Nested Prefabs for Unity3d

Reference

Bento Studio

SUMMARY

This document presents the Nested Prefab Editor functionalities. You will learn what is possible to do with it and what are the limitations.

Sample

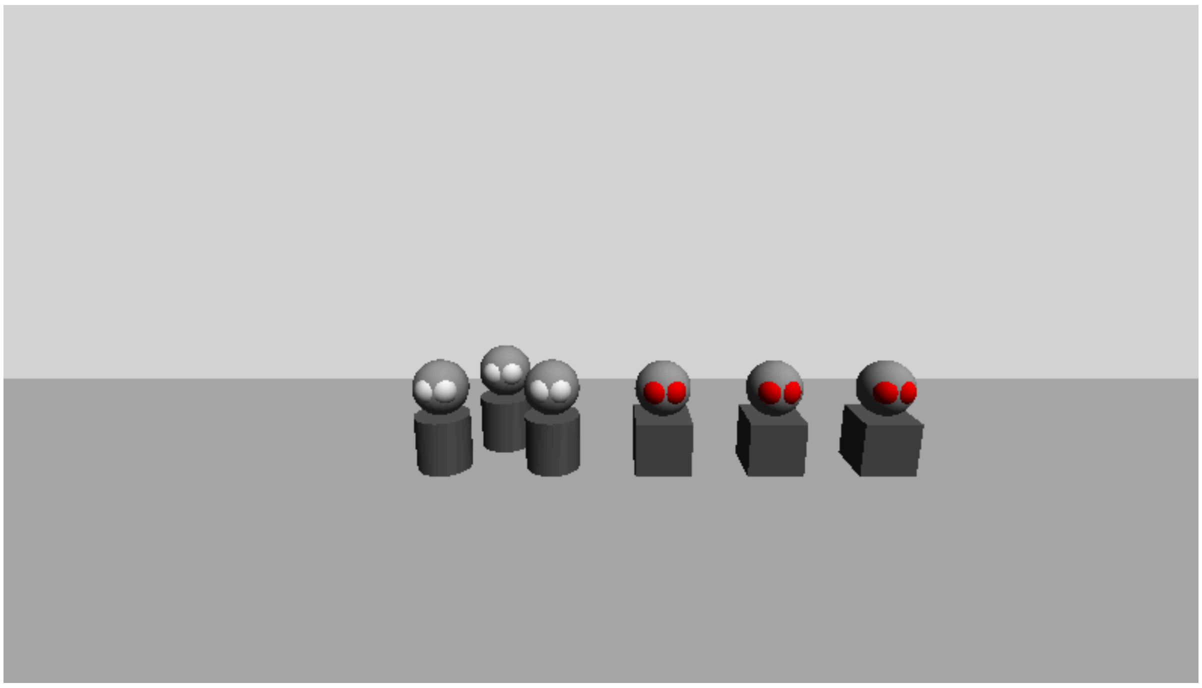
If you want to get started quickly take a look at the sample scene.

You can find it at **NestedPrefab > Sample > scene_NestedPrefabSample.**

To see how this scene was created, you can follow the tutorial you will found in the other PDF at **NestedPrefab > Documentation > NestedPrefabTutorial.pdf**

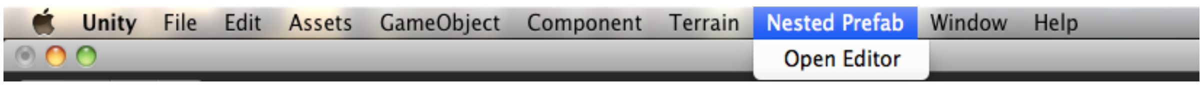
The tutorial will teach you the basics of creating prefabs nested in other prefabs.

It's recommended to read the tutorial first and come back later read the reference.



Opening the editor

You can open the **Nested Prefab Editor** by clicking on **Nested Prefab > Open Editor** in the main bar.



Hierarchical Prefab VS Classic Unity Prefab

Hierarchical prefabs and classic Unity prefabs are very much alike. Hierarchical prefabs can do all that Classic Unity Prefabs can do. But the workflow is not exactly the same.

Things that work exactly like Classic Unity Prefabs

- Hierarchical prefabs can be drag and drop into the scene or the hierarchy panel
- Hierarchical prefabs instances can use the "Select" button of the Component Inspector to find their prefab.
- Hierarchical prefabs can be duplicate (Cmd-d on Mac or Ctr-d on Windows)

Things that need the use of the Nested Prefab Editor

- Hierarchical prefabs use the "Apply" and "Revert" buttons of the **Nested Prefab Editor** instead of the classic "Apply" and "Revert" buttons of the Component Inspector.

Things that cannot be done

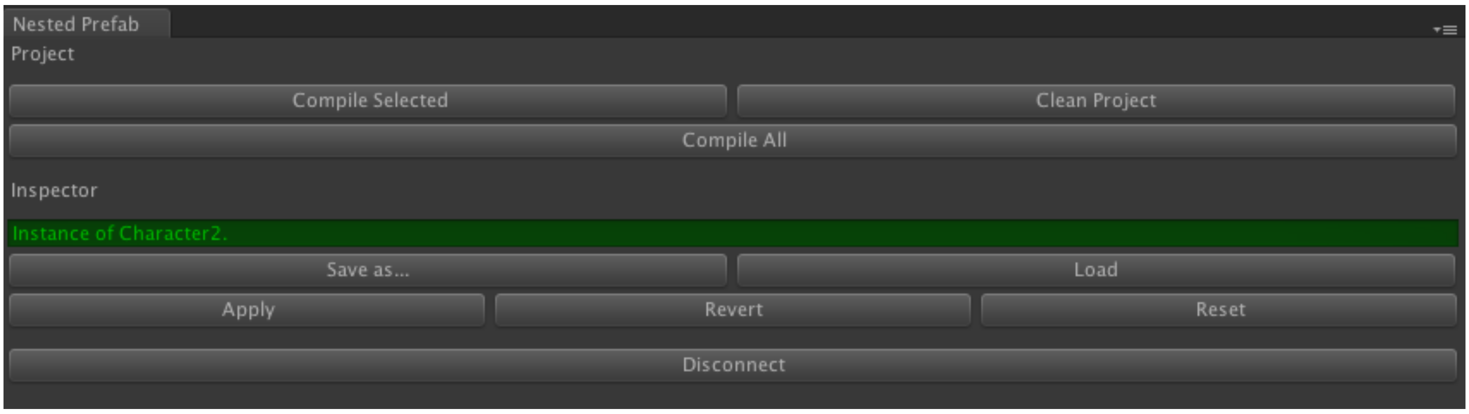
- Hierarchical cannot be drag from the scene to the project panel. It will no longer be a hierarchical prefab. Use the "Save as…" of the **Nested prefab Editor** button instead.
- Don't use the classic Component Inspector "Revert" button on a hierarchical instance. It will destroy all the nested prefabs of the instance. If you ever make this mistake, just click on the **Nested Prefab Editor "**Revert" button.
- **Don't use the classic Component Inspector "Apply " button** on a Hierarchical prefab instance. All the prefabs instances will no longer be Hierarchical prefabs; all the nested prefabs links will be lost. **Be very careful it's irreversible!**

Things only hierarchical prefabs can do

- Hierarchical prefabs can have both kind of prefabs (Hierarchical and Classic) nested in, without breaking their prefabs link.

