# Designing a Community to Support Long-term Interest in Programming for Middle School Children

Kyle J. Harms, Jordana H. Kerr
Washington University in St. Louis
1 Brookings Dr.
St. Louis, MO 63130
kyle.harms@wustl.edu
jhkerr@wustl.edu

Mark Santolucito
Amherst College
Department of Computer Science
Amherst, MA 01002
msantolucito13@amherst.edu

Terian Koscik
Grinnell College
1127 Park Street
Grinnell, IA 50112
koscikte@grinnell.edu

Michelle Ichinco
Tufts University
Department of Computer Science
161 College Ave.
Medford, MA 02155
michelle.ichinco@gmail.com

Alexis Chuck
Pomona College
Department of Computer Science
333 North College Way
Claremont, CA 91711
alexis.chuck@pomona.edu

Mary Chou, Caitlin L. Kelleher
Washington University in St. Louis
1 Brookings Dr.
St. Louis, MO 63130
mchou@wustl.edu
ckelleher@cse.wustl.edu

## ABSTRACT

To facilitate long-term engagement in programming for middle school children, we developed the Looking Glass Community. The Community includes both a website and integrated access to community resources within the novice programming environment, Looking Glass. We discuss how we designed the Community to support engagement by providing a source for initial ideas, support for learning new skills, positive feedback, and role models.

## Categories and Subject Descriptors

H.5.2 [**User Interfaces**]: Training, help, and documentation.

## General Terms

Design, Human Factors.

## Keywords

programming for children, independent learning, social networking.

## 1. INTRODUCTION

The last decade has witnessed a severe decline in undergraduate enrollment into technology-related fields in the U.S., although recent trends have shown a slight upward shift [19]. Total enrollment in U.S. undergraduate computer science programs increased 10% [19]. Although encouraging, this slight rise in enrollment will not meet the rising demand for computing skills in tomorrow's job market [18].

Many rising college students have already identified their own interests and aptitudes long before enrolling into a degree-awarding program. Middle school is a critical time when many children, especially girls, decide whether they are interested in pursuing math and science-based disciplines [3]. However, few middle schools can provide such opportunities due to a lack of time in the curriculum and a lack of K-12 teachers with a computing background. Programming environments that enable and motivate middle school students to independently explore programming concepts could serve as a stand-in for the lack of computer science classes offered at the middle school level. However, in order for these environments to be effective substitutes, children will need to use these tools for extended periods of time to develop their programming skills.

Long-term engagement is essential for beginners to cultivate adequate knowledge and an appreciation of programming. Novice programmers are not able to grasp advanced programming concepts instantaneously. Middle school students exploring computer science outside the traditional classroom need an engaging learning environment in which they feel challenged and encouraged; otherwise, they might lose interest before any impact has been made on their understanding and appreciation of computing. A programming environment to support independent learners must help to provide motivation and learning support.

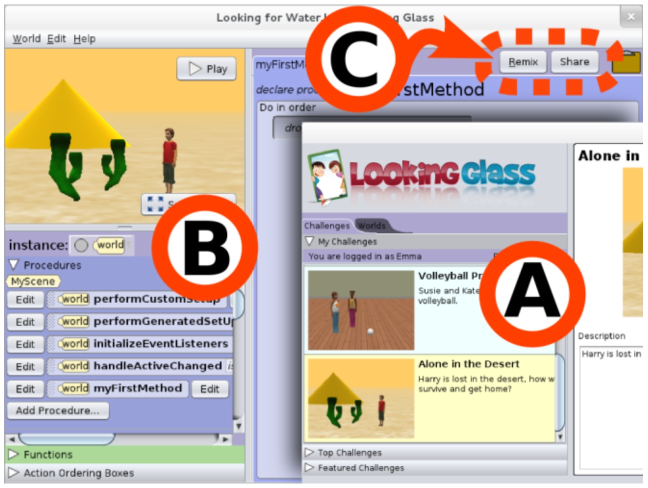


**Figure 1. When opening Looking Glass, the user is first presented with the (A) challenge selection dialog. Once opened, the challenge shows in the (B) IDE. A user can later (C) remix or share their story with the Community.**

To encourage long-term engagement in programming for middle school children, we have chosen to add a community to the novice programming environment, Looking Glass [13]. We have designed the Looking Glass Community to help users come up with ideas for programs, provide opportunities to learn new skills, facilitate a feedback mechanism for encouragement, and develop role models for future growth. We have built the Community to provide a means for users to be exposed to more advanced programming concepts and to develop a continued interest in programming. We describe our design decisions for developing our novice programming community as well as the intended results of our decisions. We also discuss our future plans: evaluating the Community and adding additional learning support.

