Evaluating visual programming environments to teach computing to minority high school students

Conference Paper · December 2013

CITATION
1

READS
67

4 authors, including:

Marvin Andujar
University of South Florida
31 PUBLICATIONS **81** CITATIONS
SEE PROFILE

Luis Felipe Jimenez
Kean University
5 PUBLICATIONS **13** CITATIONS
SEE PROFILE

Patricia Morreale
Kean University
78 PUBLICATIONS **381** CITATIONS
SEE PROFILE

EVALUATING VISUAL PROGRAMMING ENVIRONMENTS TO TEACH COMPUTING TO MINORITY HIGH SCHOOL STUDENTS *

Marvin Andujar, Luis Jimenez, Jugal Shah, and Patricia Morreale
Department of Computer Science
Kean University
1000 Morris Ave., Union, NJ 07083
marvinandujar@gmail.com, {jimluis, shahj, pmorreal}@kean.edu

ABSTRACT

This paper discusses a nine-month research study to determine which of two programming environments, Alice or Android App Inventor, have a greater impact on high school student's interest and learning in computer science. Both interfaces were taught to high school students in this comparative study, and an assessment has been done to determine which environment is superior for teaching computing to high school students. Furthermore, this paper describes how both environments increase student interest in computer science, but the participants retained more information when taught in Alice.

INTRODUCTION

An established pre-college enrichment program, designed to reduce the dropout rate of Latino students in high school, increase academic skills, and encourage higher education, was selected as the site for a comparative study using two visual programming languages, Alice and Android App Inventor. The pre-college program has a 100% high school graduation rate and a 90% college application rate. While a high majority of students enter college from this program, and 35% indicate an interest in majoring in a STEM field, less than 2% of those students selected computer science as their major area of study at university, a disproportionally small percentage given the talent and size of the total student population. In hopes of increasing student interest in college study in

---

computer science, the programming environments Alice and Android App Inventor were taught to the students over two semesters, in hopes that student's interest in computing would increase through exposure to computing, and lead to an increase in the number of students interested in computer science study at the university level. The experiment was designed to teach two grades of students one interface initially for the fall semester, and another interface for the spring semester, with the grades alternating over semesters (Fig. 1). The 9th and 11th graders received instruction in Alice during fall and Android App Inventor during spring. The 10th and 12th graders had Android App Inventor initially in the fall followed by Alice during spring semester.

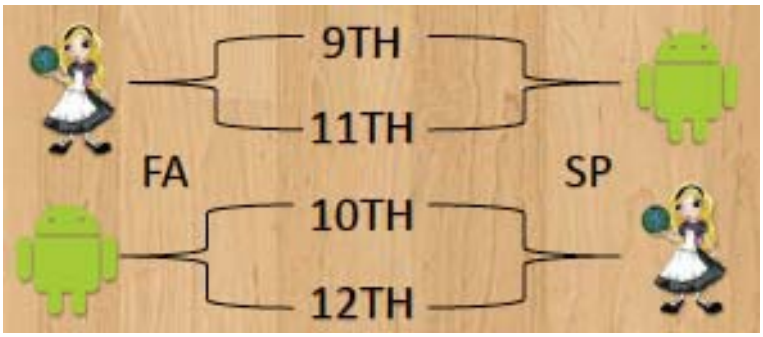

Figure 1. Fall and Spring teaching environments, by high school year.

PROGRAMMING ENVIRONMENTS

The Alice and App Inventor programming environments used for teaching basic computing concepts and early programming approaches. Both environments include visual interfaces.

Alice

Alice is a 3-D drag-drop programming environment created by Carnegie Mellon University to attract and retain more students in computing, particularly females. Designed to teach the enjoyment of programming without the need to remember any syntax [1], Alice was used to teach 9th and 11th graders during fall semester.

The Alice interface (Figure 2) is composed of the following visual elements:

1. An object tree displays a list of objects in the current Alice world, allowing students to select objects.
2. The scene editor allows students to place objects in their 3D worlds.
3. The events section is where the students use events to associate methods with mouse clicks, objects collision, etc.
4. The details area displays methods, functions, and data for the selected object.
5. Students can build programs by dragging methods from the details area [2].

