Indian Journals of Science and Technology, Vol 9(20), DOI: 10.17485/ijst/2016/v9i20/94488, May 2016
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Android-based Mobile Device as a Programming Environment in a School Programming Class

Spartak Razakhovich Sakibayev*, Bela Razakhovna Sakibayeva and Darkhan Bolatovich Toibazarov

Department of Physics and Mathematics, Zhetysu State University, Taldykorgan, Kazakhstan; spartakrz2000@gmail.com

Abstract

Background/Objectives: This article conducts a study on using Android-based mobile devices in school-level programming classes and the impact of their use on learning outcomes. **Methods:** The authors propose hypotheses upon which they create a methodology of building a simple programming environment to run on mobile devices and targeted towards the use in a classroom. **Findings:** The authors have found optimal ways of building simple programming environment on Android-based mobile devices. They also provide the description of the actual implementation of such an environment. A small dialect of BASIC programming language is used as the primary language of this environment. The environment itself is completely browser-based. The interpreter for this environment is written in JavaScript which is supported by all major Android-based web browsers. The environment is easy to use and implement and provides a transparent graphical user interface which automates the process of editing the program source code. It has been shown that introducing the Android-based mobile devices into programming classes can streamline the entire learning process in the sphere of programming. The authors have made an emphasis on the benefits which mobile technologies can bring to programming classes. The real-life example of using mobile programming environments in a classroom has been provided. **Application:** Usage of Android-based mobile devices in programming classes will promote realization of the E-Learning concept, increasing the efficiency of programming teaching process in a school environment.

Keywords: E-Learning, Mobile Programming Environment, Java-based Interpreter

1. Introduction

Mobile devices have become one of the priority directions in the development of modern computer technologies. In many spheres applications on mobile devices can be successfully used as a valid alternative to traditional desktop applications. The number of mobile users of some Web resources equals or even exceeds the number of traditional desktop users. Mobile devices are widely used in buying and selling goods and services over the Internet, making on-line reservations, personal and business planning, working with electronic mails and visiting favorite web sites. In addition, mobile devices have gained a wide acceptance in the school environment as effective teaching tools.

According to the Pew Research Center survey, 73% of the teachers reported using mobile technology in their classrooms, either through their own instruction or by allowing students to use it to complete assignments. The use of mobile technology in a classroom has many types. Mobile devices can be used as E-Readers supporting many popular formats. They are used to work with interactive educational programs. They can also be used as means of connecting to a cloud containing educational material, visiting educational web resource or playing educational multimedia files. In general, educators admit big positive advantages that mobile technologies bring to the educational process. Rick Allen[1] stated that mobile devices are the vehicles which can transform the 12-K learning for the 21st century.

*Author for correspondence

"…Education experts reason that these powerful small computers motivate students; provide constant access to the wealth of knowledge, tools and experts on the web; and are cheaper and more plentiful than laptops or desktop workstations"[1]. But despite this level of recognition by educational experts, still there is one area in a school curriculum where the use of mobile technologies has never had any effective practical implementation. This area is the use of mobile devices in programming classes. The significance of this research is determined by the fact that so far there have been no articles dedicated to the usage of Android-based mobile devices for programming tasks in a classroom.

2. Literature Review

Unfortunately, so far there have been no articles dedicated to the usage of Android-based mobile devices for programming tasks in a classroom.

3. Method of Analysis

Traditionally, mobile devices have never been considered suitable for programming tasks. First, this is determined by the limited computing resources. Second, due to the lack of traditional keyboard and small screen sizes the process of entering source code becomes inconvenient and error-prone experience. And third, because of the first two reasons teachers will meet with the lack of programming language implementations designed specifically for mobile devices. The situation here resembles the situation as it was in the 1980s, when it generally was considered impossible to do any serious programming using the newly-invented microcomputers.

The programming problems given at school-level programming classes are usually of elementary mathematical nature. They may be about finding the roots of an elementary equation or carrying out some arithmetical calculation. The process of solving this type of problems does not require an extensive resource-taking compilation process. Besides programming these problems would require the usage of only a limited number of commands and operators of the programming language. And the process of entering them on a mobile device can be easily automated. Most major mobile browsers support JavaScript out-of-the-box, so it becomes possible to use this language as a back-end when solving programming problems on mobile devices in a classroom. For example, it makes possible to create a simple interpreter for minimal programming language targeted exclusively towards solving classroom problems.

This article attempts to demonstrate one of the approaches to make an Android-based mobile device a suitable tool for use in programming classes. We describe the actual Android-oriented programming environment used in some of our local schools. This programming environment is browser-based but does not require any web server to run. It uses a simple interpreter, which implements a minimal subset of a BASIC programming language designed specifically for solving school-level problems. The interpreter itself is written in JavaScript supported by most major mobile browsers. The process of entering operators and commands of the programming language is completely automated and optimized for touch screen interfaces specific to mobile devices. The description of the programming environment is accompanied with example problems which can be solved with its help. Also the actual statistical data are provided to prove the positive impact this mobile-based programming environment has on the efficiency of programming classes.

