TDDT

GROUP: NULL

Programmierpraktikum

Project 7

Title page

Title	Group Project - TDDT
Author	Christian Karamann (24 21 98 8) Marvin Krause (24 14 15 7) Elmedin Turic (24 21 50 6) Patrick Remy (23 85 48 0) Pascal Fischer (23 84 94 4)
Learning Arrangement	Programmierpraktikum
Institute	Heinrich Heine Universität
Location of Institute	Düsseldorf, Germany
Learning Arrangement	Computer Science
Study Phase	2. Semester
Academic Year	2016
Date of Completion	14.07.2016
Language	English

Table of Contents

1. Open Questions:

To begin with, open questions that appeared doing the process of creating this application, should be addressed. On top of that, specific aspects need to be discussed, that influence the way, the project has been designed.

Features:

To begin with, the group needs to specify, which features they want to use. In total, there are three possible choices. These features are explained in the Project description. The group decided on Babysteps and Tracking. The reason, why they decided to go for Babysteps, is that it is very useful for the user later on. It should help the user, learn how to write the most minimalistic code possible. On the other hand, Tracking is useful for the teacher to see, where the students were struggling the most. Those are the reasons for the groups' choices.

Storage necessary:

The next aspect that needs to be clarified is the necessity of a class that stores the input, done by the user. The group decided to implement a feature, where they could store all the input the user did, so that if the user closes the application and re-opens it, the user can continue working on his code.

Type of input:

Next, the group needs to decide, what type of file the input file should be. Like in the example handed to us by the teacher, the group decided to use an XML-file as an input type.

Autosave:

The group decided to implