Prefab Fixer - Manual

Unity has once more misplaced your GUIDs?
Your prefabs fear that not even reimport-all can help them?

Don‘t panic!
PrefabFixer is here to help.

Table of contents

1. Replacing Objects

PrefabFixer supports two differenct commands: **Fix** and **Replace**.

Btw.: You can not only execute this on prefabs. You can replace any object with any prefab if you like. That's very useful if you have duplicated some stuff (like a button) and only later decide to make it a prefab.

Here is what **Replace** does:

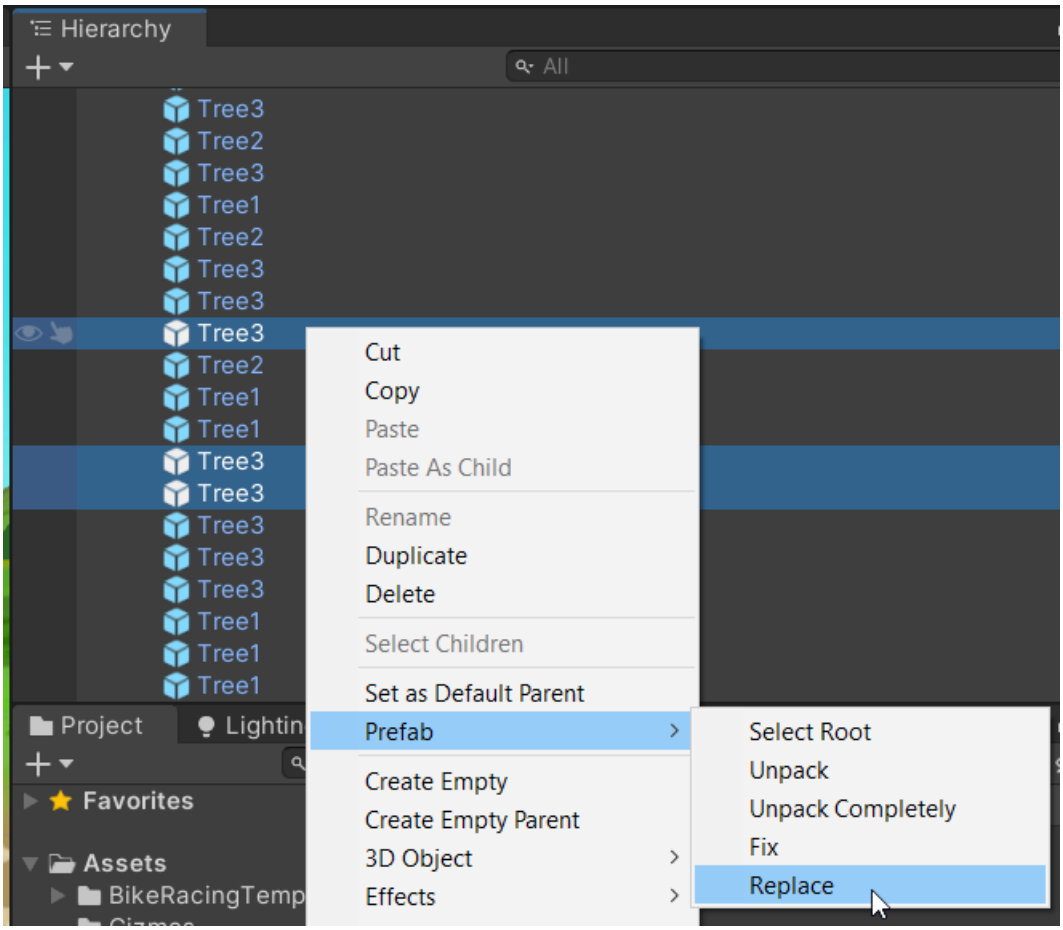
1. For each selected object it takes the transform data (position, scale, rotation, hierarchy index) and memorizes it.

