saved in an external modeling package, the Prefab and all of its instances in your scene will be synchronized automatically.
3. Contrary to Unity’s built-in updater, Parent / Child hierarchies and (Sub id-) Materials will also be synchronized.
4. Any Components added to a Prefab or instance will remain attached during synchronization. This is especially valuable for GameObjects that contain custom Scripts, Colliders, Animations or Particle Systems.

Merlin’s Prefab Lab makes our lives a little easier. Hopefully it will do the same for you!

Installation

Important: Before purchase and installation, please note that you require a Unity Pro License to use *Merlin's Prefab Lab*. Under the hood, the plug-in accesses the import pipeline, which is only possible with the Pro version.

After purchase in the Unity Asset Store, *Merlin's Prefab Lab* will be automatically added to your current project.

Adding *Prefab Lab* to a new project

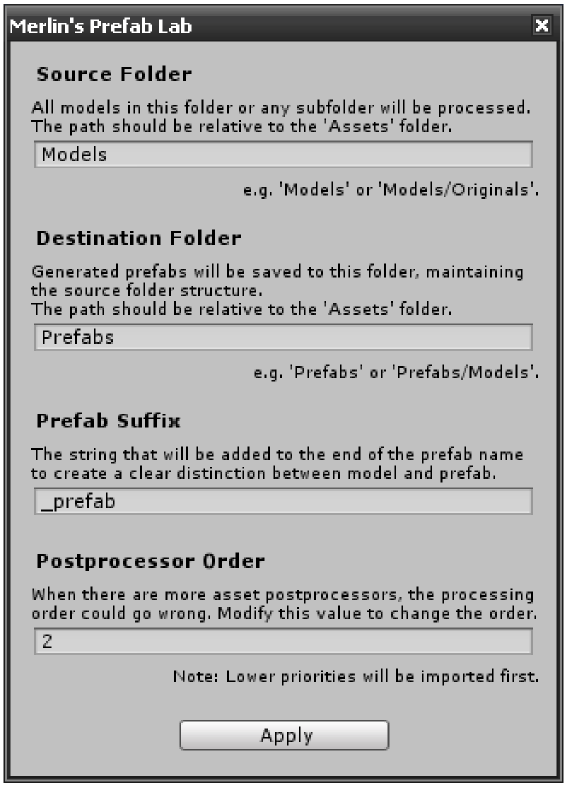
When starting a new project, adding *Merlin's Prefab Lab* is as easy as ticking its box in the 'Packages' list.

Adding *Prefab Lab* to an existing project

Open the 'Unity Asset Store' window. Then press the icon to go to the page that lists all of the Assets you purchased. Finally, locate *Merlin's Prefab Lab*, and choose 'Import'.

Setup

Merlin's Prefab Lab is very easy to set up. You can access its properties window by selecting 'Merlin -> Prefab Lab' from the menu bar.



Source Folder

All models in this folder or any subfolder will be processed. The path should be relative to the 'Assets' folder.

Models

e.g. 'Models' or 'Models/Originals'.

Destination Folder

Generated prefabs will be saved to this folder, maintaining the source folder structure.
The path should be relative to the 'Assets' folder.

Prefabs

e.g. 'Prefabs' or 'Prefabs/Models'.

Prefab Suffix

The string that will be added to the end of the prefab name to create a clear distinction between model and prefab.

_prefab

Postprocessor Order

When there are more asset postprocessors, the processing order could go wrong. Modify this value to change the order.

2

Note: Lower priorities will be imported first.

Source Folder

This is the most important property. Make sure that you change its value to the directory in which you keep all of your 3D model Assets.*

From now on, instead of adding Assets or hand-made Prefabs to your scene, always use the auto-generated Prefabs from the Destination Folder.

Now whenever you update a 3D model source file, all of its associated Prefabs and instances will be perfectly synchronized. Congratulations!

* Alternatively, you can create a new ‘Models’ folder in your project’s ‘Assets’ subdir and start saving your 3D models to that folder.

Destination Folder

This is the folder in which your auto-generated Prefabs will be stored. You may change this path to something else if you like.

Prefab Suffix

To distinguish between Asset and Prefab, a suffix is automatically added to the generated Prefabs. You may change this string to something else if you like, or leave it blank.

Postprocessor Order

Merlin’s Prefab Lab affects the Asset import pipeline. If any other plug-ins are active that also affect the pipeline, the value ‘Postprocessor Order’ controls in which order they are executed. A low number means *Prefab Lab* will run early during the import process, a higher number means later.

Note: This is an advance feature. In most cases, its value will not be used.