Prefabs

Since the Unity workflow is built around prefabs, I figured I'd document some specific use cases for the many features introduced in the Unity LTS 2019 release which added support for prefab variants and nested prefabs. On this page, I'll cover some good practices and features these prefab features provide.

I'd highly recommend heading over to devassets.com to grab some of the assets you see featured across the Unity pages on Knoats. They are entirely free and give you a lot to work with when learning. If you can afford it, I would recommend donating to the developers. Not only does this unlock more assets you can get with the package you donated to, but it shows support to the developer that organized all of these great assets in one place for you to learn with.

Positioning Prefabs

Being relatively new to Unity, I began by grabbing some assets off the Unity Store. Like most free assets on the store, these did not come entirely assembled for me and required me to work a bit to get things in a state that is usable for even the most basic games. This has been a good learning experience, and required no scripting, so if you are new to Unity and not quite ready to script, doing this will give you experience creating prefabs, working with materials, shaders, lighting, textures, and much more.

At first when creating a prefab of an object that exists within your scene, you may see something like the below when opening the prefab to edit

This is clearly not the orientation that we expect this `SciFi_Rover` vehicle to have when initially placed in our scene. To fix this, be sure you are editing the prefab in the prefab editor and NOT directly within your scene. Then adjust the transform to be in the orientation desired.

Below, we see the initial transform settings

First, set all but the `Scale` of your object to `0`. Shown in the screenshot below, there will be many cases that this does not produce the desired results, so we still need to modify the transform

further

After making some adjustments, the object's final orientation within the prefab editor is seen below

And the final transform properties of the root GameObject are now much cleaner -

---