WATCH: Using App Inventor as a Gateway to Engage African-American Students in Computer Science

ABSTRACT

Workshop for African-American Thinking Computationally and Historically (WATCH) is a program designed to leverage students' interest in computing through mobile application development to help them learn History and Computer Science. This article reports an on-going research project that investigates how historical museum site visits and computational thinking influence the learning processes of rural African American and Latino(a) students. The goal is to foster critical and creative reasoning and help students to become knowledge and artifact producers not just consumers. The study combines historical analysis, fieldwork, and computational artifact creation into a pedagogical approach. This paper will concentrate on the computer sciences aspect of the program, students' attitudes toward computing and app development, and students' conceptual understanding of computing and computer science concepts. Overall, the authors found that students enjoyed the process of creating mobile applications with App Inventor; learned about variables, if-then statements, and conditionals; and increased their interest in computer science and engineering.

Categories and Subject Descriptors

K.3.2 [Computers & Educations]: Computer & information Education – Computer Science education.

General Terms

Design and Human Factor

Keywords

Computer Science education, African American Students, MIT App Inventor, Outreach, Computational Thinking, Computational Thinking Across the Curriculum, Historical Thinking

1. INTRODUCTION

Tools like MIT App Inventor, Scratch, and Alice aim to make programming enjoyable and accessible to novices [8,3]. The difference, and perhaps an important reason for the attention MIT App Inventor has gained, is that it lets people create apps for smartphones. Given the popularity and ubiquity of mobile phones among today's generation of students, App Inventor seems to hold great potential for attracting a new generation of students to computing and computational thinking. With the rise of interest in computational thinking and the lack of computer science classes available for students in middle and high school, we also saw App Inventor as an opportunity to embed computational thinking and mobile app development into a history course.

This paper will discuss our approach to exploring computational thinking through the Workshop for African-Americans Thinking Computational and Historically (WATCH) Program. WATCH is an on-going project aimed at leveraging students' interest in computing though mobile application development to help them learn History and Computer Science. The goal is to foster critical and creative reasoning and help students to become knowledge producers not consumers. This paper will present the curriculum used in the program and answers the following questions: What are rural African American and Latino's students' attitudes toward computing? And how can their interest in mobile technology be leveraged to increase their interest in computer science?

2. EXPERIMENTAL DESIGN

2.1 Participants

This research focused on 30 African-American and Latina/o high school students from rural low socio-economic backgrounds. There were two classes of 15 students each, with a total of 30 students. Across the two classes there were a total of 11 boys and 19 girls, 27 of whom identified as African American, two as Latino, and one as mixed-race.

The study took place during a summer pre-collegiate program for high school students at a major research university in the southeastern portion of the United States. The summer program recruited students two years prior to the study from the five lowest achieving and underfunded high schools in the state. Students were selected based on family income, academic promise, and the potential for being the first generation to attend college. At the time of the study the students were in their final year of the three-year residential summer enrichment program held at our university.

2.2 Study Design

The goal off the study was to engage rural African American students in historical and computational thinking through mobile app design. The study combines historical analysis, fieldwork, and computational artifact creation into a pedagogical approach. We were particularly interested in better understanding how the computer sciences aspect of the program influenced students' attitudes toward computing and app development and students' conceptual understanding of computing and computer science concepts.

Conference'10, Month 1–2, 2010, City, State, Country.

The study was divided into eight sessions over three weeks with a total of 75 minutes per period. Figure 1 illustrates how the sessions were broken down. The first two sessions dealt specifically with historical thinking. The third session was the plantation visit. The fourth session encompassed students debriefing about the trip and storyboarded to begin conceptual development of their apps.

The fifth, sixth, and seventh sessions focused specifically on computational thinking and the development of mobile apps. The activities in this session will be further in section 2.1. The eighth and final session was dedicated to helping students contextualize their thinking about history and why the apps were developed.

