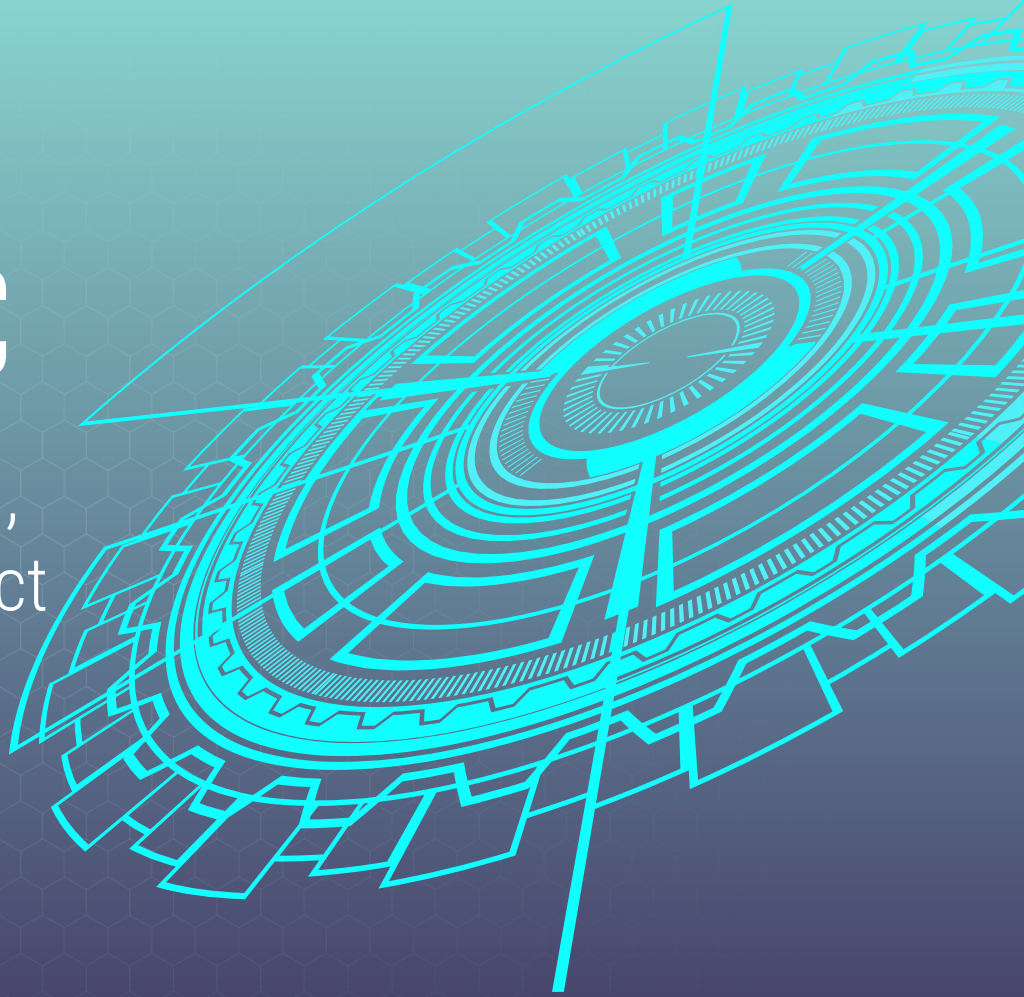
Week 18 GWC

2D Gamekit Walkthrough, Sprite Creation, and Project Work!



Katie Tooher and Carmen Seda

Where We're At

Next week, we are doing a hardware demo, so you'll only have about an hour to work on your projects.

The week after that is Spring Break, so there's no class.

Class on the 17th and 24th

Graduation on the 31st

Women In Tech Shout-Out of the Week: Melanie Saunders

- Former Deputy Associate Administrator of NASA
- Acting Chief Financial Officer of NASA
- Chairs the NASA Mission Support Council
- Meritorious Presidential Rank Award
- 2 NASA Outstanding Leadership Medals
- NASA Exceptional Service Medal
- Silver Snoopy

Warm Up

Ask a user what their age is and tell them whether they are old or not.

Ex:
50 -> You’re a grandpa!
10 -> You’re a baby!

Moonguy

nwoida.wix.com/moonguy

Creating Your Own Sprites!

