Welcome to issue 5.

The game to create this month is that game you will recognize from google. The one with the little dino running. That game looks so simply but yet holds a lot of scripts to make it all work. The dino can jump as the ground scrolls underneath him. A distance counter is running and the best high score is saved. Cactuses are spawned to be jumped over. You can see all of this created here:

https://www.youtube.com/watch?v=8ofA8qxIffY

Ready let's inspect those scripts. I will explain the parts you are new to as for familiar scripts you should know them if you read the previous issues.

Cactus Script.

3 different cactus prefab objects are created. They all have a box 2D collider and a tag called Cactus.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Cactus : MonoBehaviour
{
```

A public float is created for the moving speed. A reference to the Game Manager script is made and called GM.

```
public float speed;
private GameManager GM;
```

In the start function we make sure that the script we made a reference to is found.

```
void Start()
{
GM = FindObjectOfType<GameManager>();
}
```

When in the GM script the bool Playinggame is set to false the object is destroyed. (No longer needed) else we make the cactus move to the left with our speed variable. Time.deltaTime makes sure is uses this same speed on different systems. This al is placed in the Update function that is checked continuously. The speed can be set in the Inspector's view.

```
void Update()
{
if(GM.Playinggame == false)
{
Destroy(this.gameObject);
}
transform.Translate(Vector2.left * speed * Time.deltaTime);
}
}
```

Clouds Script.

1 cloud object was created than duplicated. They all use this script to make the move to the left and when going offscreen they return on the right side. All is the same as the cactus script accept the last (Teleport) part.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Clouds : MonoBehaviour
{
public float speed;
private GameManager GM;

void Start()
{
GM = FindObjectOfType<GameManager>();
}

void Update()
{
if (GM.Playinggame == true)
{
transform.Translate(Vector2.left * speed * Time.deltaTime);
```

When the position of the cloud object on the x-axis is smaller than -3 it get's repositioned to a new vector3 on the other side of the screen. You can take this coordinates from the inspector's view. The speed can be set in the Inspector's view.

```
if(transform.position.x < -3)
{
transform.position = new Vector3(3, transform.position.y, transform.position.z);
}
}
}
}
```

The dino character is a sprite with a box 2D Collider and a Rigidbody 2D. The tag name Given is Player. This gameObject is then stored as a prefab with the script.

Dino Script.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Dino : MonoBehaviour
{
```

A bool variable is create called enableJump so when it's true the player will be able to jump and when it's false the player cannot jump. A JumpForce float variable is created and set to 500. Next the reference to the GameManager script is made again.

```
public bool enableJump;
public float JumpForce = 500;
private GameManager GM;
```

2 audio clips are defined that the dino will use Jump and Hit.

```
public AudioClip Jump,hit;
```

The rigidbody2D of this game Object get's defined and is called rb.

```
Rigidbody2D rb;
```

In the start function the Rigidbody2D we defined is found as well as the script we referenced to.

```
void Start()
{
GM = FindObjectOfType<GameManager>();
rb = GetComponent<Rigidbody2D>();
}
```

When the dino collides with a name tag Ground (Make sure to name tag the scrolling ground sprites this way) it will set the enableJump bool variable to true.

```
public void OnCollisionEnter2D(Collision2D collision)
{
if (collision.gameObject.tag == ("Ground"))
{
enableJump = true;
}
```

When the dino collides with a name tag Cactus it will play the audio clip we defined earlier at a point near the camera. It will start the function GameOver that is in the referenced other script. (GM). The gameObject itself gets destroyed.

```
if (collision.gameObject.tag == ("Cactus"))
{
AudioSource.PlayClipAtPoint(hit, new Vector3(0, 0, -10));
GM.GameOver();
Destroy(this.gameObject);
}
}
```

In the update function when the space bar key is pressed and if the enableJump bool variable is set to true (And onl.y then) The jump sound we defined plays at a point near the camera. The Rigidbody2D defined as rb uses the Vector2.up to go up by using the JumpForce variabl. The dino no longe touches the ground tag when it's in the air so the enableJump bool variable is set to false.

