Empowering Pre-Service Teachers to Utilize Programming in the Classroom

KANSAS STATE UNIVERSITY

Scott Bell
Kansas State University
Computing and Info. Sciences

Angie Miller
USD-383 School District

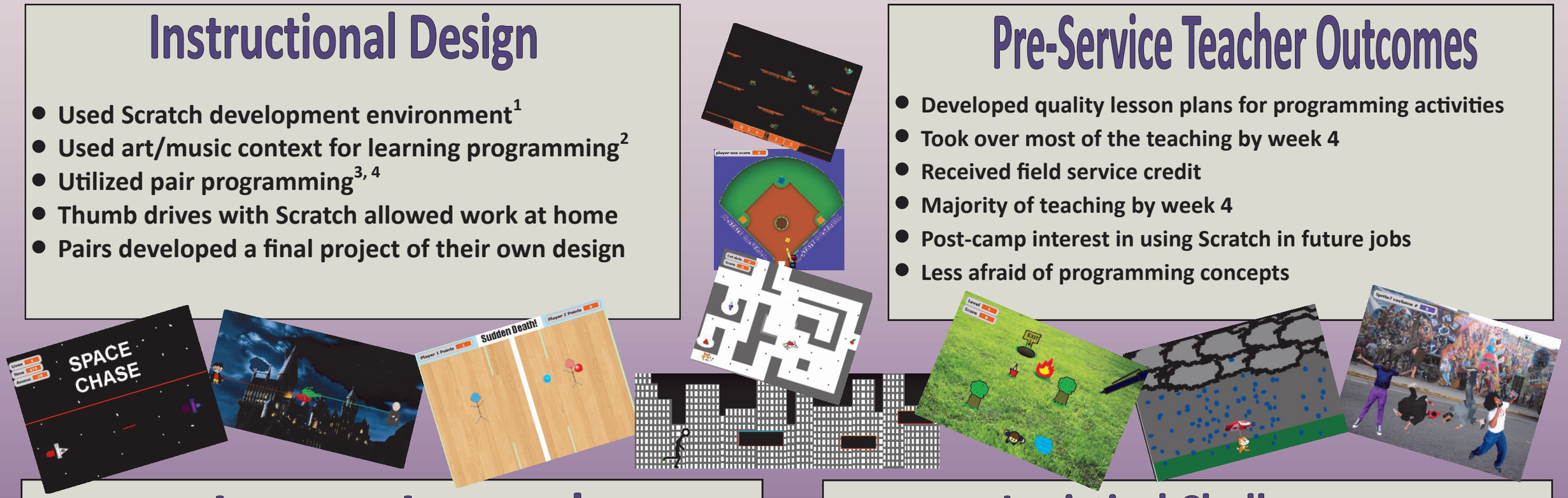
Tim Frey
Kansas State University
College of Education

Eugene Vasserman
Kansas State University
Computing and Info. Sciences

Question: How can we help K-12 teachers learn to teach programming?

Instructional Design

- Used Scratch development environment[1]
- Used art/music context for learning programming[2]
- Utilized pair programming[3, 4]
- Thumb drives with Scratch allowed work at home
- Pairs developed a final project of their own design



Pre-Service Teacher Outcomes

- Developed quality lesson plans for programming activities
- Took over most of the teaching by week 4
- Received field service credit
- Majority of teaching by week 4
- Post-camp interest in using Scratch in future jobs
- Less afraid of programming concepts

Lessons Learned

- 2 students is optimal and 3 is better than 1
- clear objectives vital with multiple teachers
- small demos/projects > big demos/projects
- Pre-service teachers picked up Scratch quickly
- Students gravitated towards video games even though we encouraged other project ideas
- Lot to learn about working with students from K-12 teachers

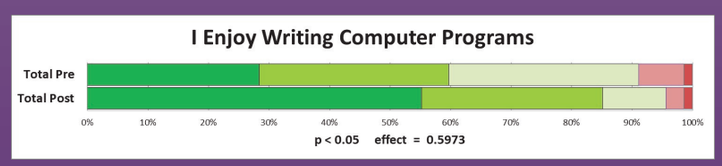
Logistical Challenges

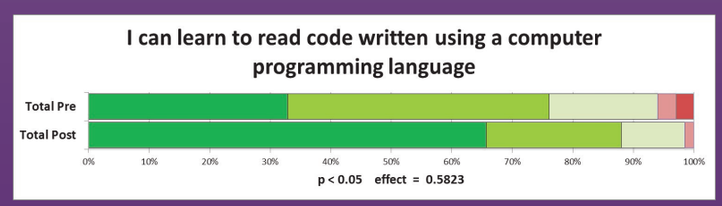
- Summer camp: 4 weeks, 4 days/week, 2.5 hours/day
- 15-20 new students each week
- 5 pre-service music and art teachers
- 1 in-service middle school math teacher
- Pre-service teachers gradually took over teaching
- Pre-service teachers introduced to Scratch 10 days before camp!