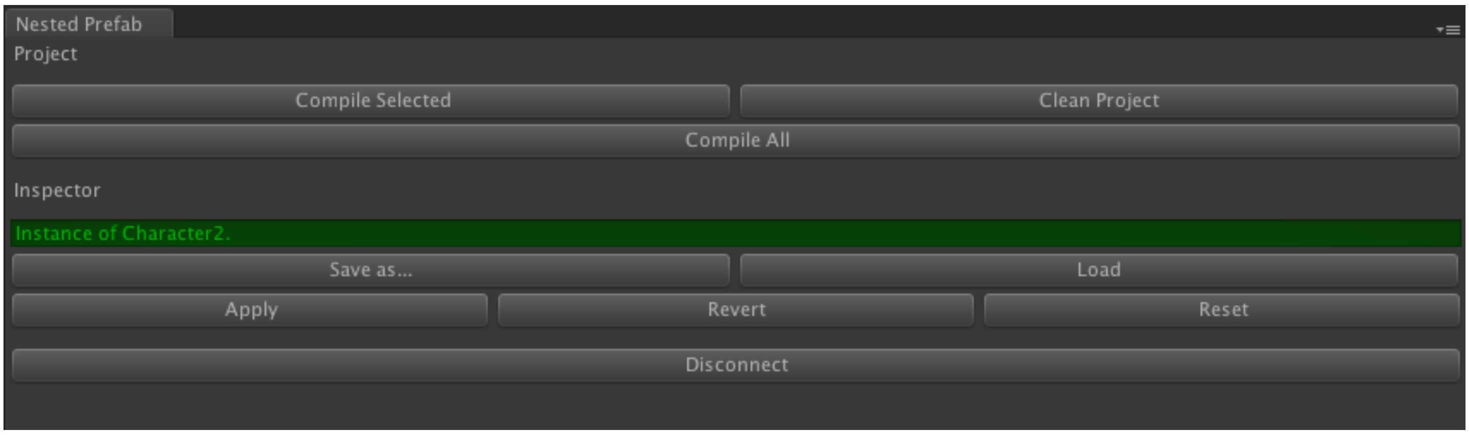
Things you will be compelled to do

-To instantiate a hierarchical prefab by script you will need to use the **HierarchicalPrefabUtility.Instantiate** method, check the **Can be instantiated** Box on the **HierarchicalPrefabInstance** component and compile the prefab using one of the **Compile** button on the nested prefab editor.

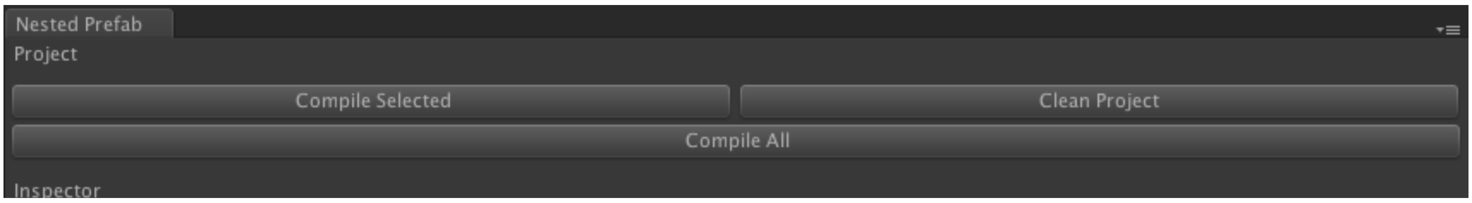


Instance of Character2.

Nested Prefab Editor overview



Project



The project section is all about compiling the Hierarchical Prefabs to be able to instantiate them by script. (See ***Instantiate a Hierarchical Prefab by Script*** for more information)

Compile Selected

Compile only the selected hierarchical prefab. Use that for rapid prototyping. It will only update the selected hierarchical prefab but not the other hierarchical prefab that contain this one.

Compile All

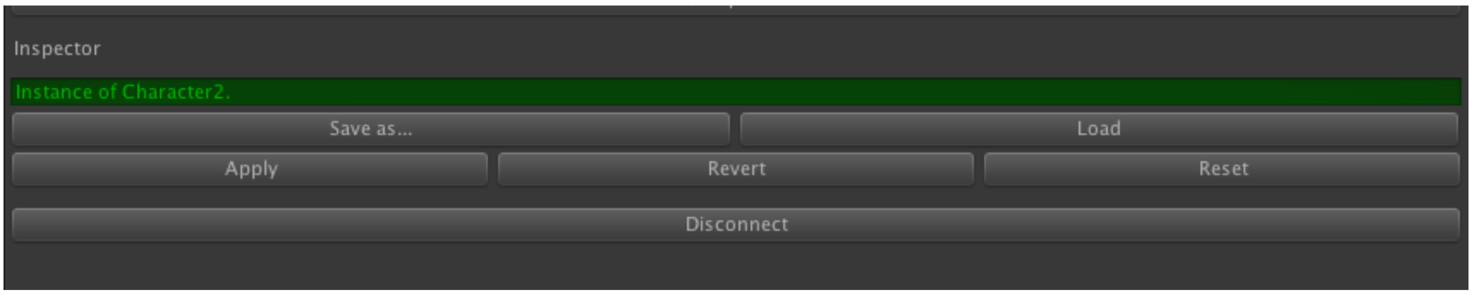
Compile All the Hierarchical prefabs. (Marked as **Can Be Instantiated**)

Clean Project

When a Hierarchical Prefab is compiled a Classic Unity Prefab snapshot of it is created in **NestedPrefab > Compiled.** Use Clean Project to remove all the Compiled prefabs.