The novelty of our study is in its attempt to propose a solution to the problem of the low rate of using mobile devices for programming purposes in school programming classes.

Our study is based on four research hypotheses which were proposed to determine whether Android-based mobile devices provide a suitable environment for school-level programming tasks.

3.1 H1

Mobile device computational resources are enough for school-level programming tasks**.** It is generally accepted both by scientists and engineers that mobile devices are not suitable for programming purposes in general. They refer to different technical factors which prevent mobile devices from becoming a valid platform for programming on. The most principal of these factors is the lack of enough computational resources (CPU, memory and disk space) needed for successful compilation process. While these considerations are true in a general sense, they are not necessarily valid when applied to programming tasks commonly performed at a school level. Typical school programming problems do not require complex and

resource-assuming process of compilation. Due to their nature they can be solved (and better be solved) using small interpreted languages, e.g. some interpreter based on JavaScript which is supported by major mobile browsers. This leads to the next hypothesis.

3.2 H2

Javascript is a good back-end for mobile device-based programming language interpreters**.** Though it is still impossible to compile on mobile devices, school programming classes can make use of the JavaScript language, supported out-of-the-box by all major mobile browsers. Instead of using it directly, schools can develop their own custom interpreters in JavaScript, that is, use JavaScript as the back end. The typical school-level programs do not require complex computations so using JavaScript-based interpreters would be enough for usage in a classroom. For example, in classrooms they can use some minimal subset of a BASIC programming language which would play the role of a simple front-end to JavaScript interpreter built in mobile browser. This custom language would use only the minimal number of commands and operators, so the process of their type can be easily automated. This leads to the next hypothesis.

3.3 H3

The mobile touch screen interface does not decrease the process of source code editing**.** One of the reasons why mobile devices are not chosen as a platform to do programming on is their small form factor, touch screen orientation and lack of traditional keyboard. These are the factors that many programmers consider to be major obstacles in the process of editing source code. However, this problem can be effectively diminished by building an interface which would automate the process of adding operators and commands of the programming language into the source code editor.

3.4 H4

The mobile-based programming environment is more transparent and user-friendly to school users than traditional IDEs**.** As we have already mentioned before, the school-level programming classes deal mostly with the problems having elementary mathematics nature. Their solution requires only the most minimal subset of any programming language. But how are they solved in a typical programming class? Users are provided with IDEs, whose functional capabilities extend the level of the programs they are used for. Also it must be mentioned that in the eye of an average school user, IDEs have very complicated and non-friendly user interface. Complex system of drop-down menus, cluttered desktops, multiple toolbars and many different specialized windows within the IDE's environment do not contribute to the effectiveness of the educational process. Here traditional mobile interfaces, due to their minimalism, would be of more appeal to students.

4. Results

4.1 General Information

Built-in JavaScript support by most major mobile browsers makes it possible to create a programming environment based on a simple Basic-like interpreter which uses JavaScript as a back-end. This environment is targeted towards solving small school-level programming problems and automates the process of editing source code. It has a basic functionality allowing users only to save and run their programs which would be sufficient for most classroom programming activities.

The front-end language for this environment is a small Basic-like language which is written in JavaScript. The source code written by students is translated into JavaScript and then the resulting JavaScript code is running within browser. The process of editing source code is automated and the user enters the language command through the means of graphical user interface.

4.2 Language Description

The language for the Android-based programming environment is designed to be simple and implement only the most basic features. For the purpose of simplification and taking into account the nature of problems solved in the school-level classroom it is necessary to make the language devoid of some advanced features of a full-fledged language implementation. For example, our Android-based programming language does not support subroutines and graphics programming.

For the purpose of simplification the variables can be assigned only 1-letter names and they start with a '$' prefix, for example, *$A*. You can assign a value to a variable in a traditional way: $A=10. Our JavaScript interpreter

translates this assignment to var $A = 10. If the variable is explicitly assigned no value, then interpreter considers it to be assigned NULL value.

Input is accomplished through INPUT operator. For example, to read to variable $N you use INPUT $N. JavaScript interpreter translates this into var $N = window.prompt ("N=", "0").[2] For the purpose of simplification with INPUT you can read data only for one variable at a time. For example, if you want to get input of two variables, you would use two separate calls of INPUT.

To output data you use the PRINT operator. For example to display some text we use PRINT "some_text". JavaScript interpreter translates it into window.alert("some_text")[2]. To print some variable you use PRINT $var_name. JavaScript interpreter translates it into window.alert($var_name). PRINT operator is made to print only one variable a time, so printing several variables in a row requires separate calls of PRINT for each variable.

