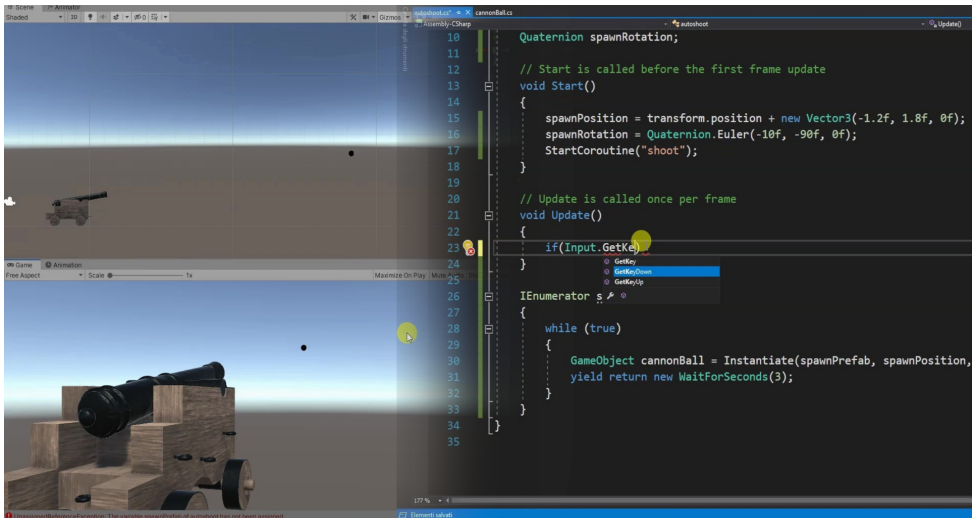


*This PDF is available for free; feel free to share it!*

[Subscribe to my YT channel](#) & turn on notifications to get updates on new videos!



Hello everyone!

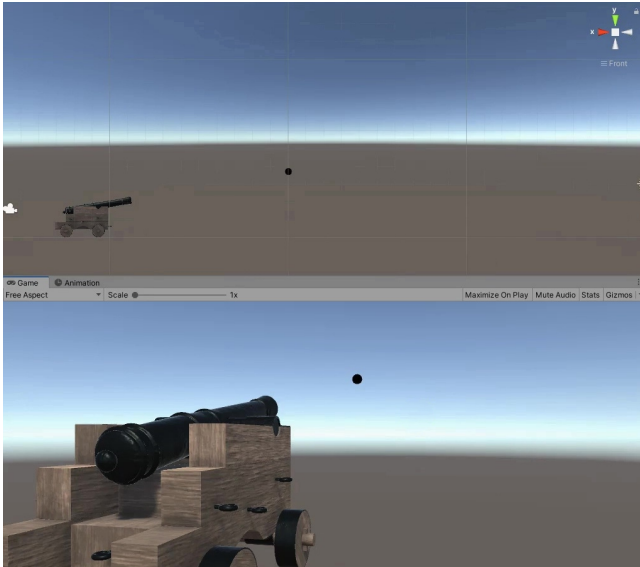
Welcome to this basic tutorial on using **Coroutines** in **Unity**.

**Coroutines** are **functions** that can be performed **concurrently** with the main game execution cycle.

In this tutorial, I will use a practical example to explain the concept and also introduce the **WaitForSeconds** function, which makes it possible to wait for the execution of an operation for a specified number of seconds.

This tutorial assumes that you have a basic understanding of **C# programming in Unity**. It was recorded using the 2022 version of Unity, but **Coroutines** are available in many programming languages and previous versions of Unity.

Our goal is to **make the cannon fire a cannonball Prefab** (a simple black sphere) **automatically every 3 seconds** until an external event occurs that interrupts this routine.



In this case, the external event will be **the user pressing the S key ("Stop")**.

To start, we need to **create a script for the cannon** and call it "*autoshoot*".

Within the main class, I define three variables to specify the starting point of the ball with respect to the cannon:

```
public GameObject spawnPrefab;
```

I will link the cannonball prefab in the Unity Editor to this public variable

```
Vector3 spawnPosition;
```

```
Quaternion spawnRotation;
```

I calculated this point and the initial orientation, so I initialize these two values within the **Start** function:

```
spawnPosition = transform.position + new Vector3(-1.2f, 1.8f, 0f);
```

```
spawnRotation = Quaternion.Euler(-10f, -90f, 0f);
```

