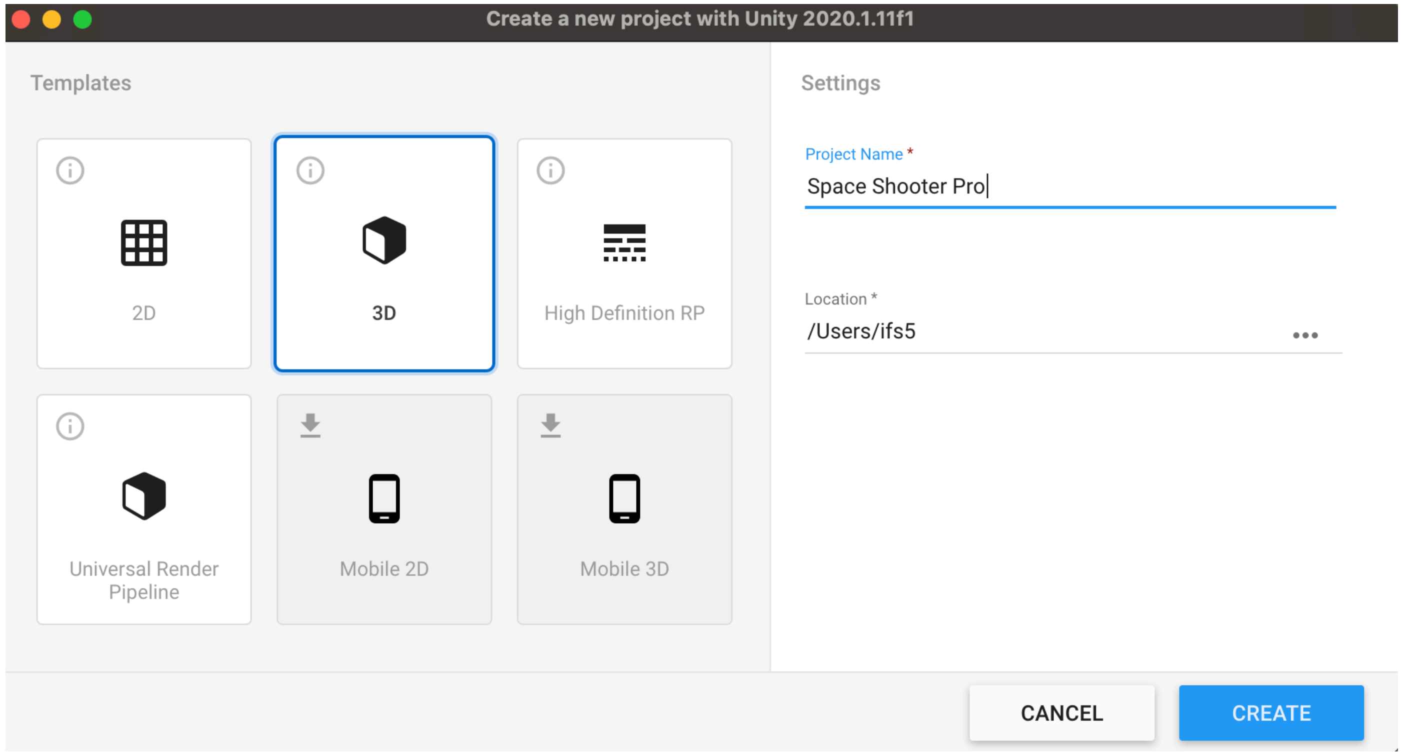
Game Programming

Lecture I

Izzet Fatih Senturk


Create a new project with Unity 2020.1.11f1

Templates

Settings

Project Name *
Space Shooter Pro

Location *
/Users/ifs5

Create a New Project

Professional Layout

- All windows are customizable. Select layout and select default. Take game view and move it under scene view. Go to layout and select tall layout. Select game view and move it under scene view. Move game view above at the same level with project (in the middle). For mobile games, move game view to the right side of the scene view. We cannot view more than one folder now. In the project window there is a little drop down menu at the right side, select one column layout: switch from view mode to list mode. Select tall and save layout: Professional Layout

Player Setup

- Delete the sample scene and create a new scene
  - Click file and new scene. Select don't save
  - To save the new scene press Ctrl+S and save it under Scenes: "Game"
- Create a 3D Cube, change the object name as "Player"
- Create a new material
  - Create a new folder under assets: "Materials"
  - In the Materials folder, create a new material: "Player_mat"
  - Make the color blue (from albedo) and assign it to the "Player"
- Make a black background
  - Go to the main camera. Change "Clear Flags" from "Skybox" to "Solid Color". Then set the background color to black
- Change the material color to white or gray
- Increase the light intensity from Directional Light.

