Teaching Control Structures Using App Inventor

Kevin Krul, 3682846

30 ECTS

Supervisors

P. van der Hoeven

P. Drijvers

Coordinator

A. Bakker

Freudenthal Institute, Utrecht University

Teaching experiment done at secondary education Roncalli, Bergen op Zoom

Possible journal

ACM Transactions on Computing Education

Audience

Interested in computing education

Abstract

Teaching programming has always been a difficult subject, especially control structures. The introduction of Initial Learning Environments like GameMaker and Alice made learning these control structures easier and more fun. A new environment in this area is App Inventor, created by Google and now maintained by MIT. App Inventor allows the user to create apps for the mobile operating system Android. This research aims to answer whether App Inventor really makes learning control structures more fun and easier. A series of lessons has been created, which teaches the basic control structures using App Inventor. This series of lessons was tested at a Dutch secondary school and resulted in positive results but different opinions among the students.

Teaching Control Structures Using App Inventor

1. Introduction

Since secondary schools have introduced informatics lessons, teaching how to program has been a big difficulty. It takes students a lot of time to understand new ways of thinking necessary to learn how to program. Basic control structures such as conditions (if, if-else) and loops (while, for) are difficult to understand and to implement for novice programmers (Milne & Rowe, 2002).

There are several possibilities for making these first steps into the world of programming easier and more enjoyable. One possibility is visual programming in which visual elements are represented by squares and other shapes are labeled with commands (Cranor & Apte, 1994). These shapes can then be dragged into a graph representing the application, only allowing correct placement of the blocks. Applications in which students can learn to program using visual programming are called Initial Learning Environments (ILE) (Fincher, Cooper, Kölling & Maloney, 2010). A few examples of well-known ILEs are GameMaker and Alice, both widely used at Dutch secondary schools.

MIT App Inventor is a new ILE using the visual programming system. Originally released by Google in 2010, it is now part of the MIT Center for Mobile Learning. It allows the user to create software applications for the Android operating system. These applications are commonly named apps. The Android operating system runs on many mobile devices like smartphones and tablets. Using App Inventor enables students to easily create applications for their smartphone, this may be a positive motivational factor while learning to program. Combining this motivational factor with the advantages of using an ILE, App Inventor might be a good introduction to programming and teaching the basic control structures.

2. Theoretical Framework

There are various reasons why learning how to program is difficult for students. Cognitive theories about this problem include the students' inability to problem-solve (McCracken et al, 2001) and the difficulty of understanding the syntax and semantics of programming code (Robins, Rountree & Rountree, 2003). Low motivation was found to be one of the major reasons why students drop out of programming courses (Kinnunen & Malmi, 2006). Creating new courses and tools to solve these problems is something which has been put much effort into over the past few years. In the following paragraphs we will look into two possible solutions, using an ILE and programming for mobile phones.

As explained in the introduction, an ILE uses visual elements instead of programming code, eliminating the problem with difficult syntax stated in the first paragraph of this section. Alice, an ILE created at the Carnegie Mellon University of Pittsburgh, was tested at different colleges and shown to be successful during the first Computer Science course. Students using Alice scored higher grades than students who used traditional methods (Moskal, Lurie & Cooper, 2004). The ILE GameMaker also showed good results when used for teaching programming concepts such as control structures (Hoganson, 2010). His research concludes that GameMaker is a good way to teach control structures and even more advanced programming concepts like object-oriented programming.

Programming for mobile phones is a recent development in the area of teaching informatics. Mahmhoud and Dyer (2007) used BlackBerry mobile phones in the curriculum at the University of Guelph. Programming for BlackBerry devices is done in the Java programming language. Java is normal programming language, not an ILE. Research by Mahmhoud & Popowicz (2010) explains that using mobile phones in an introductory course has less coding limitations than other approaches like game-design and robotics. Android and iOS are gaining market share fast, decreasing the popularity of BlackBerry devices. Students are more motivated to develop for the former two operating systems than they are for the latter (Engelsma & Dulimarta, 2011).

Now we have seen that using an ILE and programming for mobile phones both have a positive effect on the results and motivation of the students we can combine these by using App Inventor.

Before the release of the App Inventor a pilot program was taught at eleven colleges and at one high school (Abelson, Chang, Mustafaraj & Turbak, 2010). The results of this pilot program have unfortunately not been made public. Gestwicki & Ahmad (2011) used App Inventor in combination with studio-based learning. This combination showed good results and noted that even advanced control structures were successfully introduced to the students. As these results are obtained by using a studio-based learning environment it is not possible to apply the same test results to traditional classroom education. Wolber (2011a) had his students create applications with App Inventor for real customers. This particular link to the real word motivated his students.

If App Inventor can be used to teach control structures, it would be an effective new way for students to learn control structures and introductory programming. Combining the positive results from using an ILE with the increased motivation from smartphone development.
In order to find out if this is indeed possible, we would need to answer three research questions:

1. Can App Inventor be used to teach the loops and conditions control structures?
2. Does the use of App Inventor in class motivate students?
3. Which are decisive factors in the motivation and achievement of students when using App Inventor?

3. Methods

3.1 Designing a Series of Lessons

As there is no existing teaching material which is suitable for usage during this research we are going to design series of the lessons for use in this research. Only the first iteration of the design research will be done. The research questions will be answered in this design research. The series of lessons needs to be in the Dutch language, which is the main language used on the secondary school which is going to be used for the teaching experiment. The second requirement for the lessons is that they focus more on the control structures than on the use of App Inventor and the Android operating system possibilities.

The control structures we are focusing on during this series of lessons are:

- if statement
- while loop
- for loop

We have used a conceptual analysis to define the most important aspects of these control structures, visible in Figure 1. These aspects will be used as keywords to determine if the students did indeed learn the structures as we intended when analyzing the gathered data.