You normally don't need to use this at all when using Hierarchical Prefabs as each **Compile All** take care of removing unused compiled prefabs. But it can be handy to use this if you want to stop using the Hierarchical Prefabs and be sure all the temporary files are disposed off.

Inspector



The Inspector section of the **Nested Prefab Editor** allows you to manage your Hierarchical Prefabs and their instances.

Save as...

Choose a location to save any Game Object as a Hierarchical Prefab.

It's the main way to create Hierarchical Prefabs. You can also duplicate them (Save an instance under a different name and/or folder).

An advanced feature is the possibility of saving a new object as an already used Hierarchical Prefab to replace it completely everywhere (including in the other Hierarchical Prefabs containing it). **Cautious this operation is irreversible!**

Load

Replace the currently selected game object by a Prefab. (Either Classic or Hierarchic)

Revert

Revert the Hierarchical Prefab Instance to its prefab state.

Note that when used on a nested hierarchical prefab this **doesn't erase the overrides** made by its hierarchical prefab parent. (See ***Instance Overriding*** for more information)

Reset

Reset the Hierarchical Instance to its prefab state.

Note that when used on a nested hierarchical prefab this **erase the overrides** made by its hierarchical prefab parent. (See ***Instance Overriding*** for more information)

Disconnect

Disconnect a Hierarchical Prefab Instance from its Hierarchical Prefab. This allows an object to not be an instance of a hierarchical prefab anymore.

This action can't be UNDO. Use the Load Button to remake the object an instance of a hierarchical prefab.

Note that when used on a **Hierarchical Prefab** (The resource saved in the Assets folder not a scene instance) this action is possible but very risky as it's irreversible.

Instantiate a Hierarchical Prefab by Script

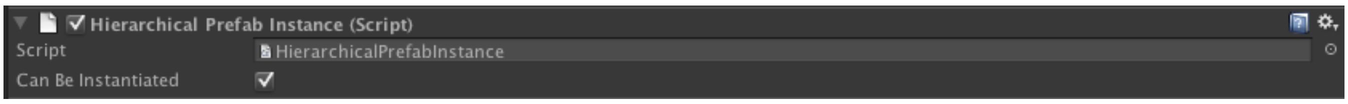
New Instantiate Method

```
using UnityEngine;
using System.Collections;

[AddComponentMenu("HierarchicalPrefabSample/HierarchicalPrefabSampleSpawner")]
// Sample class used to spawn a hierarchical prefab with the special instantiate method
public class NestedPrefabSampleSpawner : MonoBehaviour
{
    // The prefab to spawn
    public GameObject prefabToSpawn;

    // Update is called once per frame
    private void Update()
    {
        // If the mouse is clicked
        if(Input.GetMouseButtonUp(0))
        {
            // Spawn the prefab at the spawner position
            HierarchicalPrefabUtility.Instantiate(prefabToSpawn, transform.position, transform.rotation);
        }
    }
}
```

To Instantiate a hierarchical prefab you can't use the ***GameObject.Instantiate*** method you have to use the ***HierarchicalPrefabUtility.Instantiate*** instead.