Aspect Ratio

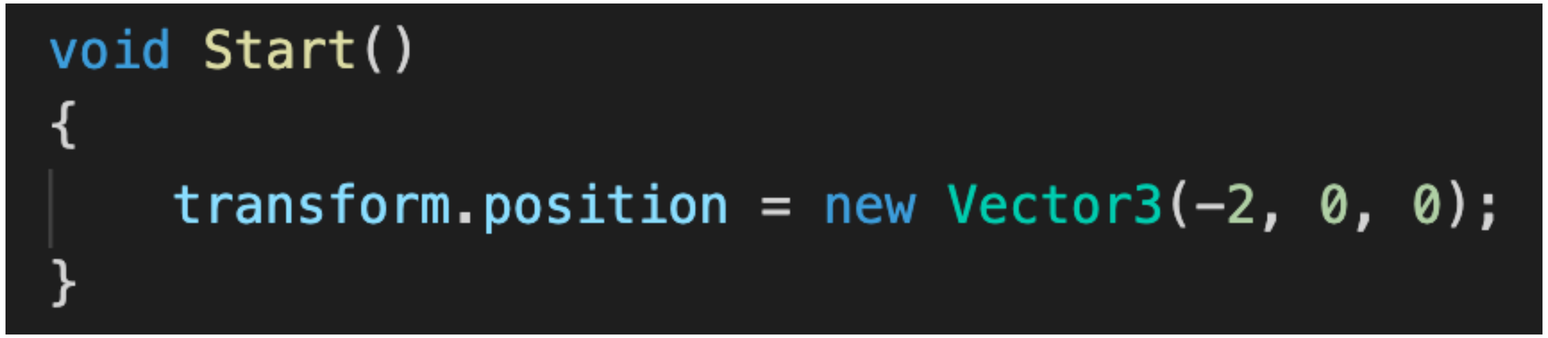
- Change the aspect ratio from “Free Aspect” to traditional HD setting: “16:9”

Starting Player Position

- Create a new C# script
  - Create a new folder under assets: “Scripts”.
  - In the new folder create a new script: “Player_sc”
  - Add the script to “Player”.
- Access current position: Position in Transform component.

Starting Player Position

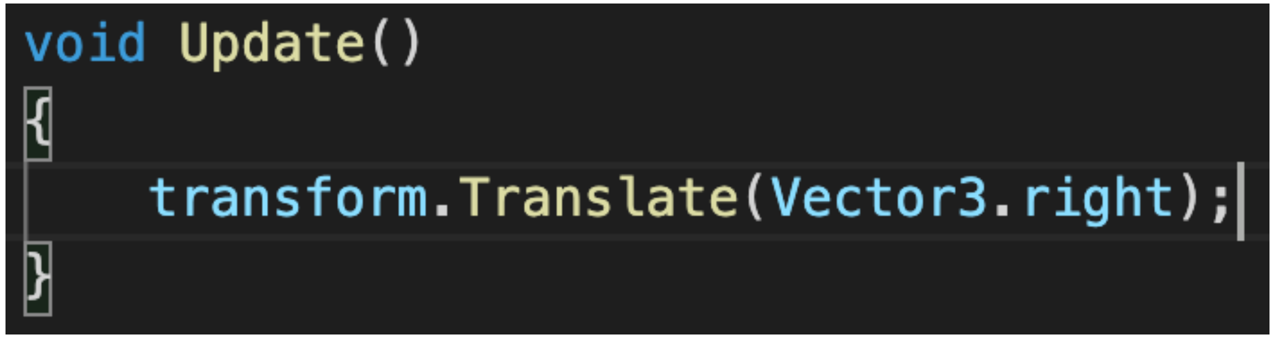
- Update the start position by accessing transform component
  - Set x to -2



```
void Start()
{
    transform.position = new Vector3(-2, 0, 0);
}
```