8 sessions over 3 weeks **75 mins / session**	
Week 1 **(3 sessions)**	Students learned how to think historically by exposing them to contextualization, the process of sourcing, and historical fieldwork.
Week 2 **(3 sessions)**	Helping the students review the data they collected from their fieldwork, select historical focus of app, develop a historical narrative app using MIT App Inventor.
Week 3 **(2 sessions)**	Finalizing their applications and presenting them.

Figure 1. Classes' breakdown

3. RELATED RESEARCH

3.1 Computational Thinking

Jeannette Wing describes computational thinking as the *mental tools* that allow us to make the best use of our availability to thinking critically [10,11]. Therefore, computational thinking is not about thinking like a computer, but rather thinking about problems from a computational perspective, emphasizing the ideas of what is computable and how computation works. This perspective includes exploring all aspects of the problem, considering the complexity of the problem, and finding an optimal solution that can be achieved with the available resources. Computational thinking has also been defined as a skill that is as fundamental as being able to read, write, and do arithmetic, involving solving problems, designing systems, and understanding human behavior by drawing on the concepts fundamental to computer and as an approach to problem solving that draws on the techniques used in computer science [6, (9) 9, 10].

While there isn't a standard definition for Computational Thinking, efforts have been made by the joint Computer Science Teacher Association (CSTA) & International Society for Technology in Education (ISTE) task force [2] to define and operational definition of computational thinking. CSTA & ISTE define computational thinking in terms of problem solving through the formulating of problems, organization and analysis of data, representation of data through abstractions, algorithmic thinking, and identifying, analyzing, and implementing solutions through combining steps and resources [2]. For the purpose of this study, we use this operational definition because it outlines the process and the product of what computational thinking might look like in a K-12 classroom.

3.2 App Inventor

MIT App Inventor has been presented at a number of recent workshops and has already created considerable buzz within the Computer Science education (CS ED) community. For example, a recent study was conducted to evaluate how effective and suitable App inventor was as a platform for bringing computational thinking to K-12 students [8]. In the study conducted by Morelli et al, students work by themselves to create and design a series of mobile apps, by the end of the course they were able to solve and fix most of the bugs on their application. Furthermore, the students created how-to tutorials to later teach the instructors how to create their mobile apps. The students' enthusiastic attitude and obvious pleasure to show their "cool apps" was remarkable. Students' successful experience reinforced Morelli's hypothesis that App Inventor could be successfully used in a high school and in introductory college course to attract students in computer science[6]. In another research study the combination of two pedagogic factors: the used of App Inventor for Android and an adoption of studio-based learning revealed positive student response to this approach and the participants demonstrated an increase in academic achievement [1]. The experimental approach found a positive effect on minority undergraduate students' motivation, creativity, achievement and attitude towards CS [1]. From these two studies, it is clear that App Inventor is a powerful tool for supporting mobile app development and for increasing students' interests perceptions of computing.

MIT App Inventor like many other MIT programming environments (e.g., Scratch and Starlogo) are specifically designed to engage students in deep thinking about disciplinary content in addition to programming [7, 8]. The Constructionist framework for which these technologies were designed support the notion that people learn particularly well from constructing artifacts and is built upon the constructivist theories of learning [5]. Moreover, the tools have been designed intentionally to shift students from being consumers of technology to producers of technology. This alignment between Constructionist tools, Constructivism, and Historical Thinking is the point from which we sought to incorporate MIT App Inventor as the tool through which to align computational thinking and historical thinking.