Make an animated sprite for your game using piskel

Where to do it: https://www.piskelapp.com/

Show before you start (importance of grid size)
https://www.youtube.com/watch?v=k0jlBbmwvGk

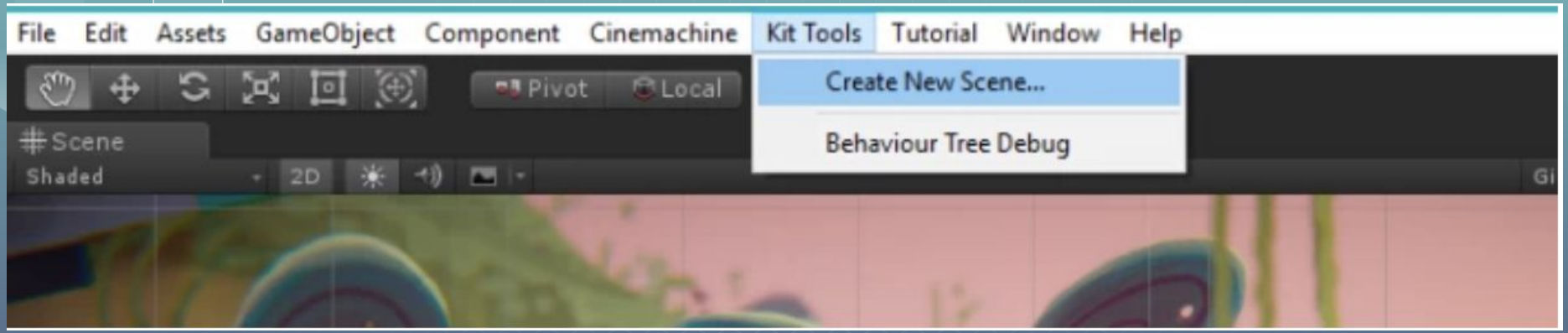
Tilemaps for 2D games

- 2D games in Unity use the tilemap feature which is a rapid prototyping feature that allows you to “paint” on a game architecture
  - Each tile in a tilemap represents one sprite of your 2D game
- Per each tile asset there are 4 main properties that you see in the inspector window
  - Preview: Displays a visual preview of the Tile with the selected Sprite.
  - Sprite: Select the Sprite texture to be rendered on this Tile. Click the circle icon to the right to open the object picker window to select from the available Sprite Assets, or drag a Sprite directly onto this box.
  - Color:Tints the Tiles placed on this Tilemap with the selected Color. When set to white, the Tiles are rendered with no tint.
  - Collider Type: Defines the shape of the Collider generated for the Tile.

Collisions

- To create events in your game (death/points) you need collisions
- Colliders: are components that define the shape of a GameObject for the purposes of physical collisions used in 2D game worlds
  - Essentially you are placing boundaries around a sprite so that when that boundary is touched, it triggers some event
  - It is invisible and doesn't need to perfectly match the sprite shape
- In Unity for 2D games there are box colliders and circle colliders based on which shapes you're using and what type of physics you want to imitate
- Physics Materials can be applied in order to create realistic collision and object interactions between sprites.

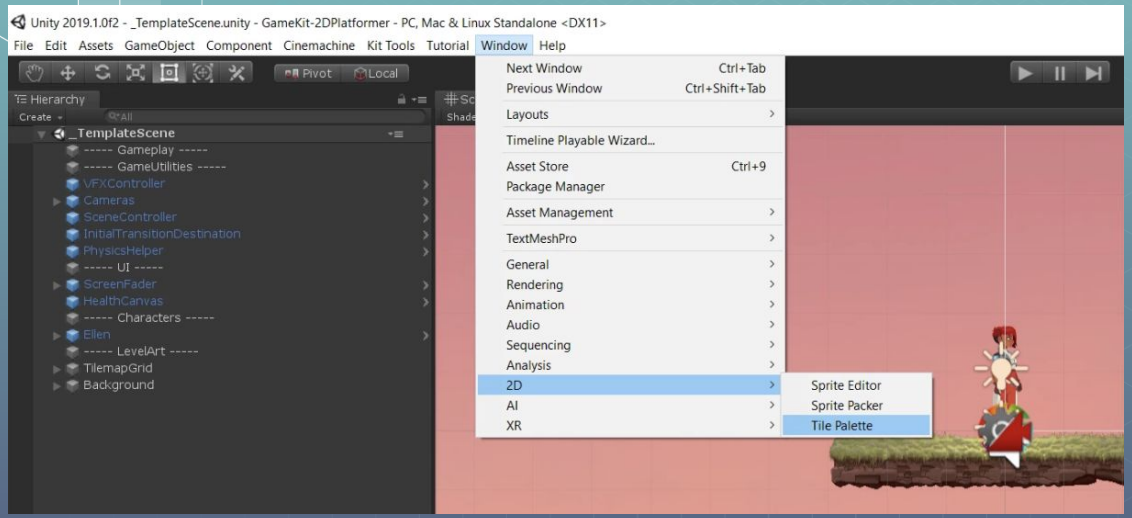
2D Gamekit Walkthrough



- First we want to create our own level, do so by clicking Kit Tools > Create New Scene..
- Name it whatever you'd like

