Learning to Design Computer Programs in an Elementary School Setting

Crystal Noel
Computer Systems Lab 2008-2009

January 23, 2009

Abstract

The goal of this project is to illustrate the value of designing 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.

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 found that 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 gave 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. (Sylvan) 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, a skill that is important when learning basic writing skills. Building a foundation in project work 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. 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 are the students that were involved in my study. The students were looking for challenges beyond the simple exercises provided to the students, which I gave to them by guiding them through the process of creating their own projects.

3 Development Sections

The bicyclists in these groups consist of three components: physical simulation, locomotion controller, and navigation controller (Figure 3). The physical simulation is dened by equations of motion that represent a hierarchy of rigid body parts and the rotary and telescoping joints that connect them. The equations of motion for the bicyclist were formulated using a commercially available package.12 A characters locomotion control ler computes how to actuate its joints in order to move at a specied desired velocity. Due to kinematic and dynamic constraints, the locomotion controllers cannot instantaneously eliminate errors between a characters desired velocity and its actual velocity. These limitations to a characters maneuverability, or mobility constraints, are realistic and intuitive to the user, but they make it more difficult for navigation control lers to compute a desired velocity for each character that accomplishes group behaviors, obstacle avoidance, and path following.

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 Technology Teacher 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.

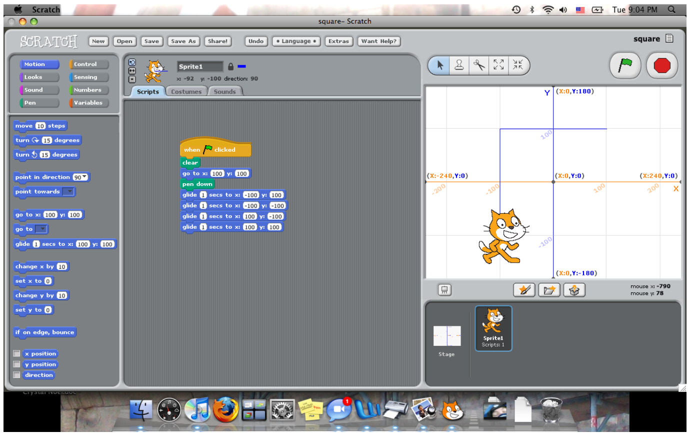
3.1.1 Lessons

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. (Insert sample picture of white board with instructions from Mr. Allard)

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 the students were trying 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, too. Lessons ended with the students saving their projects in a common folder, so that they were able to continue work the next week

3.1.2 Topics

We began the school year with a basic introduction to the Scratch program. We taught basic vocabulary like sprite, stage, background, and import. We



```
when [flag] clicked
clear
go to x: 100 y: 100
pen down
glide 1 secs to x: -100 y: 100
glide 1 secs to x: -100 y: -100
glide 1 secs to x: 100 y: -100
glide 1 secs to x: 100 y: 100
```

Figure 1: Screen shot of Project One

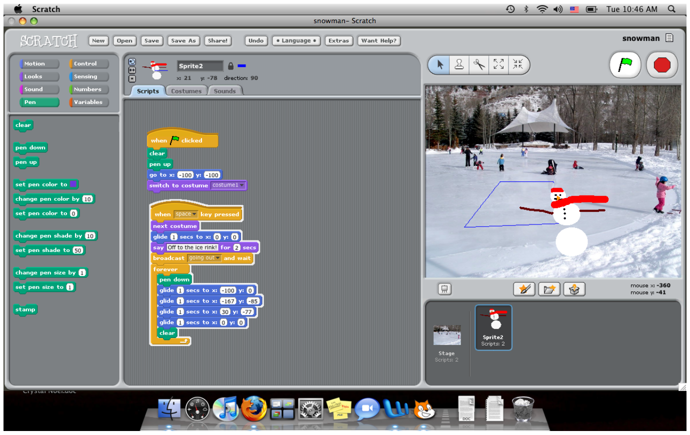
introduced the different types of scripts (commands): control, motion, and looks.

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 help us 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).

The second project, title Winter Wonderland Project, 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).



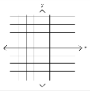
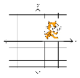
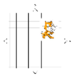
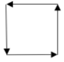
```
when [flag] clicked
clear
pen up
go to x: -100 y: -100
switch to costume costume1

when space key pressed
next costume
glide 1 secs to x: 0 y: 0
say Off to the ice rink! for 2 secs
broadcast going out and wait
forever
    pen down
    glide 1 secs to x: -100 y: 0
    glide 1 secs to x: -167 y: -85
    glide 1 secs to x: 30 y: -77
    glide 1 secs to x: 0 y: 0
    clear
```

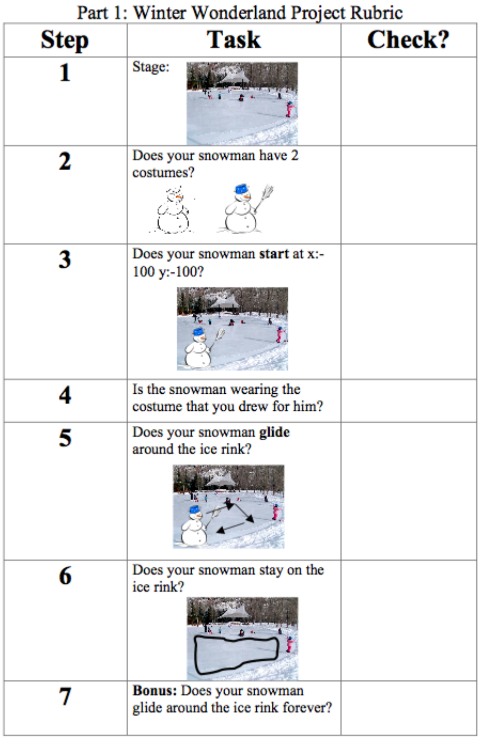
Figure 2: Screen shot of Winter Project

4 Results and Discussion

5 Appendices

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?	

6 References

(not yet in APA format)

Scratch: a platform for sharing user-generated programmable media, Monroy Hernndez
Integrating Aesthetic, Engineering, and Scientific Understanding in a Hands-On Design Activity, Sylvan
What Video Game Making Can Teach Us About Literacy and Learning: Alternative Pathways into Participatory Culture, Peppler and Kafai
Beyond Access: A Comparison of Community Technology Initiatives, Chapman and Burd
System Blocks: Learning about Systems Concepts through Hands-on Modeling and Simulation, Zuckerman
Creative Coding: Programming for Personal Expression, Peppler and Kafai