Scratching Middle Schoolers’ Creative Itch

Joel C. Adams
Department of Computer Science
Calvin College
Grand Rapids, MI 49546
+01-616-562-8666
adams@calvin.edu

ABSTRACT
Each July since 2003, the author has directed summer camps that introduce middle school boys and girls to the basic ideas of computer programming. Prior to 2009, the author used Alice 2.0 to introduce object-based computing. In 2009, the author decided to offer these camps using Scratch, primarily to engage repeat campers but also for variety. This paper provides a detailed overview of this outreach, and documents its success at providing middle school girls with a positive, engaging computing experience. It also discusses the merits of Alice and Scratch for such outreach efforts; and the use of these visually oriented programs by students with disabilities, including blind students.

Categories and Subject Descriptors
K.3.0 [Computers and Education]: General

General Terms
Measurement, Experimentation, Human Factors, Languages.

Keywords
Alice, 2-D animation, computer science, games, gender, middle school, music, programming, Scratch, summer camps, videos.

1. INTRODUCTION
Summer computing camps provide an enjoyable way to introduce young people to computing as a creative endeavor. Such camps provide a relaxed, informal setting where young students – especially young women – can use the computer to express themselves, learn problem-solving skills, have fun, and interact with peers who have similar interests.

Thanks to the broad applicability of computing, summer camp programs have been structured around Alice [1], cryptography [9], robotics [2], web-page authoring [8], and other engaging topics [5]. By providing an early, enjoyable, technology-enriched experience, such camps can provide long-term effects [3], combating the stereotypes that “computing is boring” and “computing is for geeks”. Conventional wisdom holds that these stereotypes are entrenched by high school, so many of these “computing is for geeks”. Conventional wisdom holds that these stereotypes are entrenched by high school, so many of these girls] [3] [5].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Permission to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SIGCSE ’10, March 10–13, 2010, Milwaukee, Wisconsin, USA.
Copyright 2010 ACM 978-1-60558-885-8/10/03...$10.00.

The Imaginary Worlds Camps
Each summer since 2003, the author has directed the Imaginary Worlds Camps (IWC): two self-supporting single-sex summer day camps – one for boys and one for girls – in grades 6 and up. A detailed description of these camps is given in [1].

From 2003-2008, IWC campers used Alice 2.0 (see alice.org) to create computer-generated movies, and in the process, learned about object-based programming. This approach was very successful, thanks to Alice’s integrated development environment (IDE) that lets users create sophisticated 3D animations using drag-and-drop programming. The maximum capacity for each camp is 30 campers; the boys’ camp has filled and had a waiting list each summer, while the girls’ camp has attracted an average of 15 girls each summer. The feedback has been uniformly positive, and over the years, many campers (boys and girls) have returned and attended the camp multiple years.

1.1 The Repeat Campers Problem
There are positive and negative aspects to these repeat campers:
• On the positive side, campers returning indicate success: their experience has been so positive, they want more.
• On the negative side, campers who return a second time have prior experience, and may be bored with instruction that targets novices. Serving both groups of campers within the limited resources of a summer program is a challenge, and the problem gets worse when campers return a third time.

One might simply not accept returning campers. However, this is undesirable as doing so merely turns away young people that IWC has successfully attracted to computing. It is also undesirable when a camp is not full, as is the case with the IWC girls’ camp.

A better solution might be to vary the focus of the camp from year to year. For example, instead of repeating the Alice camp each year, the camp’s focus might be shifted to a different computing-related activity that is equally fun and engaging. By doing so, all campers – first-timers and repeaters – will be novices with respect to the new activity. The challenge is to find a computer-related activity that campers will find as engaging as Alice.

One contender is Scratch (see scratch.mit.edu), which has been used in middle schools [10], after-school programs [7] and college introductory CS programs [6]. Scratch supports 2D graphics and events, has strong sound support (integrated music synthesizer / drum machine, predefined audio loops, plays WAV and MP3 files), and an innovative IDE that lets users build programs by dragging and dropping interlocking puzzle pieces. It also has a social website where users can easily share projects.
Following each morning demonstration, campers moved to the lab to begin applying what they had seen to their own projects. After a lunch break (outside, weather permitting), campers returned to the classroom for a second demo introducing new topics. Afterwards, they worked on their own projects the rest of the day.

