Retrospective Write Up

Hirsh Guha, Ryan Pope, Jordan Love, John Quitno, and Giang Nguyen

Meeting Log

First Meeting (2/12/19):

1. Location: LEEP2 Atrium
2. Members Attending: Giang, Jordan, Hirsh, Ryan, John
3. Outcomes: Divided up work, decided approach, set deadlines

Second Meeting (2/13/19):

1. Location: LEEP2 Study Room
2. Members Attending: Giang, Jordan, Hirsh, Ryan, John
3. Outcomes: Work day, finished the HTML/CSS, worked on the Gameplay functionality and finalized classes for Tiles and Boards

Third Meeting (2/15/19):

1. Location: LEEP2 Study Room
2. Members Attending: Giang, Jordan, Hirsh, Ryan, John
3. Outcomes: Since class was cancelled, we used the time as a workday in order to fix as many bugs and errors as possible.

Fourth Meeting (2/17/19):

1. Location: Google Hangouts
2. Members Attending: Giang, Jordan, Hirsh, Ryan, John
3. Outcomes: Finalized the game, and began intense stress testing for edge or corner cases, to ensure the game was ready. Also began documentation and final write ups.

Process

We went into this processing knowing that Javascript was going to be the language of choice, so our logical first step was deciding exactly how we wanted to use a language like that, and whether we wanted to use any libraries or frameworks. After discussing, we decided against any framework, proceeding with pure Javascript to write the script.

The next step was breaking the game into classes in an intuitive way in order to ensure the whole thing was object oriented, and easily extendable. It was decided that we would break the project up by a Tile class(each individual cell had to hold information), a Board(this created a grid of Tiles, calculated the value of every number, and randomly placed mines), and Gameplay(Along with helper methods, it primarily contained the code for right and left clicking, along with updating the HTML).

How Work Was Split Up

We began as such:

- **Hirsh** - Wrote the HTML and CSS
- **Jordan** - Wrote the Tiles Class
- **Ryan** - Wrote the Board Class
- **Giang** - Wrote the recursive reveal function
- **John** - Wrote the Gameplay class EXCEPT for the recursive reveal function

After this, error handling, bug fixes, documentation, and additional features was all that was left, and we split the work like so:

- **Hirsh** - Wrote the right click, the communication between JS and HTML, and added styling to cells

- **Jordan** - Generated the documentation
- **Ryan** - Fixed errors until existing functionality was usable
- **Giang** - Finished the left click, and added error handling
- **John** - Added error handling behaviour, generated documentation

Challenges Faced

One of the main challenges we faced was getting multiple classes to work together. Our understanding of javascript as an object oriented language was still in its infancy, so we worried whether it was even possible. This challenge was quickly conquered, however, as our primary issue was the order of the script tags in HTML, as Gameplay referenced Board, which referenced tile, and the script tags were in the opposite order.

Another challenge was splitting the work. With 5 people, it was difficult to divide the work into even chunks, while making sure everyone did something significant. Of course, this was overcome, but not without discussion and breaking our approach to the game down into the smallest units possible.

Features that didn’t make the demo

In the end, we did a great job of setting realistic expectations of what we would get done by the deadline, we there were no features we hoped to include that we were unable to.

In retrospect, we might have been able to add a timer, as well as a count of how many mines are remaining on screen. However, both of these things were not requirements, and were not priorities.

What we could have done different

One thing we considered was using different JS technologies such as ReactJS, NodeJS, or JQuery which would be a good tool to help with developing the interface, and perhaps given as methods that would have simplified the backend construction. However, given the size of the project, we felt it was unnecessary at this time.

There were also many shortcuts for efficiency we could have taken. We could have written it all in one class and JS file, and saved on the need for setters, getters, class communication pieces, and extra arrays. Perhaps the code would have loaded faster(which is not particularly relevant for this project). However, the fact we did separate the classes allows for greater extendability, and readability.