Teaching Programming with GameMaker Drag-n-Drop and Arduino Text-Based Code

Troy Hill
Winston Salem State University
601 S MLK JR Drive
Winston Salem, NC
919-725-6727
Thill108@rams.wssu.edu

Dr. Christna Gardner-McCune Phd
University of Florida
Gainesville, FL 32611
864-656-5862
gmccune@ufl.edu

ABSTRACT

This research looks at the possibilities of using a drag-n-drop programming language, in this instance, GameMaker to help teach middle school and high school students how to program. Drag-n-Drop programming is used to teach the basic concepts of programming like variables, if-statements, algorithms and more. The fact that they are learning to make a game at the same time is what engages them to want to learn how to program and what helps instill the concepts they learn for the long term. Learning the concepts of programming in general is what allows them to make an easy transition into text based programming. Arduino is the tool that we used to introduce the Simple C programming language. We chose Arduino because it is also an engaging tool that can be used to teach text-based programming. The Arduino Esplora is a gamepad controller that we used to work along side the games that the students made in GameMaker. Working hand and hand, the drag-n-drop concepts learned in GameMaker transitioned directly into the text-based programming of Arduino while keeping the young students engaged and interested all the while.

Categories and Subject Descriptors

General Terms

1. INTRODUCTION

Programming for a beginner can be an overwhelming experience. The concepts, syntax, different languages, and different applications can intimidate those who don't know where to start but just want to start. Another important factor comes into focus when you zero in on teaching the middle and high school age groups how to program. You not only have to focus on teaching in a clear and effective manner, but you have to keep their attention as well. In a college setting you are expected to pay attention regardless of teaching style and technique. Try the same straight forward technique with a younger audience and you quickly lose engagement from many if not all the students. Drag-n-Drop programming like that offered by GameMaker, may be the saving grace in a field where engaging younger students in a text-based programming language challenges both student and teacher.

2. METHOD

The students were taught in a classroom setting with desks, laptops, a classroom size white board and projector. Short lessons made up the majority of the group learning while hands on, one on one, teaching was our primary tool overall. Throughout all learning processes, programming concepts were reinforced heavily to encourage them to remember them long after the camp had ended. Various surveys were given for us to get a good understanding of their different levels of experience as well as the progress and understanding along the way. After one week of rigorous learning and programming the students completed and presented their games to the public.

3. CONCLUSION

When the high school students learned about our camp they had three basic ideas of what they wanted. A) They wanted to advance their programming skills in general. B) They wanted to advance or learn Game Maker skills. C) They just wanted to make their own games. One of the first questions we asked them was how did they prefer to learn. The majority 27% preferred using logic, reasoning and systems, 19% preferred words, speech and writing, and 17% preferred pictures and images visual understanding. After having a feel for how they liked to learn we next wanted to know what they already knew. 68% of the students already had exposure to programming previously with 38% of those students actively doing some form of coding on a daily basis. Additionally, 63% of the students had had either previous Game Maker experience or used another game design software before coming to the camp. This group of high school students were all motivated to learn more about computer science, many of them, determined to major in the field.

When considering how to give instruction, 24% of the students liked to learn from hands-on activities and 22% liked to read instructions before starting a project. Our instruction capitalized on those preferences. Teaching the students would consist of a half an hour lesson and then we would allow them to work in the groups. While working in their groups the staff would walk around and give hands-on help to specific problems and challenges. When teaching students that range in programming skill, from none at all to quite a lot, it's important

to maintain a balance between not leaving the beginner programmers behind, while still challenging the advanced programmers. The 30 minute lesson at the beginning was our way of getting the beginners up to speed on what they would need to progress, and the hands on help was helpful to everyone, especially the advanced programmers for helping them to progress or think computationally about their challenges. The most challenging aspect of the program was getting the students to learn and retain this information in one week. Many of the students went from no previous programming experience to a fully functional game in 5 days. I believe having a project that you can take home, show people, and continue to develop is what makes the difference. Creating a game is why the students engagingly learn the programming concepts we teach them, and the desire to improve and create more is what inspires them to retain and build upon those concepts.

There were four major things that the students stated they took away from the camp. A) How to code without any previous knowledge. B) How to code in Game Maker. C) How to use Arduino. D) Logic and Computational thinking skills. One student stated "I have never done any kind of coding prior to this program, and now I know the basics to coding in a text based language and am confident in my ability to code using Game Maker. Additionally, I have had the opportunity to learn how code is structured and how to make different aspects of a program work together to complete a task." In 5 days this student without any prior programming experience learned programming concepts in a drag-and-drop based language and applied to those concepts and as a result was able to get a basic understanding of the Arduino text based language. Applying programming concepts from a drag-and-drop platform to a text based language was the core of my research and is what reinforced constantly through the camp. The main concepts I wanted them to learn were variables, if-statements, and simple algorithms. After the camp we asked students what concepts they were able to take from Game Maker and apply to Arduino. One student stated, "I learned several concepts from these: I learned about variables, data types (from Arduino), algorithms and sprite animation." Many of the other students included those concepts, as well as if-statements and Boolean variables.