## 2. BACKGROUND

We introduce the novice programming environment, Looking Glass, as well as related online communities that have been built for use with other novice programming environments.

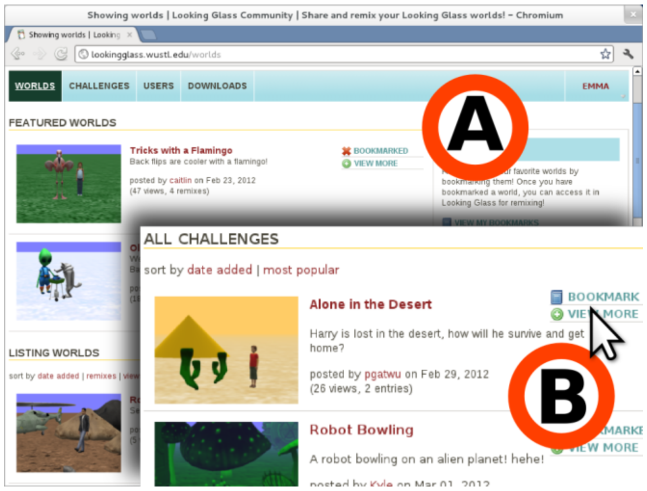
### 2.1 Looking Glass

Looking Glass is a 3D interactive development environment (IDE) designed for middle school children [13]. Looking Glass users program by dragging and dropping programming statement tiles and selecting parameters from interactive menus in order to tell a story with 3D characters. Previous research found that storytelling can motivate middle school children to program [10]. Looking Glass also has a specialized interface designed to enable non-programmers to explore and reuse code based on the 3D graphical output of the running program [4].

### 2.2 Related Work

To keep novice programmers engaged, programming systems need to provide motivation and learning support. Camps [1], workshops [10], and clubs [14] can motivate and teach programming to children; however, most middle school children to do not have access to such opportunities. Children may be able to learn new programming skills independently through tutorials [9], but most users will likely lack the motivation to follow this strategy long-term. Another approach is to remove the frustration associated with programming mistakes by using a virtual assistant which takes the blame for the user's failures [12].

Online communities are also quite popular to engage novice users in programming. The novice programming environments Scratch


Showing worlds | Looking Glass Community | Share and remix your Looking Glass worlds! – Chromium
lookingglass.wustl.edu/worlds
WORLDS
CHALLENGES
USERS
DOWNLOADS
EMMA
FEATURED WORLDS
BOOKMARKED
VIEW MORE
LISTING WORLDS
sort by date added | remixes
ALL CHALLENGES
sort by date added | most popular
BOOKMARK
VIEW MORE
A
B
Tricks with a Flamingo
Back flips are cooler with a flamingo!
posted by caitlin on Feb 23, 2012
(47 views, 4 remixes)

Alone in the Desert
Harry is lost in the desert, how will he survive and get home?
posted by pgatwu on Feb 29, 2012
(26 views, 2 entries)

Robot Bowling
A robot bowling on an alien planet! hehe!

**Figure 2. The (A) Community website with selected programs; programs are known as worlds to the end user. (B) Bookmarking a challenge for fast access in the IDE.**

[15], Greenfoot [8], and Kodu [16] all provide specialized online communities to motivate their users to program. Yet some programming environments, like Alice [2] use a traditional text-based forum to support users. Scratch, Greenfoot, and Kodu employ program sharing and feedback to keep their users continually engaged [8, 15, 16]. While these environments provide positive feedback for new programmers, prior research suggests that learners without a classroom context need 1) support for generating ideas [11] and 2) support for extending their programming skills.

Online communities can provide some support for idea generation through showcasing users' projects. Looking Glass extends this, by organizing around challenges designed to provide a story seed for users without an idea. Existing environments for Scratch and Kodu enable full program based remixing that consists of downloading an existing program, modifying it, and resharing. However, research suggests that new programmers often struggle to identify the code responsible for a given output behavior [5], limiting the potential for learning through full program-based remixing. Looking Glass includes support for users to make connections between output and the code that causes it and remix arbitrary sub-selections of code from within a program. We believe that the integration between Looking Glass and its community, coupled with our code reuse and exploration tools creates a uniquely supportive environment.

## 3. LOOKING GLASS COMMUNITY

Advanced programming skills may be difficult for novice programmers to acquire without long-term engagement and learning. Our goal is to keep users motivated and engaged through Looking Glass over a sufficient period of time to acquire these skills. In order to support this, we hypothesize that programming environments should:

- Support users in generating an idea for a program.
- Help introduce users to new programming concepts and techniques.
- Provide positive feedback from diverse sources.
- Enable beginning programmers to find a role model within the community.

Imagine Emma, a hypothetical middle school student. Emma really enjoys science class at her school, as do a lot of her friends. Some of her friends have started creating 3D animated stories using Looking Glass. Although Emma's school does not have a programming class, she hopes she that can figure out how to create her first story independently. We designed and built the Looking Glass Community to specifically address these problems for users like Emma. The Community consists of the existing Looking Glass IDE (See Figure 1) and a companion website (See Figure 2). Both of these pieces work in unison to provide solutions to the problems outlined above.

### 3.1 Supporting Idea Generation

For users to get the most out of Looking Glass, they need to be able to generate an idea for a story and then be able to realize that idea through programming. Previous research with Storytelling Alice found that users who struggle to come up with a story idea often do not engage with programming [11]. To help these users overcome these creative barriers, we have introduced the concept of challenges. Challenges are empty programs with a previously constructed scene with characters and a specific objective (See Figure 2-B). For example, a challenge might show a boy named Harry in a desert and ask users to explain how he survives.

We have made challenges a central feature of the Community. When users are browsing the website they can bookmark challenges that they might like to attempt later (See Figure 2-B). Additionally, when users open Looking Glass they are presented with their bookmarked challenges and the most active community challenges (See Figure 1-A). We hope that by providing quick and easy access to challenges, many of the users who had trouble getting started in Looking Glass will be able start programming their story more readily. Because challenges start with a pre-made scene, they may also encourage users to start programming quickly rather than spending substantial time setting up a scene.

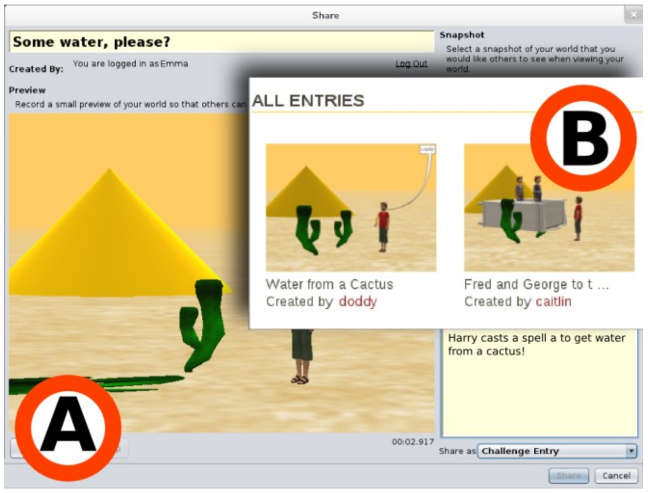
Once the challenge is opened in Looking Glass (See Figure 1-B) the user can program their story to meet the objective of the challenge. As soon as the user has finished programming their challenge submission, they can click the share button (See Figure 1-C) which will allow them to create a preview of their story for the website (See Figure 3-A). Once the program is shared with the Community as a challenge submission, the program can be seen within a list of other submissions in that challenge on the website (See Figure 3-B).

To get a better sense of how this process works, recall Emma's situation. Earlier, she was browsing the Community and found a challenge that she really liked featuring a boy named Harry who is lost in the desert, so she decided bookmark it. Later, Emma opened the IDE and selected the desert challenge from her bookmarked challenges. Emma now decides that she is going to resolve Harry's predicament by programming a story about resourcefulness in the desert. Since the challenge has already put together the characters and scene for her, Emma immediately goes to work programming her story.

## 3.2 Acquiring New Programming Skills

Emma has now programmed a basic storyline into her challenge submission. She has decided to help Harry by programming him to smash a cactus to get water. Because she has just started using Looking Glass, she has not yet discovered some of the advanced programming concepts that would enable her to create a really exciting animation. Despite this, Emma is really driven to make her story the best she can possibly make it.

To facilitate our goal of long-term engagement with Looking Glass, we need our users to feel as though they are making progress and are constantly able to improve the quality of their



Some water, please?

You are logged in as Emma

Select a snapshot of your world that you would like others to see when viewing your world.

