# How to make a video game

## Lec 2- Programming

using UnityEngine;

public class PlayerMovement : MonoBehaviour{

public Rigidbody rb;

void Start(){  
 Debug.Log(“Hello world!”);

// rb.useGravity = false;

}

void Update(){ //called once per frame

}

void FixedUpdate(){ // Better if we do physics stuff inside this method

// x y z

rb.AddForce(0, 0, 2000 \* Time.deltaTime);

//amount of time since last frame

}

}

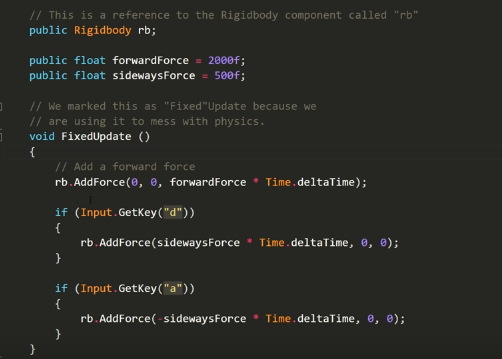
## Lec 3- Movement

To fix rotation of object when collide whit ground:

1. Player –> RigidBody –> Constraints –> freeze Rotation –> x

OR

1. New Physics Material –> Dynamic and Static friction -> 0 => Apply to Ground



## Lec 4- Camera Follow

1. Making Player object parent of camera

OR

1. Camera object script:

using UnityEngine;

public class FollowPlayer : MonoBehaviour {

public Transform player; // A variable that stores a reference to our Player

public Vector3 offset; // A variable that allows us to offset the position (x, y, z)

// Update is called once per frame

void Update () {

// Set our position to the players position and offset it

transform.position = player.position + offset;

}

}

## Lec 5- Collision Behavior

using UnityEngine;

public class PlayerCollision : MonoBehaviour {

public PlayerMovement movement; // A reference to our PlayerMovement script

// This function runs when we hit another object.

// We get information about the collision and call it "collisionInfo".

void OnCollisionEnter (Collision collisionInfo)

{

// We check if the object we collided with has a tag called "Obstacle".

if (collisionInfo.collider.tag == "Obstacle")

{

movement.enabled = false; // Disable the players movement.

}

}

}

## Lec 6- Gameplay

Create Prefab out of Obstacle => Create more Obstacle objects out of Prefab

Switch to isometric view

Create ‘Environment’ layer for Ground object and hide it so that it doesn’t get selected in the view

Edit-Snap Settings => Configure grid parameters => Press ctrl while moving object to move in grid

Duplicate Obstacles (Ctrl+D)

Change Player movement script:

if (Input.GetKey("d")) // If the player is pressing the "d" key

{

// Add a force to the right

rb.AddForce(sidewaysForce \* Time.deltaTime, 0, 0, ForceMode.VelocityChange); // The fourth parameter automatically compensates for the additional force required when momentum increases by forward force

}

if (Input.GetKey("a")) // If the player is pressing the "a" key

{

// Add a force to the left

rb.AddForce(-sidewaysForce \* Time.deltaTime, 0, 0, ForceMode.VelocityChange);

}

Add some Drag (1) (Air resistance) in the Player-RigidBody to make controls better. Will stop much faster and will attain a top speed?

Add fog from: Windows-> Lighting view-> Scene

Ctrl+P – to play

F – focus on object

Shift+Space – maximize game view

## Lec 7- Score and UI

### To improve collision detection:

Obstacle prefab-> Collision Detection -> Continuous

Same to player

Edit -> Project Settings -> Time -> fixed time step (time when unity update physics?) ->(Decrease) 0.01

Increase forces

### Score:

New text object (under UI) in hierarchy

Horizontal overflow -> Overflow

Canvas -> Scale Mode -> Scale with screen size   
+ Match height

Score.cs:

using UnityEngine;

using UnityEngine.UI;

public class Score : MonoBehaviour {

public Transform player;

public Text scoreText;

// Update is called once per frame

void Update () {

scoreText.text = player.position.z.ToString("0");

}

}

Canvas -> Pixel perfect : Can sometimes make UI elements sharper

## Lec 8- Game Over

New Object => Rename to GameManager

Add script to object => GameManager.cs

GetComponent<PlayerMovement>() => Searches for components present in the object where this class is used