2. It instantiates the prefab you have chosen (one for each object) and applies the transform from step 1.

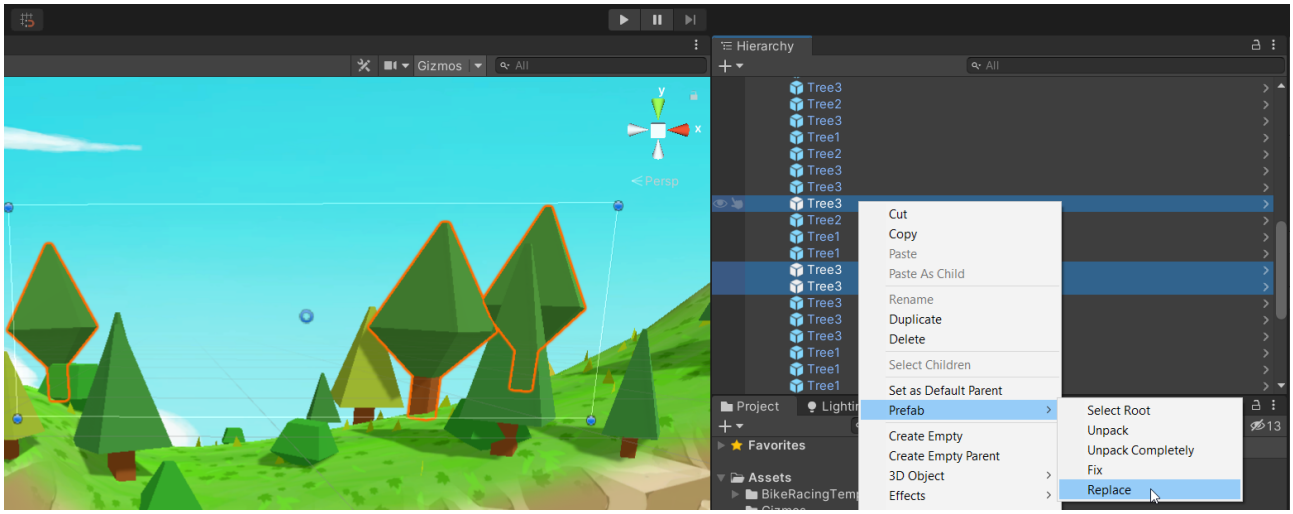
3. It searches for any references which point to the old objects and changes them to the new ones.

4. It deletes the old objects as they are no longer needed.

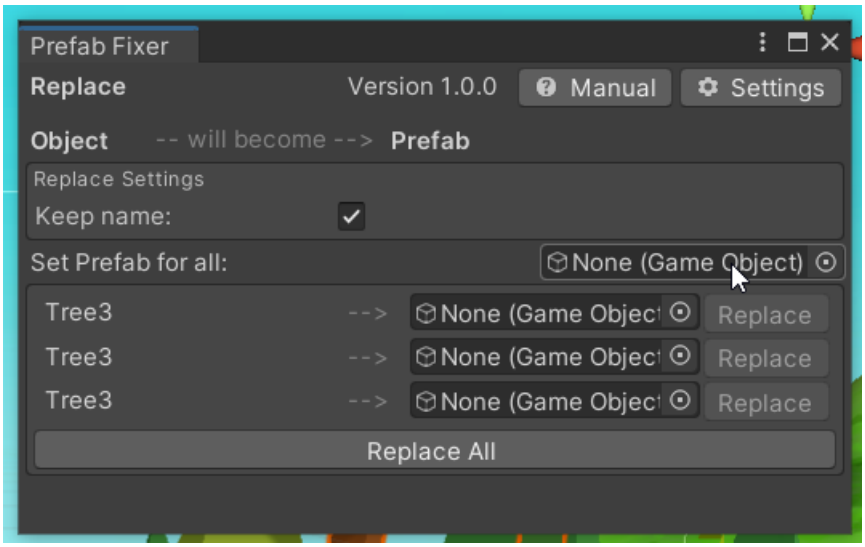
You execute it with **Right-Click > Prefab > Replace.**



Imagine you have some trees in your scene which you want to replace. In this case it's three instances of **Tree3**.

