National Center for Women & Information Technology
PROMISING PRACTICES

Snap, Create, and Share with Scratch (Case Study 5)

An Engaging Way to Introduce Computing

K-12 Education | Undergraduate

WHAT MAKES SCRATCH SO ACCESSIBLE TO NOVICES?

Scratch is a free "media rich programming environment" in which novice programmers can quickly express their creativity while learning computational thinking. Developed by the Lifelong Kindergarten group at the MIT Media Lab, Scratch is used at both the K-12 and undergraduate levels to reduce the barriers created by a programming language's abstract syntactic and semantic rules. Instead, students "snap" together several categories of "building blocks" (e.g., statements, loops, variables) to quickly generate animations, games, and art. The building blocks only snap together if they are syntactically appropriate. Students can work both individually and in small teams.

Scratch is effective as a learning tool because it incorporates several effective practices: it uses hands-on, active learning; it is visually appealing; it allows users to express their own creativity and to build on their own experiences; it gives immediate, understandable feedback; and it allows users to avoid syntax errors without focusing on minutiae, freeing them to focus on processes and concepts.

WHAT COMPUTING CONCEPTS DO STUDENTS LEARN USING SCRATCH?

Educational researchers at MIT Media Lab and University of California-Los Angeles studied Scratch scripts used in 425 programming projects created by 80 girls and boys ages 8-18 to determine which programming concepts they learned. The researchers found that all these projects used sequential execution and 90 percent used threads (multiple scripts running in parallel). About half of the projects included loops and user interaction and about a quarter included conditional statements and synchronization. A smaller set included Boolean logic, random numbers, and variables. The projects tended to include more of these concepts the longer students used Scratch.

ASSESSMENT OF SCRATCH AS TRANSITIONAL TOOL

Although Scratch was originally designed for ages 8-16, several universities are using Scratch in undergraduate courses, including Harvard, Rutgers, and College of New Jersey. Harvard researchers conducted a small classroom-based study on the use of Scratch for entry-level programming at the undergraduate level. The researchers used surveys to gather information about students' prior programming experience, their experiences with Scratch, and the ease of the post-Scratch transition into Java. Most students felt that Scratch positively influenced their ability to learn Java. Of the students who felt Scratch had no influence, all had prior programming experience.

SCRATCH COMMUNITY AND EDUCATOR SUPPORT

The makers of Scratch created a social network of sorts within the Scratch site. Users can post their project and remix others' projects; they can also discuss issues on the Scratch forum in several languages. More than 200,000 projects have been posted on the Scratch web site by novice programmers from around the world. The "top-loved" project has more than 23,000 views and 635 votes of "Love It." More than 26,000 projects have been remixed by other Scratch developers. The website also has a section especially for educators, with videos and other resources for getting started and ongoing support. Find out more here: http://scratch.mit.edu/.

SCRATCH-BASED ONLINE EDUCATIONAL COMMUNITIES

A number of educators have begun posting lesson plans and support materials to share with other teachers around the world. For example, Karen Randall, an elementary school teacher in Minnesota, has created a wiki (at http://wiki.classroom20.com/Scratch) where people can share Scratch materials. MIT Media Lab doctoral student Karen Brennan is creating an online community called ScratchEd, where educators will be able to share ideas, experiences, and curriculum plans with one another (to be launched later this year). Here are other sources of Scratch lesson plans and materials:

- http://nebomusic.net/scratch.html
- http://coweb.cc.gatech.edu/ice-gt/446
- http://www.learnscratch.org/
- http://www.lero.ie/educationoutreach/secondlevel/scratchlessonplans.html

RESOURCES

Malan, D.J., & Leitner, H.H. (2007). Scratch for budding computer scientists. *SIGCSE Bulletin (39)* 1, 223-227.

Maloney, J. H., Peppler, K., Kafai, Y., Resnick, M., & Rusk, N. (2008). Programming by choice: Urban youth learning programming with Scratch. *SIGCSE Bulletin (40)* 1, 367-371.