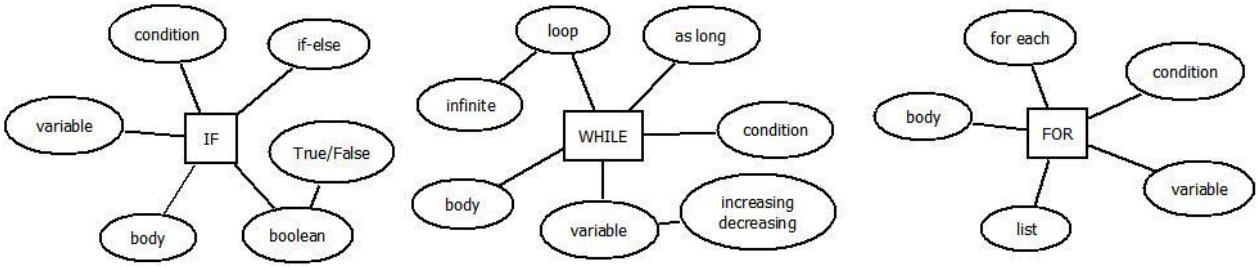

Figure 1: Concept maps of the control structures, defining the keywords

The *if* statement executes the code in his body when the tested condition is true. It is possible to create an *if-else* statement in which a second body with code is provided which is executed when the condition is false.

The *while* loop continues to execute the body code as long as the condition is true. The condition is revaluated after each iteration of the body. In some situations a variable is used for counting the number of iterations. If the condition never turns false the iterations of the body never stop, resulting in an infinite loop in which most cases the application crashes.

A *list* can contain multiple variables. The *for* loop executes the body for each variable found in the list.

Before we started designing the series of lessons, we looked into existing teaching materials for App Inventor to see if it was possible to use existing and proven material. A pilot program (Abelson et al., 2010) was taught at eleven different colleges and one high school across the United States of America. The material used in this pilot program is freely available online. Their series of lessons, in the English language, focuses on the use of App Inventor. Because of this focus it was not usable for our research, as we want the focus to be on control structures.

However, we used the first lesson from this pilot program in our series of lessons because it forms a good introduction into the usage of App Inventor by creating a small and simple app.

Previous research done on the subject of App Inventor (Gestwicki & Ahmad, 2011) used the book "App Inventor: Create Your Own Android Apps" (Wolber, 2011b). The first part of the book takes the series of lessons from the pilot program and adds more lessons. The second part of the book deals with control structures and more advanced use of App Inventor. The book gives lengthy explanations and uses good examples. A few of these examples taken from the book were used as inspiration when creating the series of lessons to be used in our own research.

During the first lesson we address event handling. This can be a user action: the click on a button or a defined event like for instance starting or closing the application. When an event occurs, the code in the corresponding body will execute. Events are the starting-base for all App Inventor applications and needs to be covered before any other concepts can be introduced. During the first lesson the variable will also be introduced. This is a returning keyword for all control structures in the conceptual analyses therefore we think it is important to introduce it early, so any problems the students may have with variables can be clarified at an early stage.

The second lesson introduces the *if* statement. The *if* statement checks if a given condition is true, if this condition is true the code in the body will be executed. This introduces the concept of a boolean as the condition is either true or false. In most cases a variable will be used inside the condition of an *if* statement.

The third lesson covers the *while* loop. The *while* loop continues to re-execute the code in the body as long as the given condition is true, this condition is checked again after each execution of the body code. An expected difficult concept for the students because of the dangers of causing an infinite loop when not preventing this in the body code. In many cases this will be prevented by increasing or decreasing a variable which is checked inside the condition.

The *for* loop, as used in App Inventor, can be used to scroll through a given list. A list is a variable containing not one, but multiple values. For each value in the given list the *for* loop

executes the body code with access to the current selected value from the list. This final control structure is introduced in the fourth lesson.

3.2 Participants

The teaching experiment took place at a Dutch secondary school called Roncalli, located in Bergen op Zoom. This school's educational system works according to the Dalton Plan (Parkhurst, 2010). This system allows the students to work at their own speed, independently and gives them more freedom of choice. Our series of lessons was created in a way which fits the Dalton Plan education. The first lesson is taught through classroom instruction, during the following lessons the students need to work independently. All subsequent lesson plans were given to the students digitally at the end of the first lesson, enabling the students to work at their own pace.

The series of lessons was tested in two different classes. Two fourth year classes, one havo and one vwo. Havo is the Dutch abbreviation for higher general secondary education and vwo is the Dutch abbreviation for pre-university education. These two levels will be named by their Dutch abbreviations for the remainder of this paper. The student's ages range from 14 to 16. When asked beforehand, each student answered he/she had no previous programming experience. This group consisted of thirteen students, ten havo students and three vwo students.

Two of the vwo students who were ahead of the curriculum schedule, possible because of the Dalton Plan education, were given the series of lessons as a pilot group, before the other students started. This allowed us to do early testing of the lessons and receive suggestions from the pilot students to change some explanations.

Just after the pilot group finished, Google transferred App Inventor to MIT. It was expected that MIT could launch the new App Inventor shortly after Google took their version offline. Sadly this was not the case. When the sample students were ready to start on the series of lessons, the MIT App Inventor was still unavailable. Thanks to the community of App Inventor users we

found a host willing to set up a private App Inventor environment which allowed us to continue the lessons. This problem pushed back the date for the first lessons with one week.