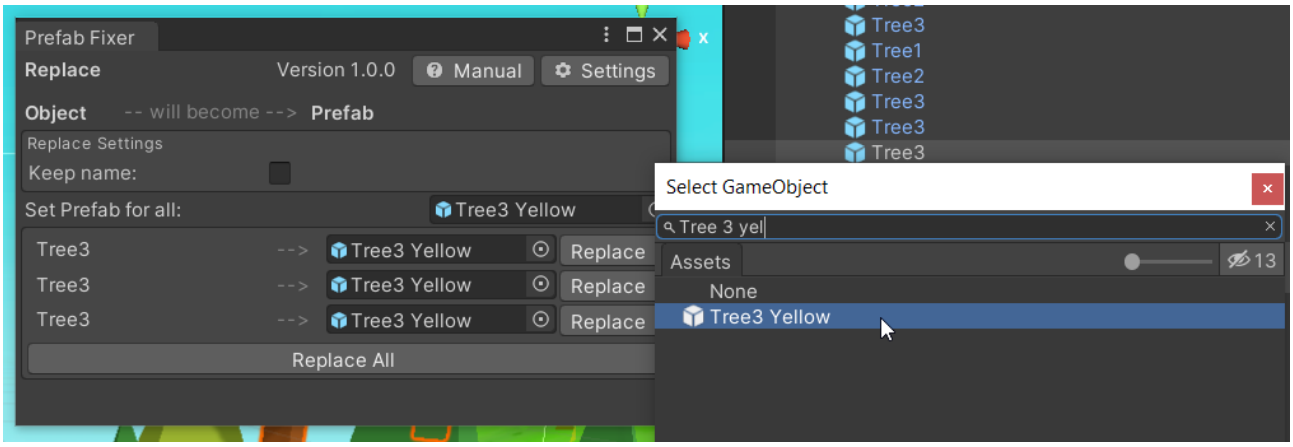


You will get this window:

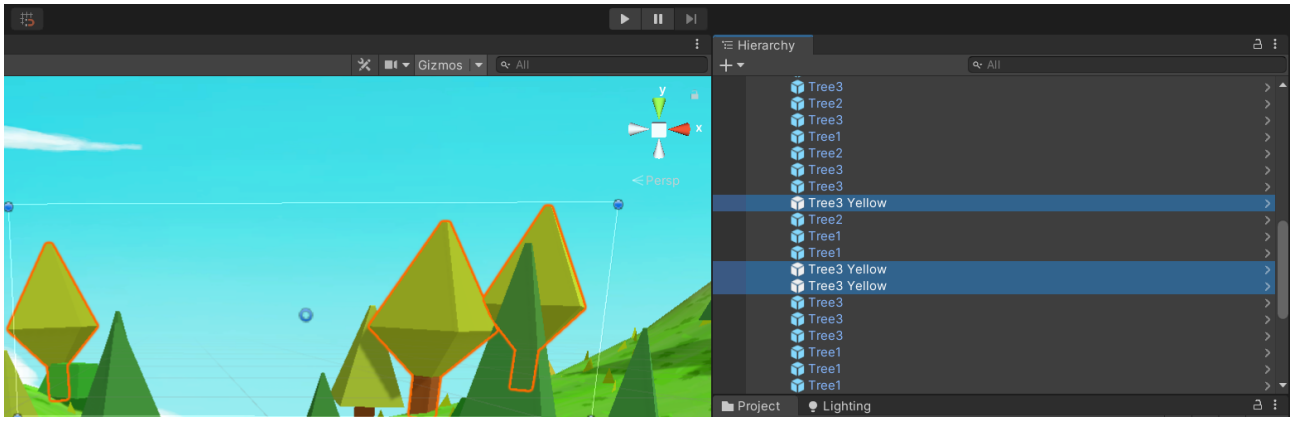


Now you can choose a replacement for each object (or you use the „Set Prefab for all" field to use the same for all of them).

In our example we want to replace them all with a prefab called „**Tree 3 Yellow**".

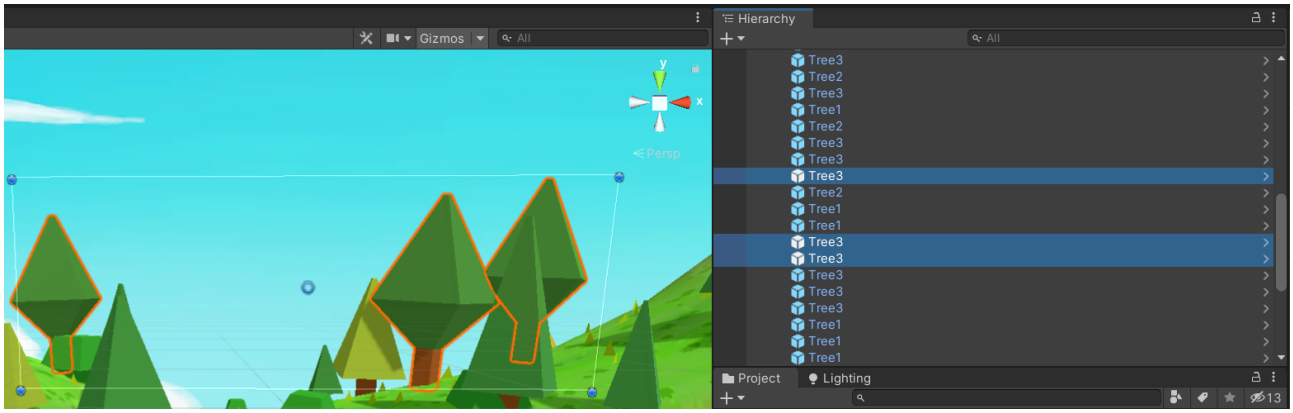


Once that is done you can hit „Replace All“ and you will see that the trees have now been replaced with the new yellow prefabs.