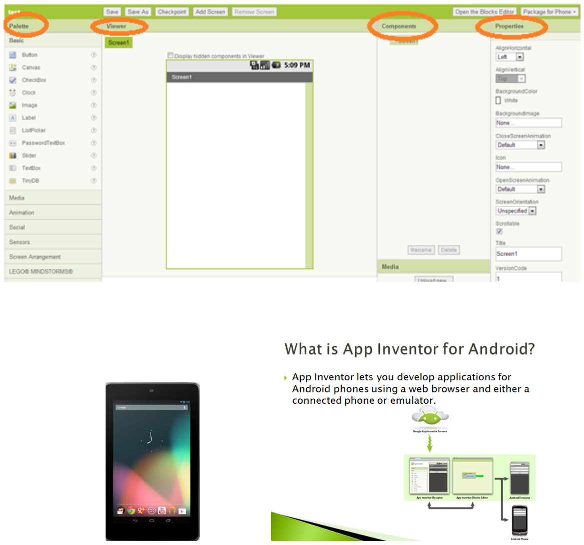
4. OUR APPROACH

We needed a unique approach in order to engage students' interests in history as many students often find history boring. Since most of the students enjoyed interacting with computers, mobile phones, apps, and other technologies, we sought to leverage students' interest in technology to foster the development of new history and computer science knowledge. In particular, we aimed to do this through providing students with opportunities to construct historical narratives and experiences building mobile applications. Using this approach, computer science activities were designed using student interests and real-world authenticity as motivational guides [5, 6]. We coordinated with the historical site to post student apps to augment the educational services already being provided to visitors. Our goal was to show students that their work had real value and that they weren't just creating apps for their class assignment. The students were given just-in-time computer science instruction [3] group activities, individual tutorial and one-on-one instructions. We used tutorials to support students at different comfort levels with MIT App Inventor and programming. This also freed the instructors up to work one-on-one with students who were having a hard time focusing and completing the tutorials.

In the next sections, we describe the three computer science sessions of this study

5. CS SESSIONS OVERVIEW

The first CS session, Session #5 focused on the fundamental principles of Computer Science (CS) to help students have a broader understanding of Computer Science and App Inventor IDE (Integrated Development Environment). The students were introduced to App Inventor and the interface components (e.g., Designer View, Block editor) through a PowerPoint presentation and a live demo of the software. In the live demo the instructor projects her screen and has students follow along on their computers. The purpose of this is for the student to get familiar with the software and to visually see the App Inventor IDE and the components that they will be using while creating their mobile application.



What is App Inventor for Android?

- App Inventor lets you develop applications for Android phones using a web browser and either a connected phone or emulator.

Image 1: Live demo, PowerPoint Presentation and tablet

During this session the students were also introduced to basic computer science concepts (i.e., variables, conditionals, and algorithms). For each of the concepts a small activity was created to ensure the students understanding of the material being taught. The students were asked if they had been previously taught the concepts that were going to be introduced and the students stated that they are not familiar with any of the new concepts being presented. Students were then introduced to algorithms and computer programs. An algorithm was described as a sequence of steps that will help you solve a problem. To help the students understand the concept, the instructor had the students describe the sequence of steps that they will have to go through for going to school each morning. Image 2 is an illustration of the algorithm diagram used in class.

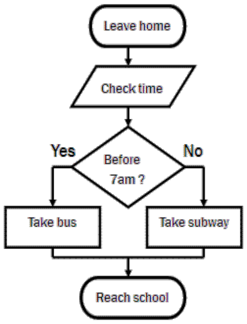
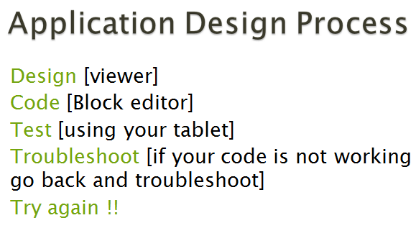


Image 2: Algorithm diagram activity

After walking through the series of steps and decisions they made from the time they woke up until they left for school, students were more familiar with the concept of algorithms.

Next, computer programs were introduced as software that deliver instructions to the computers and told the computer what to do. To reinforce this concept, students were told that people are smart but computer only follow instructions. So if a computer isn't doing what we want it to do we need to give it better instructions.

Last, the students were introduced to a mobile application design process: Design, Code, Test, Troubleshoot and Try Again. This process was implemented with the creation of the mobile application because the students might get discouraged if their application was not behaving or working like they originally planned.



Application Design Process

1. Design [viewer]
2. Code [Block editor]
3. Test [using your tablet]
4. Troubleshoot [if your code is not working go back and troubleshoot]
5. Try again !!