Record a small preview of your world so that others can

Water from a Cactus
Created by doddy

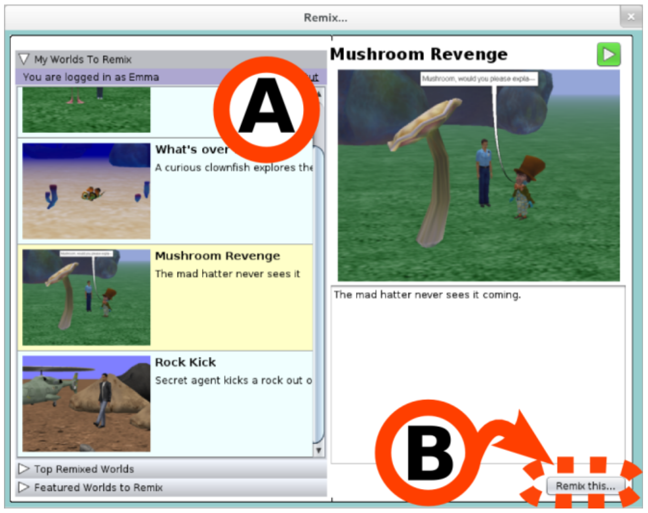
Fred and George to t ...
Created by caitlin

Harry casts a spell a to get water from a cactus!

00:02.917

Challenge Entry

Figure 3. The (A) share dialog in the IDE. Once a challenge submission has been shared, the submission will show (B) on the website with all the other submissions for the challenge.



You are logged in as Emma

What's over
A curious clownfish explores the

Mushroom Revenge
The mad hatter never sees it

Rock Kick
Secret agent kicks a rock out o

Mushroom Revenge

Mushroom, would you please expla...

The mad hatter never sees it coming.

Figure 4. The remix dialog in the IDE; The user can (A) browse bookmarked programs which they can (B) remix into their current program.

stories. We want our users to continually pursue learning advanced programming concepts such as methods and loops. We have observed during informal testing that many users will quickly reach a plateau or complacency with simple programming concepts. These users typically create simple, sequential programs containing many repetitive instructions. Few users seem to independently discover how to make use of other programming concepts to enable more exciting and compelling animations. In a traditional learning environment, a teacher or a mentor might introduce these new advanced concepts to the student. However, as previously stated, many middle school children do not generally have access to these resources.

According to previous research on remixing with a novice code selection interface, a majority of users are able to use advanced programming concepts independently in their own programs after having remixed programs which contained the same constructs [6]. We have introduced the concept of remixing programs into the Community to leverage this result (See Figure 4). Users can browse the website and bookmark the programs they like. We expect many of these programs to have compelling animations with advanced programming concepts that the user may not have been previously exposed to. Once bookmarked, users have the ability within the IDE to remix these advanced programs into their own using the code selection and remixing interface, Dinah [4].

Emma has been working on her challenge about a boy stranded in the desert. Previously, when she was browsing the website, she found a program with a smashing motion that she would like to have in her own program to help stranded Harry gather water from a cactus. Emma opens the Remix interface in the IDE and proceeds to remix the smash animation into her own program. As she selects the actions she wants to remix, Emma notices that the original programmer created the smash motion by using a loop construct. Emma notes that she may be able to use loops in her own programs to repeat actions.

## 3.3 Receiving Positive Feedback

Now that Emma has finished programming her story in response to the challenge about a boy lost in the desert, she would like to know what other people think of her work. Emma is quite proud of her story and she is ready to share it with others.

The Community enables users to easily share their stories to a website where other Looking Glass users can provide direct feedback in the form of advice and encouragement. We believe that by receiving positive feedback, middle school students will be more likely to stay engaged in programming and continue to use Looking Glass as a learning tool.

The Community provides a simple way to share programs to the website to increase exposure and feedback from the IDE. Once shared, friends and family can share a link to these stories, and fellow Looking Glass users can browse the stories and provide meaningful feedback. The website also facilitates project-oriented conversation by allowing "focused" comments on stories. A trend within the Scratch [15] community reveals comments which are often geared toward socialization amongst users and less oriented toward the shared programs. While we want our users to feel engaged in the Community, we have tried to design our feedback forms to cultivate a more program-oriented atmosphere. A comment on a shared Looking Glass program is always marked by the commenter as a question, a "like", or advice. In these three forms, a comment is expected to provide meaningful feedback for the author.