```
void Update()
{
if (Input.GetKeyDown(KeyCode.Space)&& enableJump ==true)
{
AudioSource.PlayClipAtPoint(Jump, new Vector3(0, 0, -10));
rb.AddForce(Vector2.up * JumpForce);
enableJump = false;
}
}
}
```

You can drag the sound files to use into the inspector's view. Gravity settings and JumpForce can be changed in the inspector's view to. Experiment with it and set it to your wishes.

In orde to create the distance counter and a hi-score we will use UI text elements. I create 2 text elements and called them : Counter and Highscore. I gave the canvas of these element this script.

ScoreManagment Script.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
```

Make sure to insert this UI line othewise the UI elements won't work.

```
using UnityEngine.UI;

public class ScoreManager : MonoBehaviour
{
```

2 public texts are created CounterText and HighscoreText. 2 Int variables are both set to 0 (Score and Highscore)

```
public Text CounterText;
public Text HighscoreText;

public int Score = 0;
public int Highscore = 0;
```

An audio clip to use is defined as we learned before.

```
public AudioClip Hundred;
```

A bool variable Isrunning is created and set to false. 2 float variables are created one for speed and one is Counterr to be used for scoring later in script. A final bool variable to play the sound is set to true.

```
public bool Isrunning = false;
public float speed = 3f;
public float Counterr = 0;
public bool playsound = true;
```

At start the Highscore is loaded it's a file called highscore. Bothe Text elements will show the numbers in to a String that contains 6 numbers.

```
void Start()
{
Highscore = PlayerPrefs.GetInt("highscore");
CounterText.text = Score.ToString("000000");
HighscoreText.text = "HI " + Highscore.ToString("000000");
}
```

An IE numerator is created that will set the playsound bool variable to true after 2 seconds. This is called a Coroutine.

```
IEnumerator ResetNow()
{
yield return new WaitForSeconds(2f);
playsound = true;
}
```

When playsound is true and only then,and the score is 100 it will play the soundfile defined at a position near the camera turn the bool variable playsound back to false(Otherwise the sound keeps repeating). Last it starts the Coroutine called ResetNow. (See above part script IE numerator)

```
void Update()
{
if(playsound == true && Score == 100)
{
playsound = false;
AudioSource.PlayClipAtPoint(Hundred, new Vector3(0, 0, -10));
StartCoroutine(ResetNow());
}
```

The Highscore text element is always showing.

```
HighscoreText.text = "HI " + Highscore.ToString("000000");
```

When the distance counter is no longer running and Score is bigger than the highscore it wil turn the Highscore into the score and saves it by using playerprefs to a file with the name highscore, using the int Highscore. The file is stored in :
HKEY_CURRENT_USER/Software/Unity/UnityEditor/Company/Projectname/
Remember in the start function we load it from there as it is saved here. You can find the file using Registry editor.

```
if (Isrunning == false)
{
if(Score > Highscore)
{
Highscore = Score;
PlayerPrefs.SetInt("highscore", Highscore);
}
}
```

When the distance counter runs it uses the float Counterr as it adds 1 tot it with the give speed. The Score is always equal to the Counterr as we convert this float to an int. The Counter text element is always showing.

```
if (Isrunning == true)
{
Counterr += 1*speed*Time.deltaTime;
Score = (int)Counterr;
CounterText.text = Score.ToString("000000");
}
}
}
```

The ground is actually 2 gameObjects that scroll continiously. Place the first in the middle of the screen so it overlaps the left and right side of the screen. Place the seccond as a child right next to the first. Make sure both objects get the tag name Ground. No look at the x position of this child object and make sure the parent object gets these coordinates but with a minus – in front of it. Give the parent ground object this script :

Scrolling Ground Script.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Scrollingground : MonoBehaviour
{
```

3 floats are created. Moving speed, offset and for a new position on the X axis.

```
public float MovingSpeed = 1f;
public float offset;
private float newXposition;
```

A Vector 2 is created that will be the Start position. The familiar reference to the Gamemanager is also there.

```
private Vector2 StartPosition;
private GameManager GM;
```

In the start function we make sure the object will be at the start position. (The vector2) The other script to use is found.

```
void Start()
{
StartPosition = transform.position;
GM = FindObjectOfType<GameManager>();
}
```

If the bool variable in the Gamemanager Script is set to true the game object will slide/move to the left using math repeat from the x position untill it reaches the offset. If it reaches the offset it will get repositioned back to the start position and repeats itself.