Image 3: Application Design Process

The image 3 explains how the Application Design Process is implement while using App Inventor as the students create their applications. The Application Design Process is significant in the development of the student mobile app because it give the students a design process to follow and a series of steps to track.

5.1 Tutorial Reference

Our goal was to help students personally connect to computer science and increase their interest and engagement. To achieve this goal we created a series of apps tutorials for students to learn about app development by creating two types of apps, first "My favorite Place" and their second one by creating a historical app based on their historical site visit. To help the students recognize that programming is more than creating the historical application and to ensure that students understand how to use App Inventor IDE, an application called "My favorite Place" App was created.

Each student received a "My favorite Place" tutorial that explained in detail how to create the app. In "My Favorite Place" app tutorial students found a picture of their favorite place, recorded a short video and a short introduction where they explained why the place they chose was their favorite place.

Overall, the students were extremely curious and excited about creating their first application. While the students were working on creating the app from the tutorial, the fourth author helped the students who were struggling giving them one-on-one instructions. As a result of giving the students one-on-one instructions they were more determine to finish and show off their apps to their classmate.

Image 4: Students working on "My Favorite Place" app

5.2. CS Lesson 2 - Variables and conditionals

During the second CS session, Session #6, the students were introduced to new computing concepts: variables and conditionals. These concepts were introduced because they were part of the historical app tutorial. The students were not familiar with variables but after explaining the concept, they were able to identify the variable in the tutorial. Variables were explained as being used to store data in a program and each of the variables are used for representing data of a certain type. To further connect this concept to the students, variables were refer as "bucket" where data can be place to later be retrieve by a computer program. Image 5 illustrates how the data can be place inside a "bucket" to later be retrieved and used by a program.

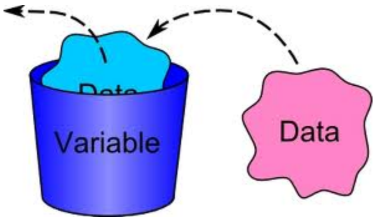
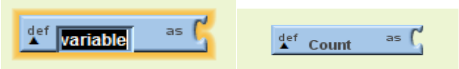


Image 5: Variable refers as "buckets"

To tie variables to App Inventor the instructor explained how to create and re-name components and variable blocks in the block editor. For example, the students created a count variable to keep track of how many times a button has been clicked as seem in image 6.

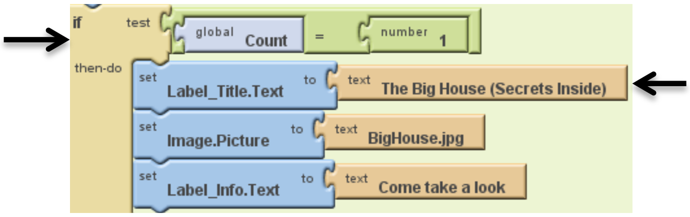


```
def variable as
```

```
def Count as
```

Image 6: Using variables in App Inventor example

For the historical app tutorial, the students needed to understand how to use an if-then-do conditional statement. Conditionals statements were defined in general as a sequence of instructions that repeats or are executed when a particular condition is met. Student used if-then-do conditional statements to control the behavior of the buttons on their app to allow user to navigate between pages. For example, the code in image 7 does the following: it tests when count variable is equal to 1, then displays the title, picture and description text of the page. A student was chosen to be ask a couple questions about the functionally of if then-to conditional statement. The student was able to successfully identity variables and talk about the functionally of the if-then-do conditional statement as describe above.

