Volume 13, No. 5
September 2015
ISSN: 1545-679X

INFORMATION SYSTEMS EDUCATION JOURNAL

In this issue:

The **Information Systems Education Journal** (ISEDJ) is a double-blind peer-reviewed academic journal published by **EDSIG**, the Education Special Interest Group of AITP, the Association of Information Technology Professionals (Chicago, Illinois). Publishing frequency is six times per year. The first year of publication is 2003.

ISEDJ is published online (http://isedjorg). Our sister publication, the Proceedings of EDSIG (http://www.edsigcon.org) features all papers, panels, workshops, and presentations from the conference.

The journal acceptance review process involves a minimum of three double-blind peer reviews, where both the reviewer is not aware of the identities of the authors and the authors are not aware of the identities of the reviewers. The initial reviews happen before the conference. At that point papers are divided into award papers (top 15%), other journal papers (top 30%), unsettled papers, and non-journal papers. The unsettled papers are subjected to a second round of blind peer review to establish whether they will be accepted to the journal or not. Those papers that are deemed of sufficient quality are accepted for publication in the ISEDJ journal. Currently the target acceptance rate for the journal is under 40%.

Information Systems Education Journal is pleased to be listed in the 1st Edition of Cabell's Directory of Publishing Opportunities in Educational Technology and Library Science, in both the electronic and printed editions. Questions should be addressed to the editor at editor@isedj.org or the publisher at publisher@isedj.org.

INFORMATION SYSTEMS EDUCATION JOURNAL

Evaluating the Effectiveness of Self-Created Student Screencasts as a Tool to Increase Student Learning Outcomes in a Hands-On Computer Programming Course

Loreen M. Powell
lpowell@bloomu.edu
Department of Business Education &
Information and Technology Management
Bloomsburg University of Pennsylvania
Bloomsburg, PA 18444, USA

Hayden Wimmer
hwimmer@georgiasouthern.edu
Department of Information Technology
Georgia Southern University
Statesboro, GA 30458

Abstract

Computer programming is challenging to teach and difficult for students to learn. Instructors have searched for ways to improve student learning in programming courses. In an attempt to foster hands-on learning and to increase student learning outcomes in a programming course, the authors conducted an exploratory study to examine student created screencasts and their impact on students' performance regarding specific learning outcomes in a hands-on programming course. This study was conducted over four semesters when an instructor taught two sections of the course per semester; one section generated self created student screencasts in-class and the other section did not. The subjects were undergraduate business students enrolled in an upper level applications/programming course at a university in Pennsylvania State System of Higher Education system. The experimental method was used to compare the differences in graded classroom activities, theory assessments, lab assessments, and final exam scores between the classes. Results showed that students who created screencasts while following along with the instructors step by step programming instructions as well as created screencast while independently working significantly ($p<.05$) performed more successful on theory assessments, lab assessments, and the final exam scores verses those students that did not.

Keywords: screencasts, programming, note taking, differentiated learning, active learning

1. INTRODUCTION

Teaching any programming course can be a challenge. However, when students don't buy the book, outline the chapter, take notes in class, review the content, redo the hands-on course material, nor have access to the computer programming application outside of class, it is impossible to successfully teach programming. Furthermore, the computer lab environment, where students learn hands-on computer programming, often makes it difficult for students to stop and take notes. In an effort to find a solution to these challenges, the authors experimented with screencasts. Screencasts are prerecorded videos that are

designed to capture the author's computer screen and narration (Udell, 2005; Lang & Ceccucci , 2014). Previous research has identified instructor created screencasts as an good instructional tool in higher education (Ashdown, Doria, & Wozny, 2011; Lang & Ceccucci , 2014; Lee & Dalgarno, 2008; Peterson, 2007; Pinder-Grover, Green, & Millunchick, 2011; Sugar, Brown, & Luterbach, 2010; Winterbottom, 2007). However, no one has examined student created screencasts as a way to enhance learning outcomes in a hands-on learning environment. This exploratory study examines student created screencasts and their impact on students' performance regarding specific learning outcomes in a hands-on programming course. This work has practical implications for computer programming faculty and practitioners alike. The remainder of this paper is structured as follows: a brief review of programming pedagogy, screencasts and video usage, the methodology used in this study, results, conclusions and limitations.