At the end of the first afternoon, the students received their flash drives, and were guided through the steps of copying Scratch and their project to it, so that they could take them home.

Each day, the students were given a reading assignment from a Scratch book [4] that Cengage / Course Technology generously provided at greatly reduced cost. This book provided the campers with a useful reference and the readings covered Scratch topics that were not covered in the live demonstrations.

The author organized the camp’s topics as shown in Table 2:

<table>
<thead>
<tr>
<th>Day</th>
<th>Morning Topic</th>
<th>Afternoon Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Scratch Overview</td>
<td>Design &amp; Storyboarding</td>
</tr>
<tr>
<td>2</td>
<td>Broadcasts &amp; Messages</td>
<td>Variables &amp; Expressions</td>
</tr>
<tr>
<td>3</td>
<td>Conditional Logic</td>
<td>Sound</td>
</tr>
<tr>
<td>4</td>
<td>Using Randomness</td>
<td>User Testing</td>
</tr>
<tr>
<td>5</td>
<td>Sharing Your Project</td>
<td>Project Showcase</td>
</tr>
</tbody>
</table>

The sequencing of these topics follows the order in which they become useful in developing a game. For example:

- In the Scratch Overview demo (#1), the author built a simple “Aquarium” animation to present the basic concepts needed to construct a simple 1-level game, including sprites, backgrounds, event loops, sound, and so on. Campers then spent the rest of the morning exploring Scratch, especially its libraries of predefined sprites and backgrounds.

- In the Design & Storyboarding demo (#2), the author sketched storyboards – the sequence of imagined screen shots (title, instructions, level 1, …) – for a multi-level game called “Cat Splat” in which the user uses arrow keys to guide a cat (with nine lives) across a series of increasingly busy highways. Afterwards, campers spent the afternoon drawing storyboards for their games or music videos. When they were done, they had to “pitch” their projects to a helper, who (if necessary) worked with the campers to improve their designs.

- In the Broadcasts & Messages demo (#3), the author showed how to use messages to organize and synchronize the actions in a multi-level game. By defining message-handlers for the stage and sprite specifying their behaviors for each level in the “Cat Splat” game, a message-broadcast could be used to initiate a given level. To illustrate, to display the title and instructions, a “level0” message-handler was defined for the stage and each sprite. The stage’s behavior was to change its background to the title screen, and each sprite’s behavior was to hide itself. Similarly, on receiving a “level1” message, the stage changed its background to a country road; the cat and a car sprites positioned and showed themselves, with the car sprite entering an event-loop that moved it back and forth along the road. Campers quickly saw the benefit of jumping directly to a given level to bypass the preceding levels.
• In the Variables and Expressions session (#4), the author added variables to the “Cat Splat” game to store the player’s level in the game, and the cat’s remaining number of lives. Asked to come up with additional uses for variables, students came up with a game timer, a player’s score, and other uses.

• In the Conditional Logic session, (#5) the author implemented “level2” as a two-way road for the cat to cross, with two cars traveling in opposite directions. Producing the desired behavior required the introduction of conditions, relational operators (< and >) and the OR operator (if the cat is touching car1 OR it is touching car2, it gets splatted). This allowed the difference between OR and AND to be demonstrated in an amusing way – when the OR was replaced by AND, the cat was invulnerable unless it was touched by both cars.

• And so on …

By letting the game development drive the progression of topics, students were introduced to the topics in an “as needed” fashion. This “just in time” pedagogy seemed to work quite well: the need to solve a game-related problem motivated the students to pay close attention as a means of solving that problem was presented.

