Intro to Unity

Computer Graphics 2020

About the Course

- Teaching Assistant - Yonatan Shamir
- Questions, complaints, compliments:
  - Moodle forum
  - yonatan.shamir@mail.huji.ac.il

TA 1

- Unity overview & interface
- Unity scripting + technical details
- Live demo
- EX0

Unity is a cross-platform game engine developed by Unity Technologies

Unity Overview

- Build applications for virtually any platform - websites, desktop computers, mobile devices etc.
- Not just for games! Real-time simulations, motion graphics, etc.
- Built-in graphics engine & physics engine
- Simplifies workflow with all kinds of assets - 3D models, 2D images, audio files & more

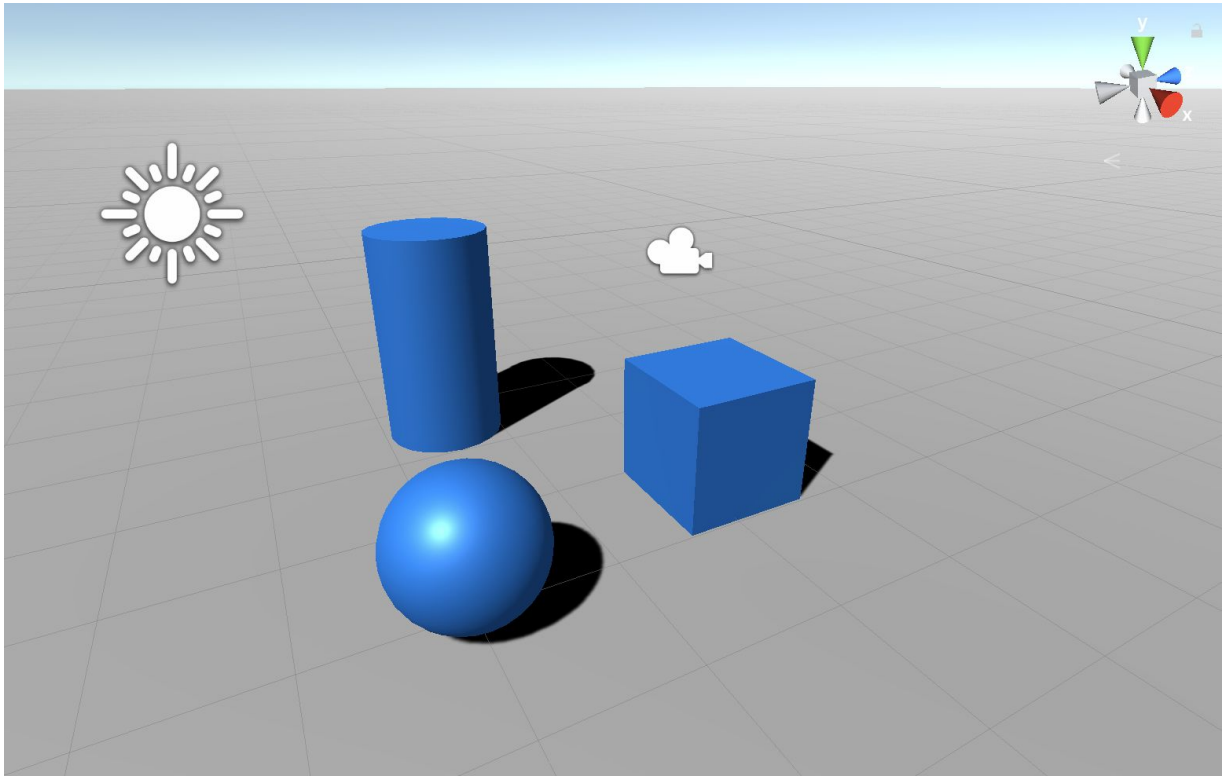

unity
MADE WITH UNITY
Sizzle Reel
Fall 2020

Why Unity for CG?

- Already implemented graphics pipeline
- Provides abstractions
- Built in GUI with interactive tools
- Industry standard
- Free to use!
- This in **not** a Unity development course

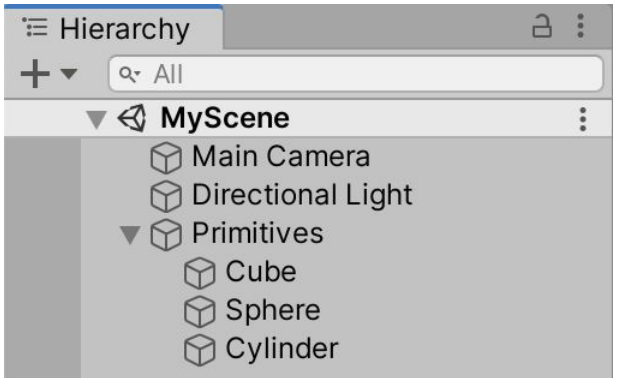
Unity Overview

- In Unity ***Scenes*** contain everything

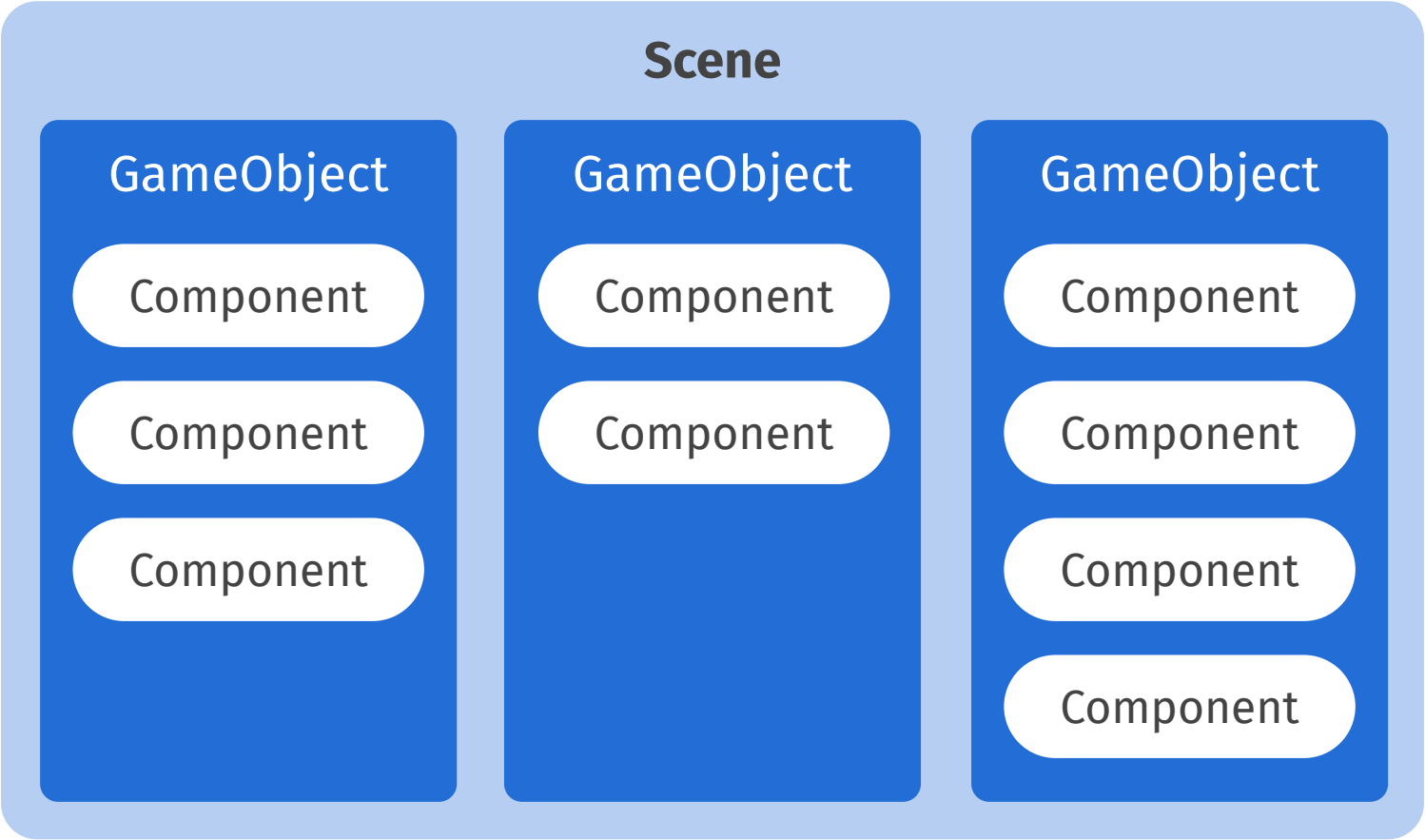


Unity Overview

- Scenes contain a hierarchy of ***GameObjects***
- GameObjects can be nested inside other GameObjects
