CS 4730: Computer Game Design
Lab 1: Introduction

Overview

For the next few weeks, you will be implementing a 2D game engine. This first week, we will start off nice and easy. You will take a look at some starter code that we have prepared for you, and get acquainted with various classes involved. It is VERY IMPORTANT that you take this seriously, so that we can hit the ground running next week for lab 2.

Pick a Programming Language

The biggest decision that you have to make is which language you'd like to implement your engine in. I am providing starter code in two languages[1]. The languages, along with a short description of the pros and cons of each, are as follows:

Java: This code base is more bloated / complicated (initially). Probably much easier to scale as you go through the labs. I have implemented every lab in Java so I can provide much more support if you are doing these in Java.

Javascript (html5 canvas): Code is simpler (especially initially), but probably a little bit more 'unwieldy' once the code base starts to grow. Web based (which is cool I guess), but I did not fully implement the labs in this engine (just the first two) so I may not be as able to give support if you get stuck. Also, I'm using ECMAScript 6 features to simplify the syntax of the code, so any browser you use must support this.

[1] If you REALLY want to use another language, you may…but I don't recommend it as you will have to start from scratch on your own, and you will receive essentially no support (or sympathy) from us if you run into issues.

Run the Code

Once you've chosen what language you'd like (I suppose you can look at both if you want to), download the respective code base and run the code.

Java: You can either import the code into eclipse, or simply run the code from the command line (your preference). The main class is called *LabOneGame.java*. There is a README file that will help here.

Javascript: Trivial to run. Simply open up index.html in a browser.

You should see a picture of Mario in the upper left corner of the "game". Really fun right!!

Understand the Code

Look through the src folder (the folders for the two languages have the exact same structure and filenames) and find the following classes:

DisplayObject.java
Sprite.java
Game.java
LabOneGame.java

For each, write a paragraph or two describing what the class does and why. Convince me that you've read the code carefully enough that you understand it's purpose and it's (very limited) functionality.

Making Mario Move

For fun, let's write a bit of very easy code to make Mario move around the screen. Adapt LabOneGame class to accomplish this. Keep the following in mind:

- update(pressedKeys) is called once per frame and provides you with the keys on the keyboard currently being pressed.
- You will want to use g.translate(x, y) inside of your draw method. This translates the canvas that is being drawn on by x and y respectively. Make sure that you move the canvas over, draw Mario, and then move it back to where it was before.
- For simplicity today, we can use new fields of LabOneGame class called xPos and yPos to store Mario's current position. Next week we will store this information in the Sprite itself (where it belongs).

Sample Game: Mario Clicker

Lastly, let's make a simple two-player game. Player one will use the keyboard to move Mario around the screen and player 2 will try to click on Mario to score points. Your game should have the following:

- Draw a number indicating Mario's health and a timer somewhere on the screen.
- The game runs for 30 seconds.
- After 30 seconds, player 1 wins if Mario's health has not hit 0. Otherwise, player 2 wins
- Tune the game (Mario speed, etc.) to make the game as interesting and fun as you can.

Turn In

You will turn in 1) a short pdf proving you read and understand the classes in the code and 2) your code-base so we can play Mario-Clicker.

Please submit only one file for submission. Zip up all of your submission into one archive. If you are using java, your archive MUST INCLUDE an executable jar file (so that we can simply run this file and test your game out quickly). If you do not know how to do this, then read the README file inside of the Java starter code.