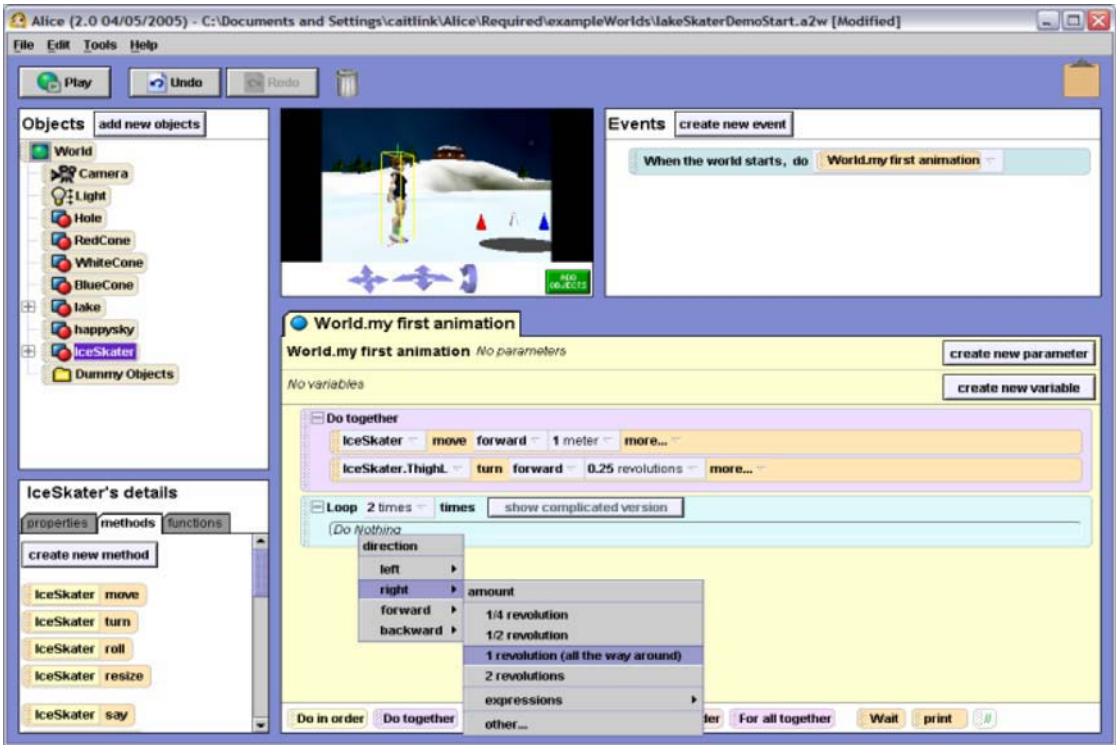

Figure 2. Alice Interface

Each group was taught two lessons using Alice. The purpose of the lessons was to teach the students basic concepts and to familiarize them with the Alice 2.2 interface. This was accomplished by getting the students involved playing with Alice's Worlds, and introducing the students to the first steps needed to build a world and populate it with objects.

Android App Inventor

Android App Inventor (AI), which was used to teach the 10th and 12th grade students, is a visual programming platform for creating mobile applications for Android smartphones. The main purpose is to give non-programmers the opportunity to build their own mobile application without needing to have knowledge of a high-level programming language [3].

Figure 3 shows the complete AI interface. This interface is broken down into the following categories:

1. The AI Designer is where the students build applications by dragging and dropping selected components such as: buttons, pictures, media files, etc. for the applications.
2. The AI Blocks Editor is used to assemble program blocks that specify how the components should behave; also, this is where the students use logic such as *if else* statements and loops. Furthermore, each component is assembled visually, fitting pieces together like a puzzle.
3. The Android Emulator is where the user sees and tests the application they have built (output).

4. The Android phone also represents the output, but this is an external device where the application can be used and not just emulated [4].

For the AI group, two lessons were also given to teach the students how to create a simple Android application using the AI environment. Through these lessons, the students become familiar with the interface and learn how to add functionality to applications.

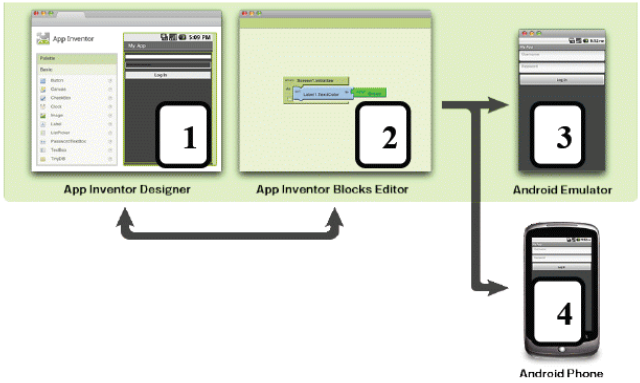

Figure 3. Android App Inventor Interface

Curriculum Instruction and Assessment

The Alice and Android App Inventor lessons were designed to provide a week-by-week skill accumulation. Parallel curriculums were developed for each visual language environment.

