Designing Computer Programs in Elementary School

Crystal Noel
Computer Systems Lab 2008-2009

June 10, 2009

Abstract

Technology and computer science are valuable tools for learning. The goal of this project is to illustrate the value of designing computer science programs and the capability of elementary school age students to articulate goals and execute them. It builds on the knowledge that elementary school students can learn to program, through work with Scratch. They can use basic programming skills to design their own projects. The students are able to develop problem solving skills and the ability to respond to feedback. This project shows that designing programs benefits a student in his or her education at the elementary level.

Keywords: Scratch, design, Cardinal Forest Elementary, problem solving, programming

1 Introduction

For the duration of the 2007-2008 school year, students at Cardinal Forest Elementary came to the computer lab during their recess time to attend a computer programming class designed and executed by Gregory Gates (Thomas Jefferson High School) and Fred Allard (Cardinal Forest Elementary). The study showed that students from kindergarten to sixth grade were able to learn the vocabulary, concepts, and skills associated with basic computer programming. [?] Scratch is designed to facilitate this process by providing a way to avoid debugging processes and syntax errors. The software is easy to understand and use; large amounts of code do not need to be memorized in order to make use of it. [?] It is also very visually oriented, making it easy for students just beginning to learn to read.

Once the basic vocabulary like a coordinate system and a continuous loop were solidified, students were able to work on basic level projects designed by one of the teachers (Gates or Allard).

In my study I explore whether or not students who have successfully completed the projects are able to move on to designing their own projects. Designing projects (e.g. a video game) teaches students the skills required to build an idea from the ground up and go through the work leading up to a finished product. It teaches problem solving skills and the ability to work with feedback to develop a project further.

While the work done by students in learning basic programming skills is valuable, the more advanced students can be taken to the next level. Students are ready to excel beyond the basic goals of learning programming language and becoming comfortable with using a computer as a tool. The computer can be a tool for personal expression through the design and execution of creative coding projects. Students can design simple games in groups or individually. My study with the students at Cardinal Forest Elementary aims to give the advanced level students the opportunity to express their creativity in building their own computer science projects.

2 Background

There has been previous discussion about the use of video games as learning tools. Peppler and Kafai, in their article What Video Game Making Can Teach Us About Literacy and Learning: Alternative Pathways into Participatory Culture, explore the learning value of actually designing the games. Their study proved that students were able to participate in today's media culture through their collaboration at computer communities, much like the one at Cardinal Forest Elementary.

Through the use of design technology, students are also able to learn about setting goals and problem solving to reach those goals. They are able to go through the entire work process of a project that they created. Problem-solving skills are valuable in school; the skills are important in math especially, which is the basis of many other subjects. Students also learn to articulate their goals, which helps them when learning basic writing skills. Building a foundation in project design is helpful when students move away from elementary level work and into the group work required in higher levels of education. [?]

Designing projects is also a chance for students to express their creativity. The arts are an integral part of elementary education, and media art is a growing part of today's culture. The design of computer programming projects allows students to express themselves in a medium that is up-to-date with current technology.

Scratch is a software program that was created in the MIT Media Lab. It provides an online community to share student work. The program makes it easy for students to learn basic programming skills because of the absence of syntax errors. The library of available bits of code, sorted by their purpose, eliminates the problem of memorizing code. The community atmosphere provides the opportunity for students to be inspired by other students' work and receive feedback on their own.

Gates and Allard succeeded in setting up the Scratch based program at Cardinal Forest Elementary. They proved that students as young as kindergarten could learn to program. Towards the end of the school year, they were also able to delve into student designed projects. Gates found that the students were not very successful in design because they spent a lot of time drawing backgrounds or sprites (characters or objects in Scratch). [?] The students that participated in the program developed an appreciation for computer science and now understand basic programming vocabulary like loop and variable. Some of the students excelled over the others; these students were looking for challenges beyond the simple exercises provided, which I gave to them by guiding them through the process of creating their own projects.

3 Development Sections

3.1 Participants

I worked very closely with Jessica Gorman and Fred Allard throughout the project. Jessica was another Senior working in the Computer Systems Laboratory. Mr. Allard is the School Based Technology Specialist at Cardinal Forest Elementary. For some sessions with the students we also had a couple of parent volunteers. The students in the program were a mix of returning and new students from primary to 6th grade. I worked specifically with the first and second grade students.

3.2 Lessons

Scratch Sessions occurred on a weekly basis with each group in the lab for about 25 minutes. The students signed in upon entering the lab and usually picked up the rubric for the current project. I wrote clear directions on the board so that students could learn to follow step-by-step directions and get into the habit of reading instructions.

We (Jessica and I) often had a student read the directions aloud to encourage students to speak up and engage in the lesson. For the majority of the school year, lessons were project based, with projects built on the skills that we wanted them to learn. Each project had an accompanying rubric that helped guide the students through completing the project. We started by instructing students through every step, but throughout the year we progressively gave less instruction, encouraging students to discover on their own and then help their classmates find the solution. Lessons ended with the students saving their projects in a common folder, so that they were able to continue work the next week

3.3 Introductory Projects

We began the school year with a basic introduction to the Scratch program. We taught basic vocabulary like sprite, stage, background, and import. We introduced the different types of scripts (commands): control, motion, and looks.

After familiarizing the students with the Scratch program, we moved into the first major topic of xy-coordinates. We used Smartbook Software NoteBook to teach the students about negative numbers and the xy-grid set up.

The first Scratch project was used to solidify the skills learned using the NoteBook software. They had to use their knowledge of x and y to choose points to make Kitty go in a square pattern. We also introduced the pen methods, making Kitty draw the square as he moved around the grid.

The first project rubric was designed for both students who could read and for the students who needed pictures (See Appendix A).

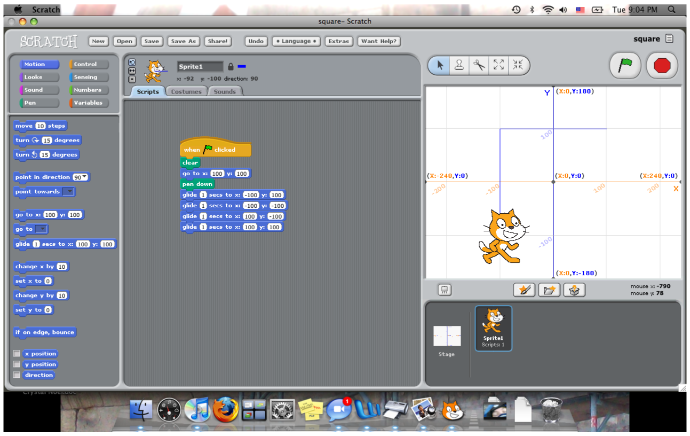


Figure 1: Screen shot of Project One

The second project, Winter Wonderland, had less emphasis on xy-coordinate skills, and gave students the opportunity to explore more of what is available when working with Scratch. We started with the connection to xy with moving a snowman around the ice rink. We then moved into more advanced skills like changing backgrounds and costumes, broadcasting, and response to keys being pressed. This project was divided into two parts: moving the snowman around the ice rink and moving the snowman from inside an igloo to outside on the ice rink. The two separate rubrics included both words and pictures (See Appendix A).

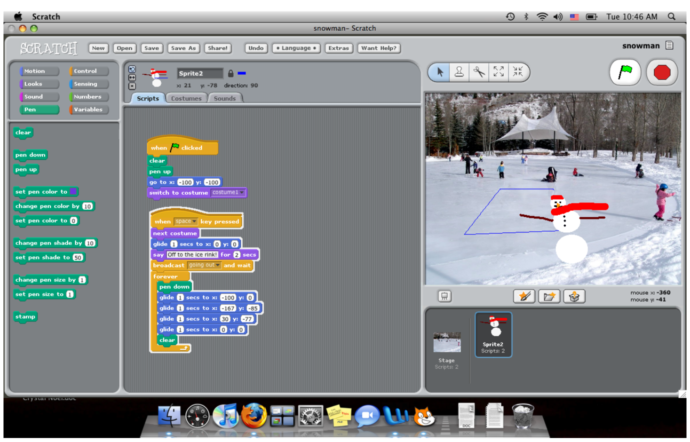


Figure 2: Screen shot of Winter Project

The third project was designed to inspire the students to think about how they could use Scratch to create games. It prompted them to think about what games they might like to create and how they could do it. The third project was Fish Game, and it was based on the game FishChomp provided in the Scratch downloadable software. The code was modified to suit the skill level of the students and incorporate ideas that had already been introduced like broadcasting and switching costumes. The game also introduced the new topic of random. The students learned what a random number was and used them to moved some of the fish around in the game. The broadcasting involved was more difficult than previous uses, so their understanding of broadcasting was increased. The rubric for this project is also included in Appendix A.

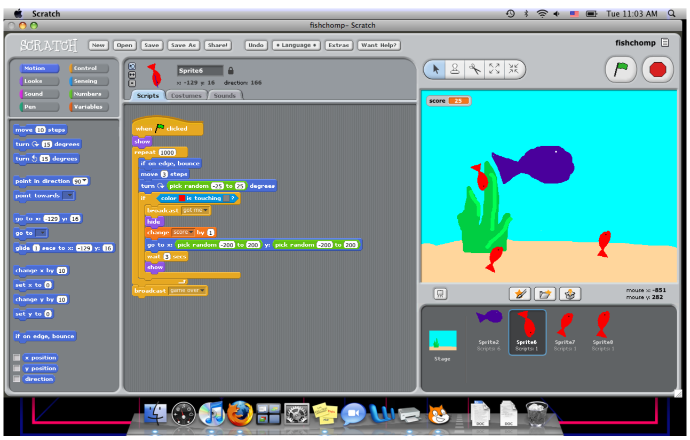


Figure 3: Screen shot of Fish Game

3.4 Student Project Design

The first graders were ready by mid-March to begin to design their own projects. We began with asking them to think of game ideas. We asked What kind of game do you want to make in Scratch? They replied with computer games that they had played online, or games that they had found in the Scratch software already. We prompted them to think about sports or arcade games as well. We then asked them to draw or write a sentence about what game they would like to make.

We narrowed down the project field into a manageable number of projects, as opposed to each student doing a different project. The final array of projects included: a story, PacMan, Super Mario, and dodgeball. We were able to place most students on the project that they had requested, whereas a few had to be moved based on their skill levels.

Our next challenge was to prompt the students to create a step-by-step layout of their projects. We created an introductory activity in which we gave them a blank rubric and asked them to fill it in regarding a very simple game based on Whack-A-Mole. The rubric had room for five steps, which was just enough to finish the project. We let students work on the rubric individually before prompting them on the correct answers. The activity taught them to think about a game in steps and create guidelines to follow in making it. We completed two activities of this kind. In the second activity, we did not even show them a sample run of the program to simulate creating their own projects.

I created simple guidelines for the story projects. The students doing the stories had the lowest skill levels, so the guidelines were designed to give them direction and expectations. I incorporated concepts that were already covered and required that they also use a script that had not been included in any previous projects. The guidelines are attached in Appendix B. The guidelines were also designed to help them realize that they could use what they already knew when starting their project. It is sometimes difficult for students to find a place to start a project, but the guidelines served as a base to push off of.

The rest of the students were given blank rubrics, much like in the introductory activities, and we pushed them to fill in the rubrics prior to completing code. We did not show them samples of their projects, which was different than what they were used to. It took prompting from Jessica and I, but for the most part they filled in the rubrics themselves. The students then worked together to arrange the scripts.

3.5 Assessments and Surveys

At the end of the school year, students completed a short assessment to see what they remembered from the year in Scratch. They were asked to draw and x-line an a y-line and label positive and negative, write a random set of numbers, and describe what broadcasting is used for. Students also were subject to a short survey about their likes and dislikes regarding school in general and the Scratch Program. The data from the surveys will be catalogued for use in future research regarding the students that participate in the Scratch program.

4 Results and Discussion

4.1 Skill Development

All students in the Scratch program were very familiar with the Scratch software by the end of the year. They were able to use basic scripts in scratch like when key pressed, glide, and change costume. They were able to see the different uses of Scratch like creating stories or games. The program successfully exposed students to the world of computer science, and I hope that it will inspire them to explore the subject further in the future.

In order for the program to exist, SOL topics must be integrated into the curriculum. Our main focus was on the coordinate system. Even if students are not masters of the subject, they have at least been exposed to it and have seen applications of the math. In the end of year assessments, many students were able to draw the x and y lines, but few were able to differentiate between them or label the correct positive and negative ends. I think with a few reminders in a future classroom, the students will be able to quickly remember what they learned in the Scratch program about the coordinate system. The students that we worked with were just beginning to develop their reading and writing skills. Throughout the year, we asked them to write sentences about their projects or read directions on the board. While reading and writing skills were not a focus of the program, these skills were inherently developed.

4.2 Student Project Design

When confronted with the process of designing and implementing a project in Scratch, the students were initially excited. They were so excited about making the project, that they were reluctant to plan before coding. Our biggest challange was to get them to sit down and fill out a blank rubric, as a plan, before beginning to code. They all knew what they wanted in their projects, but it took heavy prompting from Jessica and I to get them to write it down and quantify what they wanted to do. We reinforced the value of planning by refering back to their rubrics during the construction process.

We used their rubrics to ask them what step they wanted to work on every week. The students that were creating stories were able to easily work at their own pace and use the guidelines, which was the goal for these students. They were not advanced enough to attempt a game, but the stories that they created showcased their skills well.

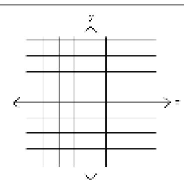
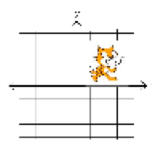
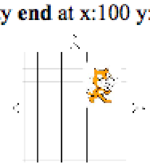
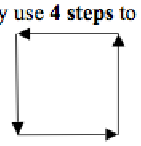
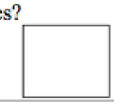
When planning for student designed projects, Jessica and I should have taken more care in helping the students choose an appropriate project. We also should have created sample programs, at least for our own reference. The PacMan game turned out to be more difficult than we anticipated, we should have started off by giving them more code to work with. The Mario game took on a new direction that we did not forsee. Students each put their own spin on the game, and we ended up with four very different versions.

For the students who did grasp the concept of brainstorming and planning, the process of designing a computer program was very beneficial. Students were able to see what projects are attainable through Scratch and the amount of thinking and work involved in creating a project. The students discovered Sratch as a creative outlet; they especially enjoyed creating their own backgrounds and sprites.

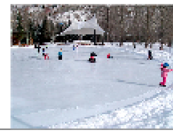
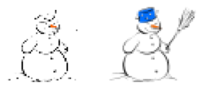
My basic conclusion is that for advanced students, designing computer programs in elementary school is a valuable endeavor. It exposes them to the world of computer science, so that they might have more interest in the future. Even for those students who did not benefit as much from designing projects learned many important concepts through their work in the Scratch program. In the future, I recommend that the program be continued at Cardinal Forest, as well as expanded to other schools. I also recommend that students with higher skills levels be singled out and taken to the next step of designing their own programs, since it is such a beneficial process.

