EARLY COMPUTING EDUCATION

ENGAGING NOVICES IN PROGRAMMING, EXPERIMENTING, AND LEARNING WITH DATA

Sayamindu Dasgupta and
Mitchel Resnick

We describe a number of new and upcoming initiatives within the Scratch project that focus on introducing programming with data to young programmers. These initiatives are placed in the context of emerging opportunities and trends in computing such as the cloud, open data initiatives, and ubiquitous sensor devices. We describe how the initiatives take advantage of these trends to enable young programmers to think not just about data, but also about the role of data in their world through computational projects that are personally engaging and meaningful.

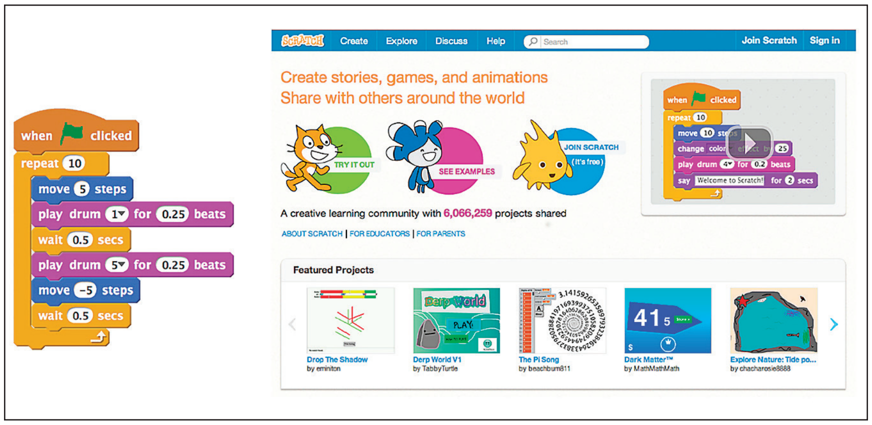
Many real-world computer-programming tasks are centered on the collection and manipulation of large amounts of data. For example, a social scientist may have to build a custom online survey tool for his research, and after spending months collecting data through the survey, he may have to program yet another tool for analyzing the results for patterns and correlations. In another example, an astronomer may have terabytes of data collected from weeks of astronomical observations, and she may need to write a custom program to analyze this data.

None of the people in the examples above are professional programmers, but yet, as a part of their day-to-day jobs, they have to write computer programs that can efficiently deal with large data sets. Understanding and managing data sets is increasingly becoming an integral part of not only our professional lives, but also of our personal lives, as we record and analyze our daily activities with fitness trackers, share data in various forms with our friends and acquaintances through social networks, etc.

Most introductory computer science and programming courses that are taught today, however, focus on the "process" aspect of programming, focusing on algorithmic concepts and related areas such as flow of control and rarely on data. While these concepts form the fundamental pillars of programming, as a side effect of this trend, toolkits for novice programmers rarely deal with data, beyond relatively simple uses of variables, lists, and key-value pairs. Also, most introductory computer science lesson plans and computational thinking frameworks do not go much beyond spreadsheets and pen-and-paper surveys when it comes to computational explorations of data.

In this article, we discuss some of the emerging opportunities that promise to enable a much wider variety of computational projects around data, especially at the introductory level. We describe these opportunities in the context of our ongoing and upcoming work with Scratch [8], a visual block-based programming language and online community developed by our research group at the MIT Media Lab. And, we describe a number of new data-oriented features in Scratch and discuss how they influence the ways in which children think about the world.

