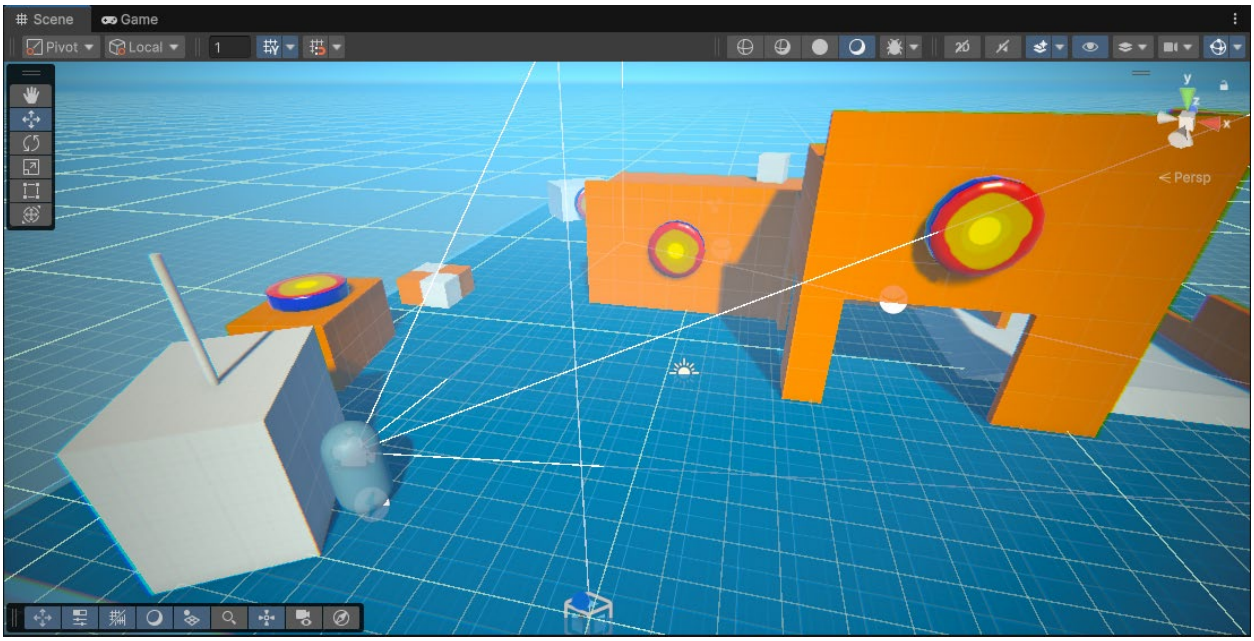
Unity Target Practice

Building a time-trial FPS level in Unity game engine.



Introduction

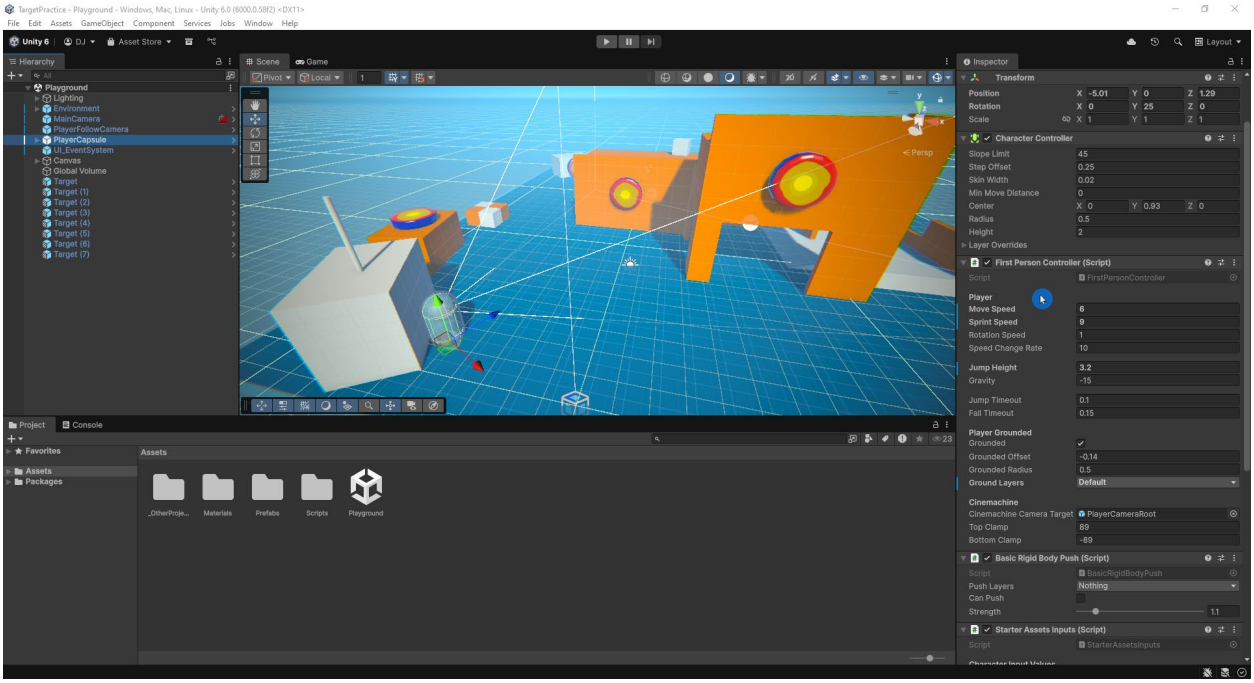
This workshop builds on a short session discussing how 'raycast' shooting works in games development. The project files are all available online here: https://github.com/djoneslec/TargetPractice . The Unity version used in this project is 6000.0.58f2. It's free to download and use unity, which is available from: https://unity.com/download