- The walkthrough can also be found here: https://tinyurl.com/uf7ek7h

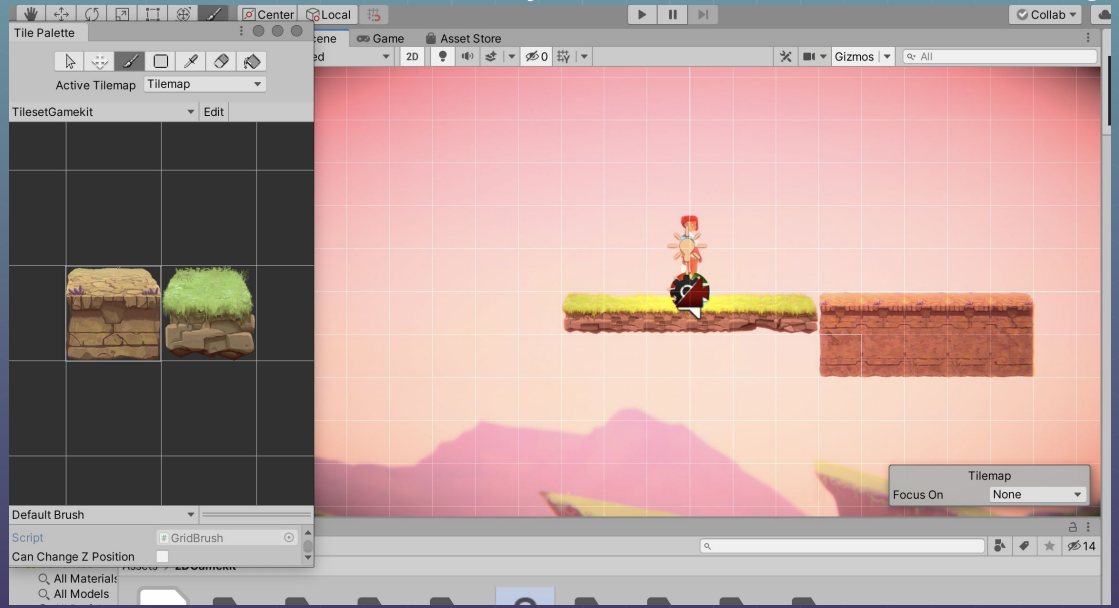
Painting Tiles



To see the tiles you can use to prototype your own game world within the 2D game kit, go to Window, 2D, tile palette

Painting Tiles

- In this tutorial, two sets of tiles are provided for you to paint with
- To paint, select which tile asset you like, then click and drag along the scene view.

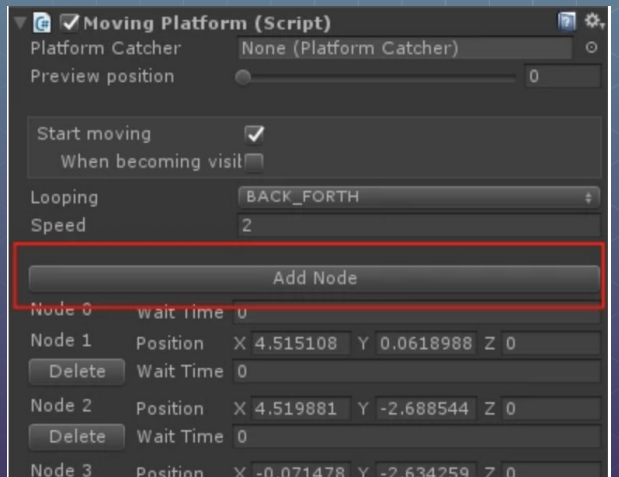


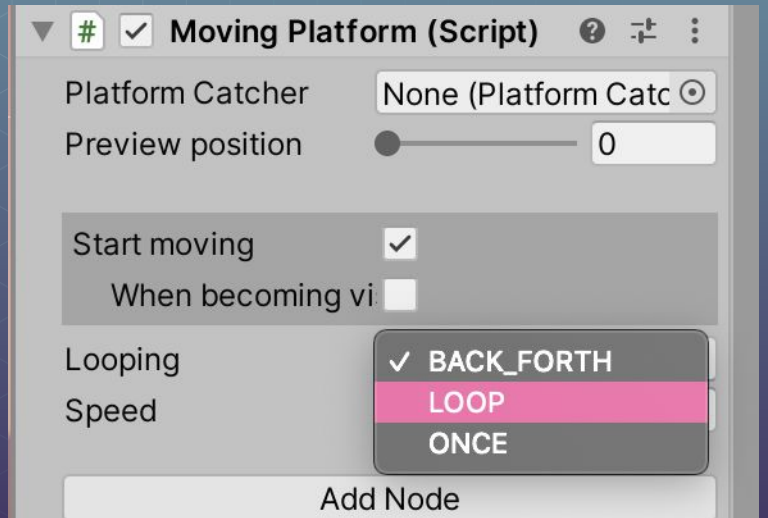
Adding Moving Platforms to the Game

In the project window, go to prefabs > Interactables
Find an interactable object you'd like to add, then click and drag it into the scene view

Adding Moving Platforms

- To make the game more challenging, give the moving platform attributes like loops
- To add a loop to your platform look at inspector view and at the Moving Platform Component to add a Node
- Add one node
- The apply a Loop in the Same area Of the inspector





Nodes

- Nodes allow you to add directional changes to your moving platform, can you guess how this one will move?

