A Networked, Media-Rich Programming Environment to Enhance Informal Learning and Technological Fluency at Community Technology Centers

NSF-ITR grant (ITR-0325828)
Final Report, November 2008

Findings

In developing the Scratch programming environment and the Scratch website, we aimed to make computer programming:

- **more accessible**, enabling learners to create computer programs by simply snapping together graphical blocks, without any of the obscure syntax or punctuation of traditional programming languages

- **more meaningful**, adding programmability to the media-rich and network-based activities that are popular in today's youth culture

- **more social**, providing young people with the opportunity to borrow and build upon one another's ideas, images, and programs

We believe that the popularity and rapid adoption of Scratch (as discussed in the Research and Educational Activities section) is an indication of the success of these design principles.

In previous annual reports, we discussed specific Findings from the first four years of the project. In this final report, we provide an overview of some of our core Findings.

1. Learning Computational and Mathematical Concepts

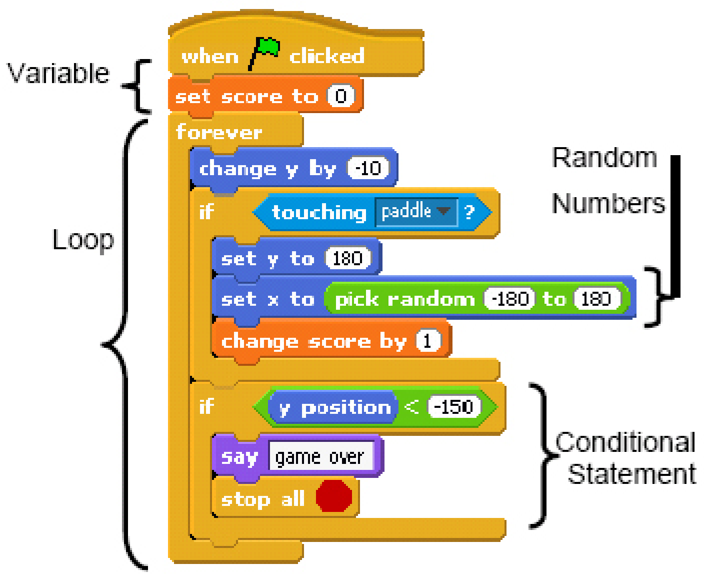
We have found that children and teens learned important computational and mathematical concepts as they created interactive projects in Scratch. We studied this issue most closely at one of the Computer Clubhouse after-school learning centers in an economically-disadvantaged neighborhood in Los Angeles.

Scratch grew to be the most widely-used pieces of software available at the Clubhouse. We did very little to explicitly "teach" programming concepts; rather, youth worked on projects of their own choosing and requested assistance from mentors when needed. Every two or three months, we organized a Scratch-a-thon during which all Clubhouse members would work on Scratch for 3-4 hours and share publicly the projects they had created. Clubhouse members used Scratch to implement wide range of media applications, including videogames, music videos, greeting cards, and animations.

The mentors were primarily students from UCLA, and most had little background in computer science or programming. The primary role of the mentors was to model and support the learning process, not to provide programming expertise. When they started, the mentors had little or no experience with Scratch, and they learned along with the Clubhouse members.

Over a two-year period, we collected a total of 536 projects created by 80 Clubhouse members (with an even gender mix of boys and girls). To get an idea of what computational concepts were learned by the Clubhouse members, we analyzed the use of Scratch programming blocks across the projects. We associated particular programming blocks with particular programming concepts, so that the use of particular blocks was viewed as an indication that the associated concept was being used in the project.

For example, the following sample Scratch script (taken from a simple Pong-like paddle game created by a Clubhouse member) shows the use of several computational and mathematical concepts: variables, random numbers, conditionals, and loops.



```
when clicked
set score to 0
forever
    change y by -10
    if touching paddle ?
        set y to 180
        set x to pick random -180 to 180
        change score by 1
    if y position < -150
        say game over
        stop all
```

Figure 1
Sample Scratch script (from Pong-like paddle game),
highlighting computational and mathematical concepts

Of the 536 Scratch projects collected at the Clubhouse, 111 of them contained no scripts at all. These “pre-scripting” projects illustrate the use of Scratch simply as a media-manipulation and composition tool. Beginning Scratch users often spend time importing or drawing images and recording sounds before moving on to scripting. Our analysis of the remaining 425 projects showed the use of a range of computational and mathematical concepts, with some concepts used very often (sequential execution: 100%) and others used only rarely (random numbers: 5%). See the Figure 2 below for a complete listing of the computational concepts and frequency of use in the analyzed Scratch projects.