Scratch has been designed especially for young programmers ages 8-15, but is used for introduction to programming across a much wider age range. Programming in Scratch is done through visual blocks, which can be snapped together (virtually) to create a sequence of instructions for on-screen graphical objects called sprites. Using Scratch, millions of children all over the world have created interactive animations, games, simulations, stories, and more. Scratch also has an online community website, where young Scratch users can share their projects, remix projects created by other users, comment on each other's projects, and "follow" each other, etc. (Figure 1).



```
when [flag] clicked
repeat 10
    move 5 steps
    play drum 1 for 0.25 beats
    wait 0.5 secs
    play drum 5 for 0.25 beats
    move -5 steps
    wait 0.5 secs
```

Scratch
Create
Explore
Discuss
Help
Search
Join Scratch
Sign in
Create stories, games, and animations
Share with others around the world
TRY IT OUT
SEE EXAMPLES
JOIN SCRATCH (it's free)
A creative learning community with 6,066,259 projects shared
ABOUT SCRATCH | FOR EDUCATORS | FOR PARENTS
Featured Projects
Drop The Shadow by eminton
Derp World V1 by TabbyTurtle
The Pi Song by beachbum811
Dark Matter™ by MathMathMath
Explore Nature: Tide po... by chacharosie8888

Figure 1: Scratch code and the Scratch online community homepage

The core design approaches of Scratch are grounded in the principles of constructionism, where Scratch enables learners to construct public entities (projects). These projects become "objects to think with," where the process of constructing these projects offers opportunities to engage in not only computational thinking, but also in "thinking about thinking" (e.g., any exercise in debugging code becomes an examination of one's own thought process) [6]. All this is situated in the context of the Scratch online community, a venue for exchanging ideas, learning from others, and collaboration with others.

DATA AND THE CLOUD

Scratch 2.0, launched in 2013, moves the code editor to the Cloud, which means that Scratch users can program and create their projects directly in the browser, and save their creations on the Scratch website directly. A consequent effect of this design is that Scratch users, when they are creating a project, or running one, are online and connected to the Internet.

In some ways, Scratch projects can be thought of mini "web-apps" running in the browser. Traditional web apps often come with support for storing data online ("in the cloud"), and, in Scratch, we designed a new feature called Cloud data that adds an online, persistent storage layer to Scratch projects.

Cloud data enables Scratch programmers to build projects that can store data online, and each Scratch project has a Cloud data-store that is shared persistently and in real-time between instances. The data storage functionality is implemented through extending Scratch data-structures (scalar variables and lists), where a given data-structure can be marked as a "Cloud variable" or "Cloud list," making it persistent. Persistence and sharedness make it possible to create projects such as surveys (where each survey choice is represented as a cloud variable), high score lists, etc.

Scratch users have created a wide variety of projects with Cloud data. One of the more popular genre of Cloud data projects consist of comparatively simple "collaborative clicking" projects, where a shared Cloud variable is incremented every time someone clicks on a button in the project, and the goal is to collaboratively reach a certain milestone (e.g., 100,000 clicks). While this type of project itself is simple in terms of programmatic complexity, the jump from traditional Scratch data-structures to shared and persistent data-structures, in terms of functional understanding, is significant.

Analysis of Cloud data projects in the Scratch website show that the single most popular use of the feature is for high-score leaderboards in games (more than 20%) [2]. This illustrates how young programmers, who will not normally work on a "data-only" project, use data to extend the functionality of a project belonging to a genre that they care about.

The "wide walls" approach of Scratch [8], where the tool is designed to engage children having a wide variety of interests and passions, gets reflected in the larger corpus of Cloud data projects. Cloud data projects include

- surveys (Figure 2);
- multiplayer games using Cloud data structures as message-passing systems;
- crowd-sourced pixel art projects where the entire Scratch canvas is turned into a grid, with grid element (pixel) states represented by a persistent Cloud list; and
- virtual currency transaction systems called "Sipcoins," inspired by Bitcoins [2].

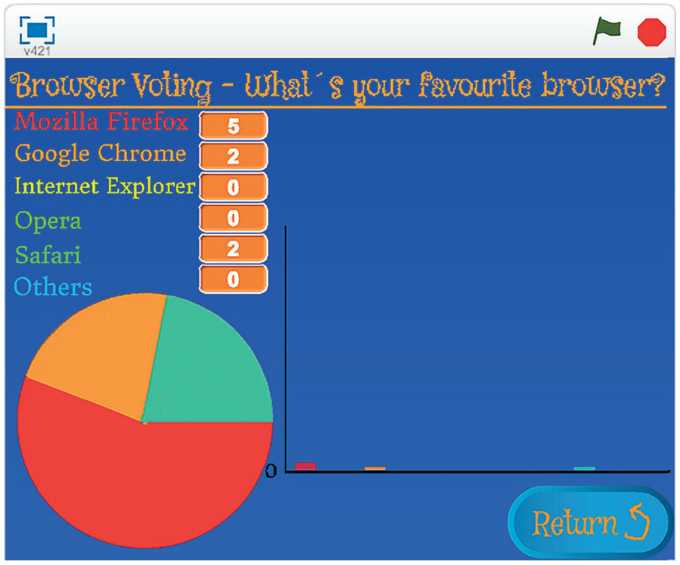
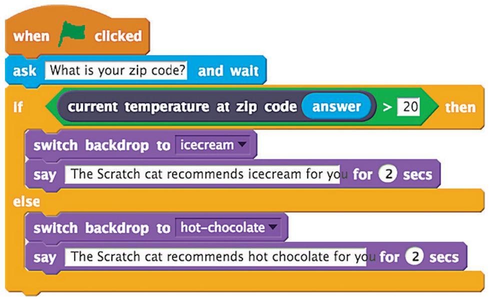


Figure 2: Cloud Data Survey Project on Favorite Browser

ONLINE DATA SOURCES

Support for persistent data, however, is only one small slice of the possibilities as we move towards programming with data in an online coding platform. Over the last few years, a large number of open-data resources have started to appear online, and these range from real-time meteorological data to rich geographical data sets made available by government agencies and municipalities. These data resources have given rise to a large number of innovative and useful tools, visualizations, etc. Additionally, a subset of these resources has also contributed significantly to the increase in richness and variety of sample data sets intended to be used for programmatic explorations of data.

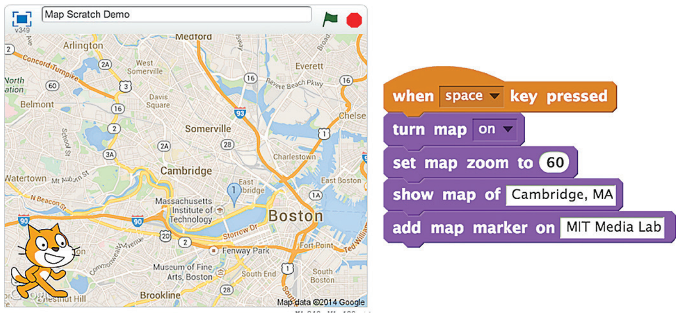
This emergence of open-data resources was one of the motivating factors behind a new feature in Scratch, known as the Extension System. The Scratch Extension System is a framework that enables anyone to extend the Scratch vocabulary by implementing custom programming blocks using JavaScript. Thus, for example, a Scratch user familiar with JavaScript can create a Scratch extension that provides blocks that fetch real-time weather data for any location on the globe, using an online web service like weather.com. This extension, shared with the larger community, can enable other Scratch users to incorporate weather information into their own projects. In a sample project that was developed to illustrate the use of the extensions system, the user running the project is prompted to enter the city that he lives in, and then, based on the current temperature in the city fetched through the extension, he is recommended ice cream or hot-chocolate (Figure 3). While still in early beta mode, we have already seen extensions from enthusiastic Scratch users that fetch the latest statistics on the recently concluded soccer world-cup, translate text using online web-services, lookup definition of words using online dictionary services, and so forth.



```
when [flag] clicked
ask [What is your zip code?] and wait
if <(current temperature at zip code (answer)) > [20]> then
    switch backdrop to [icecream v]
    say [The Scratch cat recommends icecream for yo] for (2) secs
else
    switch backdrop to [hot-chocolate v]
    say [The Scratch cat recommends hot chocolate for yo] for (2) secs
```

Figure 3: Project using Scratch Extension to fetch real-time weather data

As a large number of open datasets come with geo-encoded data (i.e., where data-points have geographical coordinates), another ongoing test feature in Scratch is map support (Figure 4). Here the background of the Scratch stage is replaced by satellite imagery or street maps from Google Maps, and objects on the Scratch stage (called sprites) are aware of geographical coordinates through blocks like "*move to location ()*." While there is a wide variety of uses of map support in Scratch (e.g., interactive tours of neighborhoods, stories of vacation trips, geographical games like Geoguessr [3]), any project that uses geo-encoded data will also benefit significantly from this feature.



```
when [space v] key pressed
turn map [on v]
set map zoom to (60)
show map of [Cambridge, MA]
add map marker on [MIT Media Lab]
```

Figure 4: Map programming blocks in Scratch

Another compelling source of data for Scratch users is the Scratch online community itself. The community is a rich source of data, both at a macro level (e.g., overall community statistics), as well as a micro-level (data on a specific project such as the number of users who have favorited it or data on a specific user such as the number of projects shared by the user). Scratch users in the past have used macro-statistics to celebrate community milestones, for example, creating projects to commemorate the millionth shared project. Through an initiative that we have just started, we plan to provide access to this data through Scratch blocks—effectively, we are making the Scratch website application programming interface (API) available from within Scratch. Through these blocks, young users of Scratch will be able to analyze and visualize their online learning activities and participation within the Scratch community. As most of the public data or metadata on the Scratch website will be available through these Scratch blocks, these blocks will enable a wide range of creative uses, ranging from extension of Scratch website features (e.g., list of projects being bookmarked by a set of users), to block analysis tools (e.g., find computational music projects by looking at the block repertoire used in each member of a set of projects).

HARDWARE SENSORS

Apart from online sources of data, another compelling class of data-sources consists of hardware sensors. In developed countries (and increasingly, in developing countries) the rise of smartphones means that everyone is carrying a sophisticated sensing device with them. A modern smartphone device can detect ambient light, acceleration, geographical location, barometric pressure, and more. Low cost sensors are also becoming ubiquitous, with platforms like the Arduino and the Raspberry-Pi emerging as easy-to-use, affordable, and readily available controllers for these sensors. The Scratch Extension system, described above, in conjunction with a hardware support browser plugin that we have developed, can be used to enable Scratch projects to gather data from such sensors [4]. This is not a new feature, as older versions of Scratch have had support for devices like the PicoBoard, which had built-in sensors, as well user-attachable ones [5]. However, with the sensor landscape dramatically changing over the last few years and with the emergence of paradigms such as the Internet of Things, novel and creative instances of bridges between the physical world and Scratch is poised to grow even more.

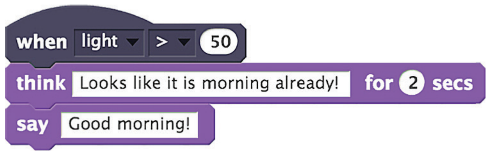


Figure 5: Scratch code using input from a light sensor

LEARNING AND UNDERSTANDING LEARNING IN A "DATA DRIVEN SOCIETY"

Learners today are growing up in a world and society that is becoming more and more "data driven." Visualization, analysis, and understanding of patterns in data are becoming integral parts and requirements of every field. As illustrated in the examples at the beginning of this article, learning to program with data is becoming an essential skill in almost any field.

With the Scratch extension system and Map Scratch, young programmers gain access to a diverse range of data sets, opening up pathways for different interests and passions. A Scratch user who is interested in soccer can incorporate authentic, real-world statistics into his fantasy soccer project, while another user who is interested in aviation can make airport control-tower simulator projects with real-world and real-time wind and cloud conditions.

[W]e strive to enable children to create projects that help them think about data and their world.

Additionally, learning itself is also changing. As we learn today, our learning activities and patterns are increasingly being analyzed and mined in ways that were simply unimaginable even a few years back. In such a scenario, it is crucial for designers, educators, and researchers to enable young learners to understand the "data driven" world around them. A 2012 issue brief on educational data mining and learning analytics from the US Department of Education [1], summarizes this need in the following terms.