Adding Events with a Door

- Click and drag the **Door Prefab** from the project view Prefabs > Interactables into the scene view wherever you'd like.
- Try to place it somewhere that it blocks the main character's path
- Then get a **pressure pad** from the prefabs (same location) and drag it into scene view on the ground next to the door

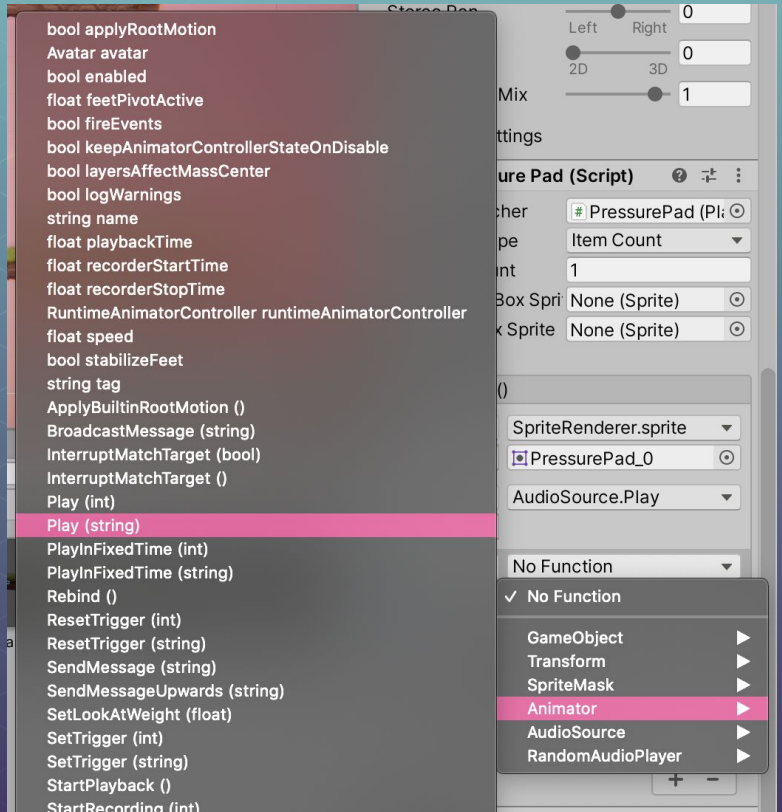
Adding Events: Door

- Some events are already defined; for example, when stepped on, the PressurePad plays a sound and lights up. Now let's connect the PressurePad to the Door.
- First Select the pressurepad prefab from the hierarchy view, then in the inspector view for that gameobject find the pressure pad component (it is a script)
- Then select the + button near the bottom of the list for On Pressed() section
- Then drag the door game object from the hierarchy Onto the None(Object) field in the event

Adding Events: Door

- In the **No Function** dropdown, find **Animator > Play(string)**
- In the text box that appeared under the dropdown, enter the text **DoorOpening**

- **Now Test!!**

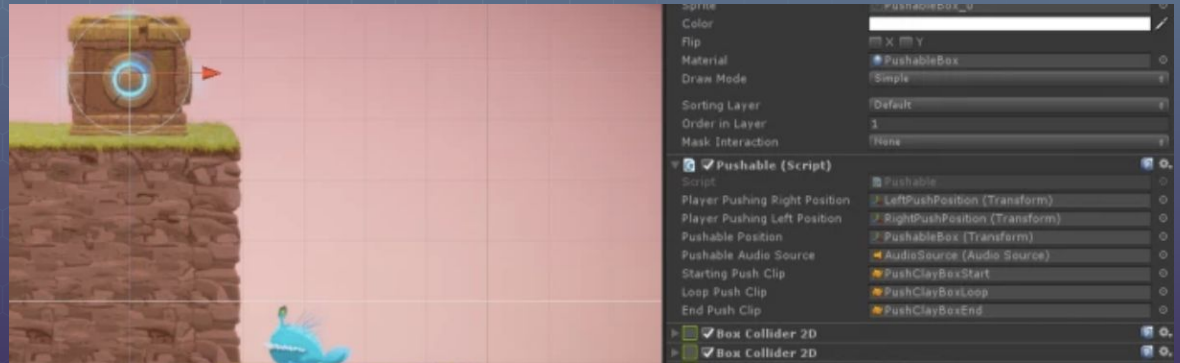


Enemy Prefabs

- For this tutorial there are 2 enemy prefabs: **Chomper** and **Spitter**
- They are located under **Prefabs > Enemies**
- They are both controlled by the **Enemy Behavior** component in the inspector view when you click on one
- All you need to do to use these is to drag and drop them into the scene
- You can use inspector view to change them around