After the first lesson all students worked at their own pace. Some students completed it within the predicted timeline of five lessons. Others finished it weeks later and started doing tasks for other subjects before continuing and finishing the App Inventor course, which is allowed at this school because of the Dalton Plan education. Two students from the havo group took a long time finishing the course: it took them two months to finish. These students had big difficulties with App Inventor and lacked the insight into the way of thinking. The teacher helped them extensively to gain the needed insight during the lessons when the other students had already finished.

3.3 Data Collecting

Five different methods of data collection were used to answer the research questions. Each of the five methods will be described in the following paragraphs, including some information about how the data was analyzed. Figure 2 shows which data were used to answer which research question.

Research question	1	2	3
Field notes	✓		✓
Progress forms		✓	✓
Final assignments	✓		
Questionnaires	✓	✓	✓
Interviews	✓	✓	✓

Figure 2: Cross-table of data and research questions

3.3.1 Field Notes

During all lessons the students worked on App Inventor the researcher (who is also the teacher) took field notes on the questions and difficulties the students had. Notes were also taken on the interaction students had with each other, as it is customary for the students to help each other.

One of the students from the sample group finished the lessons quickly and decided to help the other students, taking some workload of the teacher. He was also asked to take notes on what he explained to his classmates. We collected information from the gathered notes on which difficulties the students had when working with App Inventor and how they solved this.
For analysis, all notes were sorted by lesson and assignment in order to construct a solid overview of the problems the students had with each control structure.

3.3.2 Progress Forms

A group of three randomly selected students from the sample group was asked to fill in a short progress form at the end of each lesson, answering two questions:

- What did I learn today?
- I liked/disliked this lesson, because...

We used the first question to track the progress of these students. To see if they understood what the meaning was of each lesson and if they had learned from these lessons what they were supposed to learn. The second question on the progress form was used to see if using App Inventor motivated the student. Collecting these forms at the end of each lesson allowed us to keep a record of the motivation during the process of the course, to see if motivation had changed. Our aim was to attempt to answer question two and three of our research using the results from these progress forms.

3.3.3 Final Assignments

The designed series of lessons contains a final assignment. This will take place in the fifth lesson of the course when all control structures have been discussed. In this final assignment the students create an application in which all control structures can be used. As with many programming exercises, there are multiple solutions to create this application, so it is possible that a student comes up with a different solution in which not all control structures are needed to successfully create the application. The application that needs to be made is a quiz application. The user is shown a question they need to answer in the textbox, when the user presses a button the given answer is checked against a list of correct answers. If answered correctly, the text on the button changes to ‘Correct!’, if answered incorrectly the text on the button changes to

'Wrong!' A second version of the application needs to be made in which two questions need to be answered and checked on the button press event. The student needs to hand in both versions of the application.

The collected final assignments will be analyzed against a checklist containing all control structures to see if each control structure was implemented correctly. If not implemented correctly, notes will be taken on mistakes made. This checklist is shown in Figure 3.

Student: Marijn	If	While	For	List	Variable	Notes	Working solution
Version 1	Correct	Not used	Incorrect	Correct	Incorrect	Used wrong variable inside the for loop	No
Version 2	Correct	Not used	Correct	Correct	Correct	Errors from version 1 are fixed	Yes

Student: Oliver	If	While	For	List	Variable	Notes	Working Solution
Version 1	Correct	Not used	Correct	Correct	Not used	Checking against the button text instead of variable	Yes
Version 2	Incorrect	Not used	Correct	Correct	Not used	Using a nested if for second question	No

Figure 3: Checklist for analyzing the final assignment

3.3.4 Questionnaires

When the final assignment handed in, the student is asked to fill in a questionnaire about his use of App Inventor during the course and his motivation. This questionnaire is based on the SMTSL (Students' Motivation Towards Science Learning) questionnaire (Tuan, Chin & Shieh, 2005). We have translated this questionnaire to the Dutch language, which is the native language of the students. The first part of in the questionnaire consists of questions about the motivation for the course. The second part of the questionnaire is adjusted to contain questions specific to the content of the course like the control structures and how the use of App Inventor has influenced the students' motivation. The complete Dutch questionnaire is available in the appendix. All

questionnaires were collected and were entered into a statistics program to obtain a good overview of the results. We used this statistics program to determine the frequencies for all given answers.

3.3.5 Interviews

Finally, when all students had completed the series of lessons, six students were interviewed by the researcher. All three of the vwo students and three randomly selected havo students. This semi-open interview was an extension to the questionnaire, gaining a deeper insight into the student's view on working with App Inventor. We asked the students to explain the control structures they had learned and if they thought working with App Inventor was difficult or easy and why this was the case. Finally we asked whether or not they liked working with App Inventor and why they liked it. The data collected with this method will be used to answer all three research questions. All interviews were audio recorded and transcribed afterwards, giving us the possibility to analyze them and quote the interviewed students in the results section of this paper.

4. Results

4.1 Field Notes Results

This section will contain the results of the field notes taken by the researcher/teacher during the lessons. Because of the Dalton Plan education not every student worked on App Inventor each time. For this reason the time range in which the students worked on App Inventor was two months. The results will be grouped by the related lesson and assignments.

All students (with the exception of the two pilot students) started at the same time with the first classroom lesson with explanations about how to use App Inventor, event handling and variables. In the first notes taken on this day we see a few questions about bugs in App Inventor, which is inevitable because we were running an experimental version from MIT on our own server. Other questions are about the usage of a variable. Although it had been explained earlier a

lot of the students seem to have trouble implementing it themselves. When the teacher explained it a second time, the students now understood it. The number of questions about the use of a variable decreased after the first lesson. Two students experienced difficulty while using the App Inventor interface during this first lesson. One of them just followed the steps from the lessons but did not know what he was doing. When given some information about how an application is executed and how App Inventor controls this, the student understood what he was doing and why.