The syntax for arrays is different from that of traditional Basic syntax. It is done purposely to simplify their usage. For example, if you want to declare an integer array of three elements you can use the following syntax:

```
$A[] = (5, 10, 15)
```

The JavaScript interpreter will translate it into:

```
var $A = [5, 10, 15];
```

An array of string values is declared in the similar way, e.g.:

```
$S[] = ("String 1", "String 2", "String 3")
```

This declaration of a string array of 3 elements is translated to the following JavaScript code:

```
var $S = [String 1", "String 2", "String 3"];
```

Our Android-based language also implements loops. But for the purpose of simplification only FOR-loops are to be implemented. They have the following form:

```
FOR $i = 1 TO 10 STEP 1
// operators
NEXT $i
```

This loop is translated to the following JavaScript code:

```
for ($i = 1; $I <=10; $i++) {
// operators.
}
```

To use FOR-loop to count down, you use the following form:

```
FOR $i = 10 DOWNTO 10 STEP -1
// operators
NEXT $i
```

This loop is translated to the following JavaScript code:

```
for ($i = 10; $I >=110; $i--) {
// operators.
}
```

The conditional control statements are implemented using IF statements. They have the following form:

```
IF $A > 10 THEN
        PRINT "A is bigger than 10"
ELSE
        PRINT "A is not bigger than 10"
ENDIF
```

or

```
IF $A > 10 THEN
        PRINT "A is bigger than 10"
ELSEIF $A = 10
        PRINT "A equals 10"
ELSE
        PRINT "A is lesser than 10"
ENDIF
```

For the purpose of simplifying the process of parsing, both ELSEIF and ENDIF are implemented as single word statements, unlike in the traditional Basic syntax. If statements are translated directly to their JavaScript equivalents:

```
if ($A> 10) {
        ...
}
```

or

```
if ($A> 10) {
        ...
} else {
}
```

This limited subset is designed to simplify the process of learning the language for solving the programming problems given in a classroom. It is functional enough to allow implementing solutions based on non-linear algorithms. For example, let us consider a simple task: A program gets an integer from user and then prints the squares of all numbers from 2 to this number. In our Android-based environment the code for solving this problem can look like this:

```
PRINT "Enter your number:"
$N = 0
INPUT $N
FOR $I = 2 TO $N STEP 1
        $S = $I * I
        PRINT $S
NEXT $I
```

This program will be translated to the following JavaScript code:

```
window.alert("Enter your number: ");
$N = 0;
$N =    parseInt(window.prompt("N=", 0));
for ($I = 2; $I <= $N; $I++) {
        $S = $I * I;
        window.alert($S);
}
```

4.3 User Interface Description

To make Android-based programming environment effective, usable and user-friendly it is not enough only to provide an appropriate and suitable programming language. It becomes of a big importance to have a well-organized GUI for the programming environment. This GUI must be minimalist and should provide all the tools to automate the process of editing the source code. Our environment consists of two main areas. One area contains a text editing control which is read-only and contains the user's code. The other area contains buttons which give the user access to code editing functions.

The button VAR opens dialog in which the user enters the variable name and its initial value. After that this variable declaration appears in the text editor. The button INPUT opens a dialog in which the user specifies the variable name. After that a new line INPUT $var appears in the text editor, where $var is the variable name specified by the user. The button PRINT opens a dialog in which the user specifies the variable name or some text. After that a new line PRINT $var or PRINT "text" appears in the text editor, with the text or variable name specified by the user. The button ARRAY activates the interactive dialog in which the user specifies the array name and the initial values of its elements. After that the array declaration appears in the text editor. Similar interactive dialogs are designed for all language elements, including IF and FOR statements. For example, button FOR activates a dialog where the user specifies the loop variable name, its initial value and increment/decrement step. After that the FOR-statement appears in the text editor. The button CLEAR serves to clear the contents of the text editor. These dialogs are designed to automate the process of entering the code and minimize the amount of time that the classroom user spends with the keyboard.

The button OPEN activates a dialog where the user is prompted to specify the file name to load. If the specified path does not exist, then the program informs the user about it. The button SAVE activates a dialog where the user is prompted to specify the file name to save to.

With the button TEST the user can start the program from within the editor's environment. The button HELP provides the user with a manual on the language constructs and the GUI elements. The manual contains the example code which the user can insert into the text editor for a testing purpose.