The Community also enables less direct forms of positive feedback. Statistics on shared programs are easily attainable, such as how many times a story has been viewed. A program that has been submitted as a submission in a challenge can also be "liked" by other users viewing that challenge; when a submission is liked, the author is directly notified, and the submission gains a higher status within the challenge. A user is also notified when his or her story has received the highest form of praise: a remix in another user's program.

Emma shares her story with the Community and quickly receives a positive comment from another Looking Glass user. Inspired by the praise, Emma begins searching for a new challenge.

### 3.4 Finding Role Models

Research suggests that a lack of appropriate role models limits girls' participation in computing based careers [17]. The Looking Glass Community affords the opportunity for users to serve as role models for each other. Users can easily browse through not only the projects that their role models have created, but also their list of influences (i.e. the projects that the role models have bookmarked and the challenges which they have participated in).

## 4. FUTURE WORK

We are currently planning user studies to evaluate the Looking Glass Community. We believe that to enhance long-term engagement with programming within Looking Glass, we need to continue to build support for independent learning. Our development plans include integrating automatic adaptive tutorials as part of the remix process based our previous work on tutorials [7]. These tutorials will provide an opportunity to formally introduce new programming concepts that are encountered through the existing remixing process. Further, we plan to integrate mentoring capabilities into the Community to provide help and assistance to further enable middle school children to participate in programming.

## 5. ACKNOWLEDGMENTS

This work was funded by National Science Foundation grant #0835438 and #1054587. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-1143954.

## 6. REFERENCES

[1] Adams, J.C. 2007. Alice, middle schoolers & the imaginary worlds camps. *Proceedings of the 38th SIGCSE technical symposium on Computer science education* (Covington, Kentucky, USA, 2007), 307–311.

[2] Alice: *http://www.alice.org*.

[3] Gill, J. 1994. Shedding Some New Light on Old Truths: Student Attitudes to School in Terms of Year Level and Gender. (1994).

[4] Gross, P. et al. 2011. Dinah: An Interface to Assist Non-Programmers with Selecting Program Code Causing Graphical Output. (2011).

[5] Gross, P. and Kelleher, C. 2010. Non-programmers identifying functionality in unfamiliar code: strategies and barriers. *Journal of Visual Languages & Computing*. 21, 5 (Dec. 2010), 263–276.

[6] Gross, P.A. et al. 2010. A code reuse interface for non-programmer middle school students. *Proceeding of the 14th international conference on Intelligent user interfaces* (Hong Kong, China, 2010), 219–228.

[7] Harms, K.J. et al. 2011. Improving learning transfer from stencils-based tutorials. *Proceedings of the 10th International Conference on Interaction Design and Children* (New York, NY, USA, 2011), 157–160.

[8] Henriksen, P. et al. 2010. Motivating programmers via an online community. *J. Comput. Sci. Coll.* 25, 3 (Jan. 2010), 82–93.

[9] Kahn, K. 1995. ToonTalk(TM)--An Animated Programming Environment for Children. (1995).

[10] Kelleher, C. et al. 2007. Storytelling alice motivates middle school girls to learn computer programming. *Proceedings of the SIGCHI conference on Human factors in computing systems* (San Jose, California, USA, 2007), 1455–1464.

[11] Kelleher, C. and Pausch, R. Lessons Learned from Designing a Programming System to Support Middle School Girls Creating Animated Stories. *Visual Languages and Human-Centric Computing (VL/HCC'06)* (Brighton, UK), 165–172.

[12] Lee, M.J. and Ko, A.J. 2011. Personifying programming tool feedback improves novice programmers' learning. *Proceedings of the seventh international workshop on Computing education research* (New York, NY, USA, 2011), 109–116.

[13] Looking Glass: *http://lookingglass.wustl.edu*.

[14] Maloney, J.H. et al. 2008. Programming by choice: urban youth learning programming with scratch. *Proceedings of the 39th SIGCSE technical symposium on Computer science education* (Portland, OR, USA, 2008), 367–371.

[15] Monroy-Hernández, A. 2007. ScratchR: sharing user-generated programmable media. *Proceedings of the 6th international conference on Interaction design and children* (New York, NY, USA, 2007), 167–168.

[16] Planet Kodu: *http://planetkodu.com*.

[17] Sacco, A. 2008. Young Girls Not Interested in IT Careers Due to Lack of Female Role Models, RIM Study Finds. *CIO*.

[18] 2011. *Computing Education and Future Jobs: A Look at National, State, and Congressional District Data*. National Center for Women & Informaton Technology.

[19] 2011. *Taulbee Survey Report 2009-2010*. Computing Research Association.