When Good Games Promote Good Programming: Scratch Camp FTW

Xavier Velasquez, Utah State University
Deborah A. Fields, Utah State University

Abstract: This study presents an overview and summary of the initial results of encouraging and evaluating novice programmers, in this case young girls, by tasking them to create their own video games in the Scratch environment. The structure included an open-ended, but authentic task allowing them to explore several challenging abstract programming concepts such as variable manipulation and conditionals. This paper describes the ways in which we are analyzing student's programming processes in terms of the tasks they were given. The goal is develop analytics for the projects that describe the student's grasp (or lack thereof) of computer science concepts.

Introduction

For many years educators and researchers have been challenging kids to learn computer science skills by helping them create their own video games. Some of the goals behind this movement involve helping kids learn organizational systems and programming with an authentic, open-ended task (Denner, Werner, & Ortiz, 2011). These types of tasks can lead to sustained engagement in programming (Kafai, 1995), but one major challenge to educators is evaluating what kids are learning when the projects look wildly different.

We took on this challenge by engaging kids in a week-long, 30-hour "Scratch Camp" where 25 girls in grades 5 through 8 participated in the development of their own original multimedia content creations. Each day, the girls were given a project with goals designed to give visibility to increasingly difficult programming concepts such as loops, events, variables, and conditionals as the camp progressed. These projects culminated in creating their own interactive video games.

In this poster we explore the results of encouraging kids' programming efforts by tasking them to create their own games within the Scratch environment. It can be a challenge to engage kids in creating good games when they are novice programmers, especially given the high quality of video games many kids play in their leisure time. Further, games typically involve relatively abstract programming concepts difficult for novices, including the use of conditionals and variables. Learning these concepts is one reason why educators often use video games as a reason for programming, but with short amounts of time available, many students often do not use these programming concepts in sophisticated ways (e.g. Denner et al, 2011).

In this study we sought to engage and evaluate novice programmers (girls in this case) in learning more sophisticated programming through making simple but authentic video games. We took advantage of games' "endogenous" qualities (Squire, 2006) by allowing the students to play and "cheat" four starter games. Most students figured out how to manipulate variables by bumping up their health instead of subtracting from it or by jacking up their score in bigger intervals. The rest of this paper describes the ways we are developing to analyze students' programming processes in ways that are authentic to the game design task. One overarching goal is to develop computational analytics for students' projects that show their learning (or lack thereof) over time. This poster represents our first step in qualitative analysis toward building an analytics for evaluating open-ended video game designs.

Productive Programming in Scratch

Through an iterative analysis process we qualitatively coded students' games several different ways to understand what they changed and what they were learning hour by hour. Data included hourly collection of projects over two days which allowed us to see changes in their design and programming decisions. As a result of this process, we developed a rubric based on the requirements for the challenge presented to the girls for their interactive video game project. The rubric categorizes each game component as the following requirements: 1) create an instruction screen, 2) develop a user interface, 3) use two or more character sprites, 4) create two or more "levels", 5) create a "game over" screen, 6) use logic to change level condition by some sensing or other means, 7) use variables. Using a tool (Table 1) in our post-design evaluation, we were able to categorize the degree to which the students were able to accomplish the tasks on a scale, and then dig into their Scratch code to connect to a range of programming skills.

Variable - Health	No variables tracking player health used.	A variable tracking player health is used.	Variable tracking player health is used and implemented in advanced ways such as triggering events.

Table 1: Example of rubric health variable execution evaluation.

One challenge for novice programmers is understanding how to use variables. At a very basic level, using variables to count values such as health and score provide feedback to players about the game. However, we found that many kids started using variables in more sophisticated ways, for instance using score and health to trigger events in the game. One student, Genevieve, created a game *Kitty Catch!* where player had to catch zombie kitty-heads descending from the sky in a basket (see Figure 1). Over time she developed ways to use increasing score values to trigger more advanced levels where the zombie kitty-heads fell faster and faster. She also showed the player their "lives" in a tangible way, changing the image for lives as the player lost one after the other (see the hearts in upper right of Figure 1). While this may seem simple, it was a non-trivial task for a novice.

The ability to manipulate variables to trigger events or state changes in the game became evident in many different forms across kids' projects after outlining the requirement in the rubric. The full poster includes specific examples of Scratch code that relate to programming triumphs accomplished by students during the camp as they worked to satisfy, improve on, and often go above and beyond each requirement for their games.

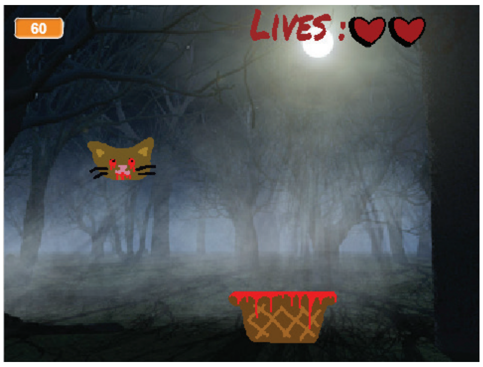

Figure 1: ***Kitty Catch!*** **Score in upper left, animated "Lives" bar in upper right.**

Implications and Next Steps

This poster shares the early stages of a project to develop and analyze student-created video games not only on a rich qualitative level but also at a computational-analytical level. One goal is to use this rubric to develop analytical measures that can track a larger group of students (75 over three camps) at more frequent time intervals (every 2-3 minutes) to see exactly when students introduce concepts such as variables and in what ways. In the full poster we compare our analytical results to our qualitative measures that better capture the aesthetics, play, and nuanced decisions of game-making. We are trying to apply this not only to games but to a variety of programs in Scratch including stories and music videos. Developing ways to authentically evaluate students' programming alongside game design could be productive for educators and researchers in other fields as well.

References

Denner, J., Werner, L., & Ortiz, E. (2012). Computer games created by middle school girls: Can they be used to measure understanding of computer science concepts?. *Computers & Education*, *58*(1), 240-249.

Kafai, Y. B. (1995). Minds in play: Computer game design as a context for children's learning. Routledge.

Squire, K. (2006). From content to context: Videogames as designed experience. Educational Researcher, 35(8), 19-29.