Playground scene

When you open the project, you should see a Unity scene file in the Project window called **Playground**. Double-click to open the scene, and you should see the level shown in the image above. This scene has been set up with a player, some geometry to walk around, and some targets.

If you play the scene, you'll be able to control the player character. Press the  button above the Scene view to start the demo, and again when you want to exit.

You can control the character with **W,A,S,D** to move, **Shift** to sprint, and **Spacebar** to jump. To adjust how the player moves, select the **PlayerCapsule** that's in the Hierarchy list on the left of the screen, and then adjust the values in the **FirstPersonController** script's Inspector window.



Input

At the moment, you can't shoot the targets because the **Shooting_Script** isn't checking for Mouse-clicks yet. Let's change that.

Double-click on the **Scripts** folder, and then double-click to open **Shooting_Script**. It will open in Visual Studio so we can edit the code.

You'll see that this script already includes lots of information and instructions on the shooting game. The following code needs to be added in-between the lines of green asterixis.

```
39      //*********************************

40

41      if ( Input.GetMouseButtonDown( 0 ) ) {
42          speaker.Play();
43      }
44

45      //*********************************
```

This tells the script to listen for left mouse-clicks, and to play a sound effect when you do. Save the script, and test the game!

Now this sound lacks variation! I like to add a small random adjustment to the pitch of the sound effect each time it plays. So, on the line before **speaker.Play();** add this:

```
39          //*******************************

40

41          if ( Input.GetMouseButtonDown( 0 ) ) {
42              speaker.pitch = Random.Range( 0.8f, 1.1f );
43              speaker.Play();
44          }
45
46          //*******************************
```

You don't want to adjust the pitch too much otherwise the sound becomes very distorted and uncomfortable. But with this, each click sounds slightly different.

Raycast

The next step is to add a **Raycast** check that looks straight ahead from the camera, through the center of the screen, and checks if something is there.

Specifically, we only care to know if we would hit a **Target**. After the sound effect lines, add this:

```
39          //*******************************

40

41          if ( Input.GetMouseButtonDown( 0 ) ) {
42              speaker.pitch = Random.Range( 0.8f, 1.1f );
43              speaker.Play();
44
45              RaycastHit hit;
46              if ( Physics.Raycast( transform.position, transform.forward, out hit, 100f ) ) {
47                  if ( hit.transform.CompareTag( "Target" ) ) {
48                      Shoot( hit );
49                  }
50              }
51          }
52
53          //*******************************
```

Just to break this down a bit more:

- **transform.position** is the position of the camera.
- **Transform.forward** is the direction the camera is looking.
- **out hit** allows us to keep a record of the object we hit.
- **100f** is the distance the ray will extend for.

The targets are already tagged as a Target, so if you save the script and return to Unity, you can test the game!

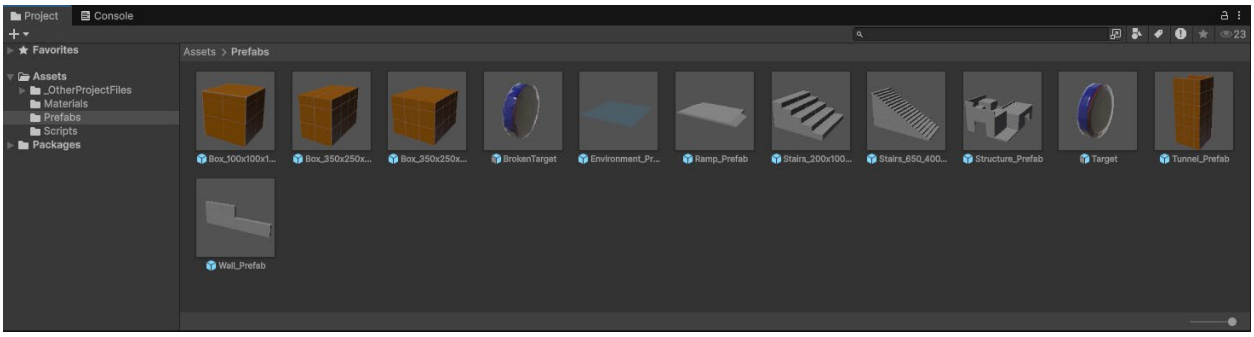
Designing the Level

Now you can improve the level by:

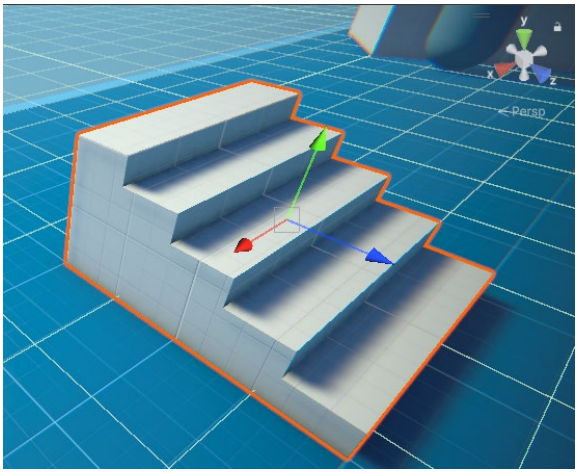
- Moving and adding Targets to the scene
- Adding geometry to the level
- Changing the target score

Adding Objects to the Level

In the **Prefabs** folder, you will find building blocks and the **Target** prefab.



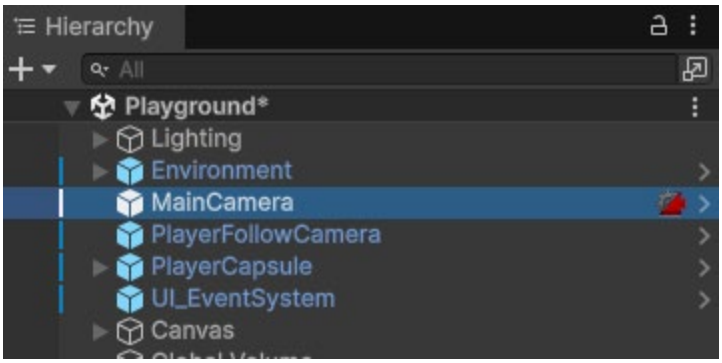
You can simply drag and drop these into the level, and then use the XYZ-arrows to move the objects around.



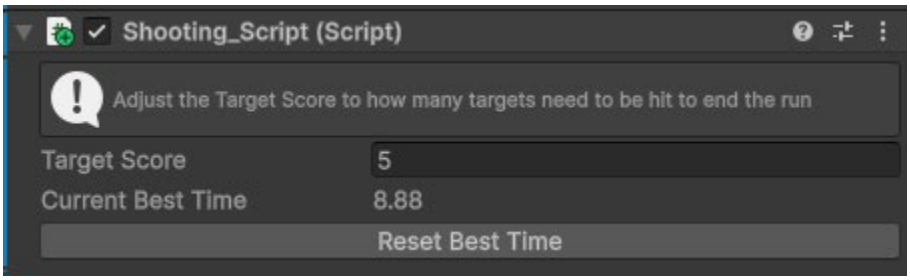
Try to create a level that requires the player to jump and run around in order to find and shoot all the targets!

Adjusting the Target Score

Currently, the game ends when you hit 5 **Targets**. If you want to change that, select the **MainCamera** in the **Hierarchy**.



You'll see the camera's Inspector panel, including the Shooting_Script. Here you can adjust the Target score and also reset your best time.



And that's it!

From here you can continue to expand on the level and test it out. Try to complete a level in the best time, then challenge your friends to do better!

Would you like to know more?

BSc (Hons) Computer Games Programming

This course develops the essential technical programming skills required to produce highly competitive and successful games. Integral to the course is the development of communication, team-working and project management skills required by all graduate employers. The course provides the foundation for entering the dynamic and diverse interactive games market on a variety of networked and stand-alone platforms, including personal computers, mobile phones and handheld computers.

<table>
<tr><td>Level 4</td><td>Games Production</td><td>Game Engine Scripting</td><td>Programming Algorithms and Techniques</td><td>Programming and Mathematics</td></tr>
<tr><td colspan="5"></td></tr>
<tr><td rowspan="2">Level 5</td><td rowspan="2">Level Design</td><td rowspan="2">Games Engine Programming</td><td rowspan="2">Programming Game Architecture</td><td>Professional Awareness</td></tr>
<tr><td>Experimental Games</td></tr>
<tr><td colspan="5">Optional Placement Module</td></tr>
<tr><td rowspan="2">Level 6</td><td rowspan="2">Advanced Group Project</td><td>Advanced AI Algorithms</td><td>Indie Game Development</td><td rowspan="2">Individual Research Project</td></tr>
<tr><td>Graphics Programming with Shaders</td><td>Advanced Concepts in Gaming</td></tr>
</table>

For more information about the course visit http://www.glos.ac.uk

Or email Daryl Jones (djones9@glos.ac.uk)