CSC 591/481 Fall 16 Homework 4: Scripting New Games

Due: 12/1/16 at 4:30pm

Overview

Your task for the fourth and final assignment is to implement a script management system, some scripted functionality, and a new game (or games) to demonstrate the resuability of your engine design. As with assignments 1–3, you will be using the Processing programming environment. Additionally, you will be building off of the code you have written for assignments 1–3. Before the due date, please submit your code and a writeup of your results, including screenshots where appropriate. The assignment is an ***individual assignment***, you are to work alone. As always, you are expected to abide by the University's Academic Integrity Policy (`http://policies.ncsu.edu/policy/pol-11-35-01`), which includes providing appropriate attribution for all external sources of information consulted while working on this assignment.

There are 125 points available on this assignment. Students enrolled in 481 are required to complete Parts 1–3 and the writeup, but may also choose to complete Part 4. Students earning scores above 100 will receive a 100 on the assignment. Students enrolled in 591 are required to complete all 125 points, and will receive a grade as the percentage of the 125 points they earn. All students are required to submit the writeup addressing the sections of the assignment they have completed.

Part 1: Scripting (30 pts)

For this part of your homework, you are to create a `ScriptManager` that will enable you to script the behavior of game objects as well as script event *handling* in your 2D platformer game. You are welcome to use the demonstration code posted on the course moodle page as a starting point.

Using the `javax.script` package, create a script manager that allows you to load and run arbitrary scripts. Using the script manager, demonstrate the capability to modify game objects during regular update cycles using scripts. Additionally, demonstrate the ability to handle events using scripts. One example of each will suffice.

Part 2: A Second Game (40pts)

At this point in the semester, you should have a very functional game engine. Congratulations! Now it's time to test its versatility. Your task for this assignment is to implement a second game using your engine code. You should make use of your engine's functionality to implement one of the two following games (or obtain prior permission from the instructor if you desire to implement a different game):

- one level of a classic Bubble Shooter. If you are not familiar with the genre, play some of the free games on `http://bubbleshooter.net/` to get a feel for how they work.
- one level of the classic Space Invaders game. If you are not familiar with the game, simply search for "space invaders" on `http://www.youtube.com/` and play some of the videos that result.

You may make these games single or multiplayer. Regardless of the choice, it must be playable and it must use your client-server architecture. You must have scripted functionality in your game as well. When implementing this second game, pay careful attention to how much reuse you get out of your core engine code. In particular, pay attention to how much functionality you can accomplish using scripting, rather than

native code. To include in your writeup, do a diff between your code for the first game and code for the second game. What percentage of the lines of code are different?

Bubble Shooter

If you choose to implement the Bubble Shooter, you should have a minimum of two colors of bubbles and there should be at least two layers of bubbles upon startup. There's no need to add additional layers during gameplay, but you should lower existing bubbles periodically as they do in a full bubble shooter implementation. To make the game logic easier, feel free to use two connected bubbles of the same color as your criteria for dropping bubbles, rather than the standard three. Your gun should rotate and fire according to your keyboard (or mouse) inputs. Also, make sure your game will end if bubbles touch the bottom of the screen.

Space Invaders

If you choose to implement the Space Invaders game, you should have a minimum of two rows of space ships upon startup. The ships should move across and down the screen as they do in the real game. Your gun turret should move and fire according to your keyboard inputs. You do not need to have defense structures typical of the first three levels on the full space invaders game.

Part 3 (591 required, 481 optional): A Third Game (25 Pts)

For this final part of the assignment you are to implement a third game. You may choose to implement the other recommended game from Part 2, or another game of your choosing. One possibility is the classic game Snake (`http://patorjk.com/games/snake/`). If you choose to implement a different it, it is probably wise to consult with the instructor prior to beginning work on it. The criteria for this part of the assignment are the same as Part 2.

Part 4 (required for all): Reflection (30 pts)

In 2–3 additional pages, please reflect on the overall design of your engine. That is a minimum of 2 *FULL* single-spaced pages. In particular, please explain how successful (or unsuccessful) you were at designing for reuse. What systems did you have to change to get your second game to work? How did you change them? If you were going to start over again to design your game engine again, what would you do differently? What would you do the same?

Writeup

Now that you have designed and implemented a script manager and a second game, write a 1-2 page paper summarizing your design. That is a minimum of 1 *FULL* single-spaced pages. It is **strongly** suggested that you do not limit yourself to only answering the questions posed in this assignment. Think creatively about what you have done. Why did you make the design decisions you made? What did they enable for this assignment? What will they enable in the future? The most successful writeups will contain evidence that you have thought deeply about these decisions and implementations and what they can produce and have gone beyond what is written in the assignment.

As an appendix to your paper, please include all relevant screenshots to support your discussion. The appendix does not count toward your 1-2 page requirement. Your points for the writeup will represent $\frac{1}{2}$ of

the points allocation for each of the above sections (*i.e.*, 20 of the 40 points for the second game section will come from your writeup).

What to Submit

By 4:30pm on 12/1/16, please upload to moodle a .zip file containing your Processing code for all games, one ore more compiled executables of your system, .pdfs of your two writeups, and a README file with instructions on how to compile and run the various versions of your code. If you elected to use the Processing IDE, you may export the Processing project as an executable to include in your .zip file as well, provided you submit executables for all three operating systems (Windows, Linux, OS/X).