- Every GameObject is made of ***Components*** that give it certain properties

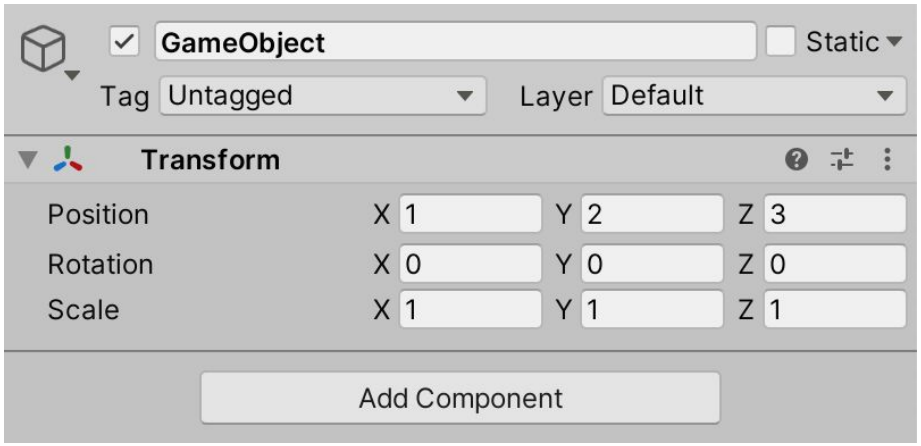


Unity Overview



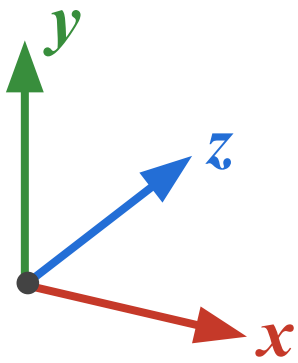
Unity Overview

- Every GameObject has a name, a flag to enable or disable it, and some other basic properties
- Every GameObject has a ***Transform Component*** that encodes its position, rotation and scale in the 3D world of the scene