FindObjectOfType<GameManager>() => Searches for scripts in the project – Not as performant as using direct reference

GameManager.cs

using UnityEngine;

using UnityEngine.SceneManagement;

public class GameManager : MonoBehaviour {

bool gameHasEnded = false;

public float restartDelay = 1f;

public GameObject completeLevelUI;

public void CompleteLevel ()

{

completeLevelUI.SetActive(true);

}

public void EndGame ()

{

if (gameHasEnded == false)

{

gameHasEnded = true;

Debug.Log("GAME OVER");

Invoke("Restart", restartDelay);

}

}

void Restart ()

{

SceneManager.LoadScene(SceneManager.GetActiveScene().name);

}

}

PlayerCollision.cs

void OnCollisionEnter (Collision collisionInfo)

{

// We check if the object we collided with has a tag called "Obstacle".

if (collisionInfo.collider.tag == "Obstacle")

{

movement.enabled = false; // Disable the players movement.

FindObjectOfType<GameManager>().EndGame();

}

}

PlayerMovement.cs

void FixedUpdate ()

{

// Add a forward force

rb.AddForce(0, 0, forwardForce \* Time.deltaTime);

if (Input.GetKey("d")) // If the player is pressing the "d" key

{

// Add a force to the right

rb.AddForce(sidewaysForce \* Time.deltaTime, 0, 0, ForceMode.VelocityChange);

}

if (Input.GetKey("a")) // If the player is pressing the "a" key

{

// Add a force to the left

rb.AddForce(-sidewaysForce \* Time.deltaTime, 0, 0, ForceMode.VelocityChange);

}

if (rb.position.y < -1f)

{

FindObjectOfType<GameManager>().EndGame();

}

}

Window -> Lighting -> Auto -> Unchecked

This automatically builds lighting in the scene on the fly. When game restarts unity doesn’t have time to automatically build.

Press Build -> To manually build and store the lighting details of the scene

If unity complains while reloading the scene => Add the scene to build settings

File -> Build Settings => Add scenes which we need to include in the game build

## Lec 9- Winning Levels

### Finish line

Create new Cube => End

Disable Mesh Renderer, Box Collider -> Is Trigger -> Checked

Change Icon so that can be displayed

EndTrigger.cs

using UnityEngine;

public class EndTrigger : MonoBehaviour {

public GameManager gameManager;

void OnTriggerEnter ()

{

gameManager.CompleteLevel();

}

}

### Winning UI

Canvas -> New Panel



GameManager.cs

using UnityEngine;

using UnityEngine.SceneManagement;

public class GameManager : MonoBehaviour {

bool gameHasEnded = false;

public float restartDelay = 1f;

public GameObject completeLevelUI;

public void CompleteLevel ()

{

completeLevelUI.SetActive(true);

}

public void EndGame ()

{

if (gameHasEnded == false)

{

gameHasEnded = true;

Debug.Log("GAME OVER");

Invoke("Restart", restartDelay);

}

}

void Restart ()

{

SceneManager.LoadScene(SceneManager.GetActiveScene().name);

}

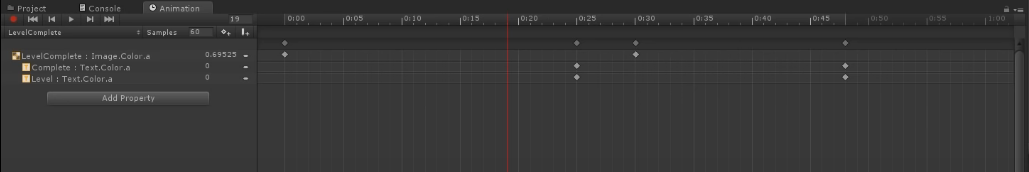
### Animation

Animator: Used for configuring when different animations happen (Flow chart for when to play different animations)

New animation => LevelComplete

LevelComplete animation -> Loop time -> unchecked

The LevelComplete animator has already added the animation => as soon as LevelComplete UI panel gets enabled the LevelComplete animation will play



### Loading Next Level

LevelComplete animation -> Add Event

For event to call a function, the function must reside on the object being animated

LevelComplete UI object -> Add LevelComplete.cs:

using UnityEngine;

using UnityEngine.SceneManagement;

public class LevelComplete : MonoBehaviour {

public void LoadNextLevel ()

{

SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex + 1);

}

}

## Lec 10 – Finishing Up