Debugging was also a huge factor in learning to program. For the beginner students it was a wakeup call for the true challenges of programming, and for the advanced students, an opportunity to gain a better understanding on the proper way to debug code. Although challenging, debugging didn't discourage any of the students and they were all able to overcome their debugging challenges. In describing how debugging has or has not helped with confidence explaining their code, one student stated, "Debugging definitely helps to explain the code. In order to debug, you have to isolate: a) the problem, b) what causes the problem, and c) steps to fix the problem. And often times I tried 3 or 4 methods to fix it, before finally making bugs go away. Since so much time and effort has to be invested in debugging, you learn exactly what your code is doing, why it does what it does, and why you can't do it another way." Another angle to look at was the difference in debugging in the drag-and- drop language of Game Maker vs debugging in the text based Arduino. The major response I expected was for them to focus more on the conceptual errors that they would come across in Game Maker, while Arduino would consist of the syntax errors from having to type out the code themselves. One student described his experience, "While debugging in Game Maker, it had to do with trying to manipulate the actions to change the errors. For instance, we had an error where we had to make Game Maker recognize we were trying to create and define a variable, but unlike Arduino, we didn't have to go into the hard code itself." As far as the difficulty of debugging in Game Maker compared to Arduino, the students were split nearly even. I found this interesting as I had expected them to consider Arduino much more of a challenge to debug due to the text code. However after learning the concepts in Game Maker and recognizing those concepts in Arduino, many students reported that debugging in Arduino only came down to minor syntax errors. "Debugging in Arduino was extremely easy because it was just syntax errors. Debugging in Game Maker however was harder because I had to figure out why my code was wrong and how to fix it."

We also wanted to know overall which type of programming language did they enjoy using more between drag-and-drop and text based programming. I expected the majority of the students to go with drag-and-drop, due to the fact that it's easier to learn and use many students agreed but there were also some interesting disagreements with my expectation. A good example of a student stating that drag-and-drop was more enjoyable is, "I prefer drag-and-drop programming, because it is easier to use and there is less room to make mistakes, which makes debugging easier." While I agree with that statement I also find the counter argument just as true. When another student was asked what language they prefer, there response was, "Text based coding. It is what real computer science majors use and has more educational value than drag and drop programming." This reinforces that fact that for beginners, a drag-and-drop learning system is advantageous for its ease of use and understanding. However once those programming concepts are understood and the programmer wants to spread their wings a little, text based programming is a preferable route, especially when education and industry become factors. Therefore by connecting the two, you supply a bridge between those who want to learn and those who want to improve or progress their learning. We directly asked students to tell us how they utilized different concepts in their programming to see if they were able to learn the concepts and

apply them. When asked how the students used variables in their program, one student replied, “To store values, like whether or not the player has a certain weapon, and how much health the player has.” The most important part of that statement is that the student understands that variable’s main purpose is to store values, accompanied by the fact that they were able to use this concept to program parts of their games. When asked how if-statements where used in their games, one student replied, “I used if-then statements in correlation with variables, assigning events and actions to varying sets of parameters.” This is an example of a student combining two concepts, variables and if-statements, and understand how to combine those concepts with game programing concepts to progress their game.

4. REFERENCES

[1] Brock, J Dean, Rebecca F Bruce, and Susan L Reiser. "USING ARDUINO FOR INTRODUCTORY PROGRAMMING COURSES: A TUTORIAL*." 129-30. *SIGCSE*. Web.

[2] C. Y. Cheung, Joey, Grace Ngai, Stephen C. F. Chan, and Winnie W.Y. Lau. "Filling the Gap in Programming Instruction: A Text-enhanced Graphical Programming Environment for Junior High Students." 276-80. *SIGCSE*. Web.

[3] Hoganson, Dr. Ken. "TEACHING PROGRAMMING CONCEPTS WITH GAMEMAKER*." 181-87. *SIGCSE*. Web.

[4] W.B. Li, Frederick, and Christopher Watson. "Game-Based Concept Visualization for Learning Programming." 37-42. *SIGCSE*. Web.

[5] Werner, Linda, Jill Denner, and Shannon Campe. "Pair Programming for Middle School Students: Does Friendship Influence Academic Outcomes?" 421-26. *SIGCSE*. Web.

[6] Werner, Linda, Shannon Campe, and Jill Denner. "TChildren Learning Computer Science Concepts via Alice Game-Programming." 427-32. *SIGCSE*. Web.

[7] Zhang, Jinghua, Emanuel Smith, Elvira R. Caldwell, and Matthew Perkins. "LEARNING AND PRACTICING DECISION STRUCTURES IN A GAME *." 60-66. *ACM*. Web.