Moving Player

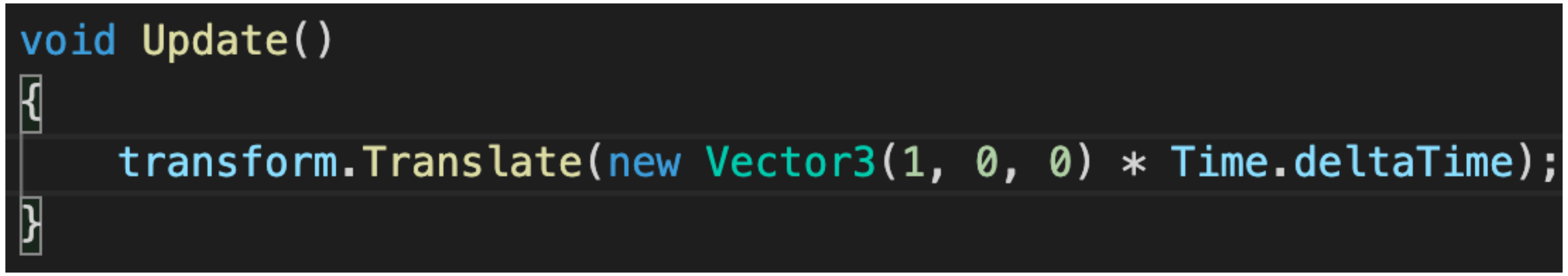
- Use translate function to move the object
  - transform.Translate(new Vector3(1, 0, 0));
  - transform.Translate(new Vector3(-1, 0, 0));



```
void Update()
{
    transform.Translate(Vector3.right);
}
```

Movement Speed

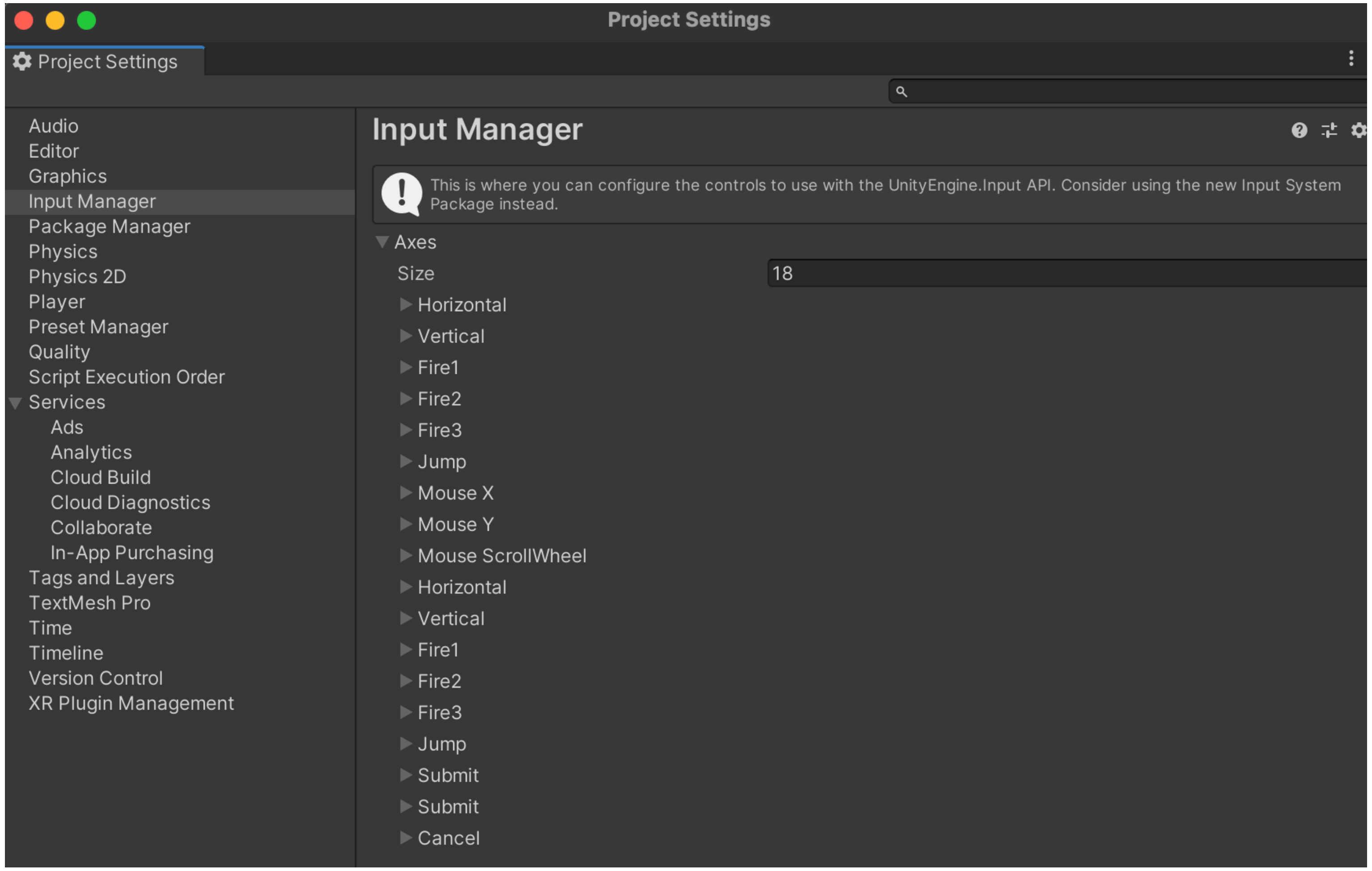
- 1 is 1 meter per frame and there are 60 frames per second.
  - We want to change it to 1 meter per second.
- Time.deltaTime indicates the time it takes from previous frame to this frame
- How can we move 5 meters per second? Create a speed variable.
  - public float speed = 3.5f;
  - Public vs Private?
  - [SerializeField]



```
void Update()
{
    transform.Translate(new Vector3(1, 0, 0) * Time.deltaTime);
}
```