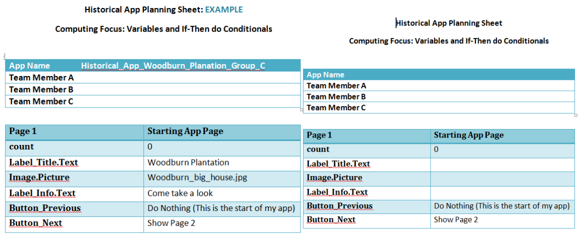


```
if      test  global Count  =  number 1
then-do
        set Label_Title.Text to  text The Big House (Secrets Inside)
        set Image.Picture to  text BigHouse.jpg
        set Label_Info.Text to  text Come take a look
```

Image 7: If statement in App Inventor

5.3 From storyboarding to historical app development

During lesson 3, students were assigned to create a storyboard base on their ideas for the historical application after reviewing the pictures they collected at the historical site. The planning sheet was designed to connect students' storyboard to the structure of their historical app. The planning sheets consist of each page of the application being broken down by title, picture, description text and navigation buttons. Image 8 is an illustration of an example and blank *historical* App planning sheet.



Historical App Planning Sheet: EXAMPLE

Computing Focus: Variables and If-Then do Conditionals

App Name	Historical_App_Woodburn_Planation_Group_C
Team Member A	
Team Member B	
Team Member C	

Page 1	Starting App Page
count	0
Label_Title.Text	Woodburn Plantation
Image.Picture	Woodburn_big_house.jpg
Label_Info.Text	Come take a look
Button_Previous	Do Nothing (This is the start of my app)
Button_Next	Show Page 2

Historical App Planning Sheet

Computing Focus: Variables and If-Then do Conditionals

App Name	
Team Member A	
Team Member B	
Team Member C	

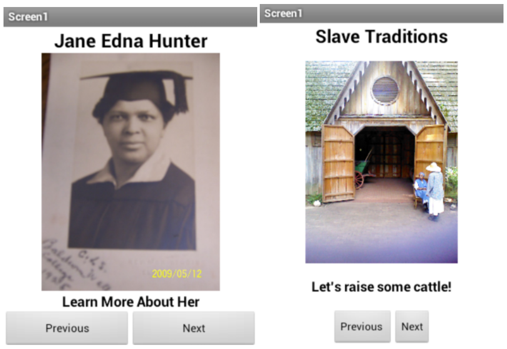
Page 1	Starting App Page
count	0
Label_Title.Text	
Image.Picture	
Label_Info.Text	
Button_Previous	Do Nothing (This is the start of my app)
Button_Next	Show Page 2

Image 8: Illustration of an example and blank Historical Planning sheet

Some students struggled with the transition but Author #1 explained the historical app planning sheet again and after that quick explanation everything went well.

The students were given the Historical Application tutorial that explained in detail how to create their app. Overall, the students had enough exposure to the App Inventor IDE and the location of all the components as well as the computer science concept that were used during the tutorial.

On the other hand, other students were engaged and enthusiastic about finish their historical app. Several students were not as engaged as the instructors had hoped, for this reason one-on-one instructions given to those students and verbally instructed part of the tutorial. By doing this the students were able to continue successfully without help



Picture 9: Two Historical Apps

6. DATA COLLECTED

For this study, we collected data using a series of surveys. Attitude and perceptions of computing surveys were constructed using an adaptation of the 5-point likert scaled computational attitudes and perception survey presented in Ericson & Macklin 2012 [4]. In addition, we designed pre and post surveys using open-ended and likert scaled qualitative questions to get a better understanding of students' prior programming experiences, expectations of the program, experiences in the program, and understanding of computer science and computational thinking, and interests in pursuing a degree in computer science.

To assess students' computer science conceptual understanding and their understanding of App Inventor, we designed a two "Checking Your Understanding" surveys. These surveys used multiple choice questions and questions that asked students to circle pieces of code that represented variables, loops, conditionals, and to describe the functionality of code using screenshots of code in the blocks editor and screenshots of the App Inventor web interface. Students were given a "Checking our Understanding Survey - Part I" at the beginning of Session #6 after completing the My Favorite Place App. Students were given a "Checking our Understanding Survey - Part II" at the beginning of Session #8 before they presented their final apps.

The data was analyzed using simple means and standard deviations to provide a qualitative understanding of students' experiences due to the small sample size. Results are reported in percentages and in indicate the number of students represented by those percentages.