2. LITERATURE REVIEW

Computer programming is one of the longest standing components in information technology/computer science degree programs. Computer programming requires students to logically understand abstract concepts, algorithms and data structure design, along with problem solving, testing, and debugging code (Wang, 2010). This subject matter has presented on-going teaching challenges and student learning difficulties (Sleeman, 1986; Ebrahimi, 1994; Jenkins 2002; Kinnunen et al. 2007; Mow, 2008; Nikula, Gotel, & Kasurinen 2011). Hence, it is no secret that teaching programming is a difficult task. The programming pedagogy literature provides a long list of failed methods known to impede students learning. Among the list of reasons as to why programming is difficult for students to learn is the lack of hands-on experience, student follow-up, and peer-driven learning (Babb et al., 2014).

Typically, the lack of hands-on experience occurs outside the classroom as students do not have access to programming software (Mow, 2008). Many students do not purchase programming software or fail to install the free programming software on their personal computers. Hence, not having access to programming software outside of class prevents students from having the necessary hands-on student follow up/content review, a process which is similar to rewriting lecture notes outside of a course.

Long standing research by Howe (1970) reported that note taking aids in student comprehension and recall. Specifically, there is only a 5% likelihood that content material will be remembered when it is not found in lecture notes (Howe, 1970, in Longman & Atkinson, 1999). However, not all students take notes and males take less notes then females (Cooperative Institutional Research program and the Higher Education Research Institute at University of California, Los Angeles, 2008 in the Chronicle of Higher Education, 2009). In 2009, Cooperative Institutional Research program and the Higher Education Research Institute at University of California, Los Angeles (UCLA) studied 26,758 students from 457 institutions and found that only 51% of males take notes in class. More importantly, their data also showed a decline of 7.5% from the previous years study (Ruiz et al., 2010).

Darmouth College is among the many universities and colleges that have developed websites compiled of note taking resources to helps students because "students frequently do not realize the importance of note taking and listening " (Darmouth College, 2013). Hence, the decline in note taking compounded with necessary hands-on experience has made it difficult for students succeed in programming courses.

Sreencasts and Video Usage

There is a large amount of research conducted on the effectiveness of using videos in the classroom. A recent student by Geri (2011) stated that "videos may improve the achievements of students enrolled in a course" (p.231). Additionally, Shultz and Sharp (2013) studied the effectiveness of using instructor created demonstration videos in a programming course. The instructors used Adobe Captivate to create a series of 20 minute videos for the main concepts of each chapter as well as how to program those concepts in C#. They reported that 89% of students (n=35) preferred videos more then text books.

A screencast is a video capture of the desired section of your computer screen that may or may not include webcam narration, voice narration, and text captions (Udell, 2005; Lang & Ceccucci, 2014). Screencasts are similar to video lectures, E-lectures, and e-notes in that they allow students to reflect back upon content previously learned.

Currently, screencasts have been used as instructional aids via instructor narrated

PowerPoint presentations or lectures, problem-solving demonstrations and application demonstrations (Lang & Ceccucci, 2014). Existing research has shown several positive benefits including, but not limited to, student learning flexibility with asynchronous access, instructor tracking of usage, instructor reusability and increased student performance. Most importantly, statistically significant differences, correlations and percentages have been found with students using instructor created screencasts as a classroom supplement (Falconer et al., 2009; Lloyd & Roberson, 2012; Mullamphy, Higgins, Belward & Ward 2009, Pindar-Grover et al. 2011, Lang & Ceccucci, 2014).

Most of the existing research has focused on instructor created screencasts. A recent literature review by Berardi and Blundell (2014) suggested that student created course materials may have the potential to add value in hands-on experience and peer-driven learning. However, there has been little or no research conducted on student created screencasts in-class and their impact on learning outcomes in a hands-on programming course.