Input Manager

- Go to Edit – Project Settings – Input Manager – Axes



Project Settings

Project Settings

Audio
Editor
Graphics
Input Manager
Package Manager
Physics
Physics 2D
Player
Preset Manager
Quality
Script Execution Order
Services
Ads
Analytics
Cloud Build
Cloud Diagnostics
Collaborate
In-App Purchasing
Tags and Layers
TextMesh Pro
Time
Timeline
Version Control
XR Plugin Management

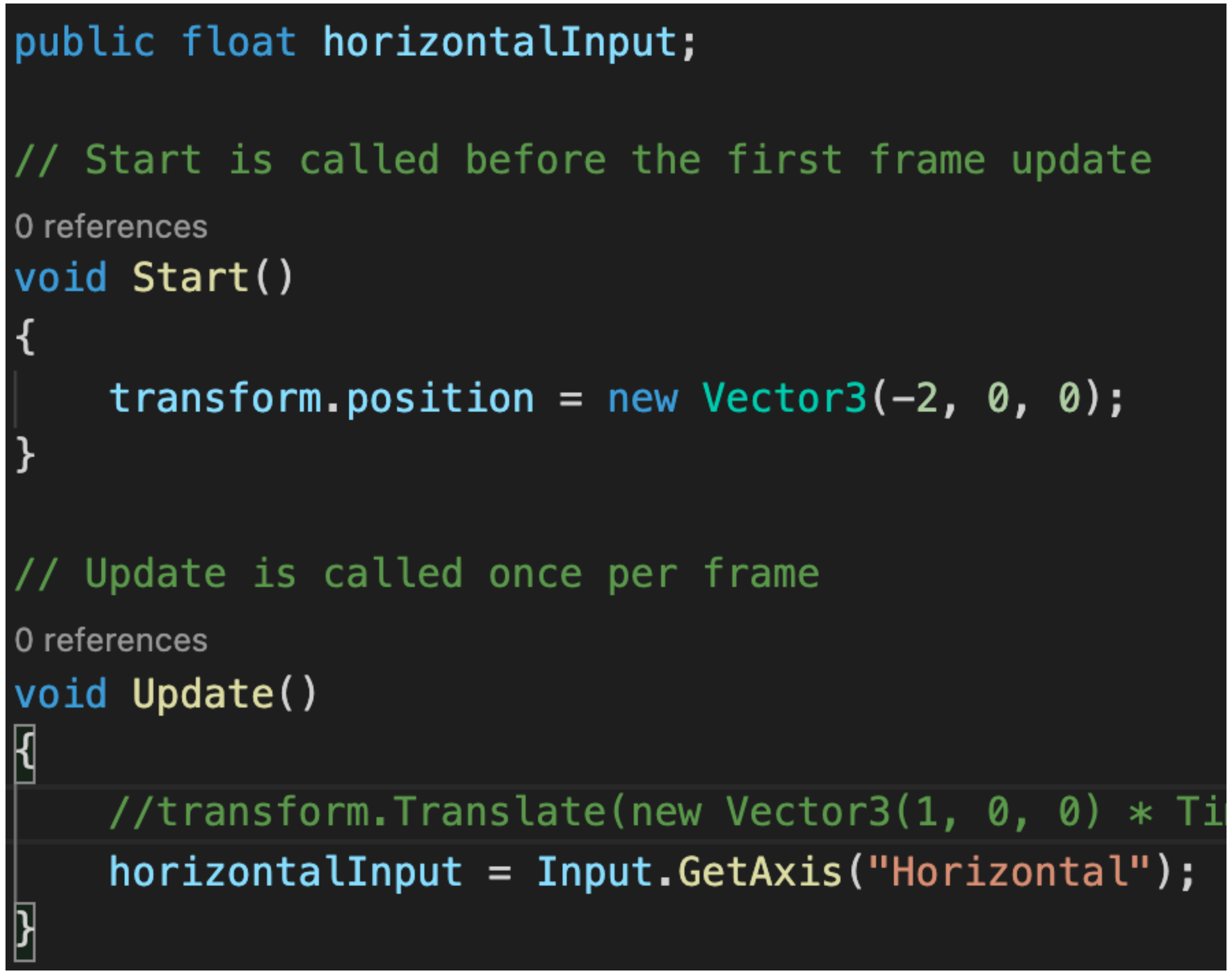
Input Manager

This is where you can configure the controls to use with the UnityEngine.Input API. Consider using the new Input System Package instead.

