Weekly log - Jere Ketonen - jxk988

Week 2:

Most of the activity this week has been done in our meetings, that we have held twice a week. In total we have sat down for 6 hours or so, talking about what kind of a project we want to do and coming up with our specifications. Then how we are going to implement the features and class hierarchy and how to organize the team somewhat.

Codewise we all sat and coded a really crude “prototype” of the game. Basically a box that moves and shoots and renders properly. We just wanted to see how to implement that and thought it would be good to do it already to get to see a bit more and see if any new problems arise and the code can be easily reused.

That’s about it for this week. I think the meetings and the coding have taken about the 9 hours a week or whatever it was we were supposed to put into the project.

Week 3:

I was in a high fever most of this week, so I did not really get much work done, as there were other deadlines I had to meet. I mostly just planned for the classes I was supposed to do and we discusses them and how to test them in our weekly meeting with the demonstrator. After this I made them with the functionality I thought they were going to require. The planning part was to think about how to implement them the best with abstract classes and how to get around some of the restriction it imposes on it.

Week 4:

This week I made the abstract superclass’ constructor take a lot of parameters, which I in general do not like to do, for I think it is hard to remember all the things you need to pass into it when you are instantiating it, but I managed to make the constructors of the subclasses fairly simple. I was doing classes for the projectiles and weapons and I think I got their relationship right. One thing I kind of did not like was having to pass the Graphics object to the Weapon.fire() method, but I couldnt really think of a way to avoid it.

I’m pretty sure I will have to tune them a bit at somepoint, when we realize more requirements that are needed from them. I did plan a bit about how to test them with JUnit and next week I will try to implement that. The planning I had done in week 3 helped me somewhat, but I could’ve been a bit more thorough. I also think we need to communicate a bit better as a team, or make a class diagram to show all the relationships between the classes and such, I think it would help a lot with the design.

Week 5:

I wrote the JUnit tests for my classes. It took some fiddling about, but wasn't too hard in the end. While doing the tests I actually came up with better ways to implement somethings, so writing the tests helped in that way aswell. In the end the things I wrote were pretty simple and basic, so I was left with a feeling of uncertainty about if they were really sufficient enough, which they were in the end.

I can see the advantage of using and writing tests even before you write the actual other code, but it will take some getting used to. If I'd manage to incorporate it into my coding routine, I don't think it would cause too much trouble although it would make the amount of work I have to do grow a bit. Hopefully it would be worth it in the end, however.

Week 6:

I went through the existing code we had and cleaned it a bit. Mainly I separated our frame's code into three parts: initialization, logic and rendering to make it a bit more clear and simple. I also implemented a timer for the game, so that it would run at 60 FPS. This will allow us to use ticks to count things and hopefully will help with the networking part aswell.

Eventhough I didnt really produce that much new code, I think the changed I made were good and will help us a lot in the future. Having a clear structure for the code is always a good thing. So now we have a part where we initialize everything. Then on every timer tick we first do the logic part, as in computing new coordinates for projectiles and ships and then in the render part we render the changes to the screen.

Week 7:

I created the background for the game, to give it the illusion that you are moving somewhere. It kind of surprised me how easy it was to do in the end. I was anticipating problems with the stars spawning at the right places and moving correctly. But it was just as easy as making a check to see if the star had reached the top edge of the screen and then just changing its Y-coordinate back to the bottom of the screen and giving it a new random X-coordinate.

Hah. Apparently I didn't think it quite through, since I noticed I had made the stars scroll the wrong way, giving you the illusion of going backwards, fixed that now.

Week 8:

I started making sprites for our ships and projectiles with the help of a friend. It was a bit hard to find a good time to do it, since he is busy and Im pretty hopeless with the stuff without help. In the end I took a coach and went over to his place and we managed to get them done in a day.

The implementation of the graphics went okay. I got the graphics loaded in fine, but I had to change existing collision detection code to get them working properly. I started by just drawing rectangles of the existing bounds around the sprites to see what was going wrong. They

seemed to be a bit in the wrong position and too big, so I just manually started changing the shapes and size of the rectangles to get them to fit the new sprites properly. I also had to make some changes to the implementation of the collision detection methods, as in where it got the bounds from and all that.

Week 9:

I created a fancier mainmenu for us to use, where the user can pick if he wants to play singleplayer or multiplayer and so forth. Basically it is just a black panel where I drew light grey outlines of a rectangle with orange text inside. Using mouse- and mousemotionlisteners I made the outlines turn orange if you hovered over a button. I also made a public static class that was designed to control what state the game was supposed to be in, as in, singleplayer, mainmenu, etc.