5. Discussion

This Android-based programming environment has been tested at programming classes in the local schools. The classes are engaged in studying the fundamentals of algorithms and programming and deal with simple programming problems of elementary mathematical nature. For example, during lessons they find the roots of elementary algebraic equations find the sum of integers within the given range or find numerically the square of basic geometric objects. That is students solve problems that do not require complicated integrated programming

environments and can be solved with the most BASIC programming means. The use of the traditional desktop-oriented development environments in this context has some drawbacks. One of them is that they provide many extraneous features for a student whose task is simply to solve a typical school-level programming problem. Students only need basic functionality – save or open a source code file and then test it. Many of the features provided by the complex menu system are not used by school-level users at all. Also miscellaneous additional tools which are usually supplied with traditional integrated environments, such as, for example, debuggers and profilers, are also ignored by the school-level users in most of the cases.

One of the most important goals in the process of using mobile platforms in educational process is to ensure the integrity of a student's working environment. That is, it becomes of importance that a student uses the same mobile device at all lessons and in doing all classroom tasks and problems. Introducing the Android-based programming environment contributes to providing a needed level of such integrity.

Almost every student shows a clear preference for using Android-based environment to solve classroom programming tasks. This environment runs on their preferable personal gadget, has a minimal user interface, automates the code writing process and enables to easily test their program. Besides it allows students to remain within the same mobile working environment at all their lessons. Students find it easier and more convenient to solve programming problems using their personal Android devices, rather than do it using desktop-oriented professional integrated environments.

6. Conclusion

Mobile devices in general and Android-based devices in particular have a potential of becoming a dominating platform for use in programming classes. The rate of growth of computing capabilities inherent in the mobile devices indicates that in the close future it will become possible to create full-featured integrated development environments targeted specifically towards the usage on mobile platforms. The situation here resembles the situation with the very first IBM PCs which, at the time of their invention, were considered absolutely unsuitable for serious software development tasks due to their limited computing capabilities. However even at their current state of development, mobile devices provide a suitable, yet a limited environment for programming at the school classroom level. Particularly they can be used for solving simple programming problems from elementary mathematics typically given in school-level programming class.

The authors have developed a methodology which makes it possible to implement basic programming environment to run on Android-based mobile devices. They also demonstrate the practical ways of implementing this methodology. A small dialect of BASIC programming language is used as the primary language of this environment. The environment itself is completely browser-based. The interpreter for this environment is written in JavaScript which is supported by all major Android-based web browsers. The environment is easy to use and implement and provides a transparent graphical user interface which automates the process of editing the program source code.

It has been shown that introducing the Android-based mobile devices into programming classes can streamline the entire process of education in the sphere of programming. The authors have made an emphasis on the benefits which mobile technologies can bring to programming classes. The real-life example of using mobile programming environments in a classroom has been provided.

7. References

1. Allen R. Can mobile devices transform education? ASCD, 2011, 53(2). 2016. Available from: http://www.ascd.org/publications/newsletters/education-update/feb11/vol53/num02/Can-Mobile-Devices-Transform-Education%C2%A2.aspx
2. Duckett J. JavaScript and JQuery: Interactive Front-End Web Development. Wiley; 2014.
3. Hosmer C, Jeffcoat C, Davis M, McGibbon T.Use of mobile technology for information collection and dissemination. Data and Analysis Center for Software; 2011 Mar.
4. Crescente ML, Lee D.Critical issues of M-Learning: design models, adoption processes, and future trends. Journal of the Chinese Institute of Industrial Engineers. 2011; 28(2):111–23.
5. Elias T.Universal instructional design principles for mobile learning. International Review of Research in Open and Distance Learning. 2011 Feb; 12(2):143–56.
6. Kahle-Piasecki L, Miao C,Ariss S.Managers and the mobile device: M-Learning and m-business - Implications for the

United States and China. Journal of Marketing Development and Competitiveness. 2012; 6(1):56–68.
7. Mobile Learning Community. Mobile Learning History. 2010. Mobile Learning For Education.Available from: www.ijceronline.com ||June||2013|| Page 101
8. Savill-Smith C, et al. Mobile learning in practice: Piloting a mobile learning teachers' tool kit in further education colleges. 2010.
9. Saylor M. The Mobile Wave: How mobile intelligence will change everything. Perseus Books/Vanguard Press; 2012. p. 176. ISBN 978-1593157203.
10. Sharma SK, Kitchens FL. Web services architecture for M-Learning. Electronic Journal on E-Learning. 2004; 2(1):203–16.
11. Singh M. M-Learning: A new approach to learn better. International Journal of Education and Allied Sciences. 2010; 2(2):65–72.
12. Meier R. Professional Android 4 Application Development. Wrox. 2nd ed. 2010.
13. Nudelman G. Android design patterns: Interaction Design Solutions for Developers by Greg Nudelman. Wiley; 2013.
14. Smith D. Android Recipes: A Problem-Solution Approach. A Press. 4th ed. 2015.
15. McLean D, Komatineni S, Allen A. Pro Android 5. Apress, 5th ed. 2015.
16. Usama Bin Aftab M. Learning Android Intents. Packt Publishing 2014.