A Mobile Scaffolding Application to Support Novice Learners of Computer Programming

Chao Mbogo
Computer Science Department
University of Cape Town
chao.mbogo@uct.ac.za

Edwin Blake
Computer Science Department
University of Cape Town
edwin@cs.uct.ac.za

Hussein Suleman
Computer Science Department
University of Cape Town
hussein@cs.uct.ac.za

ABSTRACT
Support for novice learners of computer programming can be provided by scaffolding the construction of programs. The ubiquity of mobile phones allows us to support learners whenever they wish to work on a program outside the classroom. This paper describes the development of an application that scaffolds the construction of programs on a mobile phone. The application was designed based on a five-level scaffolding framework and implemented on the Android platform.

The application scaffolds the construction of programs on a mobile device by: (i) representing a program in parts; (ii) restricting a learner to complete the program in a certain order; (iii) enabling construction of a program one part at a time; (iv) providing instructions, steps, default code to be edited, hints, and error prompts where appropriate; and (v) fading the scaffolds as the learner progresses from one successfully completed and compiled program, to the next.

Experiments are currently ongoing to test and evaluate the mobile application.

Categories and Subject Descriptors
K.3.1 **[Computers and Education]:** Computer Uses in Education – Computer-assisted instruction, Computer-managed Instruction, Distance Learning.

General Terms
Design, Human Factors.

Keywords
Mobile Application, Novice Learners, Computer Programming, Support, Scaffolding.

1 INTRODUCTION
Learning computer programming at undergraduate level has long been challenging [3]. Research from a developing country points to the importance of experimenting with new pedagogical approaches to tackle these challenges [2].

Instructional scaffolding [6] is defined as "an adult controlling those elements of the task that are essentially beyond the learner's capacity, thus permitting him to concentrate upon and complete only those elements that are within his range of competence".

ICTD 2013, Dec 07-10 2013, Cape Town, South Africa
ACM 978-1-4503-1907-2/13/12.
http://dx.doi.org/10.1145/2517899.2517941

The definition of scaffolding that this paper adopts is; "modifiable support that enables a learner to fulfill a goal". The goal for learners is effective construction of programs since programming is best learnt through doing.

The ubiquity of mobile devices provides an opportunity to use them in supporting novice learners of programming. A recent study indicates that programming directly on mobile devices is quite potent and accessible for learners who are beginning to learn programming [5].

This paper discusses a mobile scaffolding application that supports the construction of Java programs. There are several mobile Java IDEs available on the Google App store, such as DeuterIDE. However the interfaces of these IDEs mimic a PC IDE such as Eclipse and do not offer additional support that a novice learner of programming may need such as; hints for completing a program, or error prompts that indicate basic errors at the point of constructing the program.

In contrast to these IDEs, this paper describes a mobile application that scaffolds the construction of Java programs, while using design guidelines that would assist in supporting a novice learner. In addition, specific design guidelines have been implemented to address the restrictive qualities of small screen sizes in mobile phones.

The design goals of the mobile application are:

1. To support the construction of programs on a mobile device using scaffolds.
2. To provide scaffolds that adapt to the learner's level and if disabled, can be enabled by the learner at will.
3. To address the qualities of mobile phones such as small screen sizes.

To achieve these goals, the developed mobile application is designed based on a five-level framework. The framework is based on a theory-driven model [2] which has three main phases: type of cognitive learning challenge; specific learning challenge; and scaffolding guidelines. In addition to these three phases, two other phases were added in order to accommodate: (i) a model for categorizing the type of scaffolding to use; and (ii) scaffolding strategies, which are specific types of implementation approaches that can achieve scaffolding guidelines. The scaffolding strategies were then implemented on a mobile application. Experiments are currently being undertaken to test and evaluate the mobile application.

The rest of the paper is organized as follows: Section 2 describes scaffolding strategies; Section 3 describes the system implementation by using a Java program example; and Section 4 concludes the paper.

Table 1: Table that shows how scaffolding strategies fit into a five-level framework

Type of cognitive challenge	Specific learning challenge	Scaffolding type	Scaffolding guideline	Scaffolding strategy that can be implemented on a Mobile Device
Sense Making	The simple yet confusing rules of programming	Supportive	Use representation and language that bridge learners' understanding of programming.	Provide default code that the learner can edit
Sense Making	It's hard to join different parts of code into one	Supportive/Intrinsic	Organize the mobile strategy around the semantics of the programming language.	Represent a program in parts Restrict a learner to complete a program in a certain order
Sense Making	Constructing logic from programs is difficult	Reflective/ Intrinsic	Structure task and functionality by restricting a complex task by setting proper boundaries for learners.	Enable construction of a program one part at a time. Force the learner to complete 'first level' tasks before 'unlocking' 'second level tasks' and so on
Sense Making	Unclear error messages when debugging. Debugging is sometimes frustrating.	Supportive	Use representation and language that bridge learners' understanding of programming.	Prompt the learner as soon as they make a mistake in a piece of code instead of having to wait till they compile the program
Articulation and Reflection	Lack of documentation and practical examples	Reflective	Embed expert guidance about programming practices.	Provide examples that are relevant to the program part being completed.
Process Management	It takes too much time to code programs Finding ways to accom-plish a task in the shortest way possible	Supportive/Intrinsic	Organize the mobile strategy around the semantics of the programming language.	Represent a program in parts Provide steps, default code and instruction on completion of the program while using the application

2 SCAFFOLDING STRATEGIES

The scaffolding strategies were arrived at after identification of learner-cited cognitive learning challenges. These challenges were then mapped to given scaffolding guidelines and types of scaffolding. Table 1 shows how each scaffolding strategy fits into the scaffolding framework. The table shows examples of learner challenge as cited by findings from learners of programming.

To meet the design goals described in the previous section, the system is based on the following scaffolding strategies:

1. Represent a program in parts.
2. Restrict a learner to complete a program in a certain order.
3. Enable construction of a program one part at a time.
4. Providing instructions, default code, steps, hints, examples, and error prompts where appropriate.
5. Fading the scaffolds as the learner progresses from one successfully completed and compiled program, to the next.

2.1 Represent a Program in Parts

To meet the first and third design goals, the main interface is represented in five parts. These parts identify the five parts of a Java program: header comments, imports, main class, method and main method.

2.2 Restrict a Learner to Complete a Program in a Certain Order

To support the construction of programs on a mobile device, a learner is restricted to construct a program in a certain order. First, the learner is required to complete the main class declaration. The header comments are completed next where a learner is required to give author's name and describe the program. The learner can then complete the main method, and thereafter complete the methods and import parts if needed.

2.3 Enable Construction of a Program one Part at a Time

To address the small screen sizes of mobile phones, decomposition is used as a scaffolding strategy where only one decomposed 'chunk' can be worked on a time, while being able to relate to the whole part by viewing the full program. This relation would keep the learner connected to the chunks, while at the same time able to appreciate existence of the whole problem [1]. Therefore decomposition will also be used to reduce the complexity of the learning process.

2.4 Providing Instructions, Default Code, Steps, Hints, Examples, and Error Prompts where Appropriate

Instructions and steps on how to interact with the application are provided. These guide the learner on availability of menu options and how to move from one part to another. Default code is provided to reduce the cognitive load on the learner, and the learner is able to edit this in completing part of a program. Hints are part-specific and are based on standard coding guidelines on how to complete the different parts of a Java program. Error prompts are also part-specific and only pop up if a program part has an error. Examples are viewable and are related to the part of the program being completed, as opposed to examples that contain a full program.

2.5 Fading the Scaffolds

To provide scaffolds that adapt to the progression of the learner, certain scaffolds in the application fade. First-time instructions and steps on how to complete one part at a time are provided only in the first program. In the subsequent program, the learner is

notified that the steps and instructions have been disabled and they can enable them by selecting from a menu. After a learner successfully completes three programs, the interface changes from one which one part has to be completed at a time, to one which any part can be completed. A learner is able to go back to the basic interface if they wish to, by pressing on a related menu.

3 SYSTEM IMPLEMENTATION

A mobile application was developed for the Android platform based on the design strategies described in the previous section. The mobile application uses the Ideone API[1] for compilation and running of programs. This section will be explained using the Java problem example below.

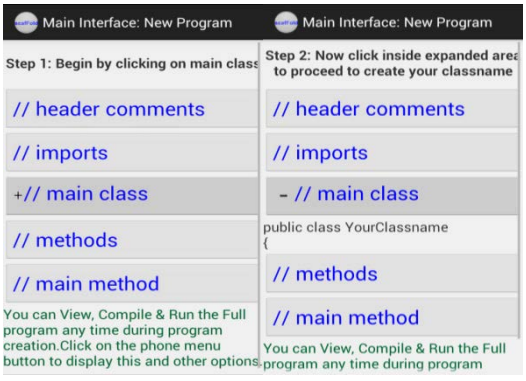
Problem: Write a program called 'Testing' that prints the words 'This works!'.