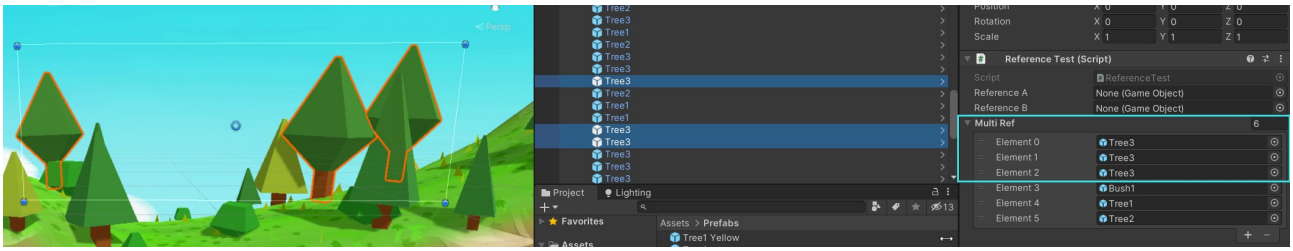
NOTICE: The name of the game objects has also changed. If you do not want that then you can disable this with the „Keep name“ option.

This is how the scene looked before (notice the difference):

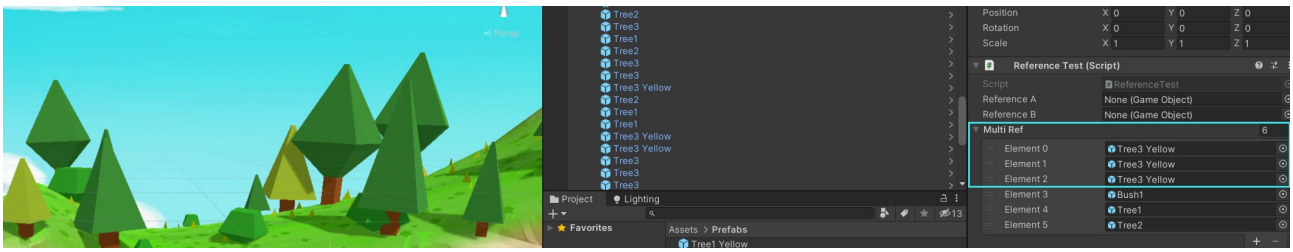


INFO: The references to the trees have also been updated in the scene.

Before:



After:



2. Fixing Objects

Fixing is a complex action (compared to replacing).

Btw.: You can not only execute this on prefabs. You can try to „connect" any object to any prefab if you like. That's very useful if you have duplicated some stuff (like a button) and only later decide to make it a prefab.

Here is what it does:

1. For each selected object (and the children of that object) it takes the serializable data of all the components and scripts and memorizes it.

2. It instantiates the prefab you have chosen (one for each object).

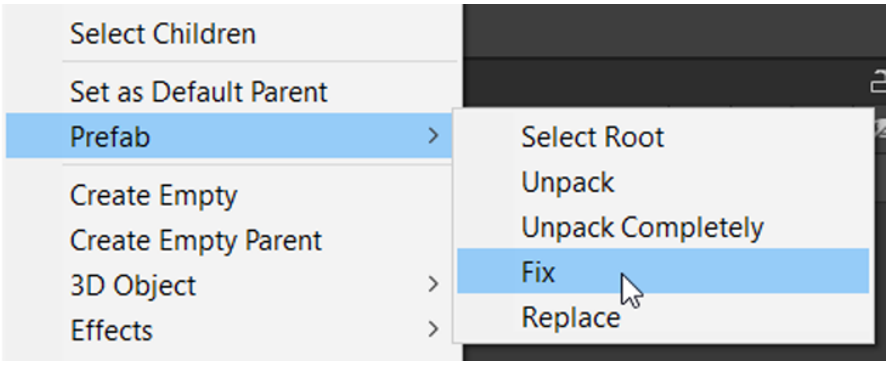
3. It then COMPARES each child and component of the old object with the new prefab instance. If it finds a match (by object name or component type) it copies over all the data from the old object to the new one. This results in „prefab overrides" which you can then use to revert things you don't need.

4. It checks if any of the children on the old object do NOT exist in the new prefab instance. If they are missing it copies them over to the new prefab hierarchy. None of your old objects are lost!