3. METHOD

The purpose of this exploratory research study is to understand the value of student's self-created screencasts as a tool to increase students' success in a hands-on application/programming course. Specifically, this study's research question is:

- In a hands-on programming course, will there be a significant difference in the classroom activities, theory assessments, lab assessments and final exam scores for students that self-create screencasts for instructor and independent hands-on programming versus those who did not?

This study was set up as an experiment for over four semesters. Each semester, one class section created screencasts and the other did not. Before starting the semester, the instructor designated one of the classes as the experimental group and the other as the control group. The experimental group created screencasts and the control group did not create screencasts. Subjects were undergraduate students enrolled in a Pennsylvania State System of Higher Education (PASSHE) University. Students were enrolled in eight different sections of an upper level applications/programming course where students learn to program with Scratch, Alice, Visual Basic, and Stencyl.

The applications/programming course met for fifty minutes three times a week. The class was structure so that students spent fifth teen minutes with theory concepts, fifth teen minutes with hands-on instruction and fifth teen minutes independent hands-on student centered learning with instructor supervision and guidance. The instructor always ensured that the students had five minutes upload their screencasts to screencast-o-matic.com before the class ended.

The same course materials (i.e. lectures, book, theory assessments, lab assessments, and final exam) were used. Each course was fifty minutes in length and followed an introduce, reinforce and apply format. Students in the experimental sections were required to record their own Screencast using www.screencast-o-matic.com. Screencast-o-matic was chosen because it was free, required no software to be downloaded and was accessible anywhere with an internet connection. Screencast-o-matic is also very simple to use and does not require multitasking difficulties or toggling between applications.

Upon the start of the first day of class students in the experimental group were asked to sign up and create a free screencast-o-matic account. By creating a screencast-o-matic account, students had immediate online access to store their self-created screencasts online. This made it easy for students to retrieve their screencasts after class. Only students themselves had access to their screencast-o-matic accounts. The instructor did not have access to student's accounts nor did the instructor ask to view student's accounts.

After the experimental group of students established their own screencast-o-matic accounts, they are given seven simple instructions on how to create a screencast:

1. Go to http://screencast-o-matic.com
2. Login
3. Click on "Start Recording"
4. Resize the recording frame to fit your programming screen.
    - To pause recording, click the universal pause icon located at the bottom left side of the screen
    - To restart recording, click the red circle icon located at the bottom left side of the screen

5. When you are finished, click "Done" at the bottom of the screen.
6. Next click "Publish to Screencast-o-matic"
7. Type in the name your screencast and click "Publish"

Students in the experimental sections were asked to record/create their own screencast while following along with the instructors 15 minute hands-on step by step classroom instructions. Students were also asked to record/create their own screencast while independently working hands-on for 15 minutes. Students did not create screencasts during the theory content or theory lecture. Each student created screencast rerecorded during the Instructor led hand-on programming session directly corresponded to chapter content in the programming text.

It is also important to note that the text book used for both the control and experimental group had narrated videos that were created by the publisher to go along with each chapter. While the instructor did not bring this to the students attention, it was presented in the book as a tool to help students.

Throughout the semester, the instructor took attendance at the beginning of class. If students attended all instructional classes they would have a total of 23 self created screencasts of instructor led hands-on programming and 23 self-created independent student learning hands-on programming screencasts. Each screencast was 15 minutes or less in duration. Each screencast only used the video recording of the students screen. Students did not use the video cam or voice recording features.

During all activities, assessments and final exam review classes, the instructor encouraged the experimental group to reference their own screencast as a helpful way to study. Data was collected via the student's assessment scores on the activities, assessments and the final exam for all eight courses.

4. RESULTS

Statistical analyses were conducted using the Statistical Product and Service Solutions (SPSS) software. Descriptive and inferential statistics, including mean, standard deviation, and two-tailed *t* tests were used to test the research question.

