Part 2: Reflection

● Did you face any technical challenges while writing the game code? Were there any JavaScript optimizations you found useful while writing the game?

I feel this game was a balance between learning about the devkit, coding a functional game and making good design decisions. Trying to balance that was a challenge. The whack-a-mole game was a good starting point but I wished I had discovered the larger games earlier as a template for learning. I find playing the game and then checking out the source to be quicker way for me to learn than just reading documentation. Also I was not able to just Google common devkit problems for a quick solution.

I tried to keep it simple and design the core game with a back-end model for the logic and a separate front view for the graphics. This extraction helped me keep the code cleaner and fix bugs easier. I used two dimensional arrays for the tiles. For board collision detection I used simple maths as a more efficient way to get the selected tile as opposed to something like attaching listeners. I also did not use any tweens for the tiles although the are shifting from point A to B. Again just using object += speed is better for performance than a bunch of tweens.

● How did you polish the game? What did you add to make the game more visually appealing or to make the gameplay feel right?

I decided to spend most of my time making the game as solid and as bug free as possible. I really tried to keep it simple so I used timer and high score element so I guess I was going for a more Bejewelled than Candy Crush. Also I added some cute Zoo Keeper style animals for visual eye candy. I also tried to game feel about right for beginners. I think 5 tiles matching is too easy personally and 6 or even 7 tiles is about right but casual game should often start easy so players are initially not turned off.

I also made it so the board refreshed when there were no more moves. This is better than just letting the player run out of moves (which is a bug) but a better solution would be to not allow the board to run out of moves in the first place by checking new the tiles being added into the game.

● Did you find anything lacking with the devkit? How would you improve our devkit to make game development easier? What did you like about it?

I think some more code examples and some more demos with viewable source code would also be handy. I guess that will come with time though. I think the documentation in general was quite thorough and the examples were enough. In terms of setting up the devkit it might be possible for people run in trouble with npm stuff as it can be more troublesome for windows especially if you do not read their websites documentation. I think the devkit is just what is needed for HTML5 as a tool for rapid cross platform game development. So I think it could combine the best parts of HTML5 and bridge the gap between what is lacks compared to other technologies.

● Did you have any plans to improve the game further?

I would like to make the selector respond to mouse/finger drag. At the moment it is only updating on mouse/finger down. Most Match 3 players play by swiping the screen. I would also like to add special effects such as tiles disappearing and a sparkle effect or something! The game would benefit from quality sound effects and music. I would like to have pick ups and bonus items also. Most people would expect there to be some kind of dialogue or story idea and It would benefit from levels or some kind of gacha element to keep players coming back. Also the game should be playable on any platform.

● Other thoughts on game development, our tech, or HTML5 in general?

I think HTML5 games have great potential. It is the technology of the past, present and future. Coming from a Flash games background I am excited about HTML5 as an open platform for casual fun addictive games that can be played online and anywhere. I always loved free casual games so I hope HTML5 really fulfill its promise to be just that and more. Although technologies such as Unity are cool and makes rapid prototyping of games easy it has recently has problems being cross platform. It used to be great for game jams for example easily viewable in the Browser.

In terms of game development am quite interested in designing Match 3 games but with extra features built into them. Recently I was thinking a lot about Match 3 game ideas. In particular I am interested in creating Match games with more action elements. Either incorporating a game above the Match board or as a tile game without the obvious board that consists of a world where matching is the core mechanic I don't like the idea of players mindlessly matching away feeling disconnected from what is happening in a game so I would like to incorporate a more addictive arcade feel to the Match 3 genre and I have some ideas for that...