Damage to Enemies

- Go to **Prefabs > Enemies** and drag a Spitter into the Scene View, place it on a lower level so that we can add a box object above to be dropped on top of it.
- Select the Spitter enemy and then in inspector look at the **Enemy Behaviour Script**
- Reduce the **View Distance** number so that **Spitter** does not shoot you immediately while you test this gameplay
- In the **Project Window** go to **Prefabs > Interactables**, and click and drag the **PushableBox** Prefab into the **Scene View**

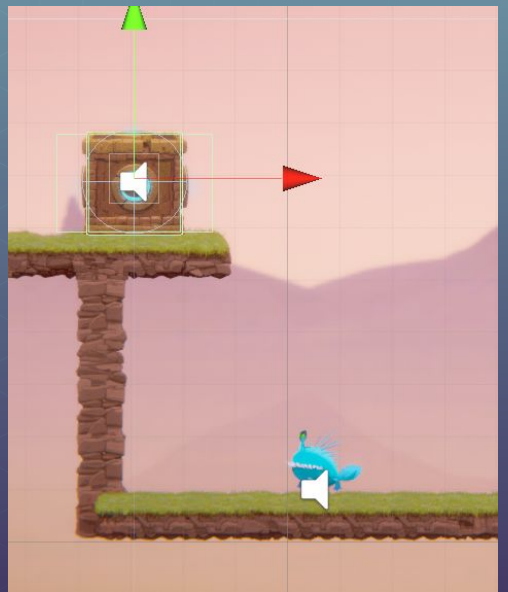


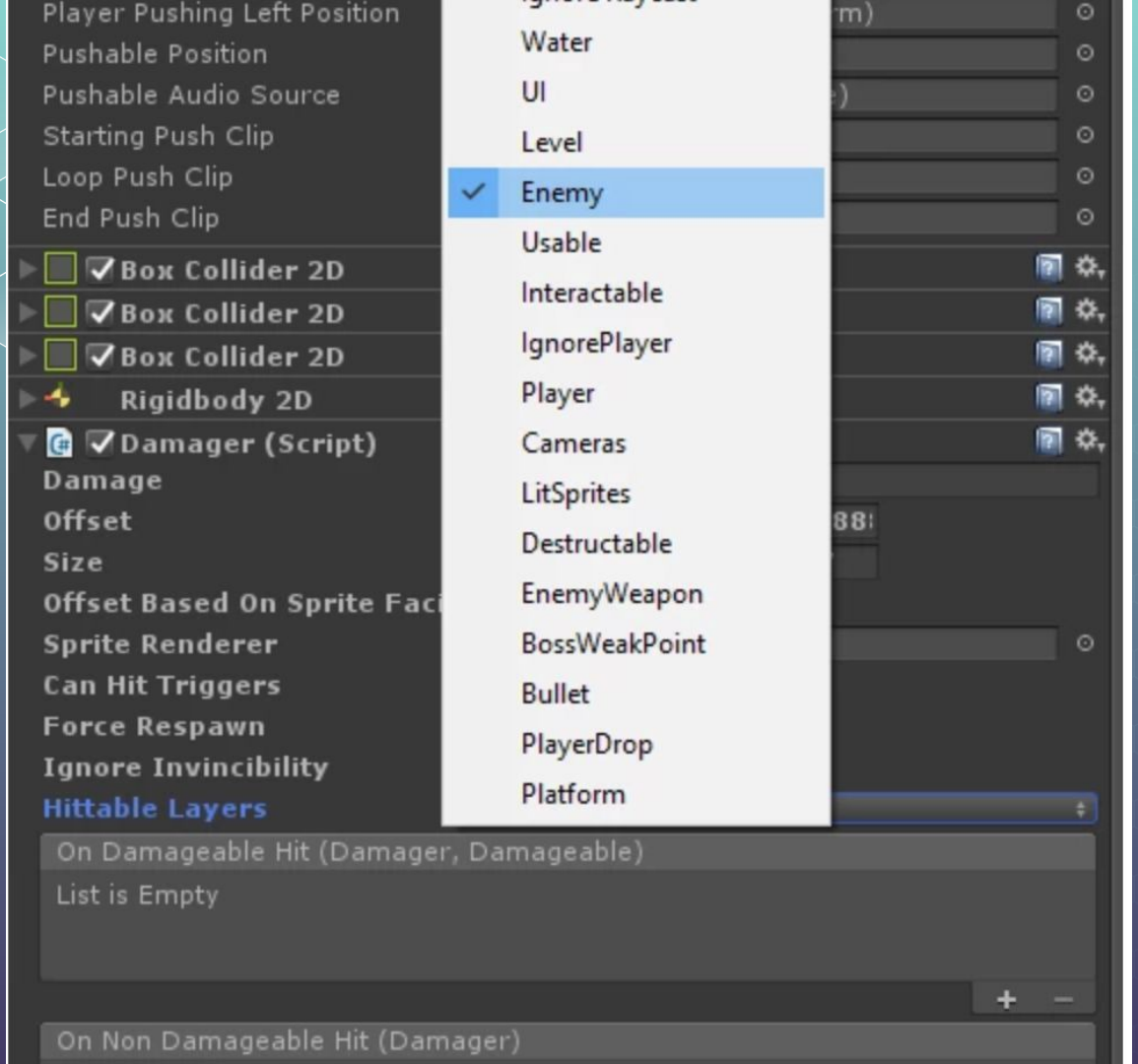
Damage to Enemies

- Select the **PushableBox** in the **Hierarchy** window, and, in the **Inspector** click **Add Component**
- In the **Search Box**, type **Damage**
- Click on **Damager** to add it to the **PushableBox**
- **Damager** Component tells any GameObject that has a **Damageable Component** on it (like a Spitter or a Chomper) to give it damage.
- In the Inspector locate the **Damager Component**
