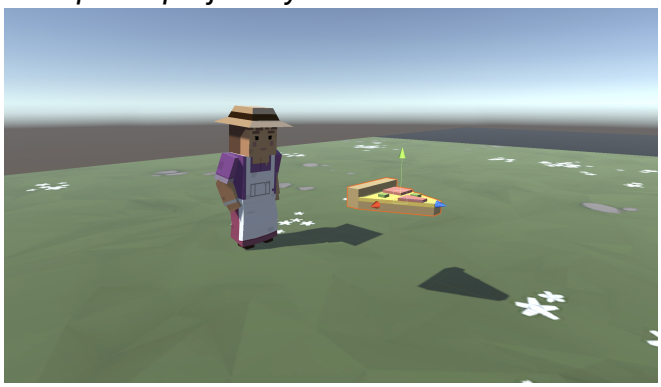
unity

2.2 Food Flight

Steps:

Example of project by end of lesson



Length: 70 minutes

Overview: In this lesson, you will allow the player to launch the projectile through the scene. First you will write a new script to send the projectile forwards. Next you will store the projectile along with all of its scripts and properties using an important new concept in Unity called Prefabs. The player will be able to launch the projectile prefab with a tap of the spacebar. Finally, you will add boundaries to the scene, removing any objects that leave the screen.

Project Outcome: The player will be able to press the Spacebar and launch a projectile prefab into the scene, which destroys itself when it leaves the game's boundaries. The animals will also be removed from the scene when they leave the game boundaries.

Learning Objectives: By the end of this lesson, you will be able to:

- Transform a game object into a prefab that can be used as a template
- Instantiate Prefabs to spawn them into the scene
- Override Prefabs to update and save their characteristics
- Get user input with GetKey and KeyCode to test for specific keyboard presses
- Apply components to multiple objects at once to work as efficiently as possible

Step 1: Make the projectile fly forwards

The first thing we must do is give the projectile some forward movement so it can zip across the scene when it's launched by the player.

1. Create a new "MoveForward" script, **attach** it to the food object, then open it
2. Declare a new ***public float speed*** variable***;***
3. In ***Update()***, add ***transform.Translate(Vector3.forward * Time.deltaTime * speed);***, then **save**
4. In the **Inspector**, set the projectile's **speed** variable, then test

- **Don't worry:** You should all be super familiar with this method now… getting easier, right?

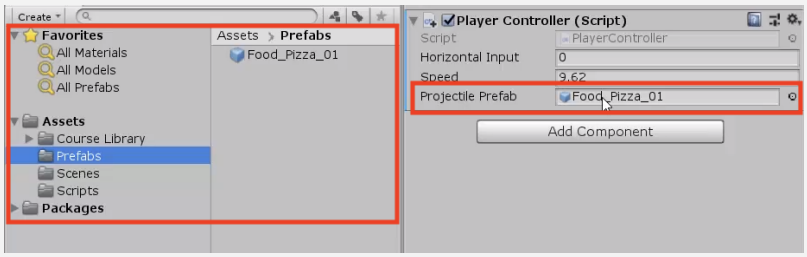
```
public float speed = 40.0f;

void Update() {
  transform.Translate(Vector3.forward * Time.deltaTime * speed);
}
```

Step 2: Make the projectile into a prefab

Now that our projectile has the behavior we want, we need to make it into a prefab it so it can be reused anywhere and anytime, with all its behaviors included.

1. Create a new "Prefabs" folder, drag your food into it, and choose **Original Prefab**
2. In PlayerController.cs, declare a new ***public GameObject projectilePrefab;*** variable
3. **Select** the Player in the hierarchy, then **drag** the object from your Prefabs folder onto the new **Projectile Prefab box** in the inspector
4. Try **dragging** the projectile into the scene at runtime to make sure they fly

- **New Concept:** Prefabs
- **New Concept:** Original vs Variant Prefabs
- **Tip:** Notice that this your projectile already has a move script if you drag it in



Step 3: Test for spacebar press

Now that we have a projectile prefab assigned to PlayerController.cs, the player needs a way to launch it with the space bar.