Axes
Size 18
Horizontal
Vertical
Fire1
Fire2
Fire3
Jump
Mouse X
Mouse Y
Mouse ScrollWheel
Horizontal
Vertical
Fire1
Fire2
Fire3
Jump
Submit
Submit
Cancel

Debug Input

- Debug "Horizontal" input



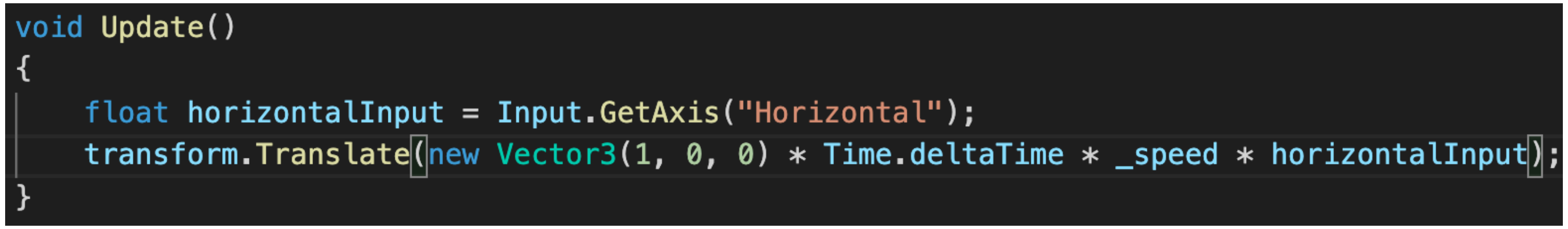
```
public float horizontalInput;

// Start is called before the first frame update
0 references
void Start()
{
    transform.position = new Vector3(-2, 0, 0);
}

// Update is called once per frame
0 references
void Update()
{
    //transform.Translate(new Vector3(1, 0, 0) * Ti
    horizontalInput = Input.GetAxis("Horizontal");
}
```