Computational Concept	**Projects Using Concept**
Sequential execution	100%
Parallel execution (threads)	88%
User interaction	54%
Looping	52%
Conditional statements	26%
Communications/Synchronization	25%
Boolean logic	11%
Variables	10%
Random numbers	5%

Figure 2
Computational concepts used in Scratch projects

These statistics seem to indicate that many Clubhouse youth, while creating Scratch projects, were able to learn and use some computational concepts (such as sequential and parallel execution, user interaction, and looping) with little support from mentors, while other concepts (Boolean logic, variables, random numbers) required more support from mentors. Indeed, this conclusion is supported by a more fine-grained analysis of the data, showing that the use of Boolean logic, variables, and random numbers increased sharply in the second year of the project, as Clubhouse members had additional interaction with mentors (and the mentors themselves had more experience with Scratch).

One short anecdote from the Los Angeles Clubhouse highlights the important role of mentors – and also the importance of context in the learning process. While visiting the Los Angeles Clubhouse, a researcher from the MIT Scratch Team met a 13-year-old boy who was creating an interactive game in Scratch. The boy didn't know how to keep score in the game, and asked for help. The MIT researcher showed the boy how to create a variable in Scratch, and he immediately saw how he could use a variable for keeping score. The boy began playing with the blocks for incrementing variables, then reached out and shook the researcher's hand, saying "Thank you, thank you, thank you." The researcher wondered: How many 8th grade algebra teachers get thanked by their students for teaching them about variables? (For more detail on our studies at the Los Angeles Computer Clubhouse, see http://web.media.mit.edu/~mres/papers/sigcse-08.pdf)

Based on our experiences at Computer Clubhouses and other field sites, we put together a description of computational and mathematical concepts that are supported by Scratch programming activities. See http://info.scratch.mit.edu/@api/deki/files/610/=programming-concepts-1.3.pdf

2. Collaboration

As children from around the world have shared projects to the Scratch website, we have been impressed not only with the number of projects (more than 250,000 projects uploaded to the website in the first 18 months) but with the level of collaboration among members of the Scratch online community. We have found that construction and collaboration go hand-in-hand in the Scratch community: children become more engaged in the construction process when they are able to share their constructions with others in a community, and children become more engaged in the community when they are able to share constructions (not just chat) with others within the community.

Members of the Scratch community collaborate with one another in many different ways. Here are some examples:

- *Remixing*. If members of the Scratch community see a project that they like on the Scratch website, they can download it, modify the programming blocks, and then upload the revised version back to the website, to share with the rest of the community. This type of "remixing" accounts for more than 15% of the projects on the Scratch website. At first, it was common for members of the Scratch community to complain when someone else "stole" one of their project. We explained that sharing is an important part of the Scratch culture, so Scratch community members should feel proud when someone else modifies or extends their work. We have continually added new features to the website to support and encourage remixing. Now, when someone remixes a project, the website automatically adds a link back to the original project. Also, each project on the website includes links to its "offspring" (projects that were remixed from it), and the "top remixed projects" are featured on the Scratch homepage.

- *Companies*. Members of the Scratch community have joined together to form "companies," enabling them to create projects that none of the individual members could on their own. An example… A 15-year-old girl from the UK, with screen name Boopaloo, created a project full of animated sprites, and encouraged others to use them in their projects. A 10-year-old girl, with screen name SonicPops, liked Boopaloo's animations and asked if she'd be willing to create "a mountain background from a bird's-eye view" for use in one of her projects. SonicPops then asked Boopaloo if she wanted to join Crank Inc., a "miniature company" that SonicPops had created to produce "top quality games" in Scratch. A few days later, a 14-year-old boy from New Jersey, screen name Wodunne, discovered the Crank Inc. gallery and offered his services: "I'm a fairly good programmer, and I could help with de-bugging and stuff." Later, a 11-year-old boy from Ireland, screen name 11Alex, was added to the Crank staff because of his expertise in "scrolling backgrounds."