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class autoshoot : MonoBehaviour
6 {
7
8     public GameObject spawnPrefab;
9     Vector3 spawnPosition;
10    Quaternion spawnRotation;
11    // Start is called before the first frame update
12    void Start()
13    {
14    }
15
16
17    // Update is called once per frame
18    void Update()
19    {
20    }
21
22 }
23
```

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class autoshoot : MonoBehaviour
6 {
7
8     public GameObject spawnPrefab;
9     Vector3 spawnPosition;
10    Quaternion spawnRotation;
11
12    // Start is called before the first frame update
13    void Start()
14    {
15        spawnPosition = transform.position + new Vector3(-1.2f, 1.8f, 0f);
16        spawnRotation = Quaternion.Euler(-10f, -90f, 0f);
17    }
18
19
20    // Update is called once per frame
21    void Update()
22    {
23    }
24
25 }
```

It is clear that we need **to implement a loop that repeats indefinitely** (unless an external event interrupts it) and includes **a delay instruction of 3 seconds** before moving on to the next iteration.

[NOTE - The cannonball prefab comes with a script which, upon startup, provides an initial forward push to the ball and sets the object to self-destruct after 3 seconds, so as not to fill either the virtual universe or the system memory with cannonballs.]

The **WaitForSeconds** instruction allows us to pause for a specified amount of time, which in our case is 3 seconds.

Now, we need to create a **WHILE** block to execute the main loop (which I'm currently writing in an empty place in the script).

```
while(true) {  
    GameObject cannonBall = Instantiate(spawnPrefab,  
    spawnPosition, spawnRotation, null);  
    yield return new WaitForSeconds(3);  
}
```

So, where should we place this WHILE block?

**It cannot be placed in Start**, as we cannot use **yield WaitForSeconds** there; plus, putting it in **Update** would not delay it as **Update** is called on every frame of the game.

We must, therefore, create a function called "**shoot**" that returns the type **IEnumerator** and includes the **WHILE** block.

This function will allow us to **start a Coroutine using the StartCoroutine instruction** with the function name as a parameter between double quotes.

Since the cannon needs to start firing immediately, we write:

```
StartCoroutine("shoot");
```

within Start immediately after initializing **spawnPosition** and **spawnRotation**.

Let's save the script and go to the editor to try the script defined up to this point.

To be able **to stop or restart the Coroutine**, let's go back to the script and add two statements.

```
autoShoot.cs cannonBall.cs
Assembly-CSharp autoShoot Start()

10 Quaternion spawnRotation;
11
12 // Start is called before the first frame update
13 void Start()
14 {
15     spawnPosition = transform.position + new Vector3(-1.2f, 1.8f, 0f);
16     spawnRotation = Quaternion.Euler(-10f, -90f, 0f);
17     StartCoroutine("shoot");
18 }
19
20 // Update is called once per frame
21 void Update()
22 {
23 }
24
25
26 IEnumerator shoot()
27 {
28     while (true)
29     {
30         GameObject cannonBall = Instantiate(spawnPrefab, spawnPosition, spawnRotation, null);
31         yield return new WaitForSeconds(3);
32     }
33 }
34 }
```

We can **stop the Coroutine** by checking if the **S** key is pressed and then calling the **StopCoroutine** function with the **"shoot"** Coroutine as a parameter.

Similarly, we can **restart the Coroutine** by checking if the **R** key is pressed and calling the **StartCoroutine** function with **"shoot"** as a parameter.

So let's write (in **Update**):

```
if(Input.GetKeyDown(KeyCode.S)) StopCoroutine("shoot");
if(Input.GetKeyDown(KeyCode.R)) StartCoroutine("shoot");
```

```
autoShoot.cs cannonBall.cs
Assembly-CSharp autoShoot Update()

10 Quaternion spawnRotation;
11
12 // Start is called before the first frame update
13 void Start()
14 {
15     spawnPosition = transform.position + new Vector3(-1.2f, 1.8f, 0f);
16     spawnRotation = Quaternion.Euler(-10f, -90f, 0f);
17     StartCoroutine("shoot");
18 }
19
20 // Update is called once per frame
21 void Update()
22 {
23     if (Input.GetKeyDown(KeyCode.S)) StopCoroutine("shoot");
24     if (Input.GetKeyDown(KeyCode.R)) StartCoroutine("shoot");
25 }
26
27 IEnumerator shoot()
28 {
29     while (true)
30     {
31         GameObject cannonBall = Instantiate(spawnPrefab, spawnPosition, spawnRotation, null);
32         yield return new WaitForSeconds(3);
33     }
34 }
35
36 }
```

After adding these statements, let's save the script and go back to the Unity editor to try it out.

That's all for this short tutorial; I hope it was useful to you!

See you soon!

---

---