The overall sample size included 225 undergraduate business students enrolled in an upper level undergraduate applications/programming course. Table 1.1 and Table 1.2 provide demographic details about the students.

Table 1.1 Class Status

	Senior	Junior	Sophomore.	Freshman
Experimental	71	27	13	0
Control	58	36	19	1

Table 1.2 Gender

	Male	Female	Total
Experimental	93	18	111
Control	88	26	114

As indicated in Table 1.1, there were 111 students in the experimental group and 114 students in the control group.

According to the instructors records, 68% of students in the experimental group attended all of the classes where students created screencasts. A total of 92% of students only missed less then two classes and 100% of students missed less then five classes.

Results indicate that there is no significant difference between the classroom activities grades for the experimental and control group. However, there was a significant difference ($p=.031$) between the theory assessment scores for the experimental group and control groups. The experimental group scored slightly higher ($M=79.61$, $SD= 6.31$) than the control group ($M=72.17$, $SD=8.44$).

A significant difference ($p=.048$) was also found between the lab assessment scores for the experimental group and control groups. The experimental group scored higher ($M=85.33$, $SD= 5.71$) than the control group ($M=73.30$, $SD=7.74$).

Another significant difference ($p=.026$) was also found between the final exam scores for the experimental group and control groups. The experimental group scored higher ($M=89.22$, $SD= 7.42$) than the control group ($M=82.27$, $SD=9.55$).

Additional analyses were conducted on gender and class status (Freshman, Sophomore, Junior, and Senior). However, there were no significant differences found.

5. CONCLUSION AND LIMITATIONS

The results indicate that by having students create their own screencast while following the hands-on instruction from the instructor as well as independent hands-on student work can be a useful tool to increase student learning outcomes. Test scores in key content areas were enhanced for those students who created screencasts versus those who did not create screencasts but may have taken notes.

This research is important information because currently fewer students are taking notes in classes. This study helps to encourage note taking via student created screencasts. By encouraging students to create their own screencasts during hands-on instruction periods, a useful tool is created for students to review hands-on class content at a later time. Additionally, by having the student create the screencast, the instructor is placing the responsibility for a successful learning experience on the student.

This study is not without limitations. This study made no attempt to control for variables that may impact student performance on activities, theory assessments, lab assessments, and the final exam other then the students in-class screencast creation. Additionally, students were not surveyed or interviewed following the course, so it is uncertain if the experimental students used their created screencasts to review classroom materials for graded activities, theory assessments, lab assessments and the final exam. Furthermore, the authors are uncertain if students in the experimental group collaborated or shared screencasts with students in the control group.

Nevertheless, this study demonstrated how student created screencasts can be used as a tool to increase learning outcomes of a hands-on programming course. Further research should better control variables for construct validity. Finally, further research should be conducted with a larger sample size from various hands-on courses in various computer lab environments.

6. REFERENCES

Ashdown, J., Doria, D. & Wozny, M. (2011). Teaching practical software tools using screencasts while simultaneously reinforcing theoretical course concepts. American Society of Engineering Education St. Lawrence Section Conference. Retrieved August 1, 2014 from http://stl.asee.org/papers_2011/Ashdown.pdf

Babb, J. S., Longenecker, B., Baugh, J. & Feinstein, D. (2014). Confronting the issues of programming in information systems curricula: the goal is success. *Information Systems Education Journal, 12*(1), 42-72.

Berardi, V. & Blundell, G. E. (2014). A learning theory conceptual foundation for using capture technology in teaching. *Information Systems Education Journal, 12*(2), 64-73.

Darmouth College (2013). Classes: note taking, listening, participation. Retrieved on August 10, 2014 from http://www.dartmouth.edu/~acskills/success/notes.html

Ebrahimi, A. (1994). Novice programmer errors: language constructs and plan composition. *International Journal of Human Computer Studies, 41*(4), 457-480.