Apply User Input

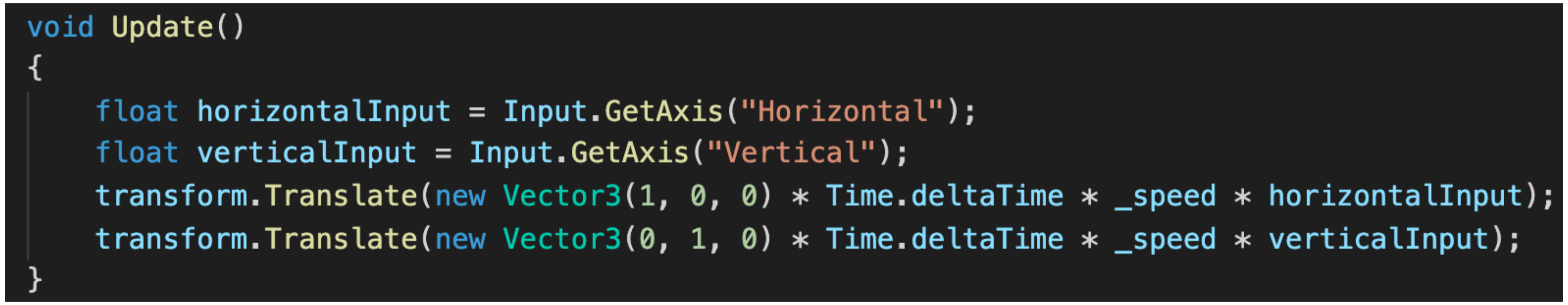
- Use "Horizontal" input in translate



```
void Update()
{
    float horizontalInput = Input.GetAxis("Horizontal");
    transform.Translate(new Vector3(1, 0, 0) * Time.deltaTime * _speed * horizontalInput);
}
```

Add Vertical Movement

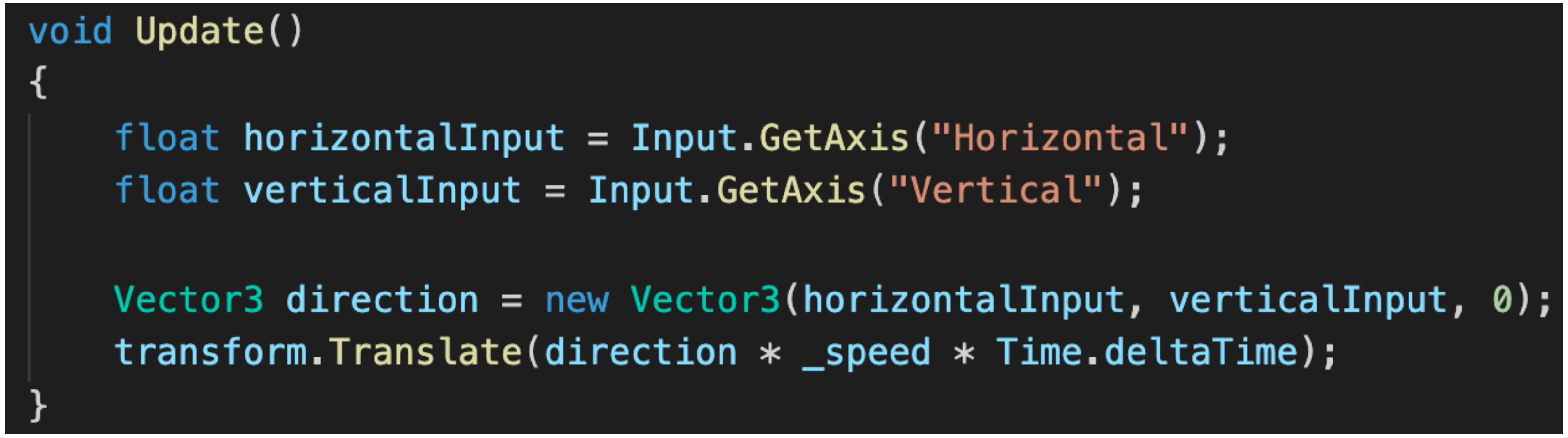
- Use both “Horizontal” and “Vertical” input in translate



```
void Update()
{
    float horizontalInput = Input.GetAxis("Horizontal");
    float verticalInput = Input.GetAxis("Vertical");
    transform.Translate(new Vector3(1, 0, 0) * Time.deltaTime * _speed * horizontalInput);
    transform.Translate(new Vector3(0, 1, 0) * Time.deltaTime * _speed * verticalInput);
}
```

Improve the Movement Code

- Define direction from “Horizontal” and “Vertical”