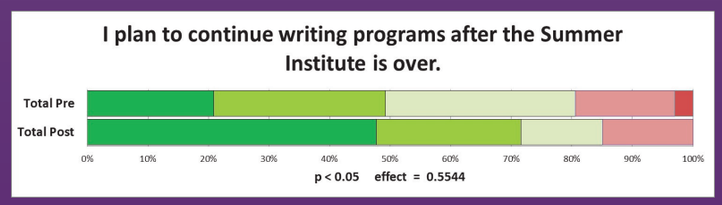
Student Survey Outcomes

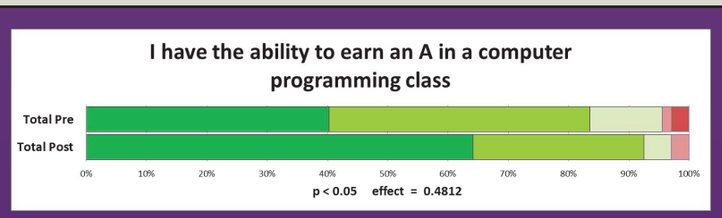
- Self-confidence increased
- Interest/enjoyment increased
- Desire to continue to program increased

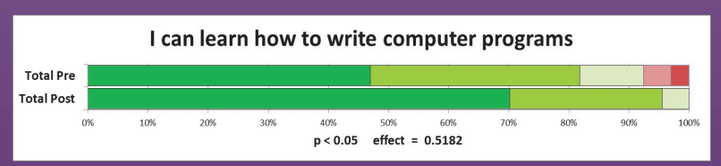
N = 52

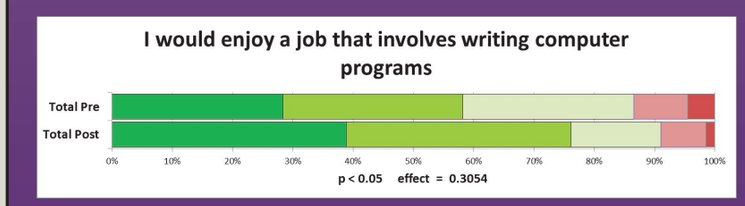


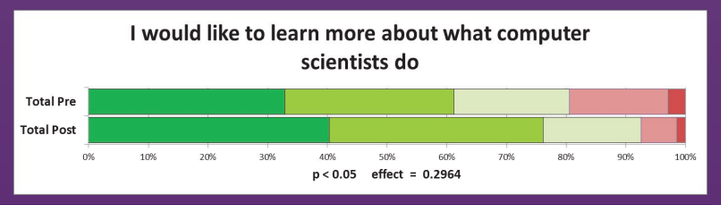












Strongly Agree | Agree | Neither | Disagree | Strongly Disagree

This project was funded by a U.S. Department of Defense Education Activity Grant (no. HE1254-10-1-0032)

References

[1] Scratch, MIT lifelong kindergarten. http://scratch.mit.edu.
[2] M. Guzdial in *SIGCSE Bull.*, 35(3):104–108, June 2003.
[3] C. McDowell, L. Werner, H. Bullock, and J. Fernald in *SIGCSE Bull.*, February 2002.
[4] L. Williams, R. Kessler, W. Cunningham, and R. Jeffries in *IEEE Softw*, July 2000.

Empowering Pre-Service Teachers to Utilize Programming in the Classroom

Scott Bell[*], Tim Frey[*], Angie Miller[++], and Eugene Vasserman[*]
[*]Kansas State University
[++]USD-383 School District

Introduction

A major obstacle in recruiting students into computing is a lack of exposure to relevant topics in K-12 classrooms.[1] Based on our experience, most teachers have little programming experience and thus are reluctant to incorporate it into their classrooms. Given todays high quality, user-friendly development environments such as Scratch,[2] they no longer need to be programming experts to introduce programming to their students in a variety of contexts such as math, science, art and music. The STEM Summer Institute provided us an opportunity to immerse pre-service teachers and an in-service teacher in a multi-week camp teaching programming to middle school students.