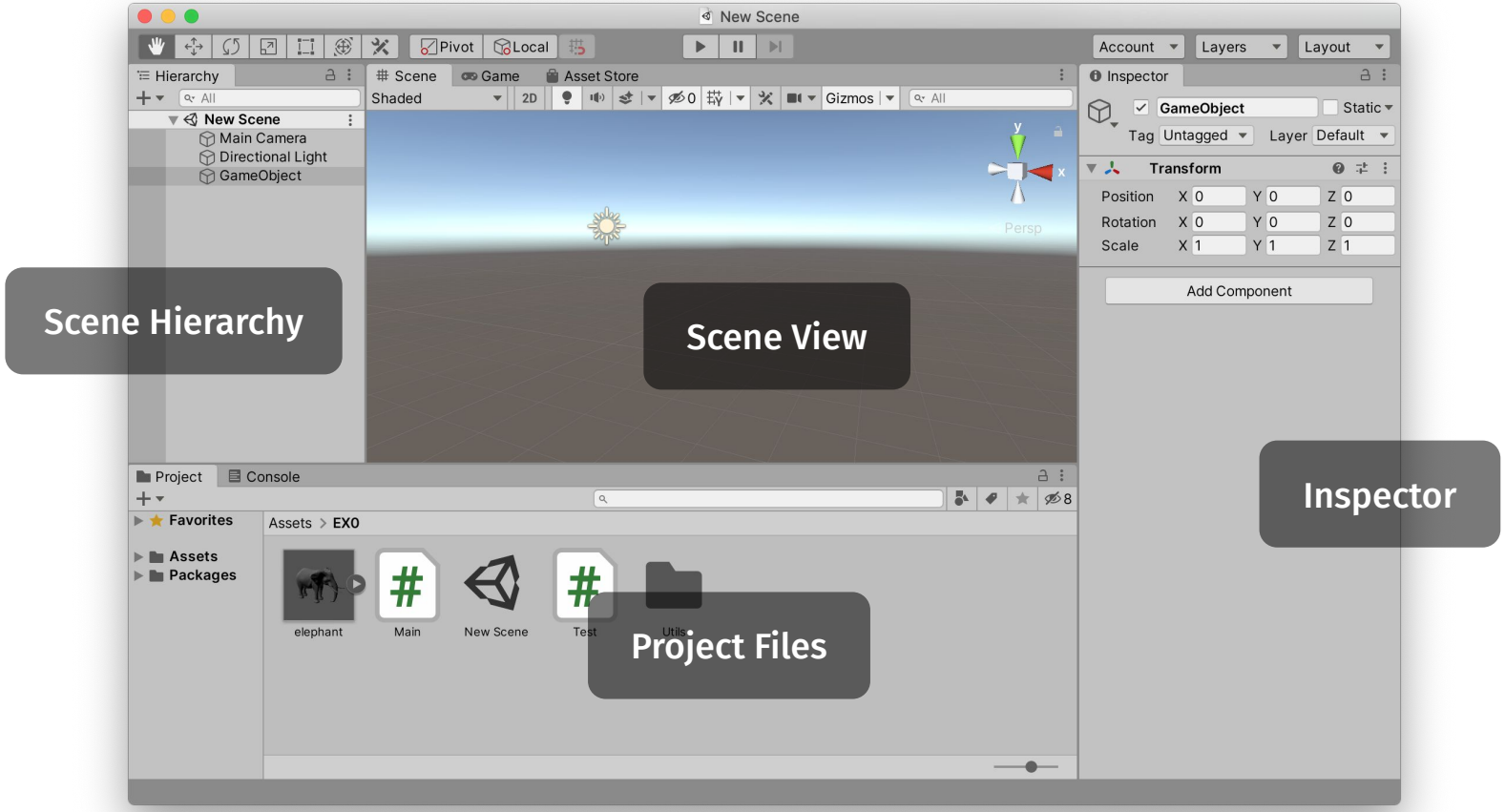


Unity Overview

- Unity uses a *left-handed* 3D coordinate system to position objects inside the scene
- x - right / left y - up / down z - forward / back
- Scenes are centered at $(0, 0, 0)$