```
void Update()
{
if (GM.Playinggame == true)
{
newXposition = Mathf.Repeat(Time.time * -MovingSpeed, offset);
transform.position = StartPosition + Vector2.right * newXposition;
}
}
}
```

As you might noticed all these scripts use and reference to the game manager Script. It's a good idea to make a gamemanager script as it can hold all your basic rules and mecanics of the game. Just create an empty game object name it Gamemanager and give it an icon you like. The icon wil change into a cog wheel. Don't worry I think this is a little mistake of unity but it does not give you any problems. Give this object this script :

GameManager Script.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class GameManager : MonoBehaviour
{
```

A reference to the scoreManager script is made.

```
private ScoreManager SM;
```

2 game object to use are defined. A bool variable is created to determine if the game is or is not playing. You can drag and drop these objects in the Inspector's view.

```
public GameObject gameOverpanel;
public GameObject newDino;
public bool Playinggame = false;
```

In the start function the game over panel is set to be invisible, the playinggame bool variant is set to false and the score management script is found.

```
void Start()
{
gameOverpanel.SetActive(false);
Playinggame = false;
SM = FindObjectOfType<ScoreManager>();
}
}
```

A game over function is created public and can be called from any other script. It turns the Bool variable in the Scoremanager script to false and makes the gameover panel visible. The playinggame is set to false.

```
public void GameOver()
{
SM.Isrunning = false;
Playinggame = false;
gameOverpanel.SetActive(true);
}
```

A restart game function is created public and can be called from any other script. It turns the Bool variable in the Scoremanager script to true after setting the counter to 0 and makes the gameover panel invisible. The playinggame is set to true and the player (Dino) is created.

```
public void RestartGame()
{
gameOverpanel.SetActive(false);
SM.Counterr = 0;
Instantiate(newDino, new Vector3(-2.053f, -1.38f, 0), Quaternion.identity);
SM.Isrunning = true;
Playinggame = true;
}
```

Pfew are you still with me ? We are allmost there. Next will be our cactus spawner. It will use the 3 cactusses stored as prefab game objects. Place an empty game object call it spawner and give it an icon you like. Give it this script :

SpawnerScript.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Spawner : MonoBehaviour
{
```

The 3 game object used to be spawned are defined. You can drag and drop these objects in the Inspector's view. A reference to the gamemanager script is made and 2 float variables are created Spawntime and pic.

```
public GameObject Cactus01, Cactus02, Cactus03;
private GameManager GM;
public float Spawntime;
public float pick;
```

In the start function the Spawntime Is set to 3 and the game manager script is found.

```
void Start()
{
Spawntime = 3f;
GM = FindObjectOfType<GameManager>();
}
```

In the spawnstuff function the variable pick will be set random between 0 and 3 depending on the number it will create a cactus at the given position. The placed cactus will use it's script to start sliding to the left.

```
public void SpawnStuff()
{
pick = Random.Range(0, 3);
if(pick == 0)
{
Instantiate(Cactus01, new Vector3(transform.position.x, -1.44f, 0),
Quaternion.identity);
}
if (pick == 1)
{
Instantiate(Cactus01, new Vector3(transform.position.x, -1.44f, 0),
Quaternion.identity);
}
if (pick == 2)
{
Instantiate(Cactus01, new Vector3(transform.position.x, -1.44f, 0),
Quaternion.identity);
}
}
```

In the update function the spawntime goes down until it reaches 0. When it reaches zero and the playing game bool variant is true it calls the Spawnstuff function. The spawntime is then set randomly 2 plus between 1 and 4. When it reaches zero and the playing game bool variant is false it resets the spawntime.

```
void Update()
{
Spawntime -= Time.deltaTime;
if (Spawntime <= 0 && GM.Playinggame == true)
{
SpawnStuff();
Spawntime = 2 + Random.Range(1, 4);
}
if (Spawntime <= 0 && GM.Playinggame == false)
{
Spawntime = 2 + Random.Range(1, 4);
}
}
}
```

Playbutton Script.

Left to do is using the buttons on mouse click. Create a play gameobject with a box 2D collider and give it this script :

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Playbutton : MonoBehaviour
{
```

That familiar reference again.

```
private GameManager GM;
```

You should really understand this start function now.

```
void Start()
{
GM = FindObjectOfType<GameManager>();
}
```

When clicked on this game Object it will destroy the game object and calls the restart game function in the game manager script.

```
void OnMouseDown()
{
GM.RestartGame();
Destroy(this.gameObject);
}
```

The retry button is on the game over panel. Make sure that this button has a box collider 2D and this script. (No need to explain it it should be clear.)

