2.1 Player Positioning

Steps:

Step 1: Create a new Project for Prototype 2

Step 2: Add the Player, Animals, and Food

Step 3: Get the user's horizontal input

Step 4: Move the player left-to-right

Step 5: Keep the player inbounds

Step 6: Clean up your code and variables

Example of project by end of lesson

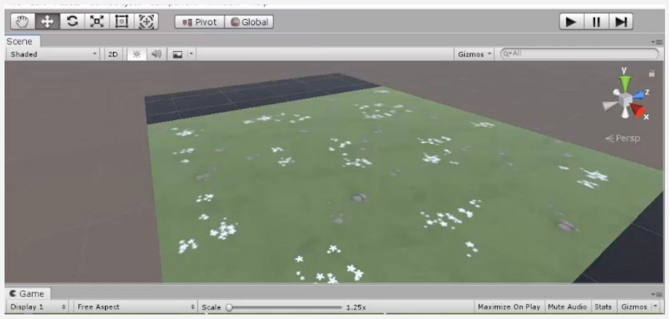
Length:	60 minutes
Overview:	You will begin this unit by creating a new project for your second Prototype and getting basic player movement working. You will first choose which character you would like, which types of animals you would like to interact with, and which food you would like to feed those animals. You will give the player basic side-to-side movement just like you did in Prototype 1, but then you will use if-then statements to keep the Player in bounds.
Project Outcome:	The player will be able to move left and right on the screen based on the user's left and right key presses, but will not be able to leave the play area on either side.
Learning Objectives:	By the end of this lesson, you will be able to: - Adjust the scale of an object proportionally in order to get it to the size you want - More comfortably use the GetInput function in order to use user input to control an object - Create an if-then statement in order to implement basic logic in your project, including the use of greater than (>) and less than (<) operators - Use comments and automatic formatting in order to make their code more clean and readable to other programmers

Step 1: Create a new Project for Prototype 2

The first thing we need to do is create a new project and import the Prototype 2 starter files.

1. Open **Unity Hub** and create a **New** project named "Prototype 2" in your course directory
2. Click on the **link** to access the Prototype 2 starter files, then **import** them into Unity
3. Open the **Prototype 2 scene** and **delete** the SampleScene without saving
4. In the top-right of the Unity Editor, change your Layout from **Default** to your custom layout

- **Don't worry:** Unit 2 has far more assets than Unit 1, so the package might take a while to import.

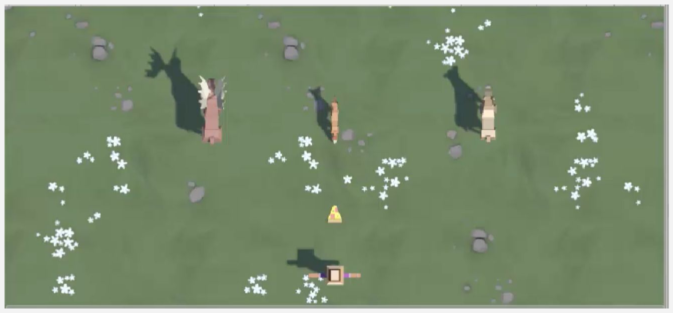


Step 2: Add the Player, Animals, and Food

Let's get all of our objects positioned in the scene, including the player, animals, and food.

1. If you want, drag a different **material** from *Course Library > Materials* onto the Ground object
2. Drag 1 **Human**, 3 **Animals**, and 1 **Food** object into the Hierarchy
3. Rename the human "Player", then **reposition** the animals and food so you can see them
4. Adjust the XYZ **scale** of the food so you can easily see it from above

- **New Technique:** Adjusting Scale
- **Warning:** Don't choose people for anything but the player, they don't have walking animations
- **Tip:** Remember, dragging objects into the hierarchy puts them at the origin



Step 3: Get the user’s horizontal input

If we want to move the Player left-to-right, we need a variable tracking the user’s input.

1. In your **Assets** folder, create a “Scripts” folder, and a “PlayerController” script inside
2. **Attach** the script to the Player and open it
3. At the top of PlayerController.cs, declare a new ***public float horizontalInput***
4. In ***Update()***, set ***horizontalInput = Input.GetAxis(“Horizontal”)***, then test to make sure it works in the inspector

- **Warning:** Make sure to create your Scripts folder inside of the assets folder
- **Don’t worry:** We’re going to get VERY familiar with this process
- **Warning:** If you misspell the script name, just delete it and try again.

```
public float horizontalInput;

void Update()
{
  horizontalInput = Input.GetAxis("Horizontal");
}
```

Step 4: Move the player left-to-right

We have to actually use the horizontal input to translate the Player left and right.

1. Declare a new ***public float speed = 10.0f;***
2. In ***Update()***, Translate the player side-to-side based on ***horizontalInput*** and ***speed***

- **Tip:** You can look at your old scripts for code reference

```
public float horizontalInput;
public float speed = 10.0f;

void Update()
{
  horizontalInput = Input.GetAxis("Horizontal");
  transform.Translate(Vector3.right * horizontalInput * Time.deltaTime * speed);
}
```

Step 5: Keep the player inbounds

We have to prevent the player from going off the side of the screen with an if-then statement.

1. In ***Update()***, write an **if-statement** checking if the player's left X position is **less than** a certain value
2. In the if-statement, set the player's position to its current position, but with a **fixed X location**

- **Tip:** Move the player in scene view to determine the x positions of the left and right bounds
- **New Concept:** If-then statements
- **New Concept:** Greater than > and Less Than < operators

```
void Update() {
  if (transform.position.x < -10) {
    transform.position = new Vector3(-10, transform.position.y, transform.position.z);
  }
}
```

Step 6: Clean up your code and variables

We need to make this work on the right side, too, then clean up our code.

1. Repeat this process for the **right side** of the screen
2. Declare new ***xRange*** variable, then replace the hardcoded values with them
3. Add **comments** to your code

- **Warning:** Whenever you see hardcoded values in the body of your code, try to replace it with a variable
- **Warning:** Watch your greater than / less than signs!

```
public float xRange = 10;

void Update()
{
  // Keep the player in bounds
  if (transform.position.x < ~~-10~~ -xRange)
  {
    transform.position = new Vector3(~~-10~~ -xRange, transform.position.y, transform.position.z);
  }
  if (transform.position.x > xRange)
  {
    transform.position = new Vector3(xRange, transform.position.y, transform.position.z);
  }
}
```

Lesson Recap

New Functionality	● The player can move left and right based on the user's left and right key presses ● The player will not be able to leave the play area on either side
New Concepts and Skills	● Adjust object scale ● If-statements ● Greater/Less than operators
Next Lesson	● We'll learn how to create and throw endless amounts of food to feed our animals!