Note that you can use ***HierarchicalPrefabUtility.Instantiate*** to instantiate any prefab (Classic or hierarchical) or duplicate scene Game Object as well. So you can entirely substitute the use of the default ***GameObject.Instantiate*** method by ***HierarchicalPrefabUtility.Instantiate.***

Flag the Hierarchical Prefab that can be instantiated



Hierarchical Prefab Instance (Script)

Script	HierarchicalPrefabInstance
Can Be Instantiated	✓

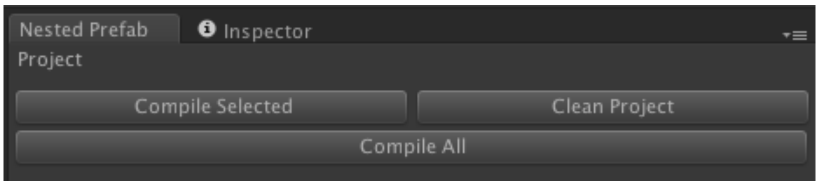
You have to flag each Hierarchical Prefab you want to instantiate by script as ***Can Be Instantiated***.

To do so, just check the **Can Be Instantiated** check box on the **Hierarchical Prefab Instance** component that you will found on the Hierarchical Prefab Root Game Object.

Compile

The last step before being able to instantiate a Hierarchical Prefab is compiling it. To do so, just use one of the compile button. Either **Compile Selected** if you only want to temporary compile and test the current prefab. Or **Compile All** if you want to compile all the instances that can be instantiated.

Before any build it's recommended to use **Compile All.**

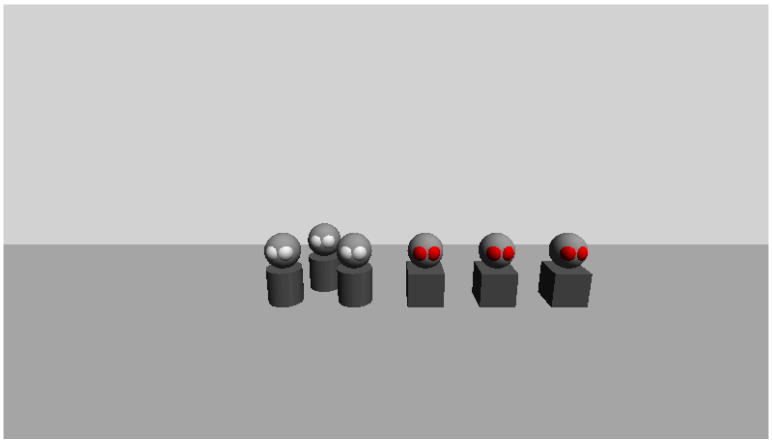


Instance Overriding

Hierarchical overrides

Like the Classic Unity Prefabs it's possible to change parameters on a prefab instance. Parameters that are not the same on the instance than on the prefab are called overrides.

This feature allows us to have several instances coming from a same prefab but with different characteristics.



In the image above the default head prefab has white eyes but three of its instances have overridden the eyes material to a red one.

Classically unity can do that on a scene with several prefab instances.

But the hierarchical prefabs add the possibility to have overrides on nested prefab.

In the image above there are several instances coming from 2 Hierarchical prefabs **Character1** (the cylindrical body one) and **Character2** (with the cube body).

Each one of them has a nested instance of the same prefab (the **Head** prefab).

But The **Character2** prefab come with an override on its nested instance of the head prefab. (Red eyes)

Thus any instance of **Character2** will have red eyes.

Warning: A prefab can only add overrides on prefab instances directly nested in him!

So if the eyes were instances of an **Eye prefab** nested in the head prefab, it will have been impossible to change their colors on a character basis. (All characters would have had the same eyes color). But in this case two different heads prefabs can override the eyes colors and thus have different eye color.

In short, overrides can only be saved on one hierarchy level. (But don't make the confusion between scene graph levels and nesting hierarchy levels)

Overriding Object References

References are object fields you can find on component into which you can drag and drop game objects or other components to link them to this component. This is extensively used in Unity.

We seen in the previous section that prefab instances nested into hierarchical prefab can be overridden. So are the references fields on their components. So nested prefabs can reference object on their hierarchical parent hierarchy. Also a Hierarchical prefab instance can set links to its nested prefabs.

But be aware that even though this operation seems to work there are many cases in which this links will be broken when updating/saving the hierarchical prefab.

What you can reference

- An Instance of prefab nested in a Hierarchical Prefab can reference it or a part of it.

What you can't reference (Or else it will break)

- A Hierarchical prefab can't reference a prefab nested in it.

- Two Nested Prefabs can't reference each other

- A nested prefab can't reference an object in top of its direct hierarchical prefab parent.