Lecture 2
A First Look at Unity

98-127: Game Creation for People Who Want to Make Games (S19)

Written by Adrian Biagioli

Instructors:
Adrian Biagioli (abiagiol@andrew.cmu.edu)
Carter Williams (ncwillia@andrew.cmu.edu)

1 Objectives

By the end of this lesson you will be able to:

- Understand what a Game Engine is, and why we use the Unity game engine
- Utilize common Unity editor functionality and navigate scenes
- Understand Unity's high level architecture: the Scene Graph and GameObjects
- Utilize the inspector to design levels, given art assets and scripts

These lecture notes were written for **Unity 2018.3.0f1**.

2 What is a Game Engine?

Video Games are a very complicated technical challenge. Think about all of the stuff that goes on in your average video game, like Call of Duty. There is a complicated *Graphics Pipeline*, which is the code that communicates very quickly with the user's graphics processor to produce complicated effects (shadows, explosions, hair, muzzle flashes... the list goes on). There is also usually a *Physics Engine* that attempts to simulate how objects would interact with each other (for example, it would figure out how a grenade bounced throughout the scene). There's also some sort of *Audio Mixer* that handles audio effects (for example, a low-pass filter when you are hit with a flashbang). Then there's the *Animation Engine*, which allows characters to move around the scene realistically. Many games include *Artificial Intelligence*, *UI Systems*, and more.

That's a lot of things that *many* games have in common! In fact, it seems like a waste of time to re-do all of these complicated systems for every game that we want to make. This is why Game Engines exist. A **Game Engine** aims to simplify game development by doing all the hard stuff for us (i.e. everything I mentioned in the previous paragraph). This allows game designers like ourselves to focus on the good stuff: how to make our game fun. In addition to this, game engines also tend to simplify *how we code* our games. Many "game-engine-less" games are simply written in C++, which can be hard for beginners. In contrast, Unity is in C#, which is much more approachable.

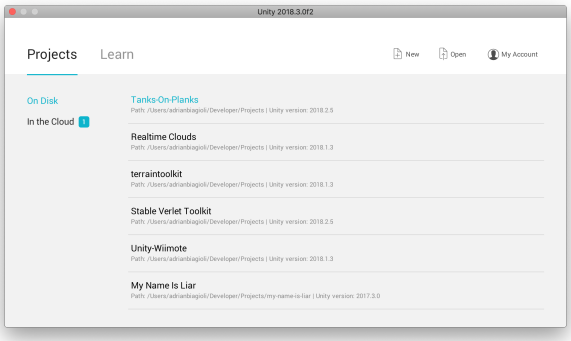
Unity is the most popular Game Engine out there right now for indie game developers, and it is the one that we will use in this course. Right now, Unity's biggest competitor is the **Unreal Engine 4** (UE4), which is developed by Epic Games (creators of *Gears of War* and *Fortnite*). UE4 rivals Unity in many respects and outshines it in some (many will argue that UE4 has better builtin graphics than Unity), but we will not use it in this class because it is generally harder to use for beginners and requires knowledge of C++. Another important difference is how you need to pay royalties if you decide to sell your game: Unity allows you to sell your game without paying any royalties, *until* your game makes $100k per year. After that point, you need to upgrade to Unity "Plus" or "Pro", which has a monthly fee. Unreal requires a 5% royalty of all revenue of your game past $3000, but has no Plus or Pro tier.

3 Getting Started with Unity

If you haven't already, download Unity:

- You can download Unity at https://store.unity.com. Click on "Try Personal" and download the installer to get the latest version. Make sure to enable Windows and Mac build support in the installer options.
- If you're a programmer running Windows or Mac, I would recommend downloading Microsoft Visual Studio. Visual Studio and Visual Studio for Mac both have great integration with Unity and are better than the default MonoDevelop: https://www.visualstudio.com.

When you open Unity, you will first see the **project selection screen**. You may need to create a Unity account—do so if prompted. Click "new project" on the top right of the window.