1. A general pre-assessment was given the first day of the program (before teaching the lessons); this assessment consisted of 24 questions that provided a background about the students and also about their prior work with computers.
2. Two lessons containing basic computer sciences concepts were created for teaching. Examples of the concepts in the lesson plans contained the lesson are: What is a computer program?, What is an object?, What are methods?, What is an algorithm?, and other basic computer concepts.
3. Two programming exercises were prepared and each exercise was given after each lesson.
    a. The first Alice exercise was a tutorial provided in Alice Interface; the purpose of given this tutorial to students was to familiarize them with the Alice windows. The second exercise was about creating their first world dragging and dropping functions and creating new methods.
    b. The Android App Inventor lessons, ideal for beginners, were given after teaching the interface. The first exercise used a media component provided by the App Inventor and created an application, adding a cat picture. In the second exercise students used a more advanced level of programming behavior to add behavior in the picture and making cat “meow”.
4. A test was given at the end of the semester to measure student’s understanding of the material.

5. Finally, a language-specific post assessment was given at the end of the each semester of teaching; this assessment helped measure the interest and engagement of the students during the teaching.

The questions for the pre- and post-assessments of were developed, using models of existing student surveys and teacher surveys that have been used in previous Teacher Workshops [1] [7] [8].

EXPERIMENT DESIGN

This experiment built on earlier research [5-8], and encourages high school students to consider computer science (CS) as a major through the use of two visual programming environments. The sample size of this study consisted of 71 high school students (67 reported data, 4 absent), from grades 9th to 12th, of Hispanic heritage from the enrichment program. All but six of the students were fluent in English. Many of the students had immigrated to the US recently; the six students who were becoming fluent in English had been in the US less than one year.

Prior the program starting, IRB approval was obtained from the university and consent forms were sent to the parents of the students for approval. Once consent was obtained, during the initial workshop in September, the researchers got to know the students by asking questions about their interests, such as their motivation for participating in the enrichment program and any plans to pursue higher education after high school, these questions were asked, so the high school students get more comfortable with the instructors. At the next workshop each student group was given an assent form, which was an agreement between the experimenters and the high school students that they understand the experimental procedures. Once the assent forms were signed, the researchers began instruction in the two programming environments.

Instructional time was weekly, on Saturday mornings, with all four grades being seen in a 45 minute teaching periods, for a total of four instruction sessions each week. While a 45 min. instruction period is normal for a regularly scheduled class in high school, this research project did not have the reinforcement provided by a high school class meeting five days a week, which yields a total weekly instructional period of 3 hours and 45 min. The high school enrichment program met once a week, resulting in a time difference between the weekend programs and regularly scheduled weekly high school class sessions of 3 hours, which is a significant amount of learning time.

At the end of the semester, a general knowledge quiz was given, asking students several general questions about computing such as: What is an algorithm?, What is a method?, What is an object?, and a few specific questions about the programming environment; this quiz score represents retention. After the quiz was administered, a post-assessment was given to the students to determine if they liked using the environments, and if they would consider majoring in computer science in college.

PRELIMINARY RESULTS

Two tests were performed to obtain the statistical difference between both groups. A two-sample proportion test was done to determine which programming environment

has more impact in the decision of students majoring in CS. It was found that there is not a significant difference between both of them ($P > 0.05$). The second test used was the Wilcoxon rank sum test, which was done to determine the difference in retention, if any. Before this test was conducted, the Shapiro-Wilk normality test was done to determine if there was normality in both groups, but there was not. It was significant and interesting to find that despite there is no difference between the decisions of students to major in CS, there is a difference between the information retention of the lessons given ($p < 0.05$). Overall, this research project data supports Alice as a better environment to teach students basic programming when compared to AI, but both environments still need to be taught more frequently (frequency of instruction), in order to increase and reinforce the interest in computer science in high school students.

Results Classified by Grade

The concluding results were broken down by grades, as each had distinct views of the programming environments they used. From the App Inventor group, 50% of the 12th grade and only 36% of the 10th grade considering a CS major. From the Alice group, more positive results were found, with 55% of 11th grade students considering CS, but only 17% of the 9th graders considering CS.

The reasons students gave for not majoring in CS are presented in Table 1. When asked "Why wouldn't you major in CS?", each student could choose more than one reason. This qualitative data illustrated that most of the students in each grade still preferred their original career choice, partly due to the limited time provided for learning computing sciences; it would take a longer time to change the minds of the students to pursue a degree in CS. Furthermore, the second most-often selected reason was the response that "I don't see myself working with computers for the rest of my life". Other reasons that some students identified were: "I don't have the computer skills needed", "I want to be an automotive mechanic", "too much technology", and "I prefer having center of attention instead of being behind a computer". Some of these reasons reflect the low confidence they have with their computer skills and others reflect self-perceptions as technology users, not technology creators or professionals.