7. STUDENTS EXPERCIENCES

At the start of the program, the majority of students did not have exposure to computer science or programming. 10 students (31%) indicated that their school offered computing courses, but only 4 students (13%) had taken business computing and desktop publishing classes at their school. These courses often do not cover programming. Despite the low enrollment in computing courses at their schools, 7 students (22%) had participated in computer programming workshops (e.g., Scratch, Alice, LEGO Robotics, or Pico Crickets, or Web Design). Of these seven students two students had previous experience designing an app and five students had experience using App Inventor. Despite their lack of experience programming prior to participating in the WATCH program, 19 students (73%) thought they could become good at computing (*M*=3.96; *SD*=1.09). 62% of students (n = 16) wanted to know more about computing (*M*=3.72; *SD*=1.13).

In addition, 26 out of 30 participants indicated that they were interested in creating mobile applications and they wanted to learn the process of developing them. 23% of students (n = 7) expressed their interest in creating an app that others could us as well as for their own personal use. 16 students (53%) indicated that they were interested in learning more about the process of developing apps and learning about computer science. The remaining students indicated that they were just interested in learning something or creating an app. These results suggest that although student lacked experience programming they were excited about learning how to create mobile application.

7.1 Student Responses and Reflection on the Process of Creating Historical Mobile Apps

After the program the students were surveyed to better understand the impact of creating their own historical mobile application. Using a Likert scaled response (5 strongly agree, 4 agree, 3 in between, 2 disagree, and 1 strongly disagree) 79% students (n=23) indicated that they―liked the computer science lessons (App Inventor) taught in the WATCH program‖ (*M*=3.96, *SD*=1.11).

The students indicated their favorite parts of the historical app creation process by checking all responses they agreed with: 67% of the students (n=20) indicated that one of their favorite parts of creating the historical app was the field trip to the plantation. 53% of the students (n=16) indicated that one of their favorite parts of creating the historical app was choosing the pictures for the app. 47% of the students (n=14) indicated that one of their favorite parts of creating the historical app was creating the app interface using MIT App Inventor‘s Design View. 30% of students (n=9) indicated that one of their favorite parts of creating the historical app was programming the logic of their app in the Block Editor. On average students checked two choices for their favorite part of the historical app creation process (*M*=2, *SD*=1.08).

The students indicated the following responses about the most challenging aspects of creating an application from a check all that apply question: 27% of students (n=8) indicated that they found ―Selecting the pictures and finding the information for the historical application challenging. 43% of students (n=13) indicated that they found ―Identifying the palette components in App Inventor‘s Designer view‖ challenging. 47% of students (n=14) indicated that they found ―Programming the components in App Inventors‘ Block Editor challenging. 37% of students (n=11) indicated that they found ―understanding the computer science concepts challenging. On average the students selected 1 to 3 challenges (*M*=1.5; *SD*=0.83);

The students indicated the following responses about the most rewarding aspects of creating an application from a check all that apply question: 63% of students (n = 19) indicated that they discovered how much they liked computer science and app development rewarding. 57% of students (n = 17) indicated that they found creating an application using their creativity and vision as rewarding. 43% of students (n = 13) indicated that they found being able to use the app they created on their android phone rewarding. 43% of students (n = 13) indicated that they found the use of the materials from the site visit in their apps rewarding. On average the students selected 2.06 rewards for working on their historical application (*SD*=1.08);

7.2 Student Engagement and Perception of App Design after participating in the program

After participating in the program, 77% of participants (n=23) agreed or strongly agreed that creating mobile apps was fun (*M*=4.03, *SD*=1.08). 74% of participants (n=22) agreed or strongly agreed that they were engaged while making their app (*M*= 4, *SD*=1.06). 86% of participants (n=24) indicated that they would be interested in creating another app. Some of their reasons for wanting to create another application included a focus on the challenging process of creating the app (e.g.,―It challenged my brain.), the outcome and potential impact of creating an app (e.g., ―It was fun and I like that people can actually download something I made. and ―So I can show others what I know how to do.‖), and the enjoyment of creating an app (e.g.,Because creating my first app was very interesting.).