2.3 The Scratch Website
A strong feature of Scratch is its website – a social website – where registered users can post projects, comment on other people’s projects, “friend” one another, group projects into galleries, and so on. The last morning, campers did a final review of their projects, created accounts at scratch.mit.edu, uploaded their projects, and added them to an IWC 2009 project gallery (http://scratch.mit.edu/galleries/view/53807/). Campers were very excited by this ability to have their creations posted publicly for anyone in the world to see, and the ability to e-mail their friends and family a URL to their project.

With the projects uploaded to the Scratch website, IWC moved to the campus theater for a “World Premier Showcase Celebration” to which the campers’ families and friends were invited. With the author standing nearby for moral support, each camper gave a brief introduction to his or her project, demonstrated it for the audience, and then talked about the biggest obstacle they had encountered during the camp. This public speaking was itself a great challenge for some of our campers; most chose to talk about technical difficulties they had encountered.

3. ASSESSMENT
The desired outcomes were that campers leave IWC having:
A. had a positive computing experience;
B. learned some basic ideas of computer programming; and
C. a positive view of a career in computing.

To assess the achievement of these outcomes, the author used a direct measure – the campers’ projects – and indirect measures including camper pre- and post-surveys. Despite the small sample sizes, IWC 2009 produced some statistically significant results, one of which we share here. The boys’ and the girls’ responses were similar; in the rest of this section, this paper focuses on the girls’ responses unless noted otherwise.

3.1 A Positive Computing Experience
The post-surveys indicate that campers had a very positive experience. Figure 1 presents the girls’ responses to the question:

How would you describe your experience at IWC?

As can be seen, 73% of the girls chose the best possible response, and 27% chose the next-best response. As a validating measure, Figure 2 presents the girls’ responses to a follow-up question:

Would you want to attend a follow-up IWC next year?

The responses remain uniformly positive, with 60%, 27%, and 13% of the girls choosing the top three responses, respectively.

3.2 Programming Basics
To assess outcome B, campers were asked:

Rate how much you know about programming?

on both the entry- and exit-surveys. Response-choices ranged from 0 (Nothing) to 5 (Expert). Figure 3 presents the shift in the girls’ answers between the two surveys:

Figure 3. Change In Programming Knowledge
Both Alice and Scratch have drag-and-drop IDEs. They differ in:

- Alice’s 3D entities are objects. Scratch’s sprites are object-like, but where Alice provides parameterized methods, Scratch provides only message broadcasts and message-handlers.
- On the post-survey, campers were asked to rate the difficulty of computer programming on a scale of 0 (Very hard) to 5 (Very easy). The average response of roughly 220 male and female Alice campers between 2004 and 2008 was 2.7. The average for this year’s 45 Scratch campers was 3.1. However, this difference was not statistically significant (p = 0.1566).
- Alice and Scratch can both be used to build games, movies, or videos. However, the prevailing metaphor for Alice projects seems to be visual story telling; Scratch seems to lend itself more to games. Given the choice of creating a music video or a game, 13 of 15 girls and 29 of 30 boys chose to create games.
- Movies and music videos have a linear structure that requires little in the way of branching behavior. By contrast, games require a great deal of conditional logic and branching behavior. Put differently, when IWC campers create movies and videos, almost none of their projects use if statements. However, when IWC campers create games, every project employs if statements and conditional logic extensively.

4.3 Computing Careers
To help IWC campers see the possibilities in computing careers, they were shown the U. of Washington’s A Day In The Life videos [11]. To assess outcome C, campers were asked:

How do you think it would be to have a job using computers?

on both the entry- and exit-surveys. Choices ranged from 0 (Very bad) to 5 (Very good). Figure 4 presents the girls’ response-shifts:

![Figure 4. Change In View Of Computing Careers](image)

53% of the girls indicated no change, 40% indicated a change of +1, and 7% indicated a change of -1. However, all 8 of the ‘0-change’ girls gave positive responses on both surveys: one chose 3 (Mildly good), four chose 4 (Good), and two chose 5 (Very good) on their surveys. In all, 14 of the 15 girls responded positively to this question on the post-survey. The average shift was +0.3, but it was not quite statistically significant (p = 0.0552).

