Extirswipe:

* Start Date:

October 2015

* Completion Date:

January 2016

* Languages & Technologies Used:

Swift, iOS

* Inspiration:

I was simply messing around with Xcode to see what I could do with it. One thing led to another, and a few months later, I had a game. (And learned how to code in Swift along the way).

* Description:

Extirswipe is a speed based matching game where you have a limited amount of time to match objects of certain categories (shapes, colors, or numbers).

* Development & Challenges:

Developing Extirswipe was particularly challenging because Swift was the first Language I had taken upon myself to learn that had significantly different syntax than Java. Instead of taking an online course in Swift or going through a tutorial, I decided to tackle the learning head on by programming what I could and researching whatever I didn’t know the syntax for. I feel like this was a much more rewarding experience than if I had just taken an online Swift course instead.

Shortcomings:

As an app, this had many shortcomings. First of all, since I was using this as more of a way to teach myself a new programming language rather than an actual app, I wasn’t following any particular design plan. This lead to an array of problems such as the game being too difficult to intuitively understand how to play, and for the game itself to lack elements that would make it interesting enough for a user to keep coming back to play more.

* External links:

Github:

BUG:

* Start Date:

August 2016

* Completion Date: B.U.G. is never complete >:)
* Languages & Technologies Used:

ATmega32u4 (Adafruit 32u4 Bluefruit), Bluetooth LE, Motors & Drivers, Ultrasonic (obstacle avoidance), camera, piezo element, C++/Arduino

* Inspiration:

Long long ago (summer of 2015), I was on the 3rd floor of an electronics mall in Taipei, Taiwan when I walk into a somewhat sketchy looking shop. To my delight, they were selling a whole bunch of cool stuff (Arduinos, Raspberry Pis, robots, etc.). After looking around a bit my eyes settled upon a plastic chassis (I had attempted hacking a really old motorized toy I had with my Arduino before with limited success). I bought the chassis with the intent of getting my Arduino to control it someday.

Toward the end of summer of 2016 (about a year later), I was bored and decided to build a very small box like robot using the chassis that I had bought. Instead of using the full chassis, I decided just to glue the gearbox motors together with a piece a cardboard resulting in a much smaller size.

I would often bring this contraption to school with me and a friend started calling it my little bug. Hence, as a joke, I named it B.U.G., the Basically Useless Gearbox.

* Description:

The Basically Useless Gearbox, or B.U.G. for short, is a robot that can dance, wander around aimlessly whilst avoiding obstacles, take pictures, play music, or follow your ever command via the Adafruit app.

* Development & Challenges:

One of the main challenges with B.U.G. is its size. I can attach so many things to it before it becomes impractical. This made powering BUG a challenge. At first I used 4 triple As and tried a few different placements and battery packs, after a time it became clear 4 triple As and their 1000 mah capacity would give my 500ma power hungry robot very little battery life. I eventually switched to 4 double As and I plan to eventually switch to rechargeable li-ion.

Shortcomings:

Not too many shortcomings in terms of living up to its name I’m afraid. However, if I were to choose something, I think that B.U.G.’s size is both a redeeming and restricting feature. It limits what I can add on to it while also challenging me to see what I can do on such a small platform.

External links:

Github:

Seat a Student:

* Start Date:

Winter 2015

* Completion Date:

Summer 2016

* Languages & Technologies Used:

Java, Machine Learning/Genetic Algorithm

* Inspiration:

One day I watched my high school computer science teacher as she slogged through completing seating charts for each of her 5 classes. She would always pass around half sheets asking students about their seating preferences (who they did/didn’t want to sit next to, preference to sit closer to windows or front of room, etc.). It seemed like such a repetitive and arduous task and it made me wonder if there was a way to get a computer to do it…

The idea had slipped out of mind for a few months until I learned about a data structure called a graph. The graph instantly reminded me of the seat a student problem: each student could be a node while each listed preference for another student would be an edge connecting the two students! I got to work coding and white boarding and I soon realized that although I had a powerful way of representing the problem

* Description:

Extirswipe is a speed based matching game where you have a limited amount of time to match objects of certain categories (shapes, colors, or numbers).

* Development & Challenges:

Developing Extirswipe was particularly challenging because Swift was the first Language I had taken upon myself to learn that had significantly different syntax than Java. Instead of taking an online course in Swift or going through a tutorial, I decided to tackle the learning head on by programming what I could and researching whatever I didn’t know the syntax for. I feel like this was a much more rewarding experience than if I had just taken an online Swift course instead.

Shortcomings:

Awful, God awful UI. (Actually UI was non-existent)

* External links:

Github: