Unity の Coroutinesと"yield WaitForSeconds"関数の概要

このチュートリアルで使用される3Dモデル： Naval Cannon (https://sketchfab.com/3d-models/naval-cannon-3407164113c2487e9391aa8d2b3902c2)

こんにちは、みなさん!

これは**Unity**での**Coroutines**の使用に関する入門チュートリアルです; それです: メインゲーム実行サイクルと同時に実行できる機能に対して行います。実践的な例を用いてトピックを扱います, コンセプトを明確にし、その間に、私はこれを利用して**WaitForSeconds**関数を使用します, これは、操作の実行を数秒間待機させます。

このチュートリアルは、**Unity**での C# プログラミングの基礎知識を既に持っている人を対象にしています。それは**Unity**の2020バージョンで記録されています。ただし、**Coroutines** (多くのプログラミング言語で利用可能) は、以前のバージョンの**Unity**でも利用でき、もちろん、以降のバージョンでも利用できます。

この例で使用しているシーンには、海軍砲の3Dモデルがあります。

私たちは、大砲の**Prefab** (単純な黒い球体、実際には)を3秒ごとに自動的に発射させたい、このルーチンを中断する外部イベントが発生するまで (このシナリオでは、外部介入は、ユーザーが S キーを押して(**"Stop"**))。

まず、大砲のスクリプトを作成し、それを**"autoshoot"**と呼びます。

メイン**class**の内部で、3つの変数を定義します:

```
public GameObject spawnPrefab
```

この**prefab**変数に、私は**Unity Editor**で大砲プレハブをリンクします

```
Vector3 spawnPosition

Quaternion spawnRotation
```

私は、大砲に対してボールの開始点を指定する最後の変数が必要です。この点と初期方向を計算したので、**Start** 関数内でこれら 2 つの値を初期化します:

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

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

問題の説明から、(外部イベントがブロックしない限り) 無限に繰り返されるループを実装し、次の反復に進むまで 3 秒間待機させるために遅延命令を含める必要があることは明らかです。

数秒間待機できる命令は**WaitForSeconds** で、これは引数として待機する秒数を取ります。

この時点で、実行するメインループがこの**WHILE**ブロック(スクリプトの空の場所に今のところ書いている)によって与えられていることは明らかです:

```
while(true) {

GameObject cannonBall = Instantiate(spawnPrefab, spawnPosition, spawnRotation, null);

yield return new WaitForSeconds(3);

}
```

*[大砲**prefab**は、起動時にボールに最初の前方プッシュを提供し、仮想宇宙とメモリを大砲で満たさないように、3秒後にオブジェクトを自滅させるスクリプトが付属しています。]*

このWHILEブロックはどこに置くべきですか?

私たちはそれを**Start**に置くことができません, プログラミング エラーが原因のいずれか (ここには、**yield WaitForSeconds**を配置できません), または、プログラムが実際にスタートで**Start** (**WaitForSeconds**なしで**Start**に入れようとしないでください、**While**ループを試してそう定義されます: **Unity**はフリーズします): 同様に、**Update**に入れることはできません...

Updateに**WaitForSeconds**を配置すると問題は解決されません。whileループなし, **Update**はゲームのすべてのフレームでまだ呼び出されるため: **WaitForSeconds**はそれを遅らせません。

したがって、このコード ブロックを関数の内部に挿入する必要があります。タイプ**IEnumerator**を帰る。この関数を**shoot**と呼びます。

Coroutine 内でこの関数を開始するには、関数名をパラメーターとして持つ**StartCoroutine**命令を二重引用符の間で使用する必要があります。大砲は直ちに発射を開始しなければならないので、我々は書く:

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

Start内で、初期後**spawnPosition**と**spawnRotation**。

スクリプトを保存し、**Unity editor**に移動して、これまでに定義されたスクリプトを試してみましょう。

スクリプトに戻って 2 つのステートメントを挿入し、**Coroutine**を停止 (S キー) または再起動 (R キー) できるようにします:

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

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

スクリプトを保存し、**Unity editor**に移動して、ここまで定義されたスクリプトを試してみましょう。

まあ、それは、この短いチュートリアルのためのすべてです!私はそれがあなたに役立つことを願っています!じゃあね！