7.3 Students knowledge retention on computer science concepts

The students took a pre and post survey was they were asked, " What do you think computing/computer science means and what does involves? In the pre-survey 15 out of 30 students responded this question and their answers were generic or under developed understanding. Some of their responses were the following: "studying the fun and adventures in computing" and " means that you working with computers and making certain things. It involves computers." However, when asked in the post-survey 30 out of 30 students answered the same question and 8 students had a well develop understanding on what computer science means and what does it involves. Here are some of the answers: "I think it means the study of computers to make then more efficient, An example would be studying an old computer, then by outing new codes to modify it, updating" and "It means that you work with technology and it involves studying of technology."

To check the students' computer science knowledge retention, the students were given "Checking Your Understanding - Part I and II" at the beginning and end of the program. One of the multiple-choice questions asked students, "What is an Algorithm?" Students were given four multiple choice options to pick from a. A collection of objects, all of the same type, b. A sequence of steps that are going to help you complete a task, c. A list of instructions to perform certain action, d. All of the above. There were two correct answers to this question, choice b and c. On the "Checking your understanding - Part I" survey only 9 out of 30 student answered the question correctly in comparison to "Checking Your Understanding II" where 20 out of 30 students correctly answered the question. This suggests that the students had a better understanding of algorithms by the end of the program. One a reason could be because each computer science concept was explained in detail and related to something the student would remember.

8. DISCUSSION & CONCLUSIONS

In general, the lack of early exposure to computer science among African-Americans and Latinos' students contributes in a significant way to the lack of students actually majoring in a computer science or engineering degree. During the course of this study students' were introduced to computer science through mobile app development and were able to learn basic computer science concepts. In addition, students demonstrated a strong interest in continuing to learn about computer science and to create mobile application. Overall, students felt challenged by creating their own applications. In the words of one participant, "It challenged my brain." Overall, we found that we were successful in leveraging students' interest in mobile technology to increase their curiosity in computer science.

9. ACKNOWLEDGMENTS

10. REFERENCES

[1] Ahmad, K., & Gestwicki, P. (2013). Studio-Based Learning and App Inventor for Android in an Introductory CS Course for non-Majors. *Proceedings of the 18th Annual Joint Conference on Innovation and Technology in Computer Science Education*, *14*, 287-292.

[2] Barr, D., Harrison, J., & Conery, L. (2011). Computational Thinking: A Digital Age Skill for Everyone. *International Society for Technology in Education*, 20-22.

[3] Cooper, S., Dann, W., & Pausch, R. (2003). Teaching objects-first in introductory computer science. *ACM SIGCSE Bulletin*, *35*(1), 191. doi: 10.1145/792548.611966

[4]Ericson, B., & McKlin, T. (2012). Effective and sustainable computing summer camps. *SIGCSE*, 289-294.

[5] Harel, I., & Papert, S. (1991). Constructionism: Research Reports and Essays, 1985-1990. *PsycCRITIQUES*, *37*(10). doi: 10.1037/031551

[6] Morelli, R., De Lanerolle, T., Lake, P., Limardo, N., Tamotsu, E., & Uche, C. (2011). Can Android App Inventor Bring Computational Thinking to K-12. *ACM Technical Symposium on Computer Science Education (SIGCSE).*

[7] Resnick, M. (1996). Beyond the Centralized Mindset. *Journal of the Learning Sciences*, *5*(1), 1-22. doi: 10.1207/s15327809jls0501_1

[8] Resnick, M., Bruckman, A., & Martin, F. (1996). Pianos Not Stereos: Creating Computational Construction Kits. *Interactions*, *3*(6), 41-50.

[9] Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., ... Kafai, Y. (2009). Scratch: Programming for All. *Communications of the ACM.*

[10] Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33. doi: 10.1145/1118178.1118215

[11] Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, *366*(1881), 3717-3725. doi: 10.1098/rsta.2008.0118