CSCI E-23a

# Introduction to Game Development

produced by CS50

Colton Ogden
David J. Malan

Harvard Extension School
Spring 2018

Assignments

Lectures

Live Stream

Office Hours

Projects

Sections

Slack

Staff

Syllabus

# Assignment 8: “Helicopter Game 3D, The Gem Update”

## Objectives

- Download Unity and get familiar with its interface.
- Read and understand all of the Helicopter Game 3D source code from Lecture 8.
- Add Gems to the game that spawn in much the same way as Coins, though more rarely so. Gems should be worth 5 coins when collected and despawn when off the left edge of the screen.
- Fix the bug whereby the scroll speed of planes, coins, and buildings doesn’t reset when the game is restarted via the space bar.

## Getting Started

### GitHub Classroom

In this course, we’ll use GitHub Classroom to distribute projects and collect submissions. To begin Assignment 8:

1. Click here to go to the GitHub Classroom page for starting the assignment.

2. Click the green “Accept this assignment” button. This will create a GitHub repository for your project. Recall that a git repository is just a location where your code will be stored and which can be used to keep track of changes you make to your code over time.
3. Click on the link that follows “Your assignment has been created here”, which will direct you to the GitHub repository page for your project. It may take a few seconds for GitHub to finish creating your repository.
4. In the upper-right corner of the repository page, click the “Fork” button, and then (if prompted) click on your username. This will create a fork of your project repository, a version of the repository that belongs to your GitHub account.
5. Now, you should be looking at a GitHub repository titled **username/assignment8-username**, where **username** is your GitHub username. This will be the repository to which you will push all of your code while working on your assignment. When working on the assignment, do not directly push to the **games50/assignment8-username** repository: always push your code to your **username/assignment8-username** repository.

## Setup

Time to pull down the starting code for Pokémon! First, on your main repository page (https://github.com/username/assignment8-username), click on the green “Clone or download” button. Copy the “Clone with HTTPS” link to your clipboard (if familiar with SSH, you can use that instead).

Then, in a terminal window (located in `/Applications/Utilities` on Mac or by typing `cmd` in the Windows task bar), move to the directory where you want to store your project on your computer (recall that the `cd` command can change your current directory), and run

```
git clone repository_url assignment8
```

where `repository_url` is the link you just copied from GitHub. You will be prompted for your GitHub username and password

Go ahead and run `cd assignment8` to enter your repository.

## Downloading Blender

First, in order to be able to import some models into our scene appropriately (and to hopefully give you a taste of what 3D modeling is all about, should you be interested), head here to download the latest version of Blender, a free and open-source 3D modeling toolkit that rivals most commercial equivalents. Using and mastering Blender is in and of itself a tremendous skill and art form and not required of this class, but do experiment if you feel so inclined! You can find some fantastic learning resources here and here! Should you wish to tinker with the models used in this project, you can find the helicopter, skyscrapers, and airplane (which I so crudely modeled) in the `Assets/Resources/Models` folder of the Unity project you've downloaded!

## Downloading Unity

You will of course need to download Unity before you can run the distro code and see your project, so do just follow the link here to download Unity's open beta. The setup is very straightforward, but you will need a Unity ID in order to use the software (which is free!), so do visit this link to create one; you should also be prompted to create a new Unity ID via the software's launcher once it's downloaded onto your computer.

Once you've downloaded and logged in to Unity, just click the "Open" button on the launcher and browse to the folder of the cloned code from the distro, and the project will open up!

But wait... nothing seems to load into the scene once you've opened it! With the project open in Unity, navigate to `Assets/Resources/Scenes`, and then select `Main` in the file browser at the bottom of the screen, double-clicking to open, and all should be loaded into the scene view!

Note: If you find that some of the models in your scene are not showing up, it's likely because you either don't have Blender installed yet (see instructions above), or you opened the project prior to the Blender installation. If you already have Blender installed and still don't see anything, do just right-click, in the Unity editor, any of the models located in `Assets/Resources/Models` and select the `Reimport All` option, which should fix missing models after a few moments of loading!

## Next-Level

Welcome to your eighth assignment! Unity is a lot to take in at once, but beneath all of the details, we'll find that this set of tools will allow us to be our most flexible and productive yet, even when coding in C#! As such, this assignment is meant less to be intensive and more just to get a grasp on navigating Unity and understanding how things work.

Your goal this assignment:

- *Add Gems to the game that spawn in much the same way as Coins, though more rarely so. Gems should be worth 5 coins when collected and despawn when off the left edge of the screen*. We have all of the pieces for this already implemented in the `Coin` and `CoinSpawner` classes, so it should suffice simply to make some new classes for the `Gem` and `GemSpawner` behaviors! In the Proto resource pack included in the `Assets` folder, you'll find a model for a gem you can use, but feel free to import your own! You'll need to make a prefab, recall, that you can attach to the `GemSpawner` component, should you choose to implement it similarly to what's in the distro. There are of course other ways to implement this behavior, so feel free to experiment with the software as a chance to learn it all the more thoroughly if curious! Do remember to make `Gems` worth 5 coins instead of just 1, and ensure they're more rare than `Coins` as well! Aside from that, they should behave identically to `Coins`, including moving automatically from right to left and despawning when past the left edge of the screen!

- *Fix the bug whereby the scroll speed of planes, coins, and buildings doesn't reset when the game is restarted via the space bar.* This one's a one-liner; note that static variables aren't actually reset upon loading a scene, so a place to check would be the `SkyscraperSpawner`, as the `speed` field therein is what actually drives the speed for `Skyscrapers`, `Airplanes`, and `Coins`! However, we won't find that this is the place where the game is reset upon pressing the space bar, and thus changing `speed` here doesn't make much sense; any guesses as to where the code for resetting the game could be located?

# How to Submit

## Step 1 of 1

1. Go to the GitHub page for your **username/assignment8-username** repository (note: this is different from the **games50/assignment8-username** repository).
2. On the right side of the screen, click the Pull request button.
3. Make sure that the "base fork" is `games50/assignment8-username`, and the "head fork" is `username/assignment8-username`.
4. Click "Create pull request".
5. On the next page, click the "Create pull request" button again.

Congratulations! You've completed Assignment 8.