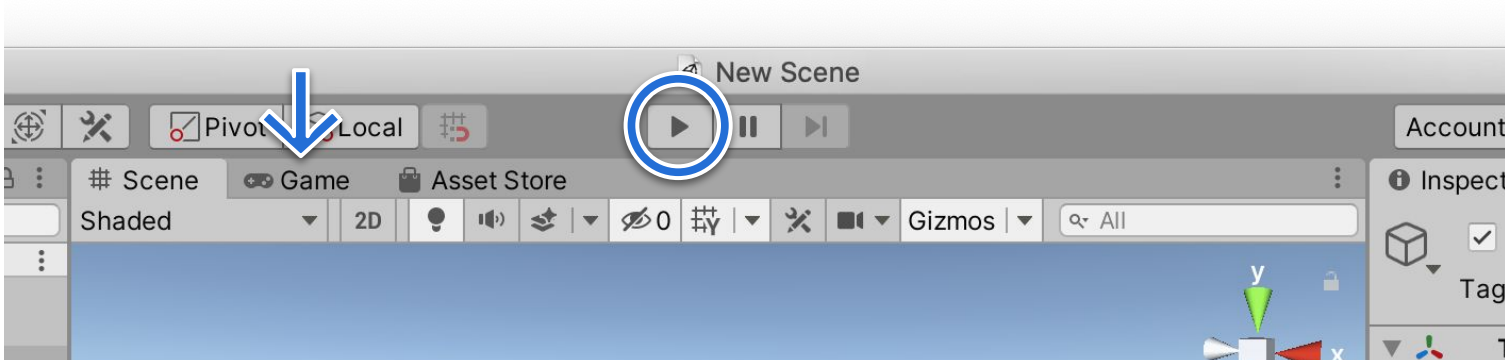


Unity Interface



Play Mode

- We can enter ***Play Mode*** by clicking ▸ play
- In Play Mode we see the scene in the Game view, from the perspective of the camera
- Scripts start running, simulations start working



Play Mode

- Once entering playmode, Unity will try to render our game at a certain ***framerate***, usually 60 fps (frames per second)
- Every frame is a single raster image that has to pass through the entire graphics pipeline - 60 images per second
- Efficiency is important!

Unity Scripting

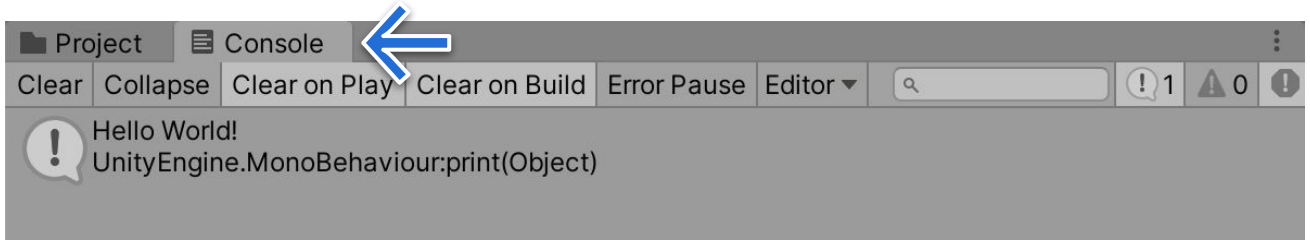
- Unity supports the C# (pronounced C-sharp) programming language natively
- C# is object-oriented
- C# is an industry-standard language similar to Java or C++, developed by Microsoft

Hello World

- A simple program that prints to the Unity console:

```
string s = "Hello World!";
print(s);
```

- After clicking ▸ play, the following will appear in the console, the tab next to the project view:



MonoBehaviour

- **MonoBehaviour** is the base class from which every Unity script derives

```
using UnityEngine;

public class MyClass : MonoBehaviour {
    // Start is called before the first frame update
    void Start()
    { ... }

    // Update is called once per frame
    void Update()
    { ... }
}
```

MonoBehaviour

- By subclassing `MonoBehaviour` we can connect to rest of the Unity ecosystem
- `MonoBehaviours` are attached to GameObjects as ***Script Components***
- We can access other components of the parent GameObject from within the `MonoBehaviour`:

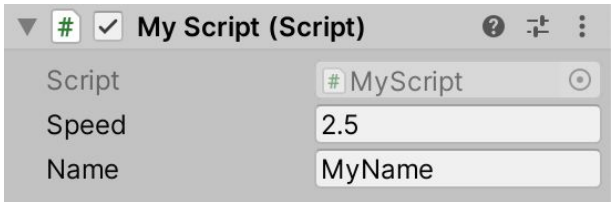
```
Transform t = GetComponent<Transform>();
t.position = new Vector3(0.0f, 0.0f, 0.0f);
```

MonoBehaviour properties

- When declaring `public` class properties, `MonoBehaviour` will automatically reveal them in the Unity inspector:

```
public float speed = 2.5f;

public string name = "MyName";
```



- We can edit properties directly from the UI!
- Works even in Play Mode, but does not save changes when exiting

`MonoBehaviour` functions

- `MonoBehaviour` provides some useful lifecycle functions that we can override, for example:
- `Start()` - Called on the frame when a script is enabled, before Update is called the first time
- `Update()` - Called every frame, if the script is enabled

Unity Vectors

- Unity has vector classes that provide intuitive functions for common vector operations
- We have `Vector2`, `Vector3`, `Vector4` - according to the number of coordinates.
- For example, `Vector3` contains 3 floats: x, y, z
- Full `Vector3` documentation:

  docs.unity3d.com/ScriptReference/Vector3.html

Vectors Example

- To declare 2 vectors $u = [1, 2, 3]$ $v = [4, 5, 6]$:

```
Vector3 u = new Vector3(1.0f, 2.0f, 3.0f);
Vector3 v = new Vector3(4.0f, 5.0f, 6.0f);
```

- A few common vector operations:

$u + v$	`u + v`	`(5.0, 7.0, 9.0)`
$2u$	`2 * u`	`(2.0, 4.0, 6.0)`
u_x	`u.x or u[0]`	`1.0`
$\|\|u\|\|$	`u.magnitude`	`3.7416`
$u \cdot v$	`Vector3.Dot(u, v)`	`32.0`

Unity Animation & Time

- Say we want to animate an object upwards, 2 units per second. In `Update()` we add:

```
t.position += 2 * Vector3.up; // shortcut for (0,1,0)
```

- `Update()` might be called at different intervals!
- After clicking ▸ play, Unity starts counting time in seconds. This can be accessed via `Time.time`
- Multiply by time in seconds since the last frame:

```
t.position += 2 * Vector3.up * Time.deltaTime;
```

Unity Keyboard Input

- Checking to see if a button is pressed is quite simple. For example:

```
1  void Update()
2  {
3      if (Input.GetKeyUp(KeyCode.Space))
4      {
5          print("Space bar was pressed.");
6      }
7      if (Input.GetKey(KeyCode.UpArrow))
8      {
9          print("Up arrow is being pressed.");
10     }
11  }
```

Demo

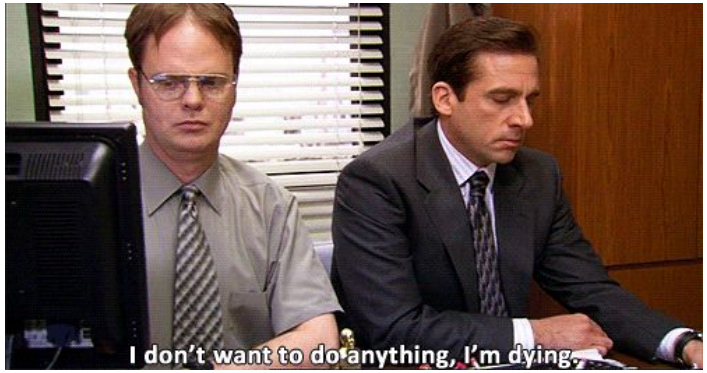
Unity UI + Scripting

Exercises

- You will be given 5 mandatory exercises
- 1 introductory exercise (not for submission)
- 50% of final grade
- Exercises must be done in **pairs**
- Finding partners - Moodle forum

Working Remotely in Unity

- Because of the current situation, working in pairs has to be done remotely
- Usually done with ***Version Control***
- There are 2 main options - Git and Unity Collab



Git

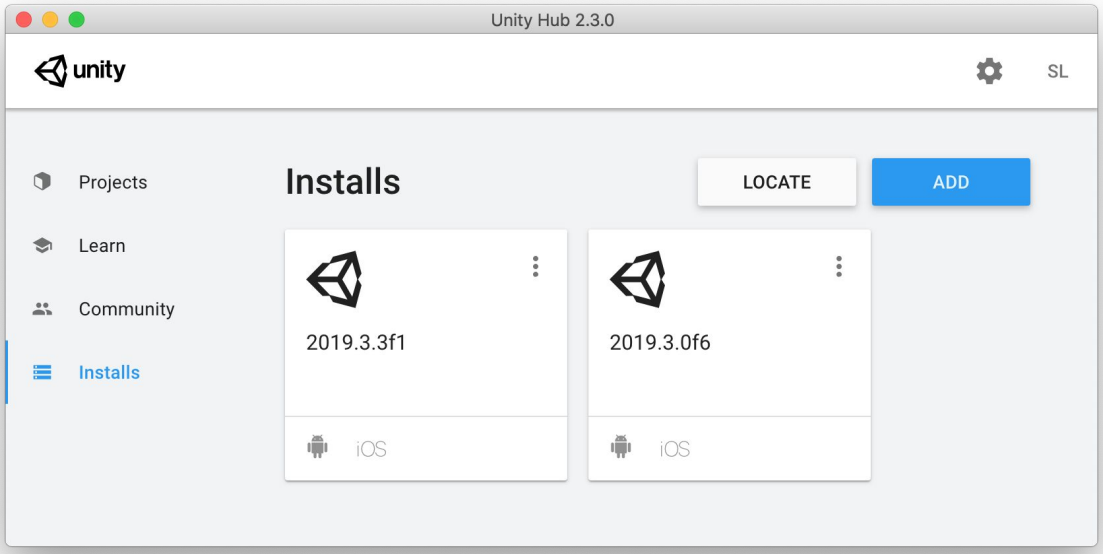
- I recommend working with ***Git***, as most of you will have to learn it at some point anyway
- Many free online resources for learning git are available:
  - www.codecademy.com/learn/learn-git
  - Try.github.io
  - and more, just google “learn git”
- Unity `.gitignore` file supplied in Moodle

Unity Collab

- Unity offers a built-in feature called ***Unity Collab*** the lets you publish & sync changes from within Unity
- Easy to learn, but quite buggy!
- Works in a similar way to Git
- I will send a short guide on how to use it

Exercises - Technical Details

- In this course we will be using **Unity 2020.1.6f1**
- You can download and manage Unity versions using ***Unity Hub***: unity3d.com/get-unity/download