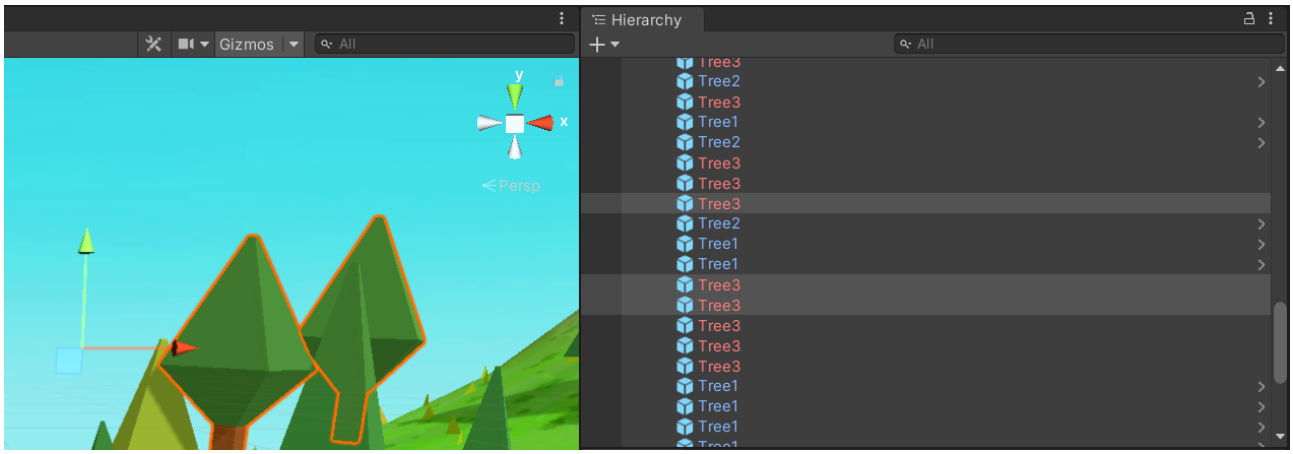
5. It goes through ALL the objects and components of the old game object. It checks if any are referenced by any script in the entire scene. If yes, then it changes these references from the old to the corresponding new object. All references will work as expected after the fix!
This step may take a while for big scenes, so please be patient. A time indicator will be shown at the bottom of the PrefabFixer window.

6. It deletes the old objects as they are no longer needed. This can be disabled in the settings (see „Delete Original").

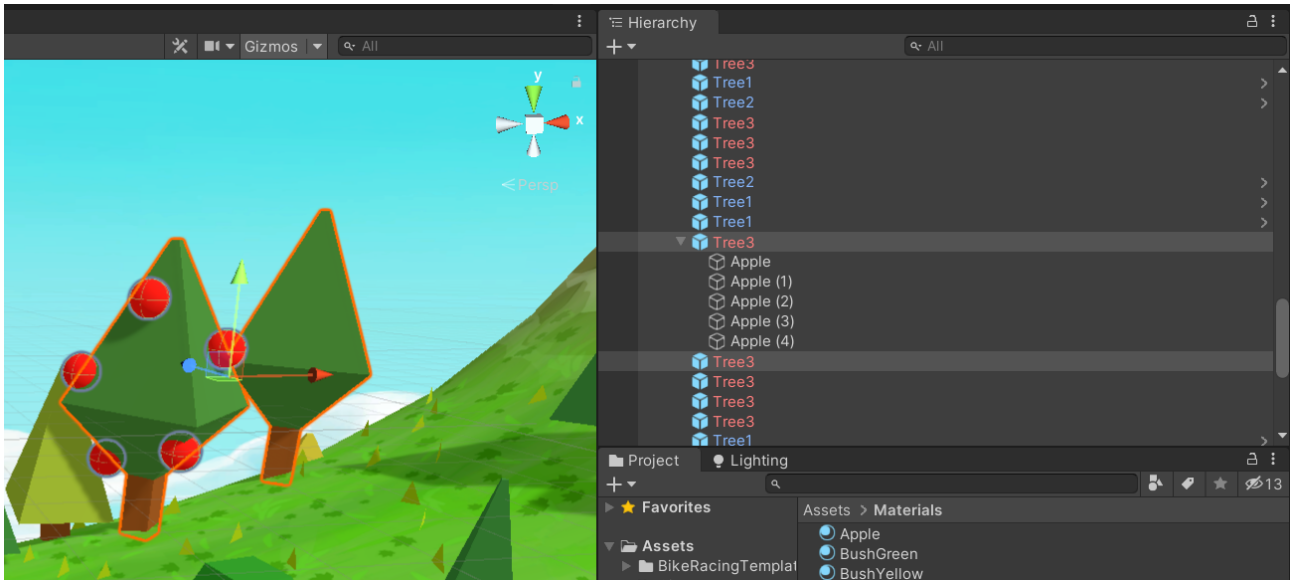
You start a fix with **Right-Click > Prefab > Fix**.