> As colleges and schools move toward the use of fine-grained data from learning systems and student data aggregated from multiple sources, they need to help students understand where the data come from, how the data are used by learning systems, and how they can use the data to inform their own choices and actions.

The Scratch website data block system attempts to address this need in the context of Scratch and the Scratch online community by enabling learners to build tools to understand their own learning and their own learning community.

In addition to the above, there are also larger topics around data with which learners need to engage. Questions and ideas about privacy and anonymity have never been as relevant in the context of learning and education as they are now. While analyzing uses of the Cloud data system in Scratch, we have started to see children, as they build projects that collect data (through surveys or crowdsourcing), begin to engage with topics like privacy and scale in their own terms, through discussions in the online community. As they move from being consumers of data-tools to creators, these children start to understand the privacy and anonymity implications of these tools from a completely new perspective. Through the process of building systems that collect data in the form of surveys, high-score lists, virtual transactions, etc. young Scratch programmers realize the nuanced ways in which they have control over the data that they collect and store.

With the initiatives that we have outlined in this article, we strive to enable children to create projects that help them think about data and their world. The initial results from these initiatives have been positive, but a lot more needs to be done. A few years ago, an 11-year-old Scratch user wrote on a public blog,

> I have made many projects. Now I have what I call a "Programmer's mind." That is where I think about how anything is programmed. This has gone from toasters, car electrical systems, and soooo much more. [7]

Through programming with Scratch, this young Scratch user was starting to see the world around him in a new way. In a similar fashion, we hope that as we design, create, and share the technologies and toolkits described in this article, we help a new generation of learners better understand the world of data they are stepping into. Ir

ACKNOWLEDGMENTS

The work described in this article has been guided by discussions with Natalie Rusk and other members of the Scratch team at MIT. Brian Silverman and Paula Bonta from the Playful Invention Company have also provided invaluable advice, along with Professor Hal Abelson from MIT CSAIL. John Maloney, Shane Clements, and Chris Willis-Ford have carried out a large part of the technical design and implementation. Financial support has come from National Science Foundation (grant numbers 1002713, 1027848, and 1417952). We would also like to acknowledge the contribution of the wider community of Scratch users, who continue to inspire us with their creativity, ingenuity, and enthusiasm.

REFERENCES

[1] Bienkowski, M. et al. *Enhancing teaching and learning through educational data mining and learning analytics: An issue brief.* US Department of Education, 2012.
[2] Dasgupta, S. "Surveys, collaborative art and virtual currencies: Children programming with online data." *International Journal of Child-Computer Interaction.* 1, 3–4 (Sep. 2013): 88–98.
[3] GeoGuessr; https://geoguessr.com/. Accessed July 2014.
[4] Idlbi, A. Y. Personalized Extensions: Democratizing the Programming of Virtual-Physical Interactions. Masters Thesis, Massachusetts Institute of Technology, 2014.
[5] Millner, A.D. Computer as chalk : cultivating and sustaining communities of youth as designers of tangible user interfaces. PhD Thesis, Massachusetts Institute of Technology, 2010.
[6] Papert, S. Mindstorms: *Children, Computers, and Powerful Ideas.* Basic Books, Inc., 1980.
[7] Resnick, M. Learn to Code, Code to Learn; https://www.edsurge.com/n/2013-05-08-learn-to-code-code-to-learn. Accessed July 2014.
[8] Resnick, M. et al. "Scratch: Programming for All." *Communications of the ACM* 52, (Nov. 2009): 60.

SAYAMINDU DASGUPTA, MITCHEL RESNICK
MIT Media Lab, 75 Amherst Street, Cambridge, Massachusetts 02142 USA
sayamindu@media.mit.edu, mres@media.mit.edu

Categories and Subject Descriptors: K.3.2 [Computers and Education]: Computer Science Education; D.3.3 [Programming Languages]: Language Constructs and Features – Data types and structures
General Terms: Human Factors, Languages, Design
Keywords: programming, data, scratch

DOI: 10.1145/2684721.2684737 © 2014 ACM 2153-2184/14/12 $15.00