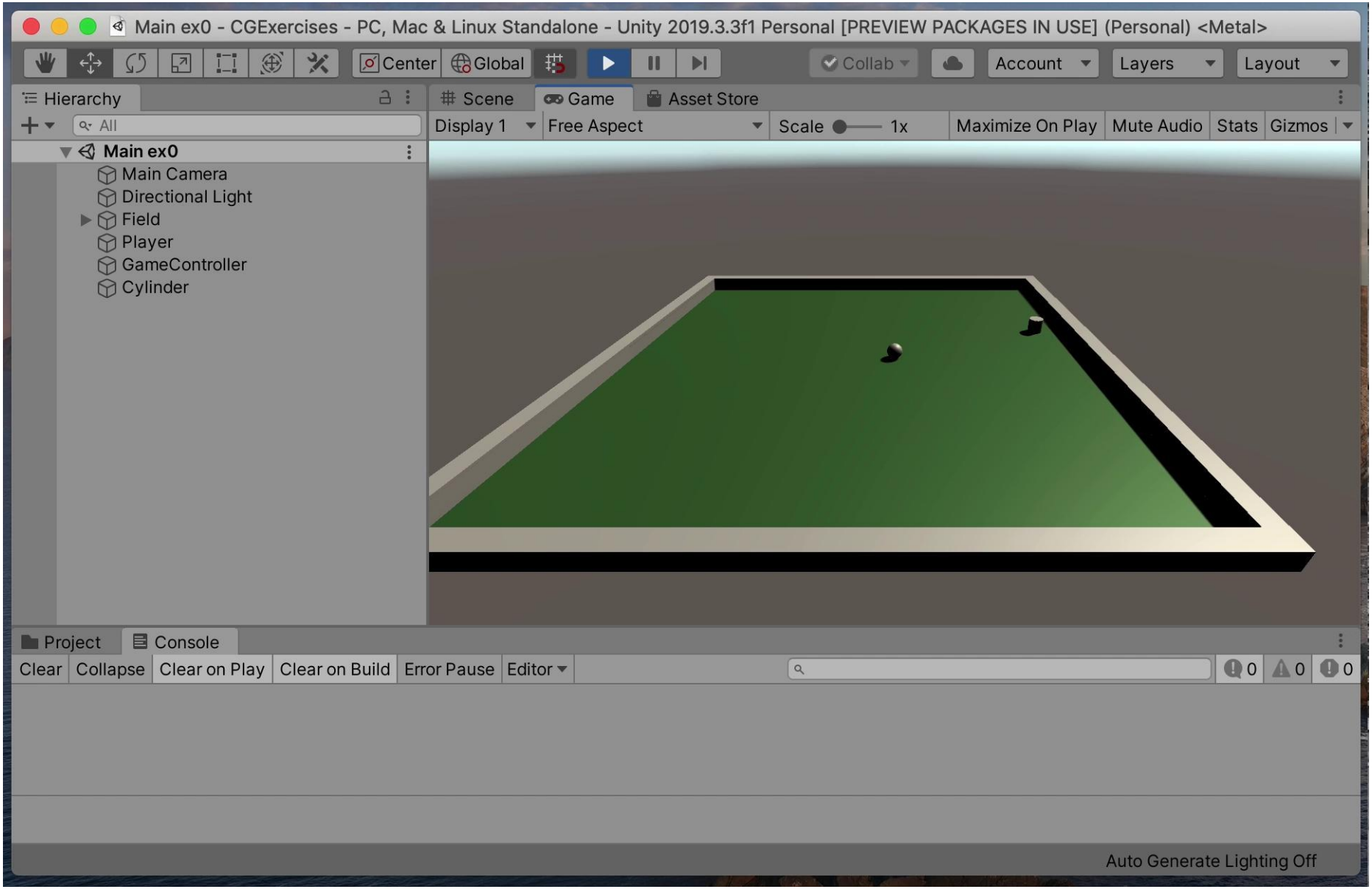


Exercises - Technical Details

- There are several IDEs that work with Unity
- It is recommended in this course to use Visual Studio that is installed by default with Unity

EX0

- This exercise is not for submission
- Learn about projects, scene navigation, scripting, and Unity in general - prepare for the rest of the course and future exercises
- Get comfortable with 3D scene navigation
- Have fun! Experiment, try playing with different settings and features


Main ex0 - CGExercises - PC, Mac & Linux Standalone - Unity 2019.3.3f1 Personal [PREVIEW PACKAGES IN USE] (Personal) <Metal>
Center
Global
Collab
Account
Layers
Layout
Hierarchy
Scene
Game
Asset Store
All
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
Stats
Gizmos
Main ex0
Main Camera
Directional Light
Field
Player
GameController
Cylinder
Project
Console
Clear
Collapse
Clear on Play
Clear on Build
Error Pause
Editor
0
0
0
Auto Generate Lighting Off

Good luck!