- *Contests*. Some members of the Scratch community have run their own contests and competitions on the Scratch website, as a way of encouraging other community members to contribute to collaborative projects. For example, a 13-year-old girl with screen name MahoAshley began using Scratch to create stories with anime characters. Other members of the Scratch community responded very positively to MahoAshley's projects (posting glowing "reviews" under her projects), so MahoAshley started posting a new project each week, like episodes in the TV series. After a while, MahoAshley decided to add a new character to her series. But instead of creating the character on her own, she posted a "contest" on the Scratch website, asking other community members to "Design Izumiko's Sister". MahoAshley reviewed all of the submissions, selected a winner, and added the new character to her series. (See the project online at http://scratch.mit.edu/projects/MahoAshley/149263). Another example: A Scratch community member created a "Can You Dance?" contest, based on a popular TV show. (See http://scratch.mit.edu/projects/angelical/64597).

- *Tutorials*. We expected that some educators would use Scratch to create "interactive tutorials" to help students learn Scratch. But we were surprised by the number of young people who created interactive tutorials. For example, when MahoAshley ran her "design a new character" contest (described above), another Scratch community member wrote a comment saying that she wanted to enter the contest, but didn't know how to draw anime characters. So MahoAshley posted an interactive tutorials (created in Scratch), with step-by-step instructions for drawing an anime character. (See the project at http://scratch.mit.edu/projects/MahoAshley/95663). Another example: Some members of the Scratch community wanted to create games with side-scrolling backgrounds, but could not figure out how to do it, so they posted questions about it in the discussion forum. Other community members began producing Scratch-based tutorials, showing the programming blocks needed to produce the side-scrolling effect.

- *Collective Experiments*. An 8-year-old girl needed to do a science experiment for school. She created a Scratch project that tested reaction time, and posted the project on the website. The project asks the user to click on a button each time the color of the screen changes, then it asks the user for some background information (their age, how often they exercise, etc.). With this project, she was able to collect data from hundreds of people, and look for correlations between reaction time and other personal attributes. Before the web, it would have been very difficult for an elementary-school student to conduct such an experiment. See http://scratch.mit.edu/projects/forcemaster/99702

3. Media Arts Education

In studies led by Kylie Peppler, we found that Scratch-based activities offer a foundation for a new "Media Arts" curriculum, integrating technology and the arts. We used Scratch as a core technology for adding "creative coding" to the Media Arts curriculum, enabling students to create dynamic, interactive works of art. We found that students make new connections to art through constructive (or programming) experiences with digital media, not just viewing or interacting with digital media. Our studies show how urban youth, as they work on interactive Scratch projects, learn about and come to understand Media Arts as mix of genres, ideas, and values that they can then use towards expressive and communicative ends.

In field studies at Computer Clubhouses, we found that youth do not necessarily need a strong foundation in visual arts before venturing into Media Arts. Youth can learn core concepts about perspective, color, shape, drawing as they work on programming projects with Scratch. Youth can use animation, constructing gestures, or higher-level aspects of the artistic practices as possible entry points. Youth can learn something from all of these experiences, as working in a digital medium allows youth to enter at whichever level interests them most.

4. Enthusiastic Response from Kids, Educators, and Parents

One of the best ways to understand the impact of Scratch is to listen directly to members of the Scratch community. Below is a message that we received from a 13-year-old girl who is an active member of the Scratch online community:

> My name is Diana and I'm 13 years old. I'm a freshman at Berkeley High School where I'm on the junior varsity cross-country team and am Freshman Class Senator. My favorite subjects are Latin and Chemistry, and I'd like to be a surgeon one day.
>
> My 8-year-old brother, Ken, and I have been active participants of the Scratch Online Community for over a year. What I really like about Scratch is the way it's possible for young kids to create interesting projects, but it's also complex enough that even adults who know many other programming languages still enjoy participating. It reminds me of some trees near our library that are always full of kids -- they have thick branches close to the ground, so that even 2-year-olds can enjoy climbing in them, but it's also interesting enough for older kids to climb. The Scratch Community, like the trees, is always full of groups of people of all ages having fun.
>
> I enjoy designing and programming games and animations with other kids, because different people have different skills, and when you work together, you can build a much better project than you can alone. I also really like talking with my friends online. We help each other with programming tips, share and modify each other's artwork, and talk about fun things in our lives. My brother joined Scratch because he really wanted to program computer games, and our mother wanted to teach him the C programming language, which was way too hard for him. The building blocks in Scratch are just right for kids. My brother has been able to program games all by himself on the Scratch website. He thinks it's really cool!
>
> When I first started on Scratch, I looked at other people's projects and made some of my own. But then I saw that some of the best games were made by groups, called "companies" on Scratch. I joined one company, but everybody was assigned a specific job and couldn't work on anything else, so if one important person didn't have time to work on a company project, nothing got done. My brother wanted to start another company to build more games, so I decided to help him. We asked another online friend of ours (Lanie) to join with us to found a new company that we called "Gray Bear Productions."
>
> Gray Bear Productions was started in June 2008. Ken asked some of his online friends to join, and Lanie advertised the new company on the forums. We started making a game called "Pearl Harbor." We didn't have assigned tasks, so anyone could work on any part of the game they liked. Some kids are good at art, some at programming in general, some at specific programming tasks like scrolling backgrounds, gravity, or shooting. Others have really good ideas for story lines and write well. Everybody could contribute in any way they felt best.
>
> Over the next few months, we all agreed on rules to help the company operate smoothly and to decide who was an official member. We usually only work on one game at a time, and all members of the company vote on the game we want to work on next. We have about 10 to 15 members. I think it's a good model for producing lots of high-quality games. After "Pearl Harbor," we made "Forest Frenzy," "Crash Scratchsanity," and "Night at the Dreary Castle"

(for Halloween). Now we're working on "Tix." Several of our games were featured on the Scratch front page. Some of us think we're the best company on Scratch! :)

What is fun about Scratch and about organizing a company to write games together is that I've made a lot of friends and learned lots of new things. I've learned a lot about different kinds of programming by looking at other games with interesting effects, downloading them, and looking at and modifying the scripts and sprites. I really like programming! Also, when I started with Scratch I didn't think I was a very good artist. But since then, just by looking at other people's art projects, asking them questions, and practicing drawing using programs like Photoshop and the Scratch paint editor, I've gotten a lot better at art. I'm even taking an art class at Berkeley High and getting an A+! I've also made lots of Scratch friends that I occasionally meet online in other games or online forums, such as Runescape or Imperion Online. Another thing I've learned while organizing Gray Bear is how to help keep a group of people motivated and working together. For example, it's important to visit our gallery often to keep people interested in the company. I also like to give positive feedback to company members to encourage them, because I remember how hard it could be to get started on Scratch and share projects.

I like Scratch better than blogs or social networking sites like Facebook because we're creating interesting games and projects that are fun to play, watch, and download. I don't like to just talk to other people online, I like to talk about something creative and new.

Below are more comments about Scratch. We have divided the comments into three categories, based on who made the comments: **kids**, **educators**, or **parents**.

What **KIDS** are saying about Scratch:

"My 8-year-old brother and I have been active participants of the Scratch Online Community for over a year. I enjoy designing and programming games and animations with other kids, because different people have different skills, and when you work together, you can build a much better project than you can alone... What I really like about Scratch is the way it's possible for young kids to create interesting projects, but it's also complex enough that even adults who know many other programming languages still enjoy participating. It reminds me of some trees near our library that are always full of kids – they have thick branches close to the ground, so that even 2-year-olds can enjoy climbing in them, but it's also interesting enough or older kids to climb. The Scratch Community, like the trees, is always full of groups of people of all ages having fun."
- Diana, age 13

"I got Scratch, started playing with it, and have been unable to stop since! Scratch is one of the most fun things I have ever tried!"
- Scratch member, age 14

"When I had first seen Scratch, I had no idea what it was, so I just moved along. But then I went back to Scratch a few more times, and I found out that it was programmed by a normal person like me! When I figured out that I could make games like that, I decided to give Scratch a try... and at my first sight, I knew I had found the programmer I had been looking for!"
- Joseph, age 13