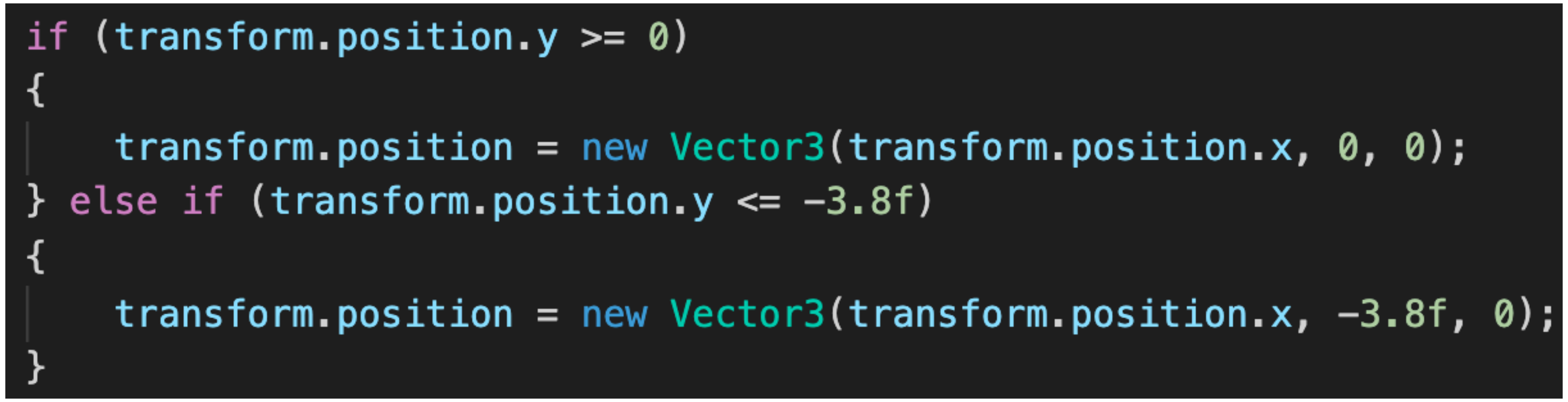


```
void Update()
{
    float horizontalInput = Input.GetAxis("Horizontal");
    float verticalInput = Input.GetAxis("Vertical");

    Vector3 direction = new Vector3(horizontalInput, verticalInput, 0);
    transform.Translate(direction * _speed * Time.deltaTime);
}
```

Player Bounds: Position Y

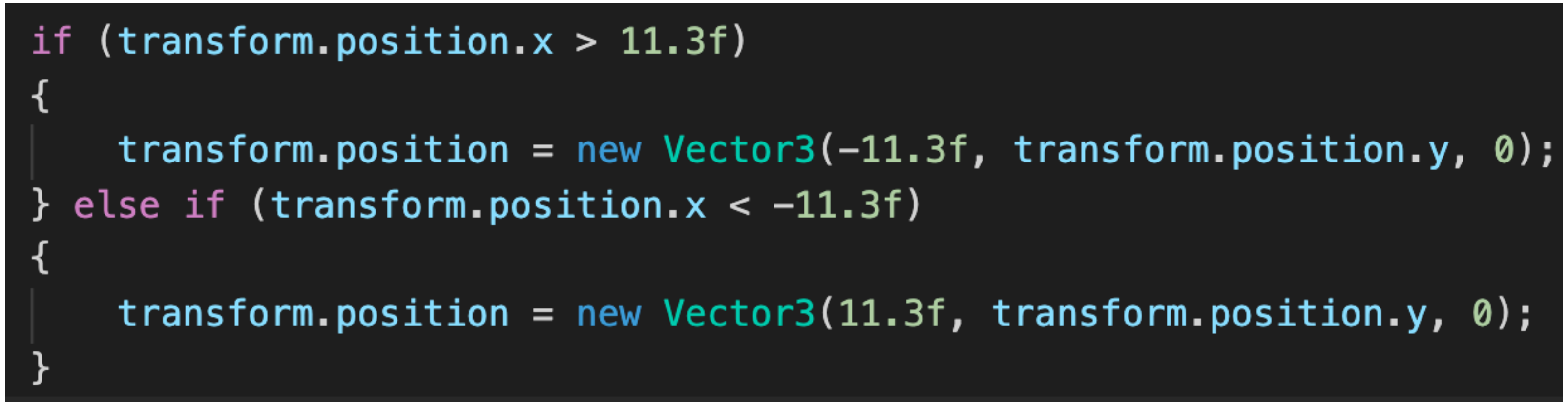
- Make sure that y position is always between -3.8 and 0



```
if (transform.position.y >= 0)
{
    transform.position = new Vector3(transform.position.x, 0, 0);
} else if (transform.position.y <= -3.8f)
{
    transform.position = new Vector3(transform.position.x, -3.8f, 0);
}
```

