EECS 448

Project 2 Battleship Retrospective

https://www.pexels.com/photo/group-hand-fist-bump-1068523/

Team: **Return Awesome**

Ethan Brenner

Kyle Kappes-Sum

Connie Li

Malena Schoeni

Archana Ramakrishnan

Log of all meetings

1. **09/27/2019. In lecture.**
    a. Attendees: Archana Ramakrishnan, Ethan Brenner, Kyle Kappes-Sum, Connie Li.
    b. Brainstormed additional features.
2. **09/28/2019. Learned Hall / voice call.**
    a. Attendees: Archana Ramakrishnan, Ethan Brenner, Kyle Kappes-Sum, Connie Li, Malena Schoeni.
    b. Brainstormed/decided on features to add; divided up work. Started listing functionality to potentially alter or improve.
3. **09/30/2019. In lecture, scrum.**
    a. Attendees: Archana Ramakrishnan, Ethan Brenner, Kyle Kappes-Sum, Connie Li, Malena Schoeni.
    b. Discussed inherited code base & how we plan to build on it. Got Dr. Gibbons' approval to refactor the inherited code base and implement a save game feature and powerups.
4. **09/30/2019. Spahr Library.**
    a. Attendees: Kyle Kappes-Sum, Connie Li.
    b. Pre-meeting meeting. Planned refactoring to increase modularity/extensibility, created class concept diagrams, including tentative planning of how to implement PowerUps and AI classes, all to pitch at the full group meeting.
5. **09/30/2019. Spahr Library.**
    a. Attendees: Ethan Brenner, Kyle Kappes-Sum, Connie Li, Malena Schoeni.
    b. Discussed & confirmed class concepts for the base game + powerups. Decided to implement AI after refactoring is mostly complete. Decided on a few program conventions. Slightly rearranged work assignments:
        i. Kyle, Connie: back-end class refactoring, integration w/ Exec. Then AI when that is done.
        ii. Malena, Ethan: front-end class refactoring (Exec). Then Powerups when that is done.
        iii. Archana: continue working on save game feature.
6. **10/02/2019. In Lecture, scrum.**
    a. Attendees: Ethan Brenner, Kyle Kappes-Sum, Connie Li, Malena Schoeni.
    b. Discussed: Progress on each area in preparation for new integrations.
7. **10/05/2019. LEEP2 G415.**
    a. Attendees: Kyle Kappes-Sum, Connie Li, Malena Schoeni.
    b. Discussed: Existing code and error fixes
        i. Kyle: Future AI integration, and debugging
        ii. Connie: Admiral development and future AI implementation
        iii. Malena: Debugging and Exec development

8. **10/07/2019. In Lecture, scrum.**
    a. Attendees: Archana Ramakrishnan, Ethan Brenner, Kyle Kappes-Sum, Connie Li, Malena Schoeni.
    b. Discussed: Current progresses and issues. Implementation of powerups and saving features given the refactoring that has occured.
9. **10/09/2019. In Lecture, scrum.**
    a. Attendees: Archana Ramakrishnan, Ethan Brenner, Kyle Kappes-Sum, Connie Li, Malena Schoeni.
    b. Discussed: progress on Powerups, general testing, makefiles, bug fixes, plans to work on remaining features.
10. **10/10/2019. Eaton Fishbowl.**
    a. Attendees: Archana Ramakrishnan, Connie Li.
    b. Discussed: Worked on Admiral, Exec, & AI.
11. **10/16/2019. Spahr Library.**
    a. Attendees: Archana Ramakrishnan, Connie Li.
    b. Discussed: Developed testing code and started the save game feature.
12. **10/18/2019. In lecture, scrum.**
    a. Attendees: Archana Ramakrishnan, Ethan Brenner, Connie Li, Malena Schoeni.
    b. Discussed: Recent progress & task divisions going forward. Planned next meeting.
13. **10/18/2019. In lab.**
    a. Attendees: Archana Ramakrishnan, Ethan Brenner, Connie Li, Malena Schoeni.
    b. Discussed: Development of AI and save game code.
14. **10/19/2019. Spahr Library.**
    a. Attendees: Archana Ramakrishnan, Ethan Brenner, Kyle Kappes-Sum, Connie Li, Malena Schoeni.
    b. Discussed: Assed current progress and issues. Developed testing code, save game, and AI.

How work was split between teammates:

We realized early on that we would have to do a massive refactor and learned everyone would be busy for Fall Break/conferences. Based on all of this information we were able to adapt our plan from the last battleship project to have a class for Ship, Grid, Admiral, AI, PowerUps, and Executive. We then determined who would work on each class: Connie worked on Ship, Kyle worked on Grid, Connie worked on Admiral, Malena worked on AI,Malena and Ethan worked on PowerUps, and Archana, Malena, Ethan, and Connie all worked on Executive.

We ended up replacing the AI class with three separate AI classes(EasyAI, MedAI, and HardAI), all of which inherited from Admiral. Additionally, we were able to use a vector to the ships instead of creating a fleet class like last time. This time we also performed smaller tests on a few of the individual classes. As a result, Kyle created a Testing class to manage all of our tests and a Test class that included a few basic items that could be inherited for each test class

we wrote. We had to shift around who worked on each class to account for availability issues and the additional time we spent working on the refactor.