The application has two main screens: Main Interface and Code Editor. The main interface is the entry point of the application and the learner is presented with the interface as shown in Figure 1(a). Any chunk with a plus sign is enabled, and on start the main class is the only one enabled. Figure 1(b) shows the main class clicked and steps are shown above, instructing the learner on what to do next.

On clicking inside the expanded area of the main class, the learner is taken to the code editing screen as shown in Figure 2 (a) where the step instruction guides them on what to do. If the learner completes the class name starting with a lower case, they get an error prompt (Figure 2 (b)). This error prompt is an example of how an error in the program gets highlighted to the learner before compilation.

On successful creation of class name and on pressing the phone back button, the main interface is displayed (Figure 3 (a)) and the program is saved onto device (Figure 3(b)). The main class is highlighted as green to indicate completion, and header comments part is now activated as is now shown with a plus sign.

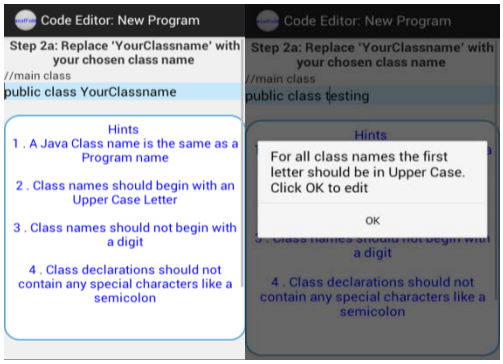
The header comment reveals the name of the program as created after creation of the main class (Figure 4(a)). On pressing phone menu and selecting to view full program, the learner is able to view the full program as at that stage (Figure 4(b)). Figure 5(a) shows the code editor when the learner selects to edit the header comment. On getting back to the main interface, the header comment is updated and main method part is now activated (Figure 5(b)).



(a) main class enabled (b) main class clicked

Figure 1. Main Interface

[1] http://ideone.com/api

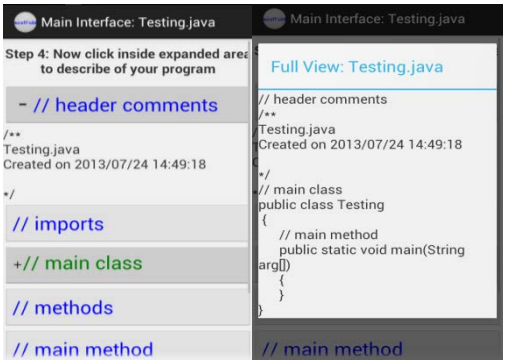


(a) editing class name (b) Error prompt

Figure 2. Code Editor

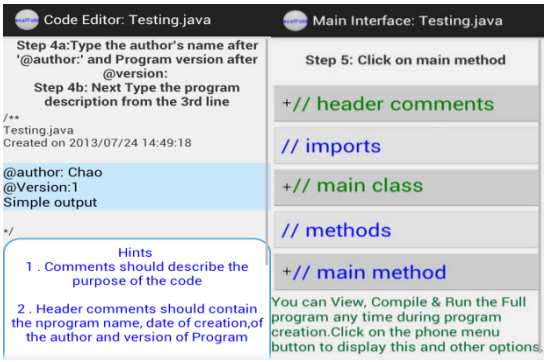
(a) main class completed (b) file saved on device

