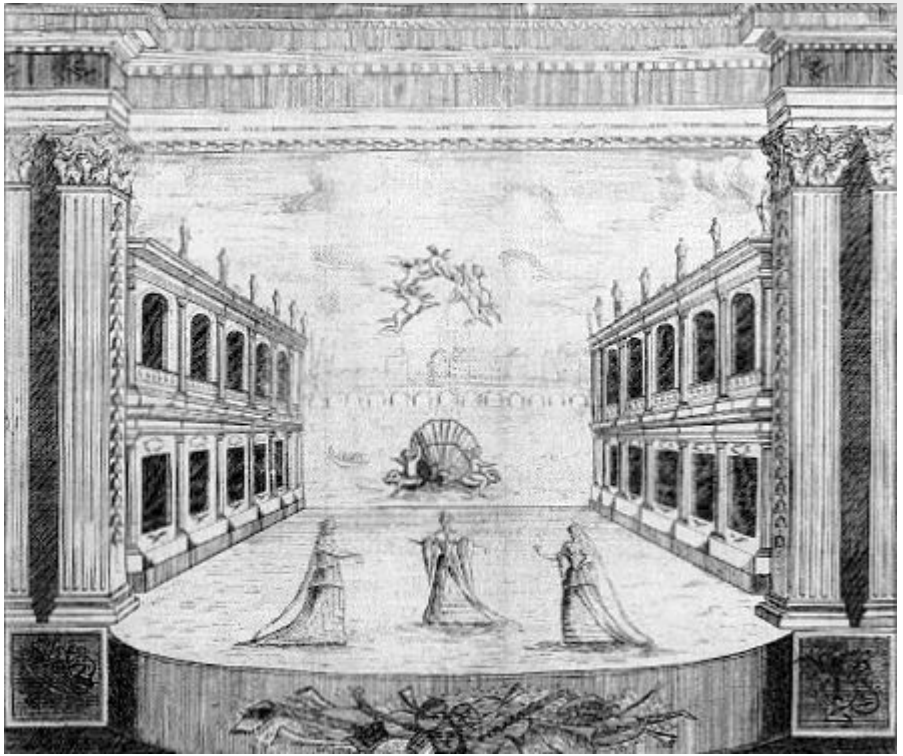
Unity Scenes & Prefabs

The stuff you didn't know you wanted to know
...I think...

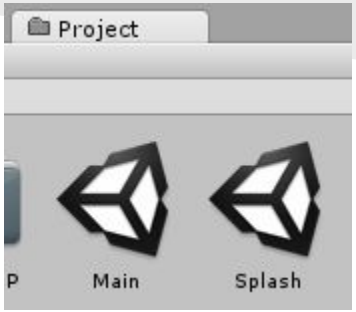
What are scenes?

- Abstract: like a scene in a play
- A staged-moment



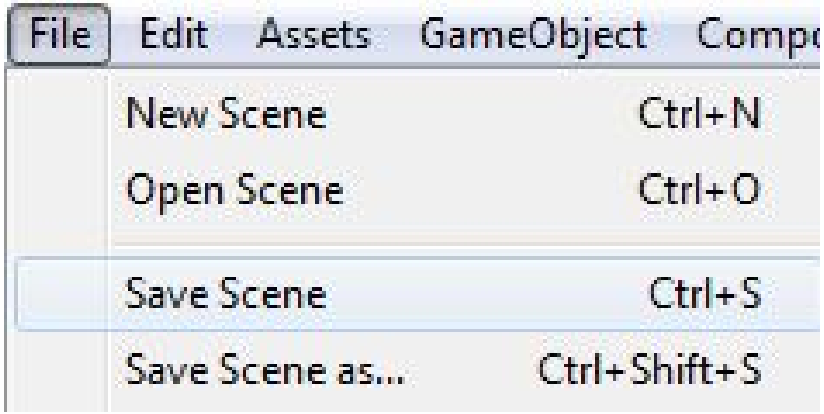
What are scenes?

- More concrete:
  - A file with a list of GameObjects in it
  - Useful for making levels, standalone menus
  - Has a *.unity file extension



Appears like this in the Project tab

How to make a scene?

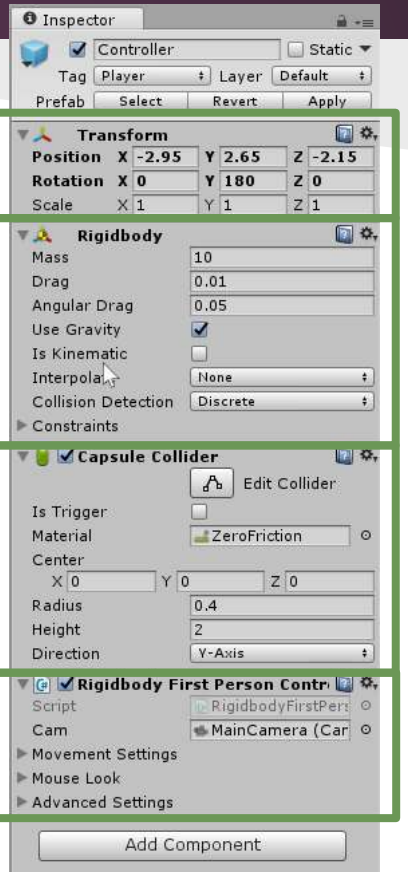


What does a scene look like?

- The Hierarchy tab is a tree of ALL GameObjects inside the scene.

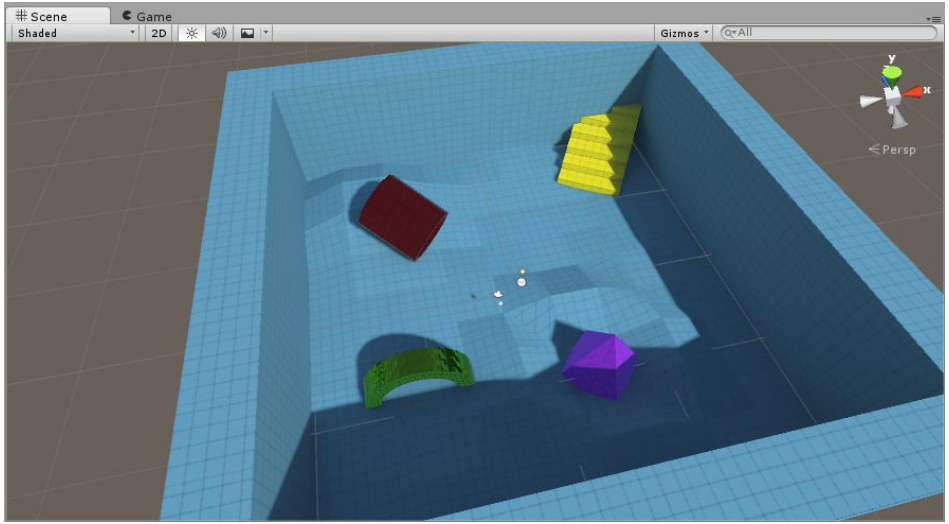
What does a scene look like?

- The Inspector tab lists all the Components attached to a GameObject.

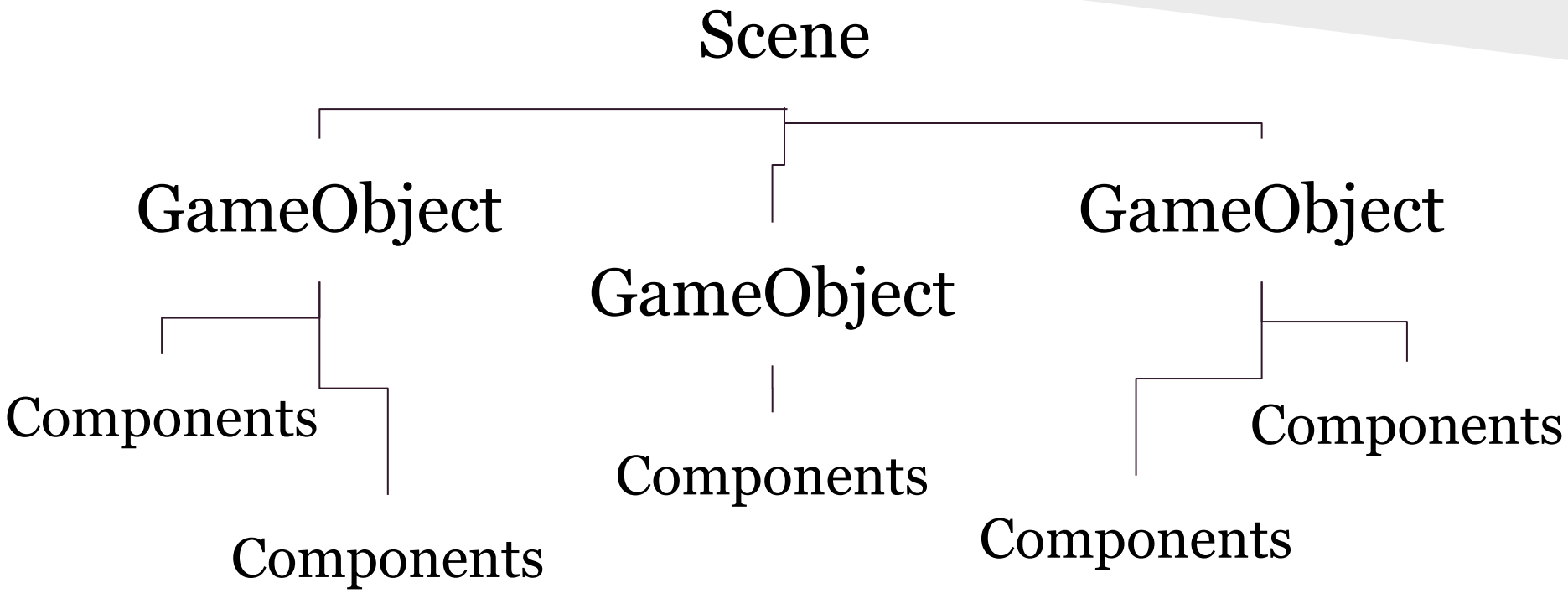


What does a scene look like?

- The Scene tab is a visual representation of the GameObjects and their Components in the scene.

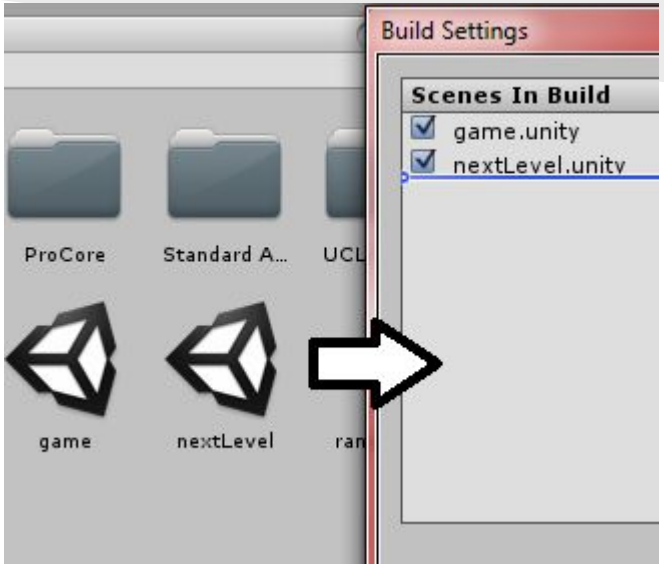


Diagram


Scene
GameObject
GameObject
GameObject
Components
Components
Components
Components
Components

Remember the build settings!

- Add scenes by dragging & dropping them into the Build Settings dialog
- The top-most enabled scene will play first!



