Programming Project #2
CpSc 8700: Data-driven Software Development
Computer Science Division, Clemson University
Introduction to Animation with SDL
Brian Malloy, PhD
September 9, 2015

Due Date:

In order to receive credit for this assignment, your solution must meet the requirements specified in this document, and be submitted, using the web `handin` command, by 8 AM, Tuesday, September 22nd, 2015. If you cannot complete the project by the due date, you can receive 90% of the grade if you submit the assignment within a week after the due date.

Specifications Checklist:

1. Write the methods to complete the singleton class Clock. These methods include update, getTicksSinceLastFrame, toggleSloMo, getFps, start, pause, and unpause.

2. Use a *graphics editor*, such as Gimp, Inkscape, Paint, to build a background sprite for your animation; the dimensions of your background sprite should be at least 854 by 480 pixels. Then, use either a Sprite or Frame class from the basic framework to draw the background.

3. Use a *graphics editor* to build a sprite from scratch; You may not use an existing sprite and modify it. Use this sprite in place of the green orb used in the basic framework. Then, modify class Manager so that it animates many sprites, instead of just one sprite. Use *std::vector* or *std::list* to contain the sprites. For example, Figure 1b illustrates 50 yellow stars.

4. Classes IOManager and Gamedata currently use the GoF approach to implement the Singleton design pattern. Modify these two classes so that they use the Meyers approach.

5. Use IOManager to print 4 lines of information on the canvas: (1) The Average frames per second and the seconds in the upper left corner of the screen; (2) The title of your project in the top center of the screen; (3) Your name in the lower left corner of the screen.

6. You should be able to modify aspects of your project by changing values in game.xml and virtually all of the constants used in your animation should come from game.xml.

7. Valgrind should reveal no memory leaks in user code.

8. Make sure that the F4 key will generate 200 frames of animation **with filenames described below**.

9. Fill in the README template, in the project directory, to summarize you and your project.

An illustration of the basic framework code is shown in Figure 1a, and you can find the source at:

`8700-2015assets/projects/2/basic`

Generating Frames for a Movie

Modify the tag, <username> in your XML file so that it matches your Clemson username, rather than mine. Before you submit your project, test the F4 key to make sure that it generates 200 frames and that the naming convention for the frames is:

```
<username>.0000.bmp
...
<username>.0199.bmp
```

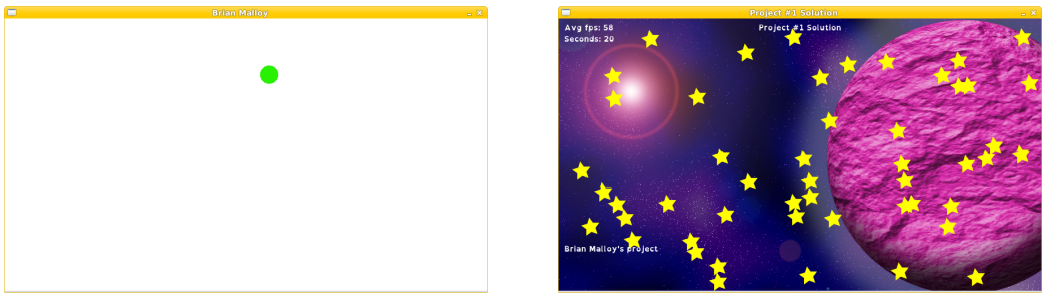
Then, when your project is ready to submit, make it clean, make sure there are no frames in the frames directory, and submit your program using the web handin facility. Our TA will run your program, generate the frames for each of your projects, and then make a movie for your project. Before you submit your project, make clean and delete any frames you may have generated.

Language and API constraints

Of course your assignment must be written using the C^{++}language and you must use the Simple Direct-media Layer (SDL) gaming API. The user code for your solution should be free of memory leaks. Since the language vehicle for the course and the project is C^{++}, you may not use the C language to code your solution to this first project. Obviously, your code may not contain C language constructs such as printf, malloc, free, and #define. I will run valgrind on your executable and there should be no errors and no memory leaks in user code. Also, your solution, when compiled, must not issue any warnings using the following compiler directives: `-W -Wall -Weffc++ -Wextra -pedantic`

Submission:

You should compress the directory that contains your code, media, README, Makefile, using either tar or zip; you may not use rar. Your assignment will be tested on a Linux Ubuntu platform; however, you should test your project on several different platforms and it should be independent of platform and language implementation.



(a) Basic Framework (b) Instructor Soln

Figure 1: The left side of the figure shows the basic framework, the right side shows a project #1 solution.