Falconer, J., DeGrazia, J., Medlin, J. & Holmberg, M. (2009). Using screencast in CHE courses. *Chemical Engineering Education*, *43*(4), 302-305.

Geri, N. (2011). If we build it, will they come? Adoption of online video-based distance learning. *Interdisciplinary Journal of E-Learning and Learning Objects, 7*(1)*, 225-234.*

Howe, M. J. (1970). Notetaking strategy, review and long-term retention of verbal information. *Journal of Educational Research*, *63*(1), 285.

Jenkins, T. (2002). On the difficulty of learning to program. In *Proceedings of the 6th Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE'01), 54-56.*

Kinnunen, P., Mccartney, R., Murphy, L. & Thomas, L. (2007). Through the eyes of instructors: a phenomenographic investigation of student success. In *Proceedings of the 3rd International Workshop on Computing Education Research Conference,* 61-72.

Lang, G. & Ceccucci, W. (2014). Clone yourself: using screencasts in the classroom to work

with students one-on-one. *Information Systems Education Journal, 12*(6), 4-14.

Lee, M. & Dalgarno, B. (2008). The effectiveness of screencasts and cognitive tools as scaffolding for novice object-oriented programmers. *Journal of Information Technology Education*, *7*(1), 61-80.

Lloyd, S. & Robertson, C. (2012). Screencast tutorials enhance student learning of statistics. *Teaching of Psychology, 39*(1) 67-71.

Longman, D. G. & Atkinson. R. H. (1999). *College Learning and Study Skills*. Belmont, CA: Wadsworth.

Mow, I. T. C. (2008). Issues and difficulties in teaching novice computer programming. *Innovative Techniques in Instruction Technology, E-Learning, E-Assessment, and Education,* M. Iskander Ed., Springer, 199-204.

Mullamphy, D., Higgings, P., Belward, S. & Ward L. (2009) To screencast or not to screencast. *Anziam Journal*, *51*(1), 446-460.

Nikula, U., Gotel, O. & Kasurinen, J. (2011). A motivation guided holistic rehabilitation of the first programming course. *ACM Transactions in Computer Education, 11*(4), 1-38.

Peterson, E. (2007). Incorporating screencasts in online teaching. *International Review of Research in Open and Distance Learning*, *8*(3), 1-4.

Pinder-Grover, T., Green, K. & Millunchick, J. M. (2011). The efficacy of screencasts to address the diverse academic needs of students in a large lecture course. *Advances in Engineering Education*, *2*(3), 1-28.

Ruiz, S., Sharkness, J., Kelly, K., DeAngelo, L. & Pryor, J. H. (2010). Findings from the 2009 administration of the first college year (YFCY): National aggregates. Retrieved August 1, 2014 from http://www.heri.ucla.edu/PDFs/pubs/Reports/YFCY2009Final_January.pdf

Sharp, J. H. & Schultz, L. A. (2013). An exploratory study of the use of video as an instructional tool in an introductory c# programming course. *Information Systems Education Journal, 11*(6), 33-39.

Sleeman, D. (1986). The challenges of teaching computer programming. *Communications of ACM, 9*(29), 840-841.

Sugar, W., Brown, A. & Luterbach, K. (2010). Examining the anatomy of a screencast: uncovering common elements and instructional strategies. *International Review of Research in Open and Distance Learning*, *3*(11), 1–19.

The Chronicle of Higher Education (2009), Men, take note. Retrieved August 1, 2014 form http://chronicle.com/article/Men-Take-Note/48317/

Udell, J. (2005). What is screencasting? Retrieved 5/20/2013 from http://www.oreillynet.com/pub/a/oreilly/digitalmedia/2005/11/16/what-is-screencasting.html

Wang, X. O. P. H. I. E. (2010). Teaching programming skills through learner-centered technical reviews for novice programmers. *Software Quality Professional*, *13*(1), 22-28.

Winterbottom, S. (2007). Virtual lecturing: delivering lectures using screencasting and podcasting technology. *Planet*, *18*(1), 6-8.