Load a scene while playing

- Can be done using “`SceneManager`”
- To access `SceneManager`, add the following line at the top of the file:
  - `using UnityEngine.SceneManagement;`

Load a scene while playing

The easy way:

- `SceneManager.LoadScene(int index)`
  - where `index` is the index listed in Build Settings (starts at 0)
- `SceneManager.LoadScene(string name)`
  - where `name` is the name of the scene file, *without* the ".unity" file extension

Pros & Cons

Pros

- It’s easy
- Uses less memory

Cons

- Suddenly pauses the game
- Can take forever
- Looks really bad

Load a scene without pausing

- `SceneManager.LoadLevelAsync(int index)`
- `SceneManager.LoadLevelAsync(string name)`
- Both return `AsyncOperation`
  - can configure whether to load the next scene automatically
  - indicates how much progress was made

Pros & Cons

Pros

- Better experience
- Progress bar support!
- Control when the level loads

Cons

- Uses more memory
- Takes longer to load

Merge scenes while playing

The easy way:

- `SceneManager.LoadLevel(int index, LoadSceneMode.Additive)`
- `SceneManager.LoadLevel(string name, LoadSceneMode.Additive)`

Pros & Cons

Pros

- Makes it easier to split open levels into parts

Cons

- Suddenly pauses the game
- No way to tell when loading ends
- No automated memory clean-up process

Merge scenes while playing

- `SceneManager.LoadLevelAsync(int index, LoadSceneMode.Additive)`
- `SceneManager.LoadLevelAsync(string name, LoadSceneMode.Additive)`
- Both return `AsyncOperation`

Pros & Cons

Pros

- Better open-world experience
- Progress bar support!
- Control when the level loads

Cons

- Uses more memory
- No automated memory clean-up process

Scenes Summary

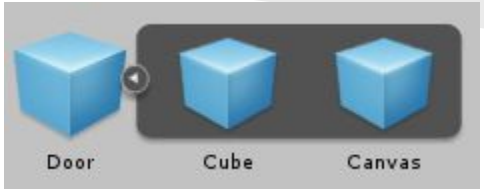
- Use `SceneManager.LoadLevel()` or `SceneManager.LoadLevelAsync()` to switch to different scenes
- Add `LoadSceneMode.Additive` as the last argument to these functions to copy objects from a scene into the one we're playing in
  - It's highly recommended to clean up the previous scene's objects where appropriate to save memory

Scenes Recommendations

- Devote at least one scene for the menus that load other scenes
- Otherwise, divide scenes by individual level
- If using open-world setup, divide scenes by world parts and use `LoadSceneMode.Additive`
- Smaller scenes are better!
  - They take up less memory, making it quicker to edit
  - Less prone to errors

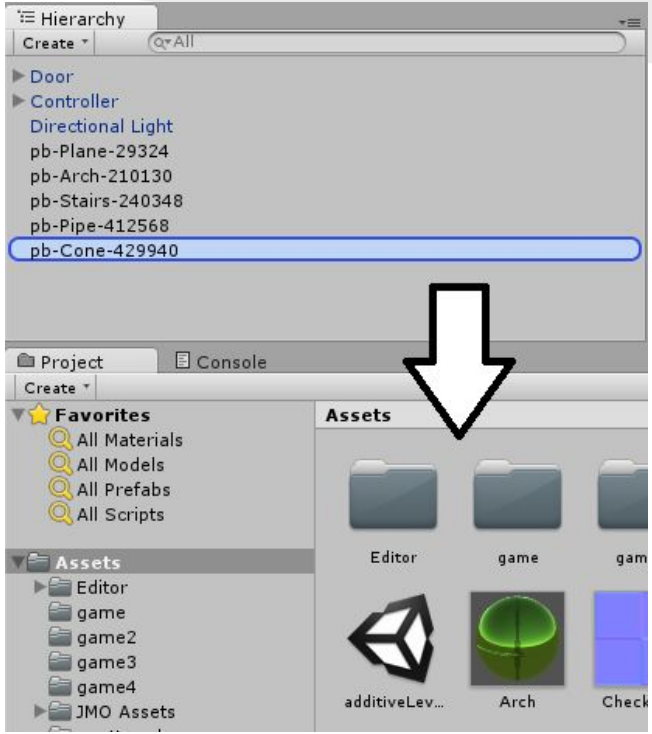
What are prefabs?