Not many questions were asked about the *if* statement during the second lesson. Only one student had some trouble, he made a condition containing only one variable, not comparing it to anything. When the teacher asked why he used an *if* statement, he realized he was missing a value to compare his variable to. The next assignment in which an *if-else* needed to be used was correctly done by many students without any help.

The *while* loop, introduced in the third lesson, resulted in more problems. The students had no trouble making the example application from the first assignment. But when they needed to use a *while loop* themselves the problems started. One of the assignments consists of displaying 10 messageboxes, each containing the next result from the multiplication table of a given number. Naturally we started the while loop with *number* * 1, counting upwards to *number* * 10. When the program was executed the first messagebox was *number* * 10 because all the boxes appeared so quickly on top of each other the next messageboxes came hidden behind the messagebox on the top. The students were asked why the messagebox containing *number* * 10 is visible first when the application is executed instead of *number* * 1 and how to fix this. Only one student directly understood what was going on and understood how he needed to adjust the *while* loop to fix this behavior. All other students had big problems understanding this, even when the teacher explained the cause they still had problems to prevent this from happening.

The next assignment of the third lesson was a tricky one as it did not need to contain a while loop but only an *if statement*. The amount of times a button was clicked needed to be counted and only on the fifth click a messagebox needed to be shown. As the starting point of the code is the *button.click event* using a *while* was not an option. Even the students who previously showed a

good insight tried creating this assignment with a *while* inside the *button.click event*. To prevent this mistake, the students need to consider exactly what happens when the application is executed. This is called lateral thinking. Some students showed big difficulty with this new way of thinking. When the teacher presented a few examples most students understood. Two students did not understand this logic after the examples from the teacher and had more trouble finishing the course. A few students asked about how a loop is stopped and how to prevent infinite looping, as this occurred in their creations.

The next lesson about the *for* loop proved to be difficult for students who previously had problems with the *while* loop. They asked questions which showed that they still lacked the insight into the required way of thinking. Besides this lack of insight, the *for* loop itself did not raise many questions from the students. One student just placed all the blocks inside the body of the *for* loop without knowing why it should or should not be in there, a problem previously seen in the first lesson. When the teacher and the student sat down together and walked through the program he created step-by-step the student saw what he was doing wrong and fixed it. Although a few questions were asked about the use of a *list*, there were no big problems here.

During the final assignment the students were required to create a quiz application. Like the other assignments, this assignment did not count for a grade. The mistakes and questions the students had about the final assignment will be discussed in the related results section. One remark we want to place here, is that a lot of students tried to solve this assignment without using a for loop, comparing a text against the *list*-variable inside an *if* statement, visible in Figure 4. As a *list* is called like a normal variable which makes placing it inside an *if* statement is possible, but when the application is executed it will not work as expected. The students just finished working with the *for* loop in the previous lesson and had already forgot when and how to use it. This clearly indicates there is a problem with the students understanding of the *for* loop.

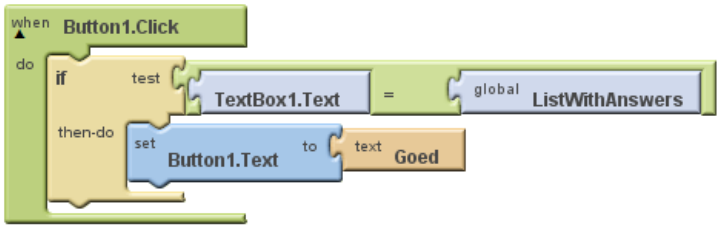

Figure 4: Incorrect use of the *if* statement with a list

4.2 Progress Form Results

Besides the notes taken by the researcher/teacher, three selected students kept records on their own progress by filling out a progress form after each lesson. The first question, asking what the student learned during that lesson, did not result in useful data. All three students simply stated which control structure they learned without providing more information.

From the second question we can see that the first student started with a good motivation on the course but when he found out it was more difficult than he had expected his motivation dropped a bit. Later he received some extra explanations from the teacher about thinking step-by-step and became motivated again because this allowed him to finish the course. The second student showed a steady motivation during the course, he answered that he liked working with App Inventor and that App Inventor motivated him. He showed a higher motivation during the final assignment because he wanted to successfully finish the course. The third student also showed a steady and good motivation. This student owns an Android phone and answered after the first lesson that the possibility to run the app on his own device motivated him. During the following lessons he answered he thought it was interesting to do because it allowed him to be creative and come up with his own solutions.

4.3 Final Assignment Results

Students needed to create a small application for the final assignment in which the user could answer a question and check the given answer to a list of possible correct answers. The student needs to create two versions of this application, in the first version only one question needs to be answered. In the second version two questions need to be answered, as shown in Figure 5.

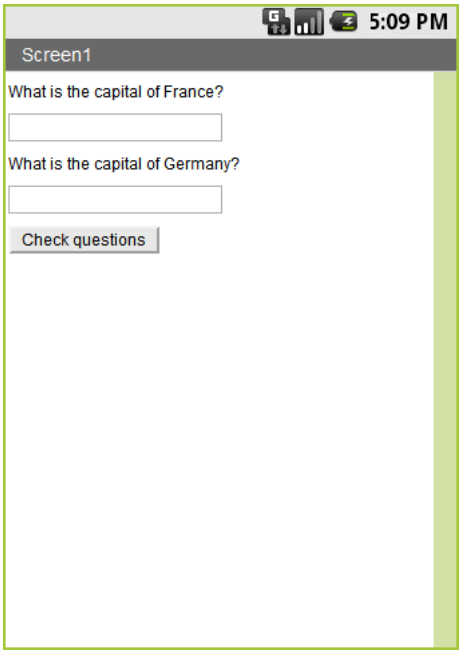

Figure 5: Final assignment

There are various ways of creating this application, The easiest way to accomplish a working solution is by creating a *for* loop which loops through the lists of

possible answers and then compare each possible answer to the given answer with an *if* statement inside the *for*'s body. Other possible solutions include a *while* loop for doing this. It is up to the students to think of their own solution to solve this assignment. The teacher did not interfere on the student's thinking process. The blocks editor in App Inventor prevents syntax errors by only allowing only the correct kind of blocks to be placed, semantic errors are not prevented. This allows us to see whether or not the student has understood the control structures well enough to avoid making semantic errors.

