BATTLESHIP WARS

Introduction

For the final assignment, I have created a game called Battleship War. The game was inspired by the all-time classic battleship board game which I played often in the past. In Battleship Wars, players were required to set their ships on the grid, and try shoot down their opponent's hidden ship. Among the many other ideas I had, I decided to go with this because it would allow me to challenge myself to make use of what I have learnt to make a different game. I was able to include most of what was taught in class (i.e. Raphael, Functions, Loops, Styling, Media Files, etc) and exercise self-learning, thinking, and exploring to figure out what needs to be done to create my game and its features.

The Process

For the creation of my game, I modified and edited a previous HTML layout I had previously used. However for the JavaScript coding, almost all codes were newly written. CSS was also used for various design aspects for my game. From the creation of my game, I was able to appreciate many of the things I have learnt in class, and challenge myself further to develop something better. Some of the key highlights and issues I had and how I manage to solve them will be discussed below.

The use of arrays was a key crucial requirement for making my game work together with the functions. To create the grids, I had used an array and a "nested for loop" which I learnt more about online on W3 (http://www.w3schools.com/jsref/jsref_for.asp). Also, instead of manually setting numbers inside the loop, I was able to create a variable that contained the number of grids I want, and do mathematical formulas that will automatically create grids for me. For example, in my coding, I set the total number of grids to be at 36. If I were to input any number (that could be rooted), the loop would automatically create the grid for me. This allows me the opportunity to make my game's grid scale dynamically, instead of numbers that are set manually. Apart from that, the dynamic variables (e.g. rootOftotalRect) had helped me in other portions of coding such as for checks, grid setup, attribute setting, and more that will be discussed below.

Moving on to gameplay, instead of having players set ships that are only 1 grid each, I have decided for players to set ships that will take 3 grids, 2 grids and 1 grid. To me, this was really tough figuring out how to plan and implement, but I have decided to do it in order to make the game more fun, and less of a chance game (of randomly selecting one point). After some considerable time planning, I was able to make it happen. The first biggest problem was deciding how to add eventListeners given that in battleship there are so many modes (e.g. set ship, battle). I did some research, and found out about removeEventListeners. However, I have decided not to use it. Instead, I used .hide() and a "Grid Blocker" (which is a Raphael object over the grids) that will prevent mousevents on the respective grids during non-playing turn. In my opinion, I feel that I do not need to remove the EventListeners since I have also incorporated "if, else if" statements to check for the respective mouse events. Examples include checking game mode, if the grid were previously marked, and if it satisfies filling the subsequently grids (if the user was setting 3 ships or 2 ships). Another problem was enabling users to switch orientation. For this, I had spent a considerable amount of time to make it happen. To allow ships to be set oriented from left to right, modulus was used. To check ships oriented top to bottom, I had to use the || (or) operator and check for the various conditions. If it satisfies, the ships will have its attributes changed depending on user action through a function. For example, if the user is setting 3 ships, ships on the right (and +1 and +2) will have its fill changed if it is in a left right orientation, or ships below (and +rootOftotalRect and +(2* rootOftotalRect)) if in a top down orientation. In short, a lot of "if, else if, else" statements were frequently used after much planning and the above functions

was only possible due to the use of arrays in creating of the grid (e.g. +1 = right side of the current grid). In addition, certain functions (such as setting attribute of grid depending on game mode) were also further split into various portions so that it can be reused accordingly depending on game mode.

Moving on, for users to enjoy the game, I have decided to employ both single player (vs computer) and 2 player (2 player local) mode. This definitely raised the difficulty of my code as I had to cater for both modes (more if statements), as well as include an artificial intelligence (AI) that could "play" with the user. To design for the AI, I had to consider both the AI setting the ships, as well as shooting the player's ship. To make this happen, I had the computer randomly generates a number from a range (which is the grid's array number), and set/shoot ships depending on the game mode. If the number was previously picked, the function would go into a loop until a new number is suitable. I also had to "force/set" the variable mouseClick to be true, so that the function could run for the AI to set ships. Lastly, I included a reveal ships function for players to see un-hit hidden ship, to cater for "fairness".

Media Files

Apart from the coding, I had also made use of images and audio for presentation, design, and effect. Media files were sourced online mainly from Flaticon.com and Freesound.org and I had utilised Adobe Photoshop and Audacity to edit the respective files to suit my game. For example, I had to "cut out" the shape for the respective top and bottom piece of "curtain" that would open and close through the game using Raphael's animation. The audio clip was also trimmed and edited to ensure synchronisation between animation of the "curtain" and sound. In the coding of the mechanics of the curtain, there were also other considerations for me. I had to plan which buttons/images variable were to be defined in the code before and after the curtain. Next, I also had to implement setTimeout. This is done so as to only allow buttons to appear/disappear after the curtain has come down/up. This allows for a better and smooth animation that makes sense. In addition, I had also decided to create buttons/instructions out of image files instead of creating them in Raphael. This allowed me more flexibility and creativity in the design of my game. Additional, I had also created a checkbox in HTML to allow users to mute the audio. I came across learning more about the checkbox in W3 (http://www.w3schools.com/jsref/prop_checkbox_checked.asp) and managed to link it to my Javascript through DOM to mute my audio files – which some players may find audio irritating.

Conclusion and Future Improvements

With the completion of my game, there are certain areas which I think could be improved in the future. For example instead of using the many "if, else" statements, I have come across on the internet the usage of "Switches" after I was almost completed with my coding. This could be implemented in my code for better efficiency. Next, to make the game more challenging for single players, I could design a smarter AI that takes into consideration the next probable grid to select. Currently, it picks the grid on random. With a better designed AI (for increased difficulty), I would allow it to detect that when it hits a ship, it would continue hitting the grids that are next to it, in hope of hitting ships that takes 2 or 3 grids. This could be done through arrays whereby numbers could be added to the array, and the computer would take a random number out from the array for its next shoot. One last area for improvement is the creation of a loading screen. This would allow users with slow bandwidth to preload all images and audio files. This then allows them a better experience while playing the game.

Overall, I had a great experience in making this game and it was satisfying to see my game work after putting in a lot of effort thinking, planning and learning new stuff and putting everything together into one piece for my game, Battleship Wars.