Background

The Institute, in its third year of operation, is a summer enrichment program made possible by a collaboration between the Manhattan-Ogden USD-383 School District and Kansas State University. The goal of the program is to increase interest in STEM subjects through interactive, hands-on activities. During this four week institute, nearly 200 sixth to ninth grade students select a different course of study each week. Students spend about two and a half hours per day over four days learning about their chosen subject for that week.

Instructional Design

In the I.Code course, Scratch was chosen for introducing basic programming and problem solving skills. Its drag and drop interface reduces the chance of major syntax problems for beginning students. This was crucial given only ten contact hours each week. The difficulty level of projects in Scratch is easy to scale to the ability level of students, which was important for a group with varied ages and backgrounds. Finally, Scratch provides a variety of ways to incorporate sound and graphics into projects, which has been shown to increase student engagement.[3]

Working in pairs,[4,5] students spent the first two days following an instructor through a sequence of demonstration projects which introduced programming constructs such as sequence, selection, and iteration. Each demonstration was followed by assigned challenges which used those constructs. We designed the demonstrations and challenges to cover a wide range of tools within Scratch and familiarize the students with the capabilities of the environment. The challenge activities required students to design then build programs that applied the concepts and tools they had just learned and included advanced challenges for those who completed the main challenge early. This pushed advanced students to continue exploring the environment and expanding their skill set. By the end of day two, most students had gained enough experience to create an initial plan for a final project that they would develop and present to the class on the fourth day. We reviewed these plans

with the students, suggesting ways they might add interesting content or avoid pitfalls. Days three and four were spent on final project design (students were required to have their project design approved prior to starting to build it) and programming followed by project presentations.

Pre-Service Teacher Responsibilities

Five pre-service teachers helped teach the course as part of their required field experience. These aspiring teachers (three in music education and two in art education) helped incorporate music and art materials into the lesson plans, demonstrating how programming can be utilized in a variety of classroom contexts. During week one, the student teachers observed the teaching of programming topics and assisted student pairs during hands-on activities. Each week, the student teachers took over an increasing percentage of the teaching responsibilities so that by week four they were responsible for a majority of the instruction. An in-service teacher supervised lesson-plan development and evaluated teaching activities (both required components of the field experience).

Lessons Learned

The one place where pre-service teachers struggled was reviewing final project designs. They lacked the experience necessary to assess the level of difficulty of a given design and the insight to suggest changes that might alleviate some of the problems within a design without changing the overall student concept. We also found that even though we attempted to avoid video games as part of the instructional material, a majority of the final projects had a game-type theme.

Outcomes

Pre-camp and post-camp surveys were collected from students to investigate changes in interest and self-confidence with respect to computer programming. Feedback from pre-service teachers was also collected through reflective discussions. Response from participants (both students and teachers) was positive. Student confidence in programming ability, enjoyment of programming, and interest in continuing to program increased. Pre-service teachers discussed ideas for incorporating more context-specific material into Scratch activities and how these could be applied in their future teaching positions. This was a promising step towards empowering new teachers with the ability to introduce programming into the K-12 curriculum and potentially increasing the number of students that consider computing fields as career options in the future.

Acknowledgments

This work was funded by a United States Department of Defense Education Activity Grant (Award Contract Number: HE1254-10-1-0032)

References

1 Lori Carter. Why students with an apparent aptitude for computer science don't choose to major in computer science. *SIGCSE Bull.*, 38(1):27–31, March 2006.

2 Scratch, MIT lifelong kindergarten. `http://scratch.mit.edu`.

3 Mark Guzdial. A media computation course for non-majors. *SIGCSE Bull.*, 35(3):104–108, June 2003.

4 Charlie McDowell, Linda Werner, Heather Bullock, and Julian Fernald. The effects of pair-programming on performance in an introductory programming course. *SIGCSE Bull.*, 34(1):38–42, February 2002. ISSN 0097-8418.

5 Laurie Williams, Robert R. Kessler, Ward Cunningham, and Ron Jeffries. Strengthening the case for pair programming. *IEEE Softw.*, 17(4):19–25, July 2000. ISSN 0740-7459.