Retrybutton Script.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class RetryGame : MonoBehaviour
{

private GameManager GM;

void OnMouseDown()
{
GM.RestartGame();
}
void Start()
{
GM = FindObjectOfType<GameManager>();
}
```

```
}
}
```

Done, enough for you to look into. The buttons could also be scripted in the game manager script. You have a global inside on how to make these kind of games now. The saving and loading of scores is good to know and you have a good view on how to call functions from other scripts. It's all up to you to expand from this basic setup, create more stuff to avoid, perhaps make the dino duck 😉

How To ?

When I was doing an other fun small project (Space invaders) I bumped into this problem. How can I make a bullet drop from one of the aliens randomly picked ? I found a lot of tutorials but none of them really dealed with this. Most of the time the aliens would get a random timer but if you use this the possibility is that they all drop at around the same time and would be impossible to avoid. I asked around in the forum and a person Kurt Dekker pointed me in the right direction. However cause of perhaps a language barrieer it was not exactly what I needed. My good friend Larry Pendleton helped me out and we created this simple working script that solved my problem.

You can see how it all works here: https://www.youtube.com/watch?v=zd0ddTxXSAY&t=10s

I created alien game objects and gave them the name tag Alien. Then I created a new empty object (make sure the Z axis is 0 like the rest of the objects) and called it Invaders. All the alien object are set as child objects. Give this script to the Invader parent game Object.

InvadersScript

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Invaders : MonoBehaviour
{
```

This is familiar stuff a float variable for the time between fire of bullets and a game object prefab that is defined as Bullet.

```
public float firecount = 3;
public GameObject Bullets;
```

This is new by using [] we can create an array of objects that we call enemyObjects we make it private as we don't need it to be public.

```
private GameObject[] enemyObjects;
```

We created a function UpdateEnemyTransforms. It will clear our previous array and assigns a new array based on current enemy count. After doing this we assign our new updated array. This way it will only use the left over enemies (Aliens) every time this function is called.

```
public void UpdateEnemyTransforms()
{
enemyObjects = new GameObject[GameObject.FindGameObjectsWithTag("Alien").Length];
enemyObjects = GameObject.FindGameObjectsWithTag("Alien");
}
```

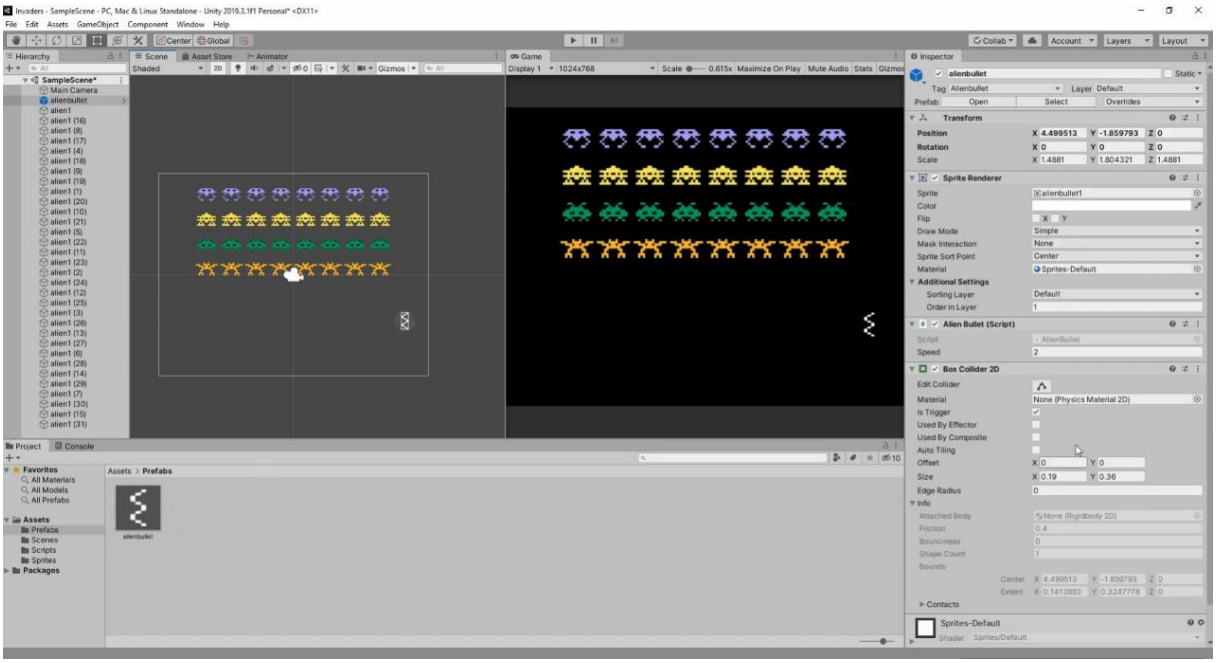
Next we created a function the gets the Vector3 (position) from one of those objects in the array and set this position. It only does that as long as the length (index) of the array is anything but zero. When there is no valid array it sets the Vector3 to zero.

```
public Vector3 GetRandomEnemyPosition()
{
if (enemyObjects.Length != 0)
{
int randomEnemyIndex = Random.Range(0, enemyObjects.Length);
return enemyObjects[randomEnemyIndex].transform.position;
}
return Vector3.zero;
}
```