Table 1. Reasons for not majoring in CS by grade

Reasons	12th	10th	11th	9th
I still like my original career choice more	83%	62%	100%	100%
I still think CS is for geeks	0%	8%	17%	0%
I don't see myself working with computers for the rest of my life	67%	38%	50%	30%
Other reasons	33%	23%	0%	10%

At the conclusion of the year, Alice users were delighted with their interface and App Inventor fared almost as well (Table 2). This is a very good assessment of the likeability of the two environments. In contrast, the mean test scores of the Alice students

on the post-assessments given to each grade, after they learned the basic concepts of CS through the programming environments, were higher than the AI group. While both environments appealed to students, greater assessed learning occurred in Alice.

Table 2. Response to the question "Did you like the programming environment?" by grade

Android App Inventor			
Groups	N	Yes	No
12th	12	11 (92%)	1 (8%)
10th	22	20 (91%)	2 (9%)
Alice			
Groups	N	Yes	No
11th	20	20 (100%)	0 (0%)
9th	12	12 (100%)	0 (0%)

Table 3 reports why each grade liked using the programming environments; students could choose more than one reason. Reasons mentioned by the students were: "it is interesting", "I like to create things that are not created yet", "I can put anything I want in the world", "It's something new", "It's a step into my career", "It's easy to use, not complicated at all", "Something new to learn, it was a great experience". The students enjoyed creating small programs using dragging and dropping. The study was slightly better at changing the perceptions of young men regarding future CS majors, than young women. There was no gender differentiation between Alice and AI. Both male and female students liked Alice, but some students from both genders did not like using AI.

Table 3. Reasons for liking the programming environments by grades

Reasons	12th	10th	11th	9th
It is fun to use	100%	67%	85%	83%
I feel comfortable using it	55%	38%	55%	33%
It is interactive and engaging	55%	24%	60%	50%
I like creating worlds	45%	57%	65%	75%
Other Reasons	36%	10%	15%	8%

SUMMARY

Previous studies have shown a lack of female student interest in CS as a major, often because of a perceived lack of interest in the subject [6, 8]. Interest in a complex subject such as CS cannot be increased overnight, or over a series of weekends, as demonstrated here. The lack of exposure to CS from an early age requires that significant amounts of time be spent teaching the environments used here, so that high school students of both genders can have meaningful exposure to the field of computer science. The work presented here suggests that students prefer the Alice environment over AI, after modest exposure. This was determined from their performance in the knowledge test and their consideration of majoring in CS after learning the interfaces. The greatest increase in interest was found in the earlier grades (9^{th} and 10^{th}), which would encourage the earliest possible introduction of computing in the high school or perhaps middle school curriculum, before career choices solidify. It is also clear that interest in computer science increases with exposure and there are not significant gender differences in the preference of Alice over App Inventor. All the students in this study found Alice to be a preferred learning environment, increasing their awareness and interest in computer science tools and techniques.

ACKNOWLEDGMENTS

This project is supported by a grant from the Computer Research Association's Committee on the Status of Women in Computing Research (CRA-W CREU).

REFERENCES

[1] Kelleher, C., Pausch, R., and Kiesler, S. Storytelling Alice Motivates Middle School Girls to Learn Computer Programming. In Proc. CHI 2007, ACM Press (2007), 1455-1464.

[2] Alice. www.aliceprogramming.net/overview/overview.html.

[3] Gestwicki, P., and Ahmad, K. "App Inventor for Android with Studio-Based Learning", *Journal of Computing Sciences in Colleges*, Vol. 27, No. 1, October 2011, pp. 55-63.

[4] Android App Inventor. www.appinventorbeta.com/learn/whatis/index.html.

[5] Beckwith, L. and Burnett, M. "Gender: An Important Factor in End-User Programming Environments?", *IEEE Symposium on Visual Languages and Human-Centric Computing*, pp. 107-114.

[6] Denner, J., Bean, S. and Martinez, J., "The Girl Game Company: Engaging Latina Girls in Information Technology", Afterschool Matters, No. 8, Spring 2009, pp. 26-35.

[7] Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., Malyn-Smith, J., and Werner, L. "Computational Thinking for Youth in Practice", *ACM Inroads*, Vol. 2, No. 1, March 2011, pp. 32-37.

[8] Margolis, J. Estrella, R., Goode, J. Holme, J. and Nao, K., Stuck in the Shallow End, MITPress, 2008.