All students came up with a variation to the above solution. None of them used a *while loop* during the final assignment. A few students tried to compare the given answer to the complete list within a single *if* statement. The teacher did help these students by pointing them back to the previous lesson in which the *for* loop had been explained. After this the *for* loop was implemented correctly by most students. Two of the students misplaced the variables that need to be connected to the *for* block, the variable in which the *list* is stored and a temporary variable in which the current selected list value is stored during the looping.

A mistake that showed up in a couple of students' work proves that these particular students have not yet gained a perfect understanding of the required lateral thinking. These students used an *if-else* for setting the text to "Wrong" or "Correct" inside the *for* loop, this way the text gets overwritten for each next value from the *list*, visible in Figure 6. A good way to prevent this from happening is using a single *if* statement inside the *for* loop to set a variable. Then, after the *for* loop has finished, check if this variable now contains the correct value and set the right text according to the value. This solution is used by most of the students. These students do show a good understanding of the necessary logic.

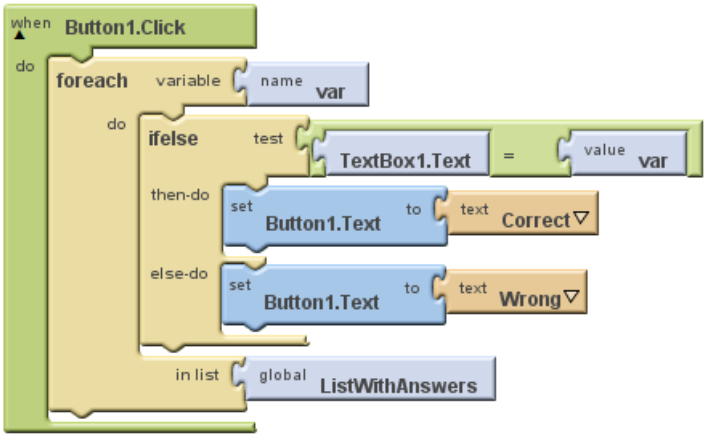

Figure 6: An incorrect solution made by several student

Two students created the second version by recreating the exact same code of the first version for a second time in the same project, resulting in a working solution but using two buttons to check the questions instead of one button. One student created a larger application with more questions and functions than originally intended. This student showed a perfect understanding of App Inventor and control structures.

4.4 Questionnaire Results

The questionnaire results give us some insight into the motivation the students had during the App Inventor course and whether or not they believe to have learned the requested control structures. The first seventeen questions are on general student motivation of learning with App Inventor, the following questions on the use of App Inventor and control structures. The complete Dutch questionnaire is available in the appendix. The answers are on a Likert scale, ranging from 1) strongly disagree, 2) disagree, 3) no opinion, 4) agree and 5) strongly agree. Each student answered the questionnaire within a week after they had finished and had handed in the final assignment of the course. As stated earlier, the sample group consists of thirteen students. The questions which play an important part in the conclusion of this research are described in text and visualized in figures. The other questions are only described in the text.

		Strongly disagree	Disagree	No opinion	Agree	Strongly agree
1	Whether the App Inventor content is difficult or easy, I am sure that I can understand it.			1	8	4
2	I am not confident about understanding difficult App Inventor concepts.	1	6	2	3	1
3	I am sure that I can correctly finish the App Inventor course.				10	3
4	No matter how much effort I put in, I cannot learn App Inventor	4	9			

Figure 7: Results of questions 1 to 4 of the questionnaire

The first four questions are about the students' ability to finish the App Inventor course and their ability to incorporate App Inventor. As seen in Figure 7 most students answer they are sure that

they are able to understand App Inventor. The second question shows that some students doubt their own ability to understand difficult concepts. However, most of the students are confident about themselves, which is also shown in the answers to the third and fourth question.

Questions five up to and including seven are on the student wishing to finish the difficult parts themselves, asking for help or rather omitting the difficult parts. All students answered that they would not skip difficult parts nor ask others for help, they would rather learn to do it themselves.

Questions eight up to and including twelve are shown in Figure 8. The views on the usefulness of App Inventor in students' daily life are divided. Most students agree that is does stimulate their thinking process. Three students disagree about the importance of learning how to solve problems. This is a concern because programming is all about solving problems. These three students clearly do not understand the logic behind programming yet. Questions eleven and twelve show no surprising results.

		Strongly disagree	Disagree	No opinion	Agree	Strongly agree
8	I think that learning App Inventor is important because I can use it in my daily life	3	4	2	3	1
9	I think that learning App Inventor is important because it stimulates my thinking.			2	10	1
10	With App Inventor, I think that it is important to learn to solve problems.	1	2		8	2
11	With App Inventor, I think it is important to participate in inquiry activities.			2	6	5
12	It is important to have the opportunity to satisfy my own curiosity when learning App Inventor			4	9	

Figure 8: Questions 8 to 12 from the questionnaire with results

The next five questions, thirteen up to and including seventeen are about when the students feels satisfied with themselves and their work. Only one student answers he would not feel satisfied when he is able finish the final assignment. One other student answer that he would not feel

satisfied when he felt confident about the content of the course. All others either agree or strongly agree. The question if they would feel satisfied when understanding the difficult parts on App Inventor gets agreed by everyone. Feeling satisfied when the teacher agrees with their solutions and ideas is also something everyone agrees with. Getting this acceptance from other students would make twelve out of thirteen students feel satisfied.