1. In PlayerController.cs, in ***Update()***, add an **if-statement** checking for a spacebar press: ***if (Input.GetKeyDown(KeyCode.Space)) {***
2. Inside the if-statement, add a comment saying that you should ***// Launch a projectile from the player***

- **Tip:** Google a solution. Something like "How to detect key press in Unity"
- **New Functions:** Input.GetKeyDown, GetKeyUp, GetKey
- **New Function:** KeyCode

```
void Update()
{
  if (Input.GetKeyDown(KeyCode.Space))
  {
    // Launch a projectile from the player
  }
}
```

Step 4: Launch projectile on spacebar press

We've created the code that tests if the player presses spacebar, but now we actually need spawn a projectile when that happens

1. Inside the if-statement, use the **Instantiate** method to spawn a projectile at the player's location with the prefab's rotation

- **New Concept:** Instantiation

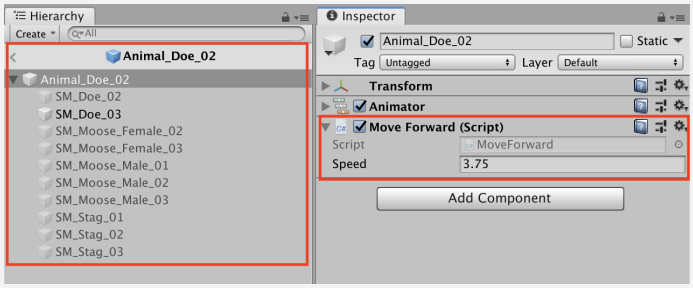
```
if (Input.GetKeyDown(KeyCode.Space))
{
  // Launch a projectile from the player
  Instantiate(projectilePrefab, transform.position, projectilePrefab.transform.rotation);
}
```

Step 5: Make animals into prefabs

The projectile is now a prefab, but what about the animals? They need to be prefabs too, so they can be instantiated during the game.

1. **Rotate** all animals on the Y axis by **180 degrees** to face down
2. **Select** all three animals in the hierarchy and *Add Component* > ***Move Forward***
3. Edit their **speed values** and **test** to see how it looks
4. Drag all three animals into the **Prefabs folder**, choosing "Original Prefab"
5. **Test** by dragging prefabs into scene view during gameplay

- **Tip:** You can change all animals at once by selecting all them in the hierarchy while holding Cmd/Ctrl
- **Tip:** Adding a Component from inspector is same as dragging it on
- **Warning:** Remember, anything you change while the game is playing will be reverted when you stop it



Step 6: Destroy projectiles offscreen

Whenever we spawn a projectile, it drifts past the play area into eternity. In order to improve game performance, we need to destroy them when they go out of bounds.

1. Create "DestroyOutOfBounds" script and apply it to the **projectile**
2. Add a new ***private float topBound*** variable and initialize it ***= 30;***
3. Write code to destroy if out of top bounds ***if (transform.position.z > topBound) { Destroy(gameObject); }***
4. In the Inspector **Overrides** drop-down, click **Apply all** to apply it to prefab

- **Warning:** Too many objects in the hierarchy will slow the game
- **Tip:** Google "How to destroy gameobject in Unity"
- **New Function:** Destroy
- **New Technique:** Override prefab

```
private float topBound = 30;

void Update() {
  if (transform.position.z > topBound) {
    Destroy(gameObject);  }}
```

Step 7: Destroy animals offscreen

If we destroy projectiles that go out of bounds, we should probably do the same for animals. We don't want critters getting lost in the endless abyss of Unity Editor...

1. Create a new **private float lowerBound** variable and initialize it = -10;
2. Create **else-if statement** to check if objects are beneath ***lowerBound***:
   else if (transform.position.z > topBound)
3. **Apply** the script to all of the animals, then **Override** the prefabs

- **New Function:** Else-if statement
- **Warning:** Don't make topBound too tight or you'll destroy the animals before they before they can spawn

```
private float topBound = 30;
private float lowerBound = -10;

void Update() {
  if (transform.position.z > topBound)
  {
    Destroy(gameObject);
  } else if (transform.position.z < lowerBound) {
    Destroy(gameObject);
  }
}
```

Lesson Recap

New Functionality	• The player can press the Spacebar to launch a projectile prefab, • Projectile and Animals are removed from the scene if they leave the screen
New Concepts and Skills	• Create Prefabs • Override Prefabs • Test for Key presses • Instantiate objects • Destroy objects • Else-if statements
Next Lesson	• Instead of dropping all these animal prefabs onto the scene, we'll create a herd of animals roaming the plain!