Jessie Jalca

Creative Coding

Katherine Bennett

December 14, 2018

Self-Assessment

Personally, the way I best learned syntax, structure, and programming concepts was by spending the time to actually program. This is especially true when it comes to syntax and structure. I don’t remember a time where I would sit down and try to memorize syntax and structure; instead, once I saw something new, I would immediately start trying to use it myself and play around with it. For concepts, I would usually take a moment to digest the new information and try to simplify it in my mind or create an analogy for how it might work, but I always went to test it out as soon as I could to see whether I got it or not. I’m a hands-on learner, and I think this shows most in my switch to P5.js.

In my last self-assessment, I had mentioned how I felt that I was a lot more comfortable with Processing than P5.js, and how I was having some trouble with P5. Since then, I’ve worked a lot more with it, and I think it really showed. Looking back, I’m really excited that I was able to pick it up and use it for my final project. Sketch 4 was the first assignment that I really tried to use P5, and I remember struggling with it a lot to figure out that jump. Sketch 5 was equally as difficult, but by the end of it, I felt a lot more confident with the way P5 worked, and I chose to use it for my final project, which I think really sealed the deal on my confidence with it now. At this point, I think I even prefer it over Processing. Using the web developer tool on Firefox really helps even more so I think than the Processing console does. And I remember that I had a few weird glitches on my Project 1, which I haven’t noticed any of the same issues with P5, so I think it flows a little better.

I like the organization of OOP, in comparison to Procedural Programming. I think the structure took some getting used to, but for the most part I think it made a lot of sense. I’d like to get better at it, because I think it would help in cleaning up my code and making everything much more organized and easier to fix later, but sometimes I think it’s easier to start out with Procedural Programming. I think mapping things out ahead of time would help to make the most of OOP, without having to code by Procedural Programming and then having to switch it over. I think sometimes, however, regardless of planning things out, sometimes it is better to work with Procedural Programming, or to mix the two together, like I did in my final project. Overall, though, I think the thing I needed to get better at is planning my code before I start coding, breaking everything down with pseudocode first. I don’t think I took nearly as much advantage of using pseudocode as I should’ve, and I think this became most evident to me while I was working on my final project.

In my final project, I actually learned a lot about learning from sample code. I used p5.play for the mechanics of my game, something that was not fully part of the class, so I was mostly on my own for figuring it out in the sense that there were no textbooks or videos or anything handed to me to help guide me through it. I really learned how to look through someone else’s code, take it apart, and piece things together about the way the library worked, even without a lot of comments in the code to fully explain everything. Now, at the end of the project, I feel like I really have a pretty good understanding of how it works. At first, I got a lot of bugs really often, but towards the end of the project, it started happening less and less. It took me a little while, but I finally remembered the web developer tool that was shown in class on Chrome, and it occurred to me that Firefox (the browser I used to test my code) would likely have one too. I quickly realized how much of a help it was; the tool was really great about pointing me towards the main problems in my code 90% of the time, easy. There were still a few things that I’d like to sort out and new ideas that I’d like to implement in the game, but for the most part, I’m pretty happy with the way it turned out. I just wish I’d had more time to work on it so that I could really get even further into it before it was due. Learning to debug a larger program—and even just creating a larger program—also helped me much better understand the importance of being aware of the way I structure my code. I experienced firsthand how much something can slow performance if it’s iterated in the wrong place.

I know I’m definitely going to keep programming. There’s something about it that I just really enjoy. The biggest thing is probably the creativity that goes behind it, particularly the sort of limitless creativity that doesn’t necessarily carry in every form of art; I like the idea of using it to bring art and craft together with technology and making really amazing projects. I’ve been thinking about going into a lot of programming-heavy fields, like game and UX design/development. This upcoming semester, I’ll be taking a leave of absence to work full-time catch up with finances, but I plan to learn HTML and CSS, as well as learn either a bit of game design with Unity or animation with After Effects.