In the update function the fire count runs to zero when zero is hit or it is lower than zero it will use the UpdateEnemtTransform to determine from wich object to spawn the bullet. After the spawn it resets the firecount back to 3.

```
void Update()
{
firecount -= Time.deltaTime;
if(firecount <= 0)
{
UpdateEnemyTransforms();
Instantiate(Bullets, GetRandomEnemyPosition(), Quaternion.identity);
firecount = 3;
}
}
}
```

You can use this way for different games. All you need to do is make all the enemies have a same tag name to spawn from.



All of the assets used in this issue can be downloaded here: https://rp-interactive.nl/magazine.html

All scripts used in issue 05 are here for you.

Cactus Script.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

///dino run jump game René Pol 01-08-2021©///

public class Cactus : MonoBehaviour
{
public float speed;
private GameManager GM;

void Start()
{
GM = FindObjectOfType<GameManager>();
}

void Update()
{
if(GM.Playinggame == false)
{
Destroy(this.gameObject);
}
transform.Translate(Vector2.left * speed * Time.deltaTime);
}
}
```

Clouds Script.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

///dino run jump game René Pol 01-08-2021©///

public class Clouds : MonoBehaviour
{
public float speed;
private GameManager GM;

void Start()
{
GM = FindObjectOfType<GameManager>();
}

void Update()
{
if (GM.Playinggame == true)
{
transform.Translate(Vector2.left * speed * Time.deltaTime);
if(transform.position.x < -3)
{
```

```
transform.position = new Vector3(3, transform.position.y, transform.position.z);
}
}
}
}
```

DinoScript.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

///dino run jump game René Pol 01-08-2021©///

public class Dino : MonoBehaviour
{
public bool enableJump;
public float JumpForce = 500;
Rigidbody2D rb;
private GameManager GM;
public AudioClip Jump,hit;

public void OnCollisionEnter2D(Collision2D collision)
{
if (collision.gameObject.tag == ("Ground"))
{
enableJump = true;
}
if (collision.gameObject.tag == ("Cactus"))
{
AudioSource.PlayClipAtPoint(hit, new Vector3(0, 0, -10));
GM.GameOver();
Destroy(this.gameObject);
}
}

void Start()
{
GM = FindObjectOfType<GameManager>();
rb = GetComponent<Rigidbody2D>();
}

void Update()
{
if (Input.GetKeyDown(KeyCode.Space)&& enableJump ==true)
{
AudioSource.PlayClipAtPoint(Jump, new Vector3(0, 0, -10));
rb.AddForce(Vector2.up * JumpForce);
enableJump = false;
}
}
}
```

ScoreManagment Script.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

///dino run jump game René Pol 01-08-2021©///

public class ScoreManager : MonoBehaviour
{
public Text CounterText;
public Text HighscoreText;
public int Score = 0;
public int Highscore = 0;
public AudioClip Hundred;
public bool Isrunning = false;
public float speed = 3f;
public float Counterr = 0;
public bool playsound = true;

void Start()
{
Highscore = PlayerPrefs.GetInt("highscore");
CounterText.text = Score.ToString("000000");
HighscoreText.text = "HI " + Highscore.ToString("000000");
}

IEnumerator ResetNow()
{
yield return new WaitForSeconds(2f);
playsound = true;
}

void Update()
{
if(playsound == true && Score == 100)
{
playsound = false;
AudioSource.PlayClipAtPoint(Hundred, new Vector3(0, 0, -10));
StartCoroutine(ResetNow());
}
HighscoreText.text = "HI " + Highscore.ToString("000000");
if (Isrunning == false)
{
if(Score > Highscore)
{
Highscore = Score;
PlayerPrefs.SetInt("highscore", Highscore);
}
}
```

```
if (Isrunning == true)
{
Counterr += 1*speed*Time.deltaTime;
Score = (int)Counterr;
CounterText.text = Score.ToString("000000");
}
}
}
```

Scrolling Ground Script.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

///dino run jump game René Pol 01-08-2021©///

public class Scrollingground : MonoBehaviour
{
public float MovingSpeed = 1f;
public float offset;
private Vector2 StartPosition;
private float newXposition;
private GameManager GM;

void Start()
{
StartPosition = transform.position;
GM = FindObjectOfType<GameManager>();
}

void Update()
{
if (GM.Playinggame == true)
{
newXposition = Mathf.Repeat(Time.time * -MovingSpeed, offset);
transform.position = StartPosition + Vector2.right * newXposition;
}
}
}
```

GameManager Script.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

///dino run jump game René Pol 01-08-2021©///

public class GameManager : MonoBehaviour
{
private ScoreManager SM;
public GameObject gameOverpanel;
public GameObject newDino;
public bool Playinggame = false;

public void GameOver()
{
SM.Isrunning = false;
Playinggame = false;
gameOverpanel.SetActive(true);
}

public void RestartGame()
{
gameOverpanel.SetActive(false);
SM.Counterr = 0;
Instantiate(newDino, new Vector3(-2.053f, -1.38f, 0), Quaternion.identity);
SM.Isrunning = true;
Playinggame = true;
}

void Start()
{
gameOverpanel.SetActive(false);
Playinggame = false;
SM = FindObjectOfType<ScoreManager>();
}
}
```

SpawnerScript.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

///dino run jump game René Pol 01-08-2021©///

public class Spawner : MonoBehaviour
{
public GameObject Cactus01, Cactus02, Cactus03;
private GameManager GM;
public float Spawntime;
public float pick;
```

```
void Start()
{
Spawntime = 3f;
GM = FindObjectOfType<GameManager>();
}

public void SpawnStuff()
{
pick = Random.Range(0, 3);
if(pick == 0)
{
Instantiate(Cactus01, new Vector3(transform.position.x, -1.44f, 0),
Quaternion.identity);
}
if (pick == 1)
{
Instantiate(Cactus01, new Vector3(transform.position.x, -1.44f, 0),
Quaternion.identity);
}
if (pick == 2)
{
Instantiate(Cactus01, new Vector3(transform.position.x, -1.44f, 0),
Quaternion.identity);
}
}

void Update()
{
Spawntime -= Time.deltaTime;
if (Spawntime <= 0 && GM.Playinggame == true)
{
SpawnStuff();
Spawntime = 2 + Random.Range(1, 4);
}
if (Spawntime <= 0 && GM.Playinggame == false)
{
Spawntime = 2 + Random.Range(1, 4);
}
}
}
```

Playbutton Script.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

///dino run jump game René Pol 01-08-2021©///

public class Playbutton : MonoBehaviour
{

private GameManager GM;

void OnMouseDown()
{
GM.RestartGame();
Destroy(this.gameObject);
}

void Start()
{
GM = FindObjectOfType<GameManager>();
}
}
```

Retrybutton Script.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

///dino run jump game René Pol 01-08-2021©///

public class RetryGame : MonoBehaviour
{

private GameManager GM;

void OnMouseDown()
{
GM.RestartGame();
}

void Start()
{
GM = FindObjectOfType<GameManager>();
}
}
```

InvadersScript.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

///Invaders Larry Pendleton & René Pol 01-08-2021©///

public class Invaders : MonoBehaviour
{

public float firecount = 3;
public GameObject Bullets;
private GameObject[] enemyObjects;

public void UpdateEnemyTransforms()
{
enemyObjects = new GameObject[GameObject.FindGameObjectsWithTag("Alien").Length];
enemyObjects = GameObject.FindGameObjectsWithTag("Alien");
}

public Vector3 GetRandomEnemyPosition()
{
if (enemyObjects.Length != 0)
{
int randomEnemyIndex = Random.Range(0, enemyObjects.Length);
return enemyObjects[randomEnemyIndex].transform.position;
}
return Vector3.zero;
}

void Update()
{
firecount -= Time.deltaTime;
if(firecount <= 0)
{
UpdateEnemyTransforms();
Instantiate(Bullets, GetRandomEnemyPosition(), Quaternion.identity);
firecount = 3;
}
}
}
```