The questions now switch from general questions on motivation to questions about the use of App Inventor, control structures and some of the keywords (from the concept maps) the students came into contact with during the course.

		Strongly disagree	Disagree	No opinion	Agree	Strongly agree
20	I understand the *if* statement				9	4
21	I understand the *if-else* statement				9	4
22	I understand the *while* loop			2	9	2
23	I understand the *for* loop				11	2

Figure 9: Question 20 to 23 from the questionnaire with results

The first two questions in this section, question eighteen and nineteen ask whether the students now know how to use and program using App Inventor, everyone agrees that this is the case.
The next four questions, shown in Figure 9, show positive results. Only two students are not sure if they correctly understood the *while* loop. The following questions asked the students if they could implement these same control structures correctly by themselves. This resulted in slightly less ‘strongly agree’ answers. Again the *while* loop gained the most uncertain answers, resulting in three students answering that they were not sure. Asking if the students understood and could implement a list and variable resulted in similar answers. This clearly points out that the students think the *while* loop is the most difficult.

		Strongly disagree	Disagree	No opinion	Agree	Strongly agree
32	I enjoyed working with App Inventor		1		7	5
33	I enjoyed creating a smartphone app		1		8	4
34	I enjoyed creating an Android app		3		6	4
35	My interest in creating smartphone apps has increased		2	2	6	3
36	My interest in informatics has increased		1	2	8	2
37	My interest in owning a smartphone has increased	2	2	2	6	1
38	My interest in owning an Android smartphone has increased	2	2	4	4	1
39	I want a follow-up course using App Inventor		2	3	5	3
40	I want to use App Inventor in my spare time		3	6	4	

Figure 10: Questions 32 to 40 from the questionnaire with results

The next questions are about the motivation gained from App Inventor. Figure 10 shows the answers. Only one student did not like working with App Inventor or creating smartphone applications. Three students disliked creating an Android application. Questions about raised interest show undecided results. Raising interest in the Android operating system is not a goal of App Inventor but it is interesting to see if a student's personal opinion about an operating system can change by developing applications for it. Only four students consider using App Inventor in their spare time, whereas eight students want a follow-up course using App Inventor, which we think is a positive outcome.

A final question asks what kind of mobile phones the students own at the moment, shown in Figure 11. Only three students have a smartphone running the Android operating system. Two students are using an iPhone and one student has a general smartphone. All other students use a normal feature phone. Now we know which phone a student has, it is

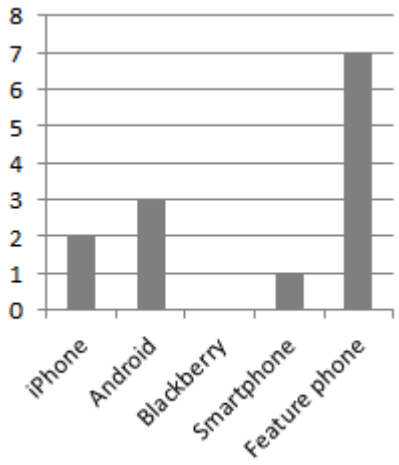

Figure 11: Phone quantities

possible to look back at the answers in the questionnaire to see if this may have influenced their answers. One of the iPhone owners did answer all questions about enjoying the usage of App Inventor and Android negatively, while in contrast to his the second iPhone owner did answer those questions positively. One of the Android owners did answer all these questions positively.

4.5 Interview Results

In this section the results of the interviews will be presented. Six students were interviewed, three vwo and three havo students. Quotes by the students will be provided to support the results. The block elements (for building the code in App Inventor) are received well, the students understand that the blocks are an easy way to learn how to program. When asked if the course was easy or difficult one of the students answered *"Easy, it is just logical thinking"*. A second student answered *"It looked simple because of the colorful blocks, but it was not simple at all"*. Some students finished their answer by saying it had motivated them to learn how to write real code after using the blocks.

The *if* statement is found the easiest of all control structures, each interviewee could explain the *if* statement without any problems. Quoting one of the students on explaining the *if* statement: "*That is when the value equals something you want. If that is true it executes the code, if not, it does nothing".* The keyword 'condition' is only correctly remembered by one student, all others can explain what it means but did not use the name.

The *while* loop is found the most difficult of the three control structures. All interviewed students had trouble explaining what it does exactly. When asked for an example on what it could be used for, some can sketch a situation in which one would use a *while* loop. Most of the examples they sketched came directly from the assignments. It was unclear to them in which other situations the *while* loop should be used. Some even came up with completely wrong situations, as the following student: "*That was the most difficult of them all. I think it was to do something during an event. For example while a button is pressed or while the background of the screen is blue"*. All students did remember the infinite loop dangers and how to prevent this from happening, but only when asked about it, one of the student brought this up himself.

The students who could remember the *for* loop found it easier than the *while*. The use of the *for* loop in App Inventor is limited to scrolling through lists, which confused a student who started coding directly after finishing the App Inventor course. Those who could remember the *for* loop explained it perfectly. Those who did not remember had huge problems on recalling when asked follow-up questions about it. Most of them thought the *for* loop compares the values in a *list*, quoting *"for each variable from the list which equals a given value it executes the code"*. The *for* loop does not compare values, it just loops through the variables in the list. In the assignments each time the *for* loop was used, it was used in combination with the *if* statement. The repetitive use of this combination may have confused the students.