"Making games is something I've always thought would be fun, but I didn't want a really complex programming language. In the beginning, even Scratch seemed complex, but after sticking with it for a while, I've been able to make some amazing games. And the fact that you can share projects so easily makes it even better. Scratch is awesome!"
- Scratch member, age 17

"Using a set of about 100 commands that can be snapped together visually, you can create just about anything and learn the fundamentals of more advanced languages. In addition, there's a whole community of people online who will give you feedback on our projects, and gladly help you with your questions."
- Scratch member, age 14

What **EDUCATORS** are saying about Scratch:

"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."
- Lecia Barker, National Center for Women & Information Technology

"Tell a story, bounce a ball, make an interactive scrapbook, take a multimedia birthday card, draw an ever-changing picture, create a book report presentation, explain physics or explain how to tie a shoe. It's possible. It's within your grasp."
- Teacher, Houston, TX

"There is a buzz in the room when the kids get going on Scratch projects. Students set design goals for their projects and problem-solve to fix program bugs. They collaborate, cooperate, co-teach. They appreciate the power that Scratch gives them to create their own versions of games and animations."
- Karen R., elementary school teacher, Minnesota

"Although designed for a younger audience, we've deployed Scratch at the undergraduate level in introductory computer science courses at Harvard College, Harvard Summer School, and Harvard Extension School. In our view, Scratch lowers the bar to programming, empowering first-time programmers not only to master programmatic constructs before syntax but also to focus on problems of logic before syntax. At the undergraduate level, then we view Scratch as a gateway for students to languages like Java."
- David M., Computer Science Instructor, Cambridge, MA

"My background is in Orff-Schulwerk teaching for music and we emphasize teaching musical concepts through playing games, storytelling, dance, and drama with child centered content. Scratch provides a natural extension to teaching programming and computer science through the same elements, games, storytelling, and simulations with child centered content. Just as we use instruments and singing to teach music, I can envision using Scratch to lay a foundation to teach computer science."
- Christopher Michaud, Elementary School Teacher, Georgia

What **PARENTS** are saying about Scratch:

"I was looking for something for my 8 year-old to start programming with and this is just perfect."
- Justin B., parent

"As a parent, I just want to say how impressed I am by Scratch. I'm trying to redirect my 9 year old son's energies from playing games, to learning more by making them...Scratch just takes all the irrelevant clutter away so that kids have the shortest distance between their concepts and accomplishing them, with the accompanying immediate gratification they are so used to these days. This leaves little opportunity for them to become frustrated and lose interest. Simply brilliant, because it's brilliantly simple."
- Jonathan A.., parent

"Unbelievable software! Already my 7 year old boy has programmed a squirrel that pops up from behind a rock, tilts his head (using whirl), and says 'Hi.' My daughter (10) is now busy programming some sort of lolcat thingy."
- Greg K., parent

"I wanted to let you know how much our daughter Geneva enjoyed the Scratch class yesterday. She was beside herself with excitement and could barely stop talking about what she had done in class until she fell asleep. Thanks so much for organizing this and including her in the class. It sounds like the perfect way to get her interested in computers that goes beyond using them to play games and draw pictures!"
- Kurt S., parent

"My homeschooled 10 year old son, Ben, (aka Grunt590) enjoys your software '100%'. After my husband discovered your software and downloaded it, programming (according to Ben) was 'as easy as pie'. He has taught himself how to use it. He says it is the best 2d program. Better then GameMaker. He says your software lets the programmer's imagination go as far as it can, is very user friendly as far as 2d software goes.
- Susan R., parent

"This is the best!! My 6 year old wants to make a video game, and I didn't know how to help him. But a friend fowarded me Scratch, and my son is thrilled. Even I was able to create something! So, just a quick thanks from a happy parent."
- Julie M., parent

"Scratch totally rocks, and I have the perfect evidence: my 8 year-old son who discovered Scratch six months ago and would, if allowed, spend eight hours a day playing with Scratch. I've been watching him turn into a programmer by stealth. From his perspective, he's not learning programming, he's playing….When I first showed Scratch to my son, I thought I'd have to spend a couple of days showing him how to use it, how to put statements into the proper pane, and so on. I gave him a quick tour and then got distracted by something. When I was ready to resume the next day, it was too late: he'd already figured it out and was merrily on his way."
- Bernard G., parent