5 Appendices

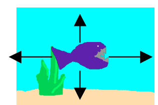
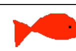
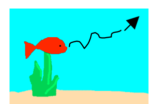
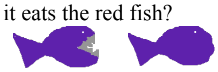
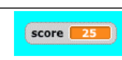
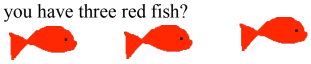
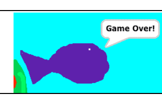
5.1 Appendix A: Rubrics

Task	**Check?**
Stage: 	
Does Kitty **start** at x: 100 y: 100? 	
Does Kitty **end** at x:100 y:100? 	
Does Kitty use **4 steps** to move in a square? 	
Bonus: Does Kitty **draw** the square as he moves? 	
Bonus: Does the square that kitty has drawn **disappear** when you start the program over again?	

Part 1: Winter Wonderland Project Rubric

Step	Task	Check?
1	Stage: 	
2	Does your snowman have 2 costumes? 	
3	Does your snowman **start** at x:-100 y:-100? 	
4	Is the snowman wearing the costume that you drew for him?	
5	Does your snowman **glide** around the ice rink? 	
6	Does your snowman stay on the ice rink? 	
7	**Bonus:** Does your snowman glide around the ice rink forever?	

Fish Game Rubric

Step	Task	Check?
1	**Stage**: 	
2	Big Fish **Sprite**: 	
3	When you **press** the arrow keys, does the Big Fish move that way? 	
4	Red Fish **Sprite:** 	
5	Does your red fish move **randomly**? 	
6	Does your red fish **hide** when the Big Fish eats it?	
7	Does your red fish **show** in a different spot after it is eaten?	
8	Does your Big Fish change **costumes** when it eats the red fish? 	
9	Does your game keep **score**? 	
10	Do you have three red fish? 	
11	Does the Big Fish **say** "Game Over"? 	

Game Rubric.pdf

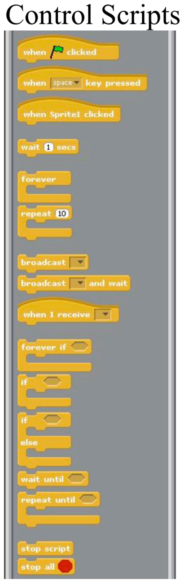
5.2 Appendix B: Project Design

Make Your Own Project!

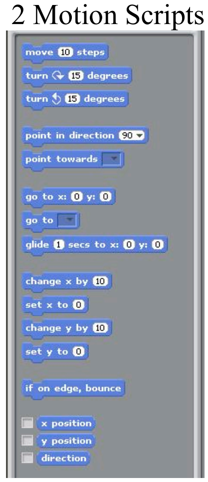
Task	Check?
1 Sprite	
2 Backgrounds	
Change Sprite Costume 1 time	

Circle:

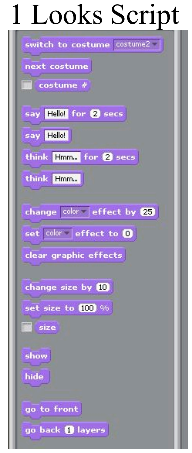
4 Control Scripts



2 Motion Scripts



1 Looks Script



Draw a rectangle around your new script.

2 **Scripts** from: **Pen, Sensing, Numbers, or Variables**

1. ____________________
2. ____________________

References

[1] Yasmin Kafai and Kylie Peppler, Creative Coding: Programming for Personal Expression.

[2] Monroy Hernndez, Scratch: a platform for sharing user-generated programmable media.

[3] Elizabeth Sylvan, Integrating Aesthetic, Engineering, and Scientific Understanding in a Hands-On Design Activity

[4] Yasmin Kafai and Kylie Peppler, What Video Game Making Can Teach Us About Literacy and Learning: Alternative Pathways into Participatory Culture

[5] Robbin Chapman and Leo Burd, Beyond Access: A Comparison of Community Technology Initiatives

[6] Oren Zuckerman, System Blocks: Learning about Systems Concepts through Hands-on Modeling and Simulation 2004

[7] Gregory Gates, Elementary Education in a Technology Age 2008