4. DISCUSSION
This section presents some IWC-related observations.

4.1 Comparing Alice And Scratch
Both Alice and Scratch have drag-and-drop IDEs. They differ in:

- Animation: Alice is 3D; Scratch is 2D. Pragmatically, Scratch projects save and load much more quickly; our Scratch Showcase Session took about ½ the time as in previous years.
- 3D animations look more polished than 2D animations. However, one of the first questions in every IWC has been, “Can we create our own characters?” Thanks to Scratch’s 2D sprite-graphics editor, we were able to answer, “Yes” for the first time in 2009. Pretty much every camper did so.
- Alice users drag code tiles; Scratch users drag interlocking puzzle pieces. Pragmatically, Scratch seems to have an easier learning curve: once novices see how to drag-drop-lock puzzle pieces and launch a script, they are off and running.
Bob loved using Scratch’s sprite editor to create new costumes for his hero and villain sprites. But he was very frustrated by not completing his project prior to the Showcase session. (He was consoled to learn that he could finish his project at home and then update his project on the Scratch website.) Bob reported that what he liked best about IWC was “making my game”; what he liked least was “not getting my game done in time.”

4.2.3 “Chris”
Chris was a 13-year-old attending IWC for the first time. His father contacted the author months before the IWC began, explained that his son was blind, but was very interested in music, and would like to create a music video. Was this possible?

The author invited Chris and his father to campus for a Scratch demo. Chris owns a computer with a Braille interface, but Scratch had no accessibility features or support for a Braille interface.

Chris’ father decided to take a week of vacation and join Chris at the 2009 IWC, to help Chris create a project by serving as his eyes and hands. Chris and his father attended each classroom session where his father watched and Chris listened to the demo. In the lab, they sat at one of the aisle workstations, where Chris would communicate his ideas for what should happen next, and his father would work at making those ideas happen on-screen. Together, they created an interesting story involving a flying saucer, a vampire bat, and mutant humans. Its sound track was a fascinating mixture of bagpipes, the X-Files theme, sci-fi sound effects, and other interesting sounds. Chris reported that what he liked best about IWC was the “sounds library in Scratch”; he left blank the question regarding what he liked least. The author takes that to mean he liked everything!

5. CONCLUSIONS
Having used Alice in previous camps, the use of Scratch in 2009 was fun. Alice’s 3D graphics tend to look more polished, but Scratch’s 2D graphics offer some advantages, including a less-steep learning curve, faster load/save times, and the ability to create your own characters. As a lead-in to Java, Alice’s provision of objects, methods, and parameters seems preferable. But Scratch’s lower complexity and strong support for game-creation (requiring branching behavior) seem to make it preferable for teaching conditional logic and branching. In short, both Alice and Scratch can be used to provide young people with a positive introduction to computing, but since Scratch seems simpler than Alice, perhaps middle schools should use it first, to introduce students to the very basics of programming. After students have mastered conditional logic and branching in Scratch, teachers could switch to Alice to introduce object-based concepts.

Students with disabilities who are able to use a mouse can use either Alice or Scratch, thanks to their drag-and-drop IDEs. Likewise, many students with learning or behavioral disabilities will find the creative and graphical aspects of both programs to be deeply absorbing. There have been almost no behavioral problems at IWC over the years, as students find both of these programs to be very engaging.

Both Alice 2.0 and Scratch are highly visual; neither provides accessibility features for students with visual impairments. Either program may be difficult to use for students with visual impairments, and impractical for blind students, unless sighted persons work with such students as their eyes and hands.

From the author’s perspective, using Scratch in the 2009 IWC was a great success. All involved had a very positive experience learning about programming, and nearly all left with a positive view of computing careers. The author hopes this record will encourage others to plan their own outreach efforts, and looks forward to reading about them.

6. ACKNOWLEDGMENTS
The author thanks Stacy Hiquet of Course PTR (a division of Cengage Learning), who provided reduced-price copies of [4].

7. REFERENCES