All interviewees, with one exception, liked using App Inventor. The interviewee who had said he had disliked App Inventor is the same as who had answered this in the questionnaire. The students who liked it explained programming for smartphones as something new, which otherwise they would never have come in contact with. From those who found it difficult, some thought that was a good thing and liked it because of that, one student did not like it because it was difficult. Two students did not want a follow-up course on App Inventor, one of them explained *"because it was difficult I had to call for the teacher every time. I would like to be able to do my work myself"*. The other students did like the idea of a second App Inventor course, one of them said "*Yes. We only made simple applications this time and I would like to create something useful"*.

Only two of the interviewed students thought about using App Inventor at home for creating an app themselves. The others did not have that interest. None of the interviewee was in possession of an android phone, each of them said this was disappointing because they could not try the app they created. One of the students said he would prefer creating a pc application with App Inventor because he would be able to test and use the application himself. The others said this would make no difference.

To the question whether or not the next year's new students should also have App Inventor in their curriculum, all of the interviewees answered a yes. They said it showed them what programming entails and how to start thinking like a programmer. One student explained it perfectly *"Yes, it is a nice beginning to learn programming. The blocks make it easy and tell you precisely which values are needed."*

5. Conclusion & Discussion

5.1 Conclusion

A conclusion will be made for each of the three research questions. Each question will be answered by using the results we have gathered from the five different data collection methods.

The first question was: Can App Inventor be used to teach the loops and conditions control structures? Concluding from the field notes we can say that it is possible. All students finished the lessons, some with additional help from the teacher. Most questions asked were on the *while* loop. Most students correctly finished the final assignment, only a few handed in an insufficient solution. Students knew how to use the control structures. The few semantic errors they made proved that the skills they developed are not yet perfect but certainly sufficient for their experience. From the answers given during the interviews we can conclude that they can recall most control structures correctly. All students had some difficulty in recalling the *while* loop. The *while* loop shows up in all methods as the most difficult to attain, the other control structures are well received.

The second question was: Does the use of App Inventor during lessons motivate students? From the field notes we can conclude that the students were motivated during the lessons, they did not get bored or started to annoy other students during the lessons. The progress forms show the same result, the motivation is high and remains high during the whole course. The questionnaire clearly concludes that the students liked working with App Inventor. With the exception of one student, all of the students answered the questions on this positively. Quoting one of the

interviewed students: *"I liked it! And by the looks of it, everyone was working on it. It's something different, different is fun"*

The third question was: What are decisive factors in motivation and achievement of students? The students realized that working with an ILE is an advantage above writing programming code. This advantage did play a part of them understanding the required thinking process to program. During an interview a student explained that he understood that the blocks were an easy way of coding, preventing them from making syntax errors. The interviewees say they liked working with App Inventor. The biggest factor in the students' positive motivation this is the idea of working on something new. It is something they did not use before. Something the students would otherwise never have come into contact with. The fact that they were creating a smartphone application only slightly raised their interest. It did not play a large part in their motivation. None of the interviewees owns an android device, which may have influenced the answers because they were disappointed they could not test the application on a real Android device.

5.2 Discussion

The conclusion shows that students attained the control structures from using App Inventor. However, when asked during the interviews to explain, some students had big problems doing so. Previously they had answered they understood the control structures in the questionnaire. Explaining something is more difficult than understanding it, but it makes us question the students' understanding of these control structures on the long-term.

Our research was performed on a respectively small sample group from one school, following one iteration of the App Inventor course. The results may differ when a bigger sample group is used from different schools, or following multiple iterations of the course allowing improvements to the designed series of lessons based on our findings.

The series of lessons will be updated based on the results of this research. This second version of the lessons will be made available in for interested informatics teachers. The school on which the teaching experiment was held will continue to use the lessons in their curriculum.

The Dalton Plan used at the school on which the teaching experiment was held entails that the students mostly worked independently on the course. For this reason the lessons in the research were also organized according to the Dalton Plan. Further research can be done using traditional group instruction, as these results may differ.

Further research can be done to see if the views on App Inventor differ when a student owns a iPhone, Blackberry or Android smartphone. As many young people have a strong preference for one of three, this preference may influence their opinion about creating Android apps. The two iPhone owners from our sample group had contradicting opinions about this in the questionnaire. One did enjoy programming for the Android operating system while the second made clear he would prefer creating apps for the iPhone. We think it is interesting to see more results on this.

References

Abelson, H., Chand, M., Mustafaraj, E. & Turbak, F. (2010) . Mobile phone apps in CS0 using App Inventor for Android: pre-conference workshop. *Journal of Computing Sciences in Colleges,* 25(6), 8-10. Retrieved from http://dl.acm.org/

Cranor, L. & Apte, A. (1994). Programs worth one thousand words: visual languages bring programming to the masses. *Crossroads - Special issue on programming languages*, *1*(2), 16-18. doi:10.1145/197149.197163

Engelsma, J. & Dulimarta, H. (2011). 8 Reasons Why You Should Use Mobile Platforms in Your CS Courses. *ITNG '11 Proceedings of the 2011 Eighth International Conference on Information Technology: New Generations* (pp. 245-250). doi:10.1109/ITNG.2011.50

Fincher, S., Cooper, S., Kölling, M. & Maloney, J. (2010). Comparing alice, greenfoot & scratch. *Proceedings of the 41st ACM technical symposium on Computer science education* (pp. 194-195). doi:10.1145/1734263.1734327

Gestwicki, P. & Ahmad, K. (2011). App inventor for Android with studio-based learning. J*ournal of Computing Sciences in Colleges*, *27*(1), 55-63. Retrieved from http://dl.acm.org/