- Short for prefabricated object
- Allows copying objects from one scene to another
- Allows *modifying* a common object in every scene (only in the editor, though)
- Has *.prefab file extension

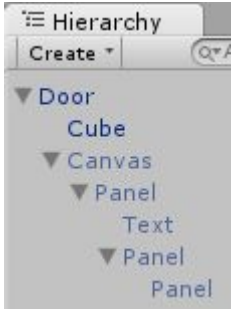


How to make a prefab?

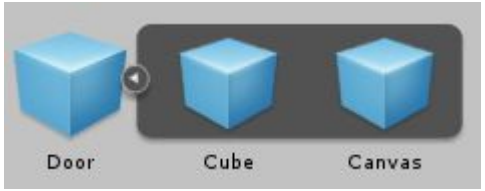
- Drag & drop an object into the projects tab



What does a prefab look like?



Appears blue in the Hierarchy dock



Appears with this icon in the Projects dock

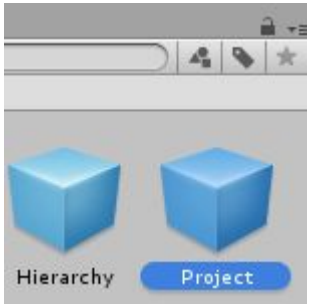
What does a prefab look like?

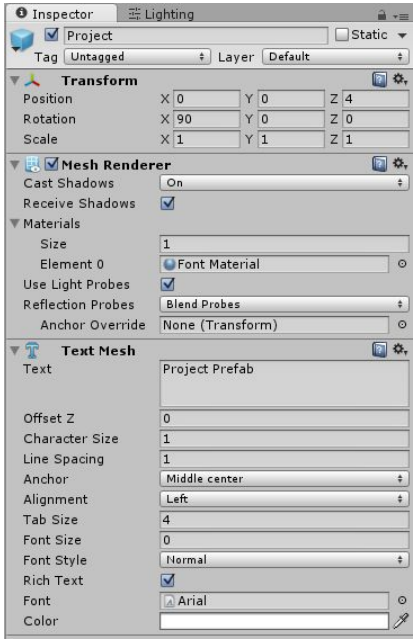
- You can now drag the prefab from the Project dock to the Hierarchy or Scene dock to create a copy of the object the prefab represents!
- Useful for copying objects in one scene to a different scene using the editor

Modifying prefabs: Project

You can modify a prefab in the Project dock

Just select the prefab in the Project dock, then make changes in the Inspector dock. You are now modifying the prefab file directly!





Modifying prefabs: Project

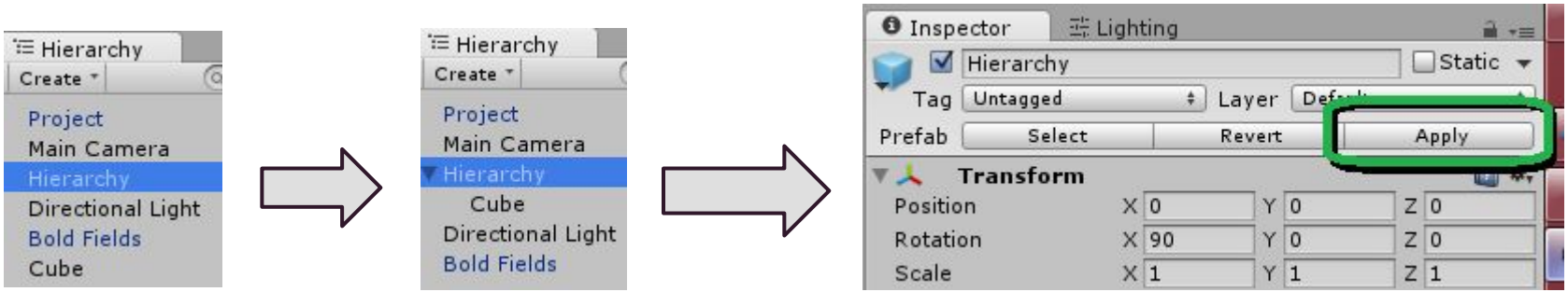
So what happens if you do that?