Figure 3. Interface on successful creation of class declaration



(a) on click of header (b) full program at this stage

Figure 4. Header and Full View

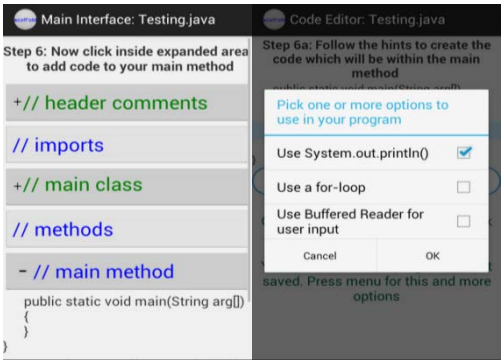


(a) editing header (b) header completed

Figure 5. Header Edit

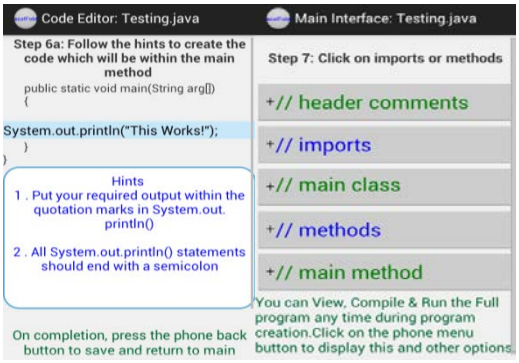
On pressing the main method button, default structure for main method is revealed (Figure 6(a)), and on pressing inside this expanded area the learner is shown some options to select (Figure 6 (b)). This problem requires display of output, hence the learner can select the System.out.println() option. This takes them back to the code editor (Figure 7(a)) and the learner can type what is required within the brackets of System.out.println(). On pressing the back button, the three completed section are all green as shown in Figure 7(b).

The completed full program can now be viewed and seen as complete (Figure 8(a)). On compilation, the output is shown in Figure 8(b).



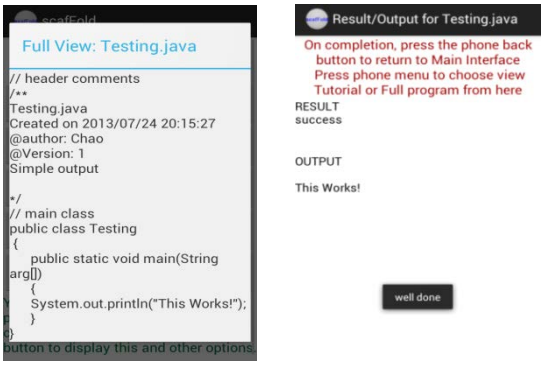
(a) main method selected (b) selection of option

Figure 6. Main Method



(a) edit output option (b) main method complete

Figure 7. Main Method Edit



(a) full program (b) compiled and run program

Figure 8. Completed Problem

4 CONCLUSION

A mobile application that scaffolds the construction of programs on a mobile device has been developed. A five-level scaffolding framework was used in order to implement the scaffolding strategies. Scaffolding is provided in the form of: representation of a program in parts; restriction of order of program completion; completion of a program one part at a time; and provision of instructions, steps, default code, hints and prompts. The application provides fading of scaffolding that can be enabled and disabled by the learner at will.

The application is currently under testing and evaluation with first year learners of Java programming. Several issues that need to be resolved as identified in early testing stages include; reduction of textual information in the code editor, more pronounced instructions on first use, usability of the code editor needs improvement and it should be possible to load a saved program in order to reuse it.

The hints, prompts and selection of options to use in program have been positively received by the learners so far. Complete results will be published in future papers.

Future work involves implementing user feedback into the first prototype and iteratively testing with learners of programming.

5 ACKNOWLEDGMENTS

This study is funded by Hasso Plattner Institute and supported by ICT for Development Laboratory at University of Cape Town.

6 REFERENCES

[1] Ackermann, E. Perspective-Taking and Object Construction. In Kafai, Y., and Resnick, M., ed., In Constuctionism in Practice: Designing, Thinking, and Learning in a Digital World. Mahwah, New Jersey: Lawrence Erlbaum Associates. , 1996.

[2] Apiola, M, Tedre, M, and Oroma, J.O. Improving Programming Education in Tanzania: Teachers' and Students' Perceptions. In 41st ASEE/IEEE Frontiers in Education Conference (Rapid City SD 2011), Session F3G.

[3] Lahtinen, E.,AlaMutka, K.,Järvinen, H. A Study of the Difficulties of Novice Programmers. In ITiCSE '05 Proceedings of the 10th annual SIGCSE conference on Innovation and technology in computer science education (2005), 14 - 18.

[4] Quintana, C, Reiser, B, Davis, E.A, Krajcik, J. Fretz , E. A Scaffolding Design Framework for Software to Support Science Inquiry. Journal of the Learning Sciences, 13, 3 (November 2009), 337-386.

[5] Tillmann, N. Moskal, M., de Halleux, J., Fahndrich, M., Bishop,J., Samuel, A., Xie, T. The Future of Teaching Programming is on Mobile Devices. In ITiCSE'12 Proceedings of the 17th ACM annual conference on Innovation and technology in computer science education (Haifa, Israel 2012), 156-161.

[6] Wood, D, Bruner, J. S, and Ross, G. The role of tutoring in problem solving. Journal of Child Psychology & Psychiatry & Allied Disciplines, 17(2) (1976), 89–100