Looking Glass: Supporting Learning using Peer Programs

Caitlin Kelleher
ckelleher@cse.wustl.edu

Lack of diversity can create unintentional problems.

How do we get a broader group of people into computer science?
One piece: remove the frustration.

Alice 2 helps retain CS majors

<table>
<thead>
<tr>
<th>Declared CS majors at Ithaca College and St. Joseph’s University</th>
<th>CS1 Grade</th>
<th>Take CS2?</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Alice Class Prior to CS1</td>
<td>C</td>
<td>47%</td>
</tr>
<tr>
<td>Alice Class Prior to CS1</td>
<td>B</td>
<td>88%</td>
</tr>
</tbody>
</table>

But these students already wanted to major in computer science.
And, there aren’t too many of them.

Middle School is potentially a high-leverage window

- Many girls turn away from math and science disciplines, including computer science, during middle school.
- Once they decide to leave, it is difficult to get them back.

For Middle Schoolers Easy Isn’t Enough

“OK, so I can make the bunny move around. But why would I want to?”
- 12 yr old girl

No matter how easy something is, people still need a reason to want to do it.

Why should middle school kids program?
Another piece: Embed programming in a motivating context.

Approach: Present programming as a means to the end of storytelling.

Storytelling Alice Demo

Enabling Storytelling

1. Add high-level animations that enable social interaction
2. Create a story-based tutorial.
3. Provide a gallery of characters and animations that inspire stories.
Generic Alice:
All objects are created equal.

Objects can:
- move
- turn, roll
- resize
- play sound

But....
It’s all about the people.
Storytelling Alice:
Focus on humanoid characters

People need to:
- communicate
- loco mote
- change posture
- attend to
- interact with

Provide custom animations that require explanation in the story

Harold T. Wireton. Crazy go nuts

Animations can be incredibly powerful in helping kids to come up with a story idea.
Does storytelling help?

Get Representative Subjects

88 Girl Scouts from within 1.5 hours of Pittsburgh Troops used participation as a fundraiser.

Isolate Storytelling Focus

Generic Alice (aka “Alice Green”) vs. Storytelling Alice (aka “Alice Blue”)

Keep the mechanical supports for programming constant.

Evaluation Workshop Structure

Control Group:
- Tutorial
- Build something to show in Generic Alice
- Take programming quiz and attitude survey
- Try Storytelling Alice
- Choose Generic or Storytelling Alice to take home
- Show a world to everyone

Experimental Group:
- Tutorial
- Build something to show in Storytelling Alice
- Take programming quiz and attitude survey
- Try Generic Alice
- Choose Generic or Storytelling Alice to take home
- Show a world to everyone

2 hrs, 15 min

30 min
Three Activities in Alice

1: Scene Layout

2: Editing Programs

3: Running Programs

Storytelling Alice users spend 42% more time programming

Time on Alice Activities

- Scene Layout
- Editing Program
- Running Program

Average Time (%)

- p < 0.001

Legend: Generic Alice □ Storytelling Alice
Storytelling Alice motivates reluctant programmers

Scene Layout vs. Program Editing

Time (% on Scene Layout)

Time (% on Editing Program)

- Generic Alice

Storytelling Alice motivates reluctant programmers

Scene Layout vs. Program Editing

Time (% on Scene Layout)

Time (% on Editing Program)

- Generic Alice

Storytelling Alice motivates reluctant programmers

Scene Layout vs. Program Editing

Time (% on Scene Layout)

Time (% on Editing Program)

- Generic Alice

Time on task is a strong predictor of learning

Scene Layout vs. Program Editing

Time (% on Scene Layout)

Time (% on Editing Program)

- Generic Alice

- Storytelling Alice
Time on task is a strong predictor of learning

Users of Storytelling Alice are more likely to sneak extra time to continue programming.

An All-Too Common Scenario

1. Ashley downloads Storytelling Alice.
2. She builds a small story and gets excited.
3. She begins planning a larger scale project.
4. Early on, she gets stuck.

Increased engagement isn’t enough either.
Looking for help.

- Parents/Friends
  - Few have experience with computer programming.
- Teachers
  - CS is rarely taught at the middle school level.
- Internet
  - Some tutorials, but finding one related to a specific question is difficult.

Looking Glass

- Focus on enabling kids to teach themselves in pursuit of their own goals.
- Keep the storytelling and social interaction motivation.
  - Using models learned from a motion library to make it easier to generate appealing procedural animations for humans.

A Looking Glass Scenario

I want them to throw paper airplanes.

Adapt toy throwing to create throwing paper airplanes.

Find code where owner throws toy for dog.

A Looking Glass Scenario

I want them to throw paper airplanes.

Adapt toy throwing to create throwing paper airplanes.

