Programming Project #4
CpSc 416/616: Data-driven 2D Game Development
Computer Science Department
Clemson University
(1) Object Pooling, (2) Projectiles, (3) Collisions,
(4) Explosions, and (5) Sound
Brian Malloy, PhD
March 31, 2014

Due Date:

In order to receive credit for this assignment, your solution must be submitted, using the `handin` command, by 8 AM, Tuesday, April 15th, 2014.

Project Specifications:

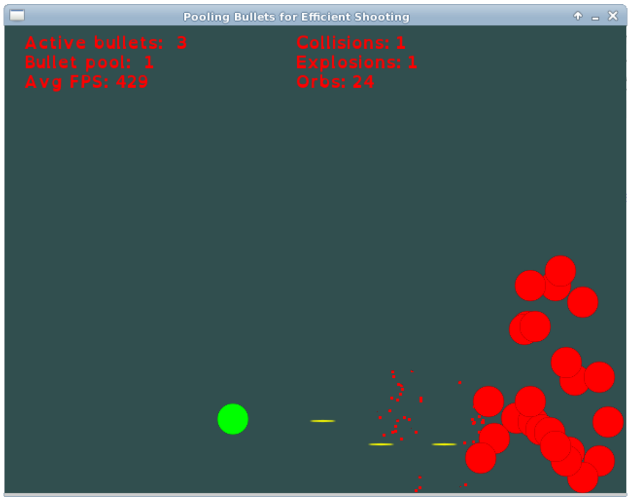
The purpose of this project is to incorporate more action and interaction into your animation so that it includes projectiles, collisions, explosions, music, and sound. Also, you should begin to develop a consistent theme that threads your animation, and to begin to enable the player to reach a conclusion. When considering your project goals, keep in mind that video games are typically built by large teams of programmers, artists, and many others, and that there are only about four weeks remaining in the semester. With this in mind, your goal should be to keep the scope of your project reasonable, and to produce a robust game that reaches a conclusion.

For this project, you are not required to draw your own sprites; however, if you use sprites that you did not build yourself, the sprites must be made available under the open source license and you must specify, in your **ascii** README, the site where you got the sprites. Also, your game must be data driven so that you read game constants from an XML file.

Your goal should be to establish a consistent theme so that your animation might begin to tell a story. The following requirements should guide this fourth project; however, if one or more of the requirements is inconsistent with the theme that you are trying to establish, please stop by during my office hour, or send me an email with times that we might meet to discuss your game idea. However, one requirement that is not negotiable is that your fourth project must demonstrate your use of object or memory pooling. For example, Figure 1 illustrates, in the upper left corner, the instructor's use of an object pool for bullets, with **3** bullets currently active, and **1** bullet currently in the Bullet pool. Notice also in Figure 1 that one collision has been detected and there is one explosion currently active. I will place 3 videos on my web page that will help you with (1) incorporating projectiles, (2) explosions with chunking, and (3) incorporating object pooling.

A summary of the requirements for this project include:

1. Demonstrate object or memory pooling (show sizes in F1/HUD);
2. Your player, after it explodes and chunks/frames are no longer drawn, should re-appear.
3. Projectiles: you don't have to shoot bullets, but you need projectiles;
4. Explosions: you can either use chunks or frames;
5. Collision Detection;
6. Sound and Music;



```
Active bullets:  3          Collisions: 1
Bullet pool:  1             Explosions: 1
Avg FPS: 429                Orbs: 24
```

Figure 1: An illustration of object pooling, with three active bullets and one bullet in the "pool"

7. A well-controlled player object, depth, and a HUD;
8. F1 displays the HUD, with information about the pool(s), the animation, and player movement.

Project Grading:

1. If you are able to submit your well-designed project, using the web handin command, by the due date, and your project meets the above requirements, then your grade begins at 90%.
2. If you are unable to submit by the due date, but you submit your well-designed project, using the web handin command, by 8 AM on Friday, then your grade begins at 75%
3. If you omit the inclusion of an **ascii** README file, 10 points will be deducted from your grade. README files submitted by email will not be accepted.
4. If your README does not include your name and **Clemson** email address, 5 points will be deducted.
5. If you fail to demonstrate object or memory pooling, 20 points will be deducted.
6. If your player fails to re-appear, 10 points will be deducted.
7. If *user code* has leaks, up to 10 points may be deducted.
8. You must submit your complete project, including media, code, and Makefile, in the proper format. If you choose to compress, assume your project is in a directory, dir; then you must zip or tar as follows:

   ```
   zip -r dir.zip dir
   tar zcvf dir.tar.gz dir
   ```

Extra:

To extend your grade: (1) incorporate an Observer, or some other design pattern, into your project; (2) Draw additional sprites that you did not include in project #3; (3) your choice.

Your assignment will be tested on a Linux Ubuntu platform using gcc C^{++}version 4.6.3, or clang version 3.0; however, you should test your project on several different platforms and it should be independent of platform and language implementation.