Kinnunen, P. & Malmi, L. (2006) Why students drop out CS1 course? *ICER '06 Proceedings of the second international workshop on Computing education research* (pp. 97-108). doi:10.1145/1151588.1151604

Mahmoud, Q. H. & Dyer, A, (2007). Integrating BlackBerry wireless devices into computer programming and literacy courses. *ACM-SE 45 Proceedings of the 45th annual southeast regional conference* (pp. 495-500). doi:10.1145/1233341.1233430

Mahmoud, Q. H. & Popowicz, P. (2010). A mobile application development approach to teaching introductory programming. *Frontiers in Education Conference*, T4F-1 – T4F-6. doi:10.1109/FIE.2010.5673608

McCracken, M., Almstrum, V., Diaz, D., Guzdial, M., Hagan, D., Kolikant, Y. B., . . . Wilusz, T. (2001). A multi-national, multi-institutional study of assessment of programming skills of first-year CS students. *ITiCSE-WGR '01 Working group reports from ITiCSE on Innovation and technology in computer science education* (pp. 125-180). doi:10.1145/572133.572137

Milne, I. & Rowe, G. (2002). Difficulties in learning and teaching programming—views of students and tutors. *Education and Information Technologies*, *7*(1), 55–66. doi:10.1023/A:1015362608943

Parkhurst, H. (2009). *Education on the Dalton Plan: -1922*. Ithaca, NY. Cornell University Library

Robins, A., Rountree, J. & Rountree, N. (2002). Learning and Teaching Programming: A Review and Discussion. *Journal of Computer Science Eduction*, 13(2), 137-172. doi:10.1076/csed.13.2.137.14200

Tuan, H., Chin, C. & Shieh, S. (2005). The development of a questionnaire to measure students' motivation towards science learning. *International Journal of Science Education*, 27(6), 639-654. doi:10.1080/0950069042000323737

Wolber, B. (2011a). App inventor and real-world motivation. *Proceedings of the 42nd ACM technical symposium on Computer science education* (pp. 601-606). doi:10.1145/1953163.1953329

Wolber, B. (2011b). *App Inventor: Create Your Own Android Apps*. Sebastopol, CA: O'Reilly Media

Appendix

		Sterk niet mee eens	Niet mee eens	Geen mening	Mee eens	Sterk mee eens
1	Of AI nou moeilijk of makkelijk is, uiteindelijk zal ik het begrijpen					
2	Ik ben er niet zeker van of ik moeilijke onderdelen van AI kan begrijpen					
3	Ik ben er zeker van dat ik de AI cursus goed kan afsluiten					
4	Hoe veel moeite ik er ook in steek, AI kan ik niet leren					
5	Wanneer AI onderdelen moeilijk zijn sla ik deze over en doe ik de makkelijke onderdelen					
6	Tijdens de AI lessen vraag ik liever de oplossing aan anderen dan dat ik zelf nadenk					
7	Wanneer ik AI onderdelen moeilijk vind probeer ik dit niet te leren					
8	Ik denk dat AI leren belangrijk is omdat ik het in mijn dagelijks leven kan gebruiken					
9	Ik denk dat AI leren belangrijk is omdat het mijn denkvermogen verbeterd					
10	Ik denk dat bij AI het belangrijk is om te leren hoe problemen op te lossen					
11	Ik denk dat bij AI leren het zelf doen belangrijk is					
12	Het is belangrijk dat ik mijn eigen nieuwsgierigheid kwijt kan bij AI					
13	Ik voel mij het meest voldaan als ik de eindopdracht goed heb gedaan					
14	Ik voel mij het meest voldaan als ik me zeker voel over in de inhoud van de lessen					
15	Ik voel mij het meest voldaan als ik in staat ben een moeilijke opdracht op te lossen					
16	Ik voel mij het meest voldaan als de docent mijn ideeën accepteert					
17	Ik voel mij het meest voldaan als de andere leerlingen mijn ideeën accepteren					

		Sterk niet mee eens	Niet mee eens	Geen mening	Mee eens	Sterk mee eens
18	Ik begrijp nu wat AI inhoud					
19	Ik kan nu programmeren met AI					
20	Ik begrijp nu de IF statement					
21	Ik begrijp nu de IF-ELSE statement					
22	Ik begrijp nu de WHILE lus					
23	Ik begrijp nu de FOR lus					
24	Ik begrijp nu de LIST					
25	Ik begrijp nu wat een variabele is					
26	Ik kan de IF statement nu zelfstandig toepassen					
27	Ik kan de IF-ELSE statement nu zelfstandig toepassen					
28	Ik kan de WHILE lus nu zelfstandig toepassen					
29	Ik kan de FOR lus nu zelfstandig toepassen					
30	Ik kan de LIST nu zelfstandig toepassen					
31	Ik kan een variabele nu zelfstandig toepassen					
32	Ik vond het leuk om met AI te werken					
33	Ik vond het leuk om een applicatie te maken voor een Smartphone					
34	Ik vond het leuk om een applicatie te maken voor Android					
35	Mijn interesse om applicaties maken is gestegen dor deze lessen					
36	Mijn interesse in informatica is gestegen dor deze lessen					
37	Mijn interesse in een Smartphone is gestegen door deze lessen					
38	Mijn interesse in een Android telefoon is gestegen door deze lessen					
39	Ik wil in de les meer leren met AI					
40	Ik wil buiten de les meer gaan leren met AI					

Mijn eigen telefoon is:

A. Een Android telefoon
B. Een iPhone
C. Een Blackberry
D. Een andere smartphone
E. Een normale telefoon