NCWIT offers practices for increasing and benefiting from gender diversity in IT at the K-12, undergraduate, graduate, and career levels.

This case study describes a research-inspired practice that may need further evaluation. Try it, and let us know your results.

ncwit.org Author | Lecia Barker
Copyright © 2008

NCWIT Investment Partners: National Science Foundation, Avaya, Microsoft, Pfizer, and Bank of America

National Center for Women & Information Technology
PROMISING PRACTICES

How Do You Introduce Computing in an Engaging Way?
with Case Study 5

K-12 Education | Undergraduate

Experience with computers between boys and girls has equalized, but boys continue to have greater knowledge of computing and programming *concepts* than do girls. Not so in biology, chemistry, or mathematics, where both boys and girls are encouraged to provide evidence of proficiency when they apply to college. High school study of these subjects familiarizes students with the content and concepts, and gives them confidence. The result is that women's undergraduate completion rates have neared parity in these disciplines.

Because IT study is elective in almost all K-12 schools, developing relevant and interesting assignments that appeal to a broader audience is recommended for:

- fostering a climate where the non-predisposed can belong both academically and socially
- recruiting students who are not predisposed to pursuing computing
- exposing fundamental computing concepts to inexperienced learners

Is prior programming experience required for students to be successful in an IT program? Most undergraduate departments would say no. That is, experience with programming is not the same as expertise in problem-solving, algorithmic thinking, or computing theory. Yet research shows that introductory courses and their embedded assignments work better for students who have *some* experience with programming.

Research also shows that students with programming experience are more confident and more successful in introductory courses than are their inexperienced peers. Students with lower grades or less confidence are less likely to persist in an IT major. What is more, when introductory courses have limited opportunities for talking to other students (e.g., collaborative learning), inexperienced students have little information on which to judge whether they belong academically in the major. Hence more women than men switch out of IT majors (most often to other sciences or mathematics).

MAKING IT MEANINGFUL

Educational researchers emphasize the importance of linking educational materials and curricular programs to students' existing knowledge and experiences. When class syllabi list topics and assignments that focus on unfamiliar concepts with limited, if any, relationship to a student's life experience or interests, she or he is unlikely to take that class. High school curricula contribute to low enrollments in college computing because, under the existing educational policy of election, computing is rarely required in secondary schools. This means that students are likely to have a narrow and inaccurate view of what IT study involves, what careers are possible, or what kind of people "do" IT. Given the very small proportion of females who study computing in high school, females are less likely to choose IT in college.

The challenge to educators at all levels is to develop engaging assignments and curriculum that can appeal to a variety of students with different learning styles, interests, socio-cultural backgrounds, and abilities, while maintaining the rigor of the discipline. Putting the concepts of computing in appealing contexts and building on existing competence can both reduce entry barriers and level the playing field for those with limited experience.

Creative assignments that teach algorithmic thinking while also calling on students' existing knowledge or interests, may serve to both recruit and retain students. When experienced and inexperienced students use non-computer-based assignments to learn computing concepts, they quickly realize that their peers with programming experience are not necessarily better at algorithmic thinking, just more experienced with programming. Building confidence through relevant and interesting assignments is a promising practice for motivating student enrollment and retention.

RESOURCES

Lecia Barker and William Aspray, "The State of Research on Pre-College Experiences of Girls with Information Technology." In McGrath Cohoon, J. and W. Aspray (Eds.) *Women and Information Technology: Research on the Reasons for Under-Representation*. Cambridge, MA: MIT Press, 2006.

Joanne McGrath Cohoon and William Aspray, "A Critical Review of the Research on Women's Participation in Postsecondary Computing Education." In McGrath Cohoon, J. and W. Aspray (Eds.) *Women and Information Technology: Research on the Reasons for Under-Representation*. Cambridge, MA: MIT Press, 2006.