- Adjust the **Offset** and **Size** to position and size the collision box. The easiest way to do this is to left-click on the words and drag left and right to scrub through values.

Damage to Enemies

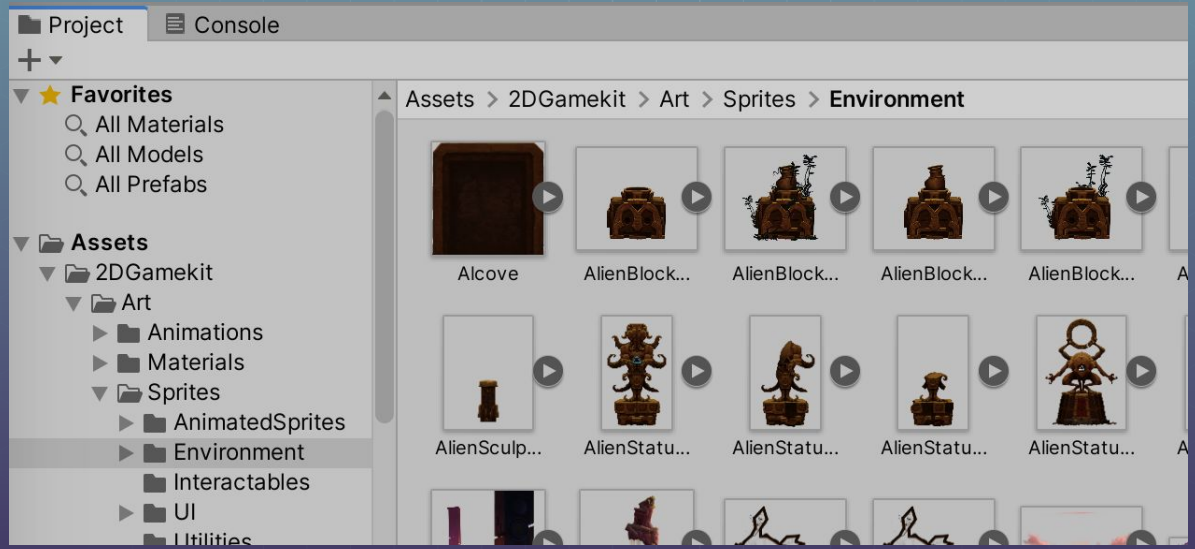
- Lastly, we need to make sure the damage is given to the right GameObjects. We separate objects into ***Layers*** in the Editor so that they can easily be found and seperated:
  - Select the **PushableBox**
  - In the Inspector, find the **Damager** Component
  - On the **HittableLayers** dropdown select the **Enemy Layer**




Player Pushing Left Position
Pushable Position
Pushable Audio Source
Starting Push Clip
Loop Push Clip
End Push Clip
Box Collider 2D
Box Collider 2D
Box Collider 2D
Rigidbody 2D
Damager (Script)
Damage
Offset
Size
Offset Based On Sprite Faci
Sprite Renderer
Can Hit Triggers
Force Respawn
Ignore Invincibility
Hittable Layers
Water
UI
Level
Enemy
Usable
Interactable
IgnorePlayer
Player
Cameras
LitSprites
Destructable
EnemyWeapon
BossWeakPoint
Bullet
PlayerDrop
Platform
On Damageable Hit (Damager, Damageable)
List is Empty
On Non Damageable Hit (Damager)