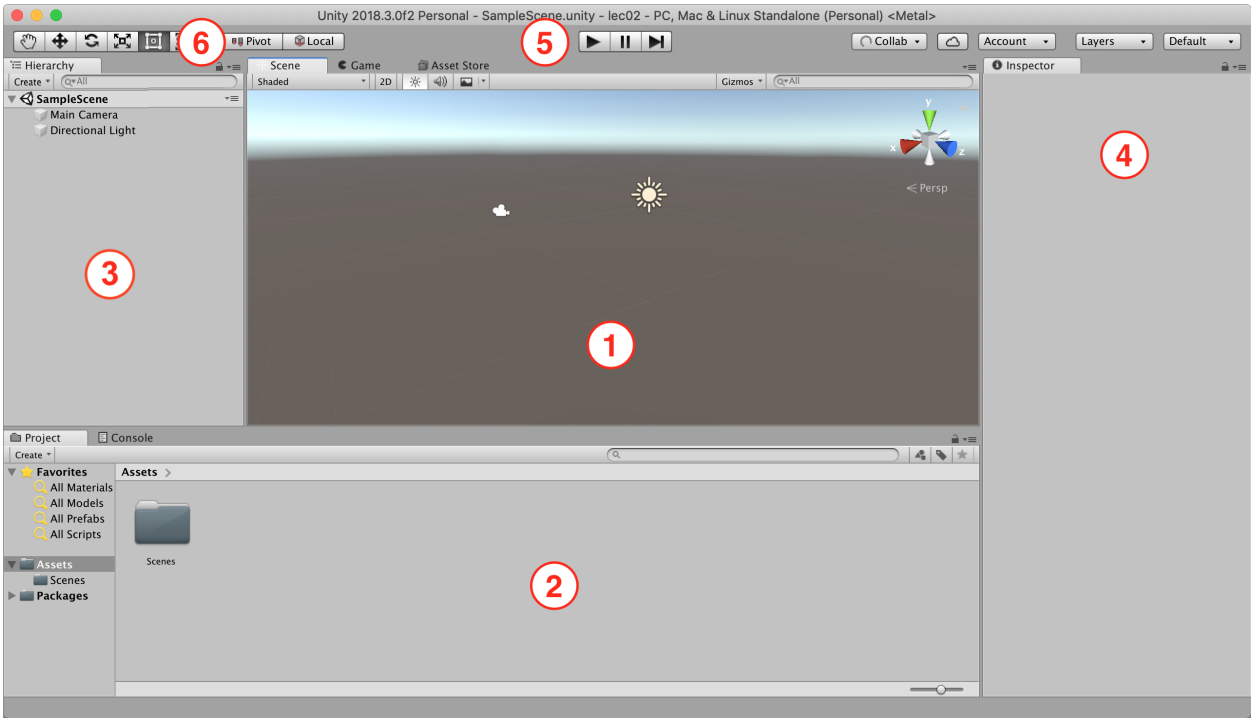


Unity Project Select Window

Enter a name for your new project ("Lecture 2 Project") and specify a location for the project folder. Make sure "3D" is checked and "Enable Unity Analytics" is unchecked. Then click *create project*. **Unity projects are saved in folders on your machine**, so if you browse to the project location you gave then you will find the Unity project files.

The Unity editor should now appear. The editor is divided into **panels**, each providing a different way to interact with your Unity project. If your editor doesn't look like the one on the next page, click *Layout* → *Default* on the top right of the editor. Notice that panels can be reorganized and redocked to other panels by clicking and dragging on the tab at the top of each pane. You can also resize panels and drag them out to their own window.

4 Basic Unity Editor Panels



This is the default configuration of the Unity editor, opening a freshly-made project. Each panel that you see in the above screenshot is labeled as follows:

1. The **Scene View** allows you to look into the game world from a perspective other than the in-game camera. You can navigate the scene view as follows:
   - **Right Click + Drag**: Look around
   - **Right Click + WSADQE**: Move camera (FPS controls). Hold shift to move faster.
   - **Left Click**: Select and manipulate objects (see item 6 below)
2. The **Project Window** is a view into the assets of your game. **Assets** are normal files that your game will use like scripts, 3D Models, textures (images), audio, animation data, scene files, and more. The project window behaves much like the Finder on macOS or Explorer on Windows.
   - In fact, these files are located in the `/Assets` subfolder of your Unity project! Try right clicking on a file in the project window and select "Reveal…"
3. The **Hierarchy View** lists all of the **GameObjects** that are currently in the scene.
   - In Unity terminology, a **GameObject** is any "thing" in your game. *Everything is a GameObject*: the player, a fence, an enemy, a weapon, a button, an NPC, a scoreboard, ... Right now there are two GameObjects in the scene, a Main Camera and a Directional Light for the Sun.
   - You can select a GameObject by clicking on it: either in the scene view or the hierarchy view.

- If you double click on a GameObject in the hierarchy view you can focus on it in the scene view.

4. The **Inspector** allows you to inspect and modify the properties of whichever GameObject or Asset is selected. For example, you can change the position of a GameObject or the name of a file. More information on how to use the Inspector will follow.

5. The **Play Button** (▶) will shift focus from the Scene View to the **Game View**, allowing you to play your game in real time! When you are in "Play Mode", you can switch back to the Scene/Hierarchy/Inspector Panels and continue to modify the scene. However, as soon as you stop playing (by clicking the play button again) your changes will revert to before you started playing, so be careful! While playing, you can also click the **Pause Button** (❚❚) to pause execution of the game (you can still make changes in the Scene View / Inspector while paused). While paused, you can click the **Advance Frame Button** (⏭) to advance by one frame.

6. The **Scene View Tools** change the currently-active manipulator in the scene view. These manipulators allow you to move, rotate, and scale GameObjects–more on this later.