Fall 2014

# Report on MatteGeni

IMT 3672 Mobile Development Project

Kjersti Lien (110262) og Bård Løne (110272)


# Index

# Introduction

During the Mobile Development Project course, we have developed an educational app for students in Elementary School. The purpose of the application is for students to solve math problems in a fun way - and encourage them to work and enhance their math knowledge at home using a tablet or a smart phone.

## *Getting started*

We started the project by picturing how we thought the app should work, what features it should have and what it should look like. Since there has been a while since we ourselves attended Elementary School, we had to find out what the curriculum was for the seven different grades. This gave us a framework as to what the math problems should (and could) involve based on what the students actually learn in school, or what they are going to learn.

## *Considerations*

First, we had to picture how we thought the app should look like and if we were going to divide the different grades into levels of difficulty. We decided that for each of the grades, from first to seventh grade, there would exist three different levels of difficulty. Dividing the grades into different levels also meant that we had to divide the curriculum in each of the grades, and split them as we saw fit between the levels, in such a way that the easiest problems would be in level one, and the most difficult problems would be in level three.

We were also considering setting up a log in feature, but decided to skip it since the purpose of the app is to practice math and not compete or show/share the scores they have gotten. Another reason for not adding log in as an option was their age; the first graders are either five or six years old and the oldest students are aged 11 or 12. The log on would therefore have been an unnecessary feature, since it would be difficult for the young ones, and they wouldn't have gotten any use for it after all. However, if we want to develop the app further, we could add log in for higher grades and add features that allows the students to play against each other – solving problems.

One of the first things we started discussing was the layout of the app and we decided that it should look fun to play with. Another consideration was that it had to be easy to use and easy to understand. This resulted in us creating a colourful prototype for the layout.

Naming the app turned out to be quite easy as well, the name "MatteGeni" is both short, easy to remember and catchy. It also gives the users and idea of what the app is all about; mathematics.

## *Layout*

The Layout xml files are all stored in the res/layout folder.

There are mainly four layouts; the main activity layout which is the first activity in the app. This layout displays texts “Velkommen”- and “Velg klassetrinn”, along with the seven buttons used to choose at which grade they want to solve the problems. The ChooseLevel(…) files are the files that shows the layout for the different levels in each of the grades. They all look the same; three buttons for choosing the levels. The only difference is that by clicking the buttons they open up the activity to the chosen level in the chosen grade.

The activity_grade(…) layout displays the math problems for each of the grades, generated by the java files in src. These problems will depend on the level they’ve chosen, but the layout remains the same.

The answers layout is the last layout and it displays the answer the user has clicked, along with the correctly generated answer. The layout also contains a button “neste” that brings the user back to the right level in the right grade, with another math problem to solve.

## *Coding mathematical problems*

In the different java files that are named for example GradeOneLevelOne, we have generated different math problems by using two generated random integers, and two random operators. These randomly generated ints and operators are used for defining a math problem. Once the problem has been created, the question/problem will be shown in a textView. Underneath the question there is a field in which the students can input their answer, and click the “svar” button. When the “svar” button is clicked the activity of checking the answers against one another is started.

Note: depending on the grades and the levels, the number of random integers and random operators will vary. Example: in grade seven, there will be four operators instead of two, since the difficulty of the math problems are higher. The students in first grade only solve problems with + and -, while the seventh graders solve problems with +, -, * and /.

## *Checking answers from user input*

We had a lot of trouble accessing the “answer” from one java file to another, since the answer is generated in the file with the math problems, and the answers from the user and the correct answer is checked against one another in the CheckAnswers.java file. We luckily found a way around this, so we could display both the calculated answer from the code and the answer from the user input.

## Files and classes

Main Activity: This is the class in which the app starts, so the layout for this class is the first thing to come up when you open the app. The main class exists of the seven buttons that they can choose between – the seven grades of elementary school. This is here we declared the onClickListeners for the buttons, and made a switch case for clicking the different buttons. If for example button 6 is clicked, a new activity will open; in this case the ChooseLevelGradeSix Java file.

ChooseLevel files: The ChooseLevel files is where they choose which level they want to start solving math problems in. There are seven off these ChooseLevel files – one for each of the grades. When they've first chosen grade 6 (in main activity), and then level 1 (in ChooseLevel), another activity will open. In this case the file GradeSixLevelOne will open, where they can solve the actual math problems.

Grade files: To continue the previous example (GradeSixLevelOne), this is of course where the actual problems are made. We have made every grade file fit with the curriculum for each of the grades, so we have in this example made problems that consists of random numbers, with max and min, fitting the curriculum for sixth grade. All of the files has from 2 to more random generated numbers, and from 2 to 4 random generated operators. The files also contains an int answer, so the program can calculate the right answer based on the generated numbers and operators.

When the random generated numbers are made, the question/problem is displayed in a textView in the layout for these files (example: grade_six_level_one.xml. Beneath the problem in the TextView, there is an editTextView, where the user can input their answer. When they've answered, they click the "svar" button, and a new activity opens: example CheckAnswersGradeSixLevelOne.

CheckAnswer files: these are they files that check the input answer from the user up against the correct answer the Grade files has calculated. The layout for this file will show first what the user has answered, and next what the correct answer is. At the button there is a button where they can click to retrieve another question in the same category as they are in. In this example, clicking "neste" will then open the GradeSixLevelOne activity again.

## Testing

All the test files is in the src/mattegeni.test package. There are only four tests which are testing the functionality of ChooseLevel files, Main Activity, Grade files and CheckAnswer file. Since the files contain almost the same code, we found it unnecessary to test them all. We also tested GUI features,

## Problems

As mentioned earlier, we had some difficulties getting the right answer to turn up. This messed a lot with the score function, so we decided that (for now), the score will not be part of the app. This means of course, that it might not be as fun as the student would like, but the purpose of the app is just to exercise solving math problems after all.

# Conclusion

Both group members took the Theory part of the course last fall (2013), so we were a bit rusty about how we actually generated the small app we did in the course last year. It took us a while to remember everything we learned about android development – how to set up buttons and layouts, where the layouts were stored, how to start new activities, how to add the activities to the Android Manifest and so on - but once we got the hang of it we were able to do a lot of work in a short period of time. Unfortunately, because of other projects and sickness we put the project on hold for a while, so there was a lot of work to be done for the two final weeks of the project.

We thought the project was a lot of fun, and we want to complete the app at another time by adding the missing component; the score function, and possibly develop the app further by adding log in functions and setting up users to play against each other.