Find code where owner throws toy for dog.
Pre-req: Record high level UI actions

1. Click and drag “Petal Beamweb walk to”
2. Drop it at the top of scene 1 method.
3. Choose “trevor” and “the entire trevor” from the menus.

Provide UI tools to help users find code they want to use in other’s programs

Every program knows the UI steps necessary to create itself.

Pre-req: Record high level UI actions

1. Click and drag “Petal Beamweb walk to”
2. Drop it at the top of scene 1 method.
3. Choose “trevor” and “the entire trevor” from the menus.

Use history for selected code to generate tutorial
A Looking Glass Scenario

First, what are the properties of programs they are likely to find?

How do novice programmers approach finding code in unfamiliar programs?
(with Paul Gross)

Finding Properties of Web Examples

- Randomly selected 15 programs from Alice.org forums

![Pie chart showing distribution of program types]

- Games: 7
- Simulations: 5
- Stories: 2
- Other: 1
Some Observed Properties of the 15 Sample Programs

- **Avg 125 Lines of Code**
- **Large Number of Objects**
- **Poor Naming Conventions**
- **Use of Concurrency**

Design Dimensions Considered in Writing Programs for Search

- **No dialog, all action**
- **Clear object names**
- **Methods and hidden functionality**
- **Long programs**
- **Lots of concurrent execution**
- **Interactivity**
- **Lots of speaking**
- **Ambiguous object names**
- **Long blocks**
- **Short programs**
- **Clear sequential execution**
- **Passivity**

Programs

<table>
<thead>
<tr>
<th>Program</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fish World</td>
<td>No text, all actions, clear object names</td>
</tr>
<tr>
<td>Woods World</td>
<td>Three concurrent ‘parts’ (methods) timed to appear sequential</td>
</tr>
<tr>
<td>Magic Trees</td>
<td>Long program, three concurrent blocks in one method, hidden method functionality</td>
</tr>
<tr>
<td>Race World</td>
<td>Interactive program, lists, events, randomness</td>
</tr>
</tbody>
</table>

How will users use example programs they find?
Expected Use

I want that part... where he bends to pick up the banana.

Tasks Mirror Expected Use

- **Bounding Tasks**
  - Denote begin, end of highlighted functionality

- **Modification Tasks**
  - Modify highlighted functionality

- 5 tasks for each of the 4 programs.

Participants

- 14 adults from the Washington University community
  - Mostly staff (summer break?)
- 12 had no prior programming experience of any kind
- 1 had limited experience with Fortran 20+ years earlier
- 1 had some experience with Matlab 6+ years earlier.

Code Search Study Process

- Code search tasks were randomly chosen.
- Participants completed as many as they could in the remaining time.
Task Data Collection

• Video/audio recording of users, screen
  – Talk-aloud protocol

This is hard for novices.

• 41% correct answers
  – 33% correct bounding
  – 72% correct modification
• Bounding task time range: 01:02 – 26:20
• Modification task time range: 01:07 – 27:29

What Users Do

How do novices approach searching for code?
Task Process Model

Process Model

What Users Focus On: Landmarks

- A specific feature in output or code

What Users Try to Find: Mappings

- A direct connection between a line of code and output function
Task Process Model

Internal Landmark Model

Searching for identified landmarks

Handling Search Failure

- Go back to the output, look for some new landmarks and try again.

- As the frustration really sets in....
  - Look at the API for the character of interest and hope that helps identify a new landmark
  - Start randomly clicking around the interface and hope.
Specific Challenges

- Building mappings is often extremely inefficient.
- Users overlook locations that may contain their target code.
- Novice users do not understand programming constructs.
  - Can lead to errors in temporal reasoning
- Users will include/exclude code based on method naming
  - Likely that they will be searching a program with poorly named methods

A new tool for helping novices find target code

(Demo)

Making Connections Between Code and Output
Helping users know where to look

Helping users know where to look

Helping users know where to look

Focusing attention can help users build an understanding of constructs
Focusing attention can help users build an understanding of constructs.

Next Steps

- Formal evaluation of the code finding tool
- We’re seeing some natural independent learning of programming constructs
  - Can we use deconstruction as a way to teach basic concepts?
- Connecting in the ability to generate the tutorials based on the UI action history.

Other applications for these kinds of tools.

Personal Anecdote (web)

- Finding functionality in unfamiliar code
- Adapting functionality from unfamiliar code

Call your relative working on a Ph.D. in Computer Science

Adapt Relevant Parts
The End-User Programming Gap

- Workers benefitting from programming
- Lack formal training
- Heavy use of examples

United States Programmer Estimates for 2012

- Professional Programmers [1]
- Employees Performing Programming [2]

Questions?

Download Storytelling Alice from www.alice.org