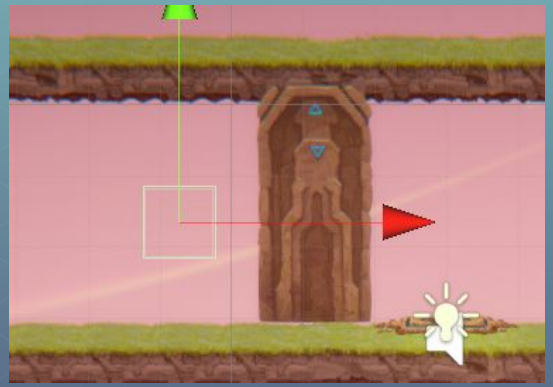
Decorating your Game Level

- This gamekit also includes many decorative sprites
- You can find all of these extra sprites to drag and drop into your game in **Art > Sprites > Environment**
- Don't forget to check the subfolders for more decorations.



Teleporting the Player

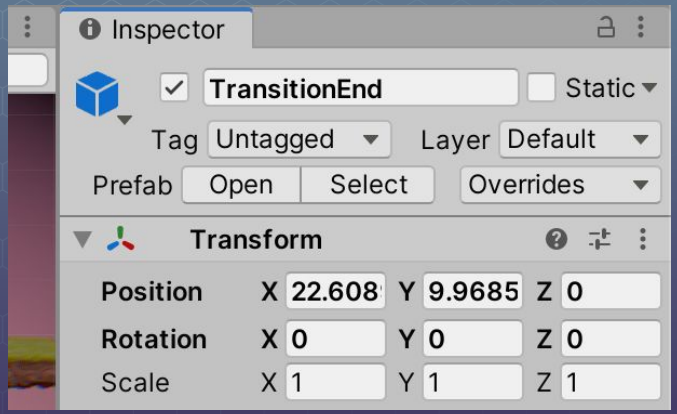
- To teleport the player within a Scene, we need to set up a **transition**. For this, we're going to make two prefabs:
  - TransitionStart
  - TransitionEnd



- First we need to setup the starting point of the transition:
  - In the Project window, go to **Prefabs > SceneControl**
  - Find the **TransitionStart** Prefab
  - Drag **TransitionStart** to the **Scene View**. Place it at a position where the player will touch the Collider (the green box) when walking

Teleporting the Player

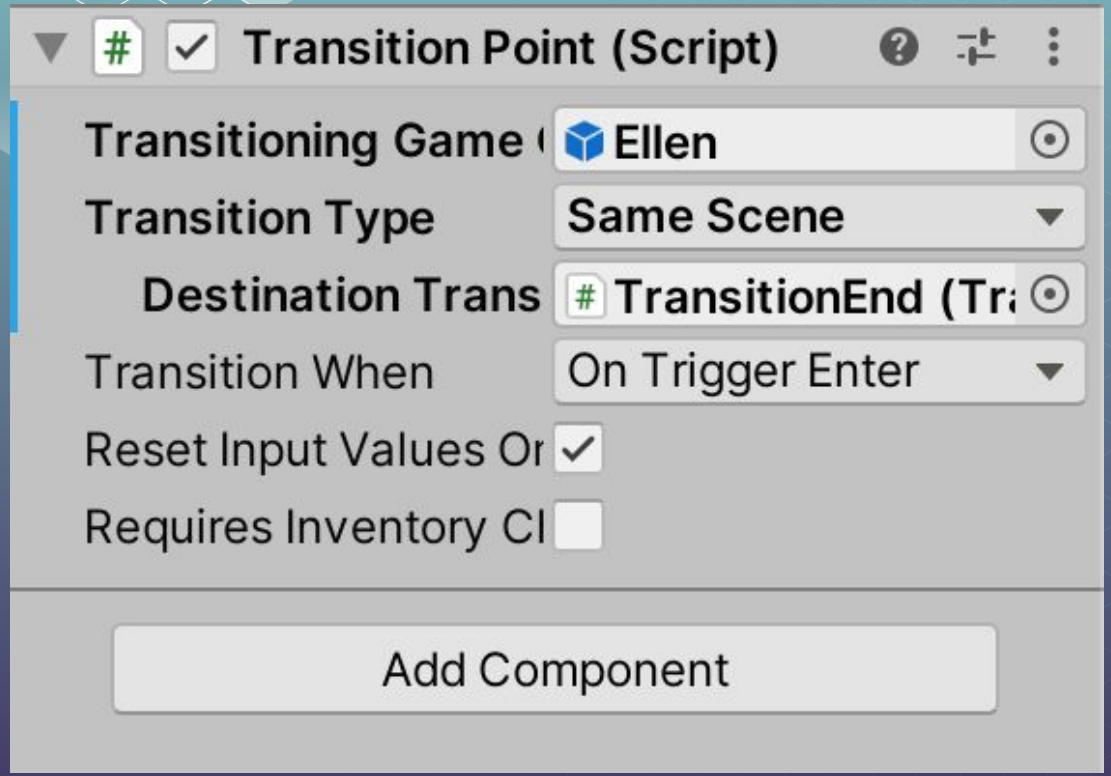
- To set up the destination;
  a. Drag another **TransitionStart** Prefab from the **SceneControl** folder into the **Scene View**
  b. In the Inspector, rename this to **TransitionEnd**