Player Bounds: Position X

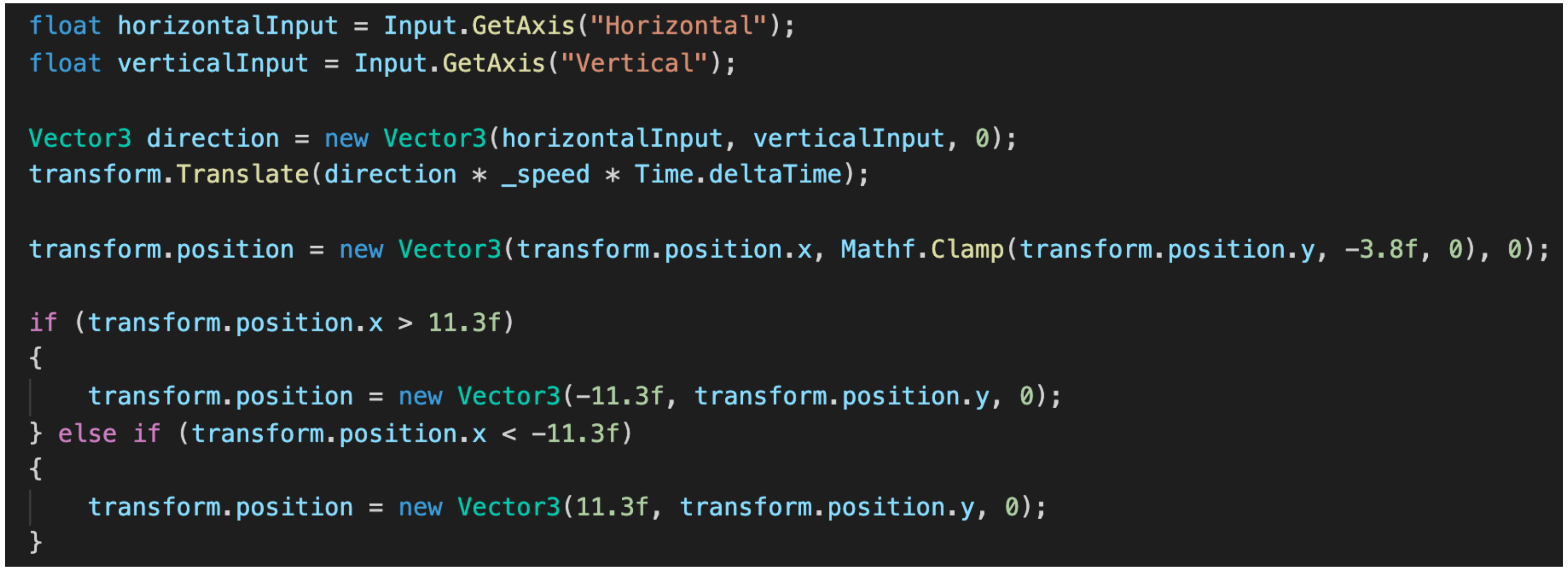
- Make sure that x position is always between -11.3 and 11.3



```
if (transform.position.x > 11.3f)
{
    transform.position = new Vector3(-11.3f, transform.position.y, 0);
} else if (transform.position.x < -11.3f)
{
    transform.position = new Vector3(11.3f, transform.position.y, 0);
}
```

Improve Bound Code

- Clamping y position



```
float horizontalInput = Input.GetAxis("Horizontal");
float verticalInput = Input.GetAxis("Vertical");

Vector3 direction = new Vector3(horizontalInput, verticalInput, 0);
transform.Translate(direction * _speed * Time.deltaTime);

transform.position = new Vector3(transform.position.x, Mathf.Clamp(transform.position.y, -3.8f, 0), 0);

if (transform.position.x > 11.3f)
{
    transform.position = new Vector3(-11.3f, transform.position.y, 0);
} else if (transform.position.x < -11.3f)
{
    transform.position = new Vector3(11.3f, transform.position.y, 0);
}
```

Code Improvement

- Add a new function to control movement: Calculate_Movement