Challenges and how they were overcome or dealt with:

1. Team Challenges:
    a. **The Refactor Challenge:** The code base we inherited greatly lacked extensibility and after an early examination, it was concluded that major refactoring was needed (it lacked any place to include AI, a modular way to add our power up and save features, or even more than two classes). Two team members were tasked with focusing on refactoring the code base only. Everyone else initially had to contribute to the refactor before they could start developing the new features. We had to determine, in the planning stage, the development pace so that the new features had existing code to use. We were able to develop everything we hoped to but it wouldn't have been possible to do in two weeks.
    b. **OOO:** The timing of this project also overlapped with a conference and fall break which reduced our ability to meet as a team. It also meant that we had to strongly consider availability when we determined what files we were working on.
    c. **Incompatible Operating Systems:** Unlike our prior project we had to consider that half of our team developed on Macs and the other half developed on PCs. As a result, we had to use different makefiles and avoid committing them. We introduced a more complex makefile that was easy to use, shorter, and more flexible to our needs. We also had to deal with errors that would only occur on one operating system.
2. Archana:
    a. **Playing catch-up and understanding the code base:** I missed almost a week of school because of the Grace Hopper Celebration. By the time I was back, a lot of code had already been written and the class relationships took me a while to understand. I attempted to write code for the AI, but I was very unsure of how I would tie into other classes like grid, ship, admiral etc. I sat down with Connie and expressed my worries. We talked about how I might be able to work on save game instead because it is a more independent part of the project and I had an intuitive idea of how to start.
    b. **Problems with reloading the game after reading from file:** I fumbled quite a bit trying to understand how to reconstruct the game after reading from file. However, it helped me get a holistic understanding of how all the classes tie in together and how the gameplay functions were implemented. Connie and Ethan helped me monumentally throughout implementation and debugging, and Kyle gave me many helpful pieces of advice and helper functions for the Grid class.
3. Malena:
    a. **OOO:** I left early for fall break for a conference in Pennsylvania and I didn't have a lot of time over break to work on it. Something I had troubles with a lot was

placing ships for AI. I was also working a bit on lots of classes and I feel like I was all over the place sometimes.

4. Kyle:
    a. **Testing Options:** In order to test out portions of Grid while the rest of Admiral and Executive were still being written, I wrote a separate test class. I designed it with some basic items for testing so that all of our class specific tests could inherit from it. I also design a Testing class to manage it all. I had originally looked into storing it all in a separate file, however this caused issues with the makefile. As a result I looked into the possibility of using a recursive make however, I could not get it to clean recursively. I did learn how to make a simpler easier to use makefile which we implemented allowing our future developments to be added really easily. I altered the main to allow for a -t flag to run the Testing file instead. Between the testing content and the alternative makefile we were able to save time and test our errors more effectively.
5. Connie:
    a. **Refactoring is hard:** I should have tried to reference the legacy code more to minimize how much I had to do when refactoring. I learned to not try to take on more work than I can do in a timely manner.
6. Ethan:
    a. **Error Checking (on Windows):** Prior to this project, I had never coded/compiled c++ on a windows machine(always preferring to do so on Linux). On multiple occasions I ran into issues that were either 1)local to windows, fine on Linux or 2) seg faults (which on windows, didn't report any error, the program just ended). I did a lot of checking back and forth between the cycle servers and my local machine to narrow down these issues, often with the help of others.

Features that didn't make the demo version:

1. **Saving AI:** Adding the ability to save a game played with both another player and the AI
2. **Torpedo power-up:** This was a power-up that we really would have love to implement. When you hit a ship with a torpedo, no matter what tile it is on, it would sink the entire ship.

Known issues: (We ran out of time to troubleshoot and fix these.)

1. **Torpedo power up:** Upon calling the function, it threw an "out of range" error that we were unable to find before the deadline

Retrospective on what the team would have done differently

1. **Better time management:** A lot of time was spent planning and implementing the individual classes. We only began to put together a functional gameplay towards the end of the project timeline. Testing and debugging should have been done in parallel to the implementation, but was mostly done close to the code freeze, which ended up being a stressful.

2. **Setting more achievable goals:** We should have been a little less ambitious with how much we planned to refactor the original code base. Extensibility to implement an AI and the power-ups would have been difficult with the inherited code. However, the refactoring took much more time and effort that we expected.
3. **Prioritize tasks:** We started off on the AI part and powerups towards the very end. We should have started early as these very higher priority. Most of the time was instead spent in refactoring the project to make it best suited to how we wanted to extend it.
4. **Better Object Orientedness:** The save game feature was implemented closer to the end and was thrown into the Executive file as 4 functions. This resulted in Executive having hundreds of lines of code and prevented us from abiding to best modularity practices. We would have liked to implement it as a separate class.

Bloopers :)

- "DID YOU VALGRIND IT?" -- Ethan
    - “Things CS majors say” -- Archana
- "So I accidently made it correctly? Cool!" --Kyle
- *plaintively* "Did you mess up my UberCommander?" --Ethan
- "NO, Illegal instruction #4" --Connie