Teleporting the Player

- Now let's link the two together:
  a. In the **Hierarchy**, select the **TransitionStart** GameObject
  b. In the **Inspector**, find the **TransitionPoint** Component
  c. Drag the **Ellen** GameObject from the Hierarchy into the **Transition Point** Component's **Transitioning Game Object** slot
  d. Set **Transition Type** to **Same Scene**
- Now to set destination
  a. Drag the **TransitionEnd** GameObject into the **TransitionPoint** Component's **DestinationTransform** slot
  b. Set **Transition When** to **On Trigger Enter**
-

Teleporting the Player

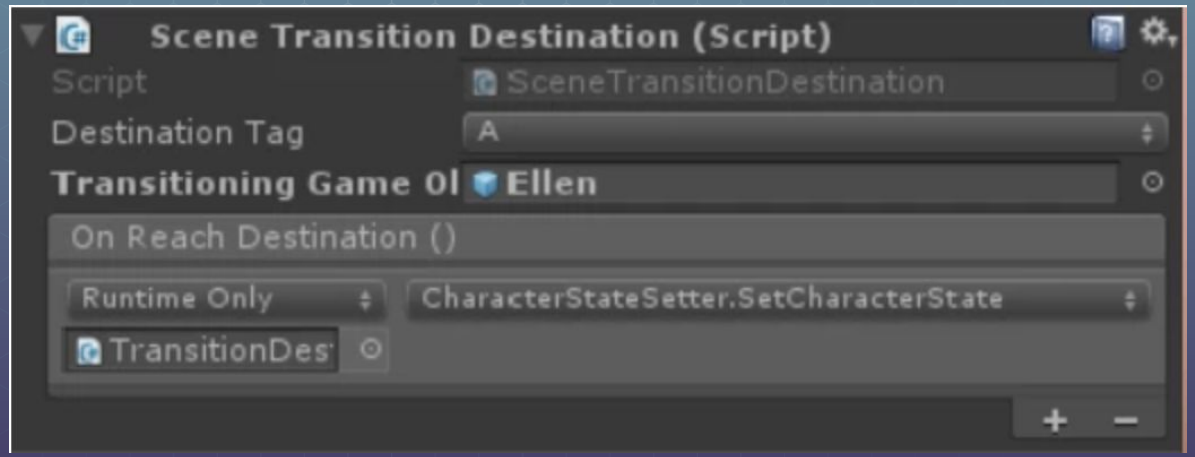


Teleporting to another Level!

- If you will be using more than one level for your game, (as I think most of you might want) You will want to be able to transition between two scenes!
- To do this you need two prefabs:
  - **TransitionStart** is the same prefab we used in the previous section; it “sends” the player to the destination. It contains a **Transition Point** Component, which defines all the properties for where the teleportation starts, and where the teleportation should take the player.
  - **TransitionDestination** is a prefab which “receives” the player. It contains a **Transition Destination** Component. Place this prefab in a different Scene, where you want the transition to finish.

Teleporting to another Level!

- To add a transition to a Scene, open that Scene, navigate to the Project window, and go to **Prefabs > SceneControl > TransitionDestination.** Place it in your Scene, in the location you want the teleporter to lead to.
- The **TransitionDestinaton** prefab contains a **Scene Transition Destination** Component:



Teleporting to another Level!

- First, set the **Destination Tag** to a letter. It doesn't really matter which letter, as long as it is the only **Scene Transition Destination Component** in this Scene with that letter.
- Next, tell it which GameObject it should expect to receive. Drag the player GameObject (Ellen) from the Hierarchy window to the **Transitioning Game Object** slot.
- Finally, make sure your destination Scene is in your Editor's Build Settings. To do this, go to **File > Build Settings** and click **Add Open Scenes**.

Teleporting to another Level!

- To set up a Starting transition point:
  - First add a new Transition Start Scene Control Prefab to where you want to transition from
  - Set **Transition Type** to **Different Level**
  - Set **New Scene Name** to the Scene you want to send it to.
  - Set **Transition Destination** Tag to the **Destination Tag** letter you set in the **Transition Destination Component**.