Let's assume that somehow we deleted the **Tree3** Prefab in a way that even GIT can not find it.

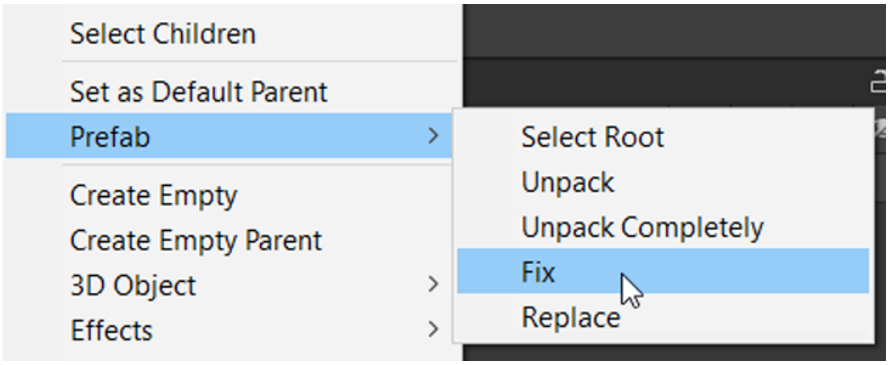


We we certainly do not want to have to place all the trees again. Especially since we have added some juicy apples to one of them:

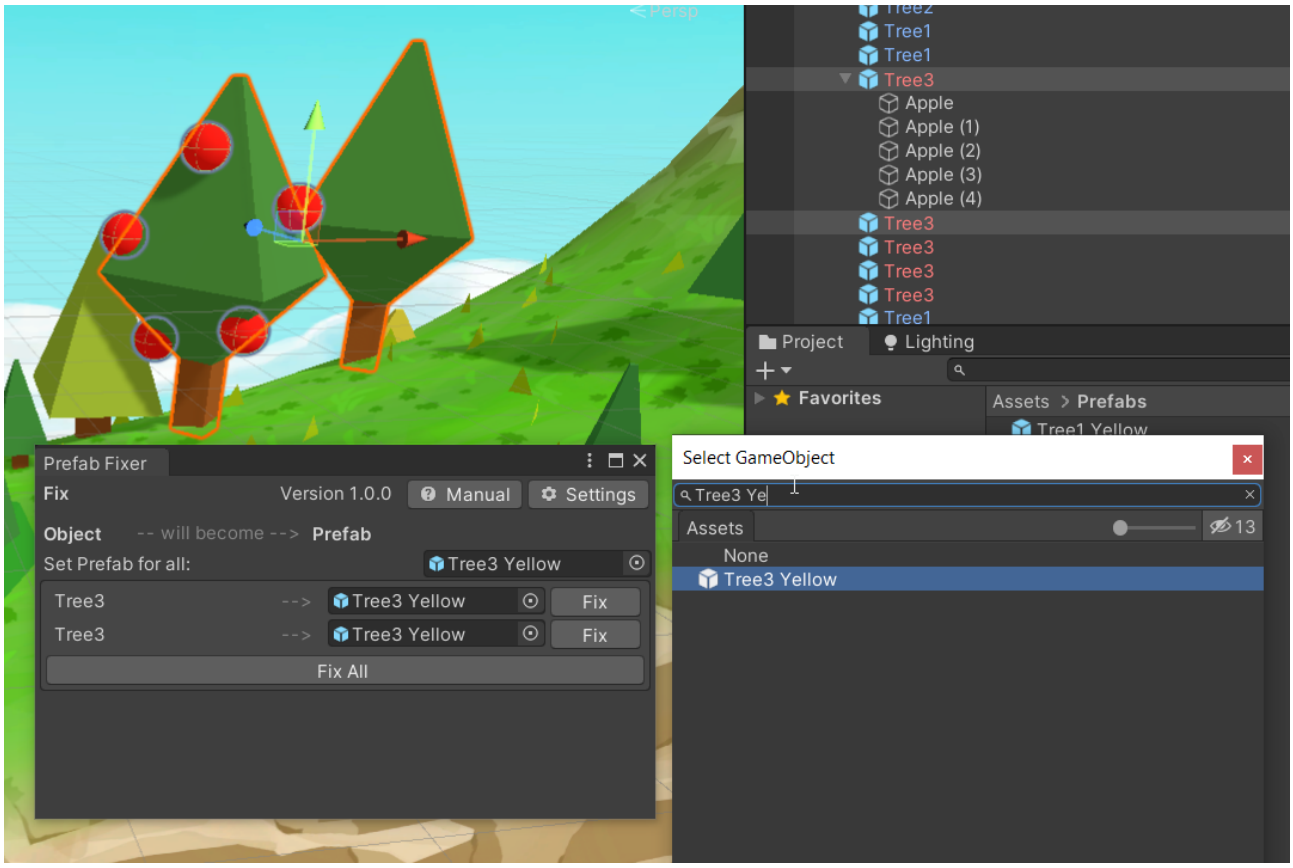


Let's try to fix these prefabs.
Luckily we still have a Prefab which is similar to the old tree. It's called „**Tree3 Yellow**".

Let's start with: **Right-click > Prefab > Fix**



The PrefabFixer window appears. Let's pick „**Tree3 Yellow**" as a basis for our fix.



After hitting „Fix All" we can check if the fix did what we expected.

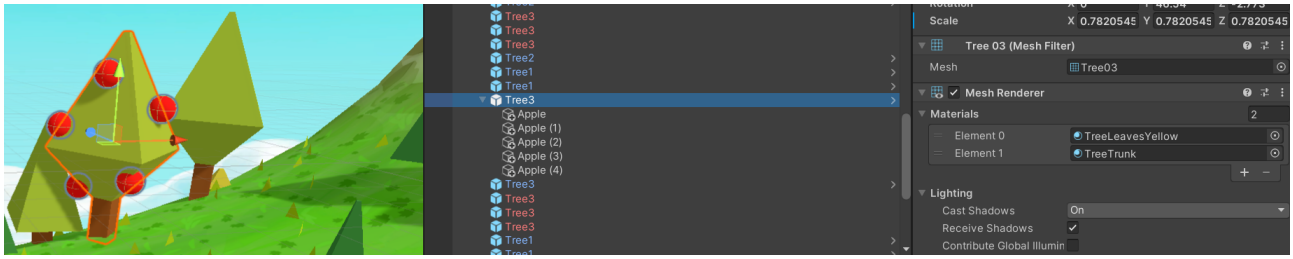
Nice, the prefabs are connected again and the apples are still there.
But wait: did we not fix it with the „Tree3 **Yellow**" prefab. Why is the tree still green?

That's because all the properties of the tree have been copied and applied as „prefab overrides".

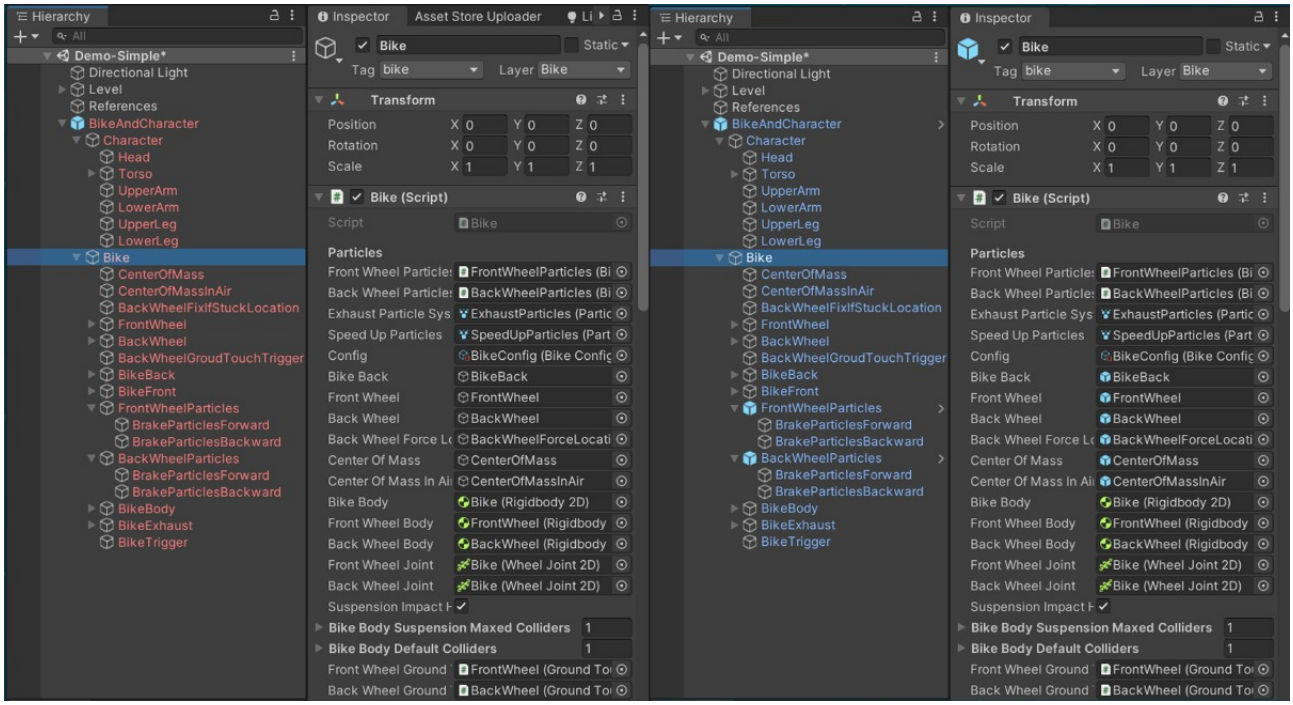
Let's revert those which we do not need.

Et voilà !

Our tree is now yellow and has some apples. We have successfully converted a formerly broken prefab (with apples) into a normal prefab (with apples).

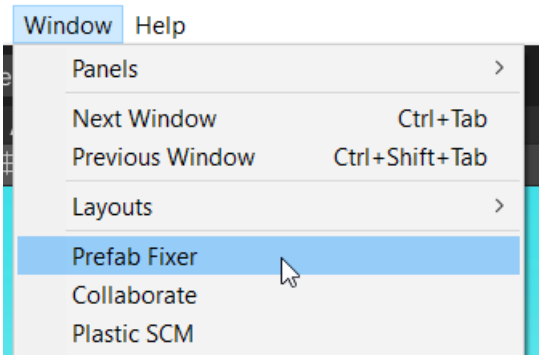


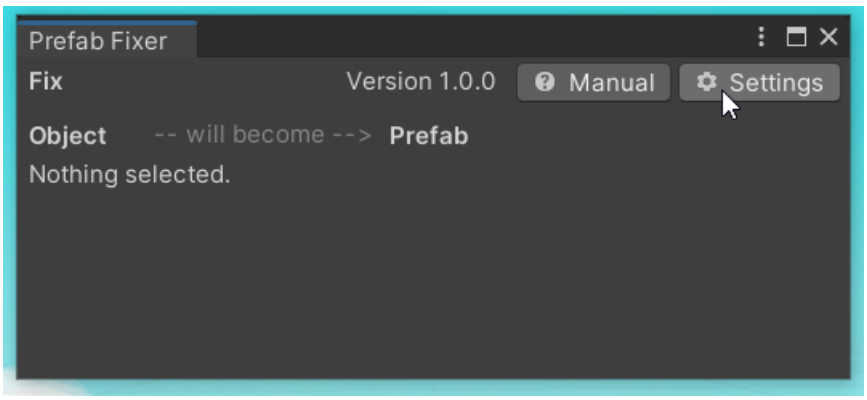
Btw.: The PrefabFixer can fix much more complex prefabs too. Give it a try ;-)

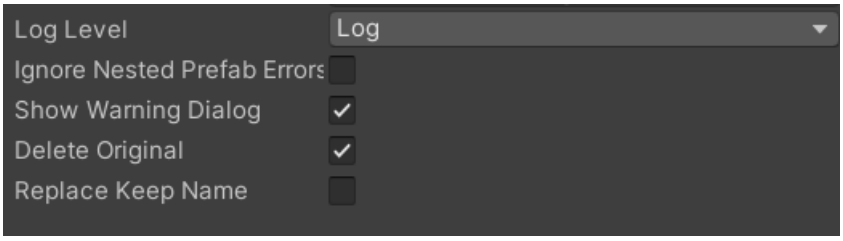


3. Settings

You can access the settings through the PrefabFixer Window (**Window > Prefab Fixer**)







Ignore Nested Prefab Errors

If enabled then broken nested prefabs will be unpacked and copied. Otherwise an error will be shown and the fix will be aborted. If you have nested broken prefabs you have to fix them first before fixing the root.

Show Warning Dialog

It's related to „Ignore Nested Prefab Errors". Set this to false to skip showing the dialog for each broken nested prefab.

Delete Original

Should the old object be deleted after it was replaced/fixed?

Replace Keep Name

Applies to „Replace" only. Defines whether the existing name or the name of the new prefab should be used.