- All scenes with that prefab in it will be updated with that modification.
- Only the components in the top-most and second-top-most objects in the prefabs can be modified.

Modifying prefabs: Scene

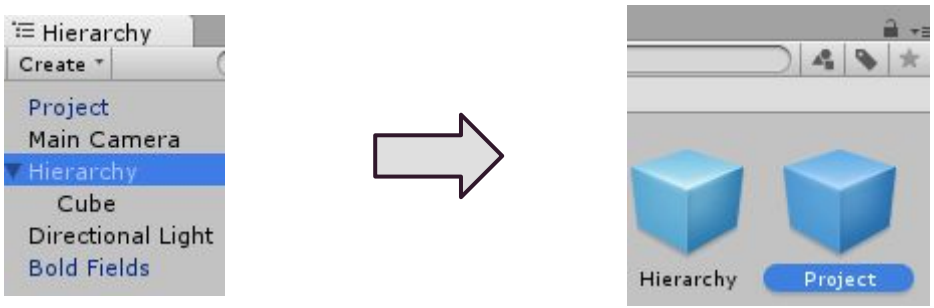
You can modify a prefab in a Scene

To do so, make any modifications in the Scene, Hierarchy, and/or Inspector on an instance of a prefab. Finally, select the prefab and click the Apply button under the Inspector



Modifying prefabs: Scene

Alternatively, you can drag an object from the Hierarchy dock into the prefab file in the Project dock to modify it.



Modifying prefabs: Scene

So what happens if you do that?

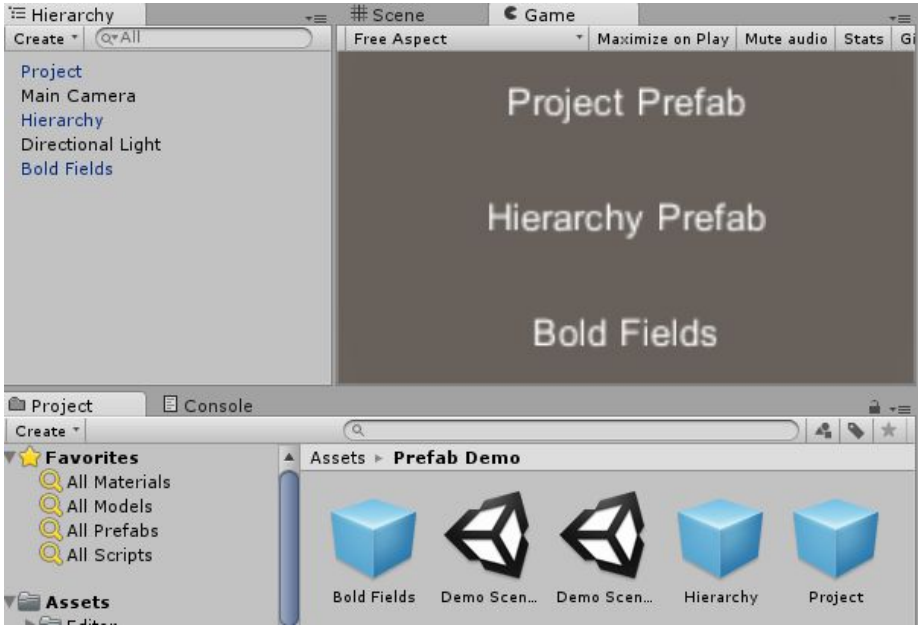
- Any new child object in the prefab will now be stored as part of that prefab
- Any bolded (i.e. modified) fields in the inspector will be stored, removing the bold effect on each of those fields
- All scenes with that prefab in it will be updated with that modification.
- With this method, you can modify any child object in that prefab, and even add or remove them!

Modified Fields in Prefabs

- If there are any modified (i.e. bolded) fields in a prefab, those values will be stored in the scene it's in.
- This means a scene can store variations of a single prefab.
- Makes no sense? Let's demonstrate:

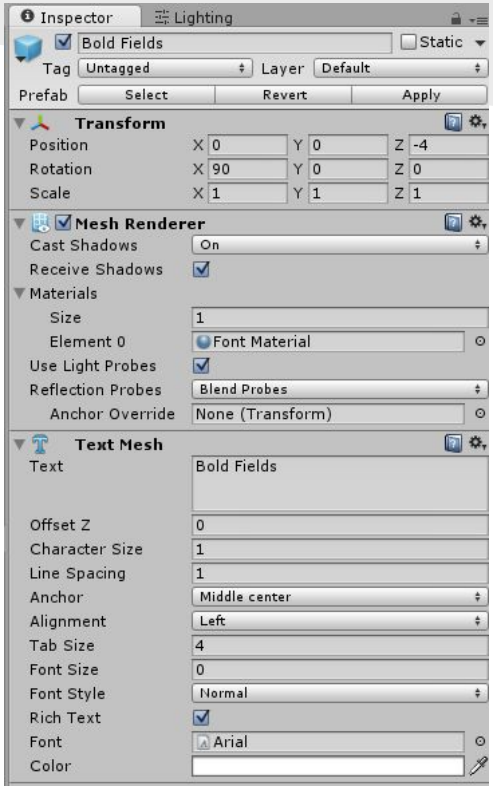
Modified Fields in Prefabs

Let's say there are two scenes with "Bold Fields" prefab:



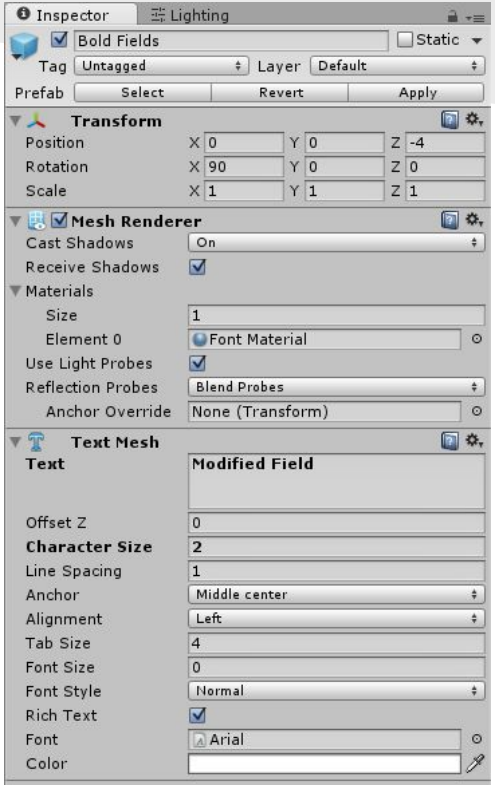
Modified Fields in Prefabs

In Demo Scene 1, none of the prefab's fields are modified, so they aren't bolded:



Modified Fields in Prefabs

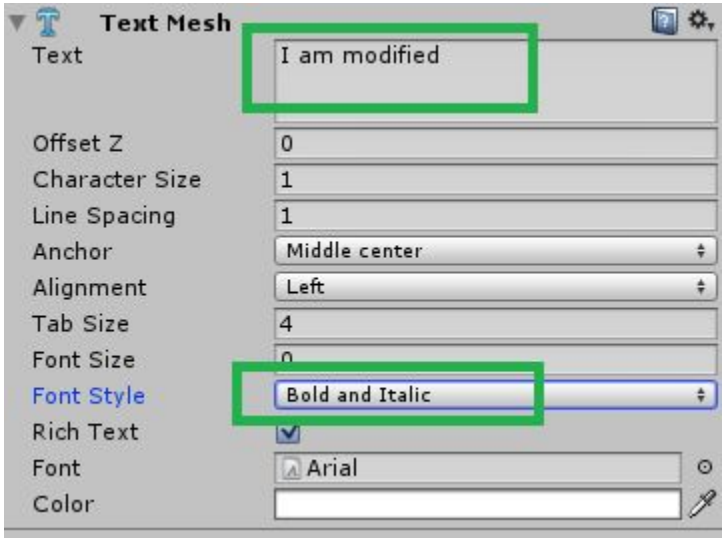
And in Demo Scene 2, the “Text” and “Character Size” fields are modified, thus appearing bolded:



Modified Fields in Prefabs

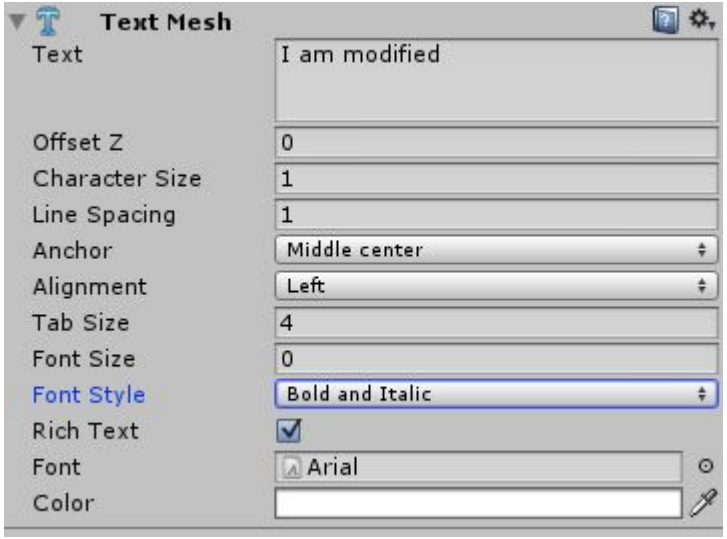
Let's modify the "Bold Fields" prefab in the Project dock

- Remember that this is supposed to modify the instance in each scene



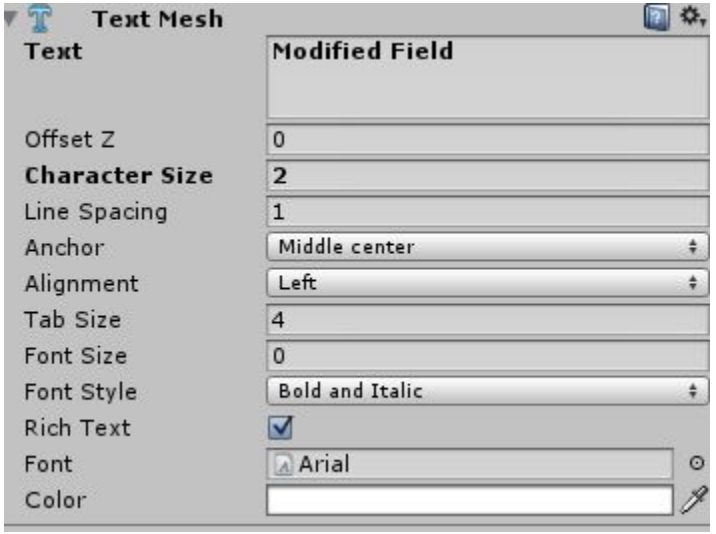
Modified Fields in Prefabs

In Demo Scene 1, if we select the “Bold Fields” object, all the fields will be modified with the new values



Modified Fields in Prefabs

In Demo Scene 2, if we select the “Bold Fields” object, the Font Style will be updated, but the bolded field Text will not!



Modified Fields in Prefabs

If you want to change a modified field back to what's stored on the prefab file, right-click and select "Revert Value to Prefab"

Prefabs Summary

- You can copy an object from one scene to another by creating prefabs
  - Once a prefab is created, just drag that file into the new scene to copy it
- Prefabs share properties between scenes

Prefabs Summary

- Changing the Prefab in the Project dock, hitting the Apply button in the scene, or dragging & dropping the object to a pre-existing prefab file causes global changes:
  - Any prefab instances in a scene will be updated with the new changes, *if* the component fields aren't bolded

Prefabs Summary

- If a field in a prefab instance in a scene is modified, those changes will remain local to the scene
  - Modified fields will *not* be updated if the prefab file changes

Prefabs Recommendation

- Great for duplicating objects repeatedly
- Useful for sharing information between scenes
- Powerful enough to create subtle variations in different scenes
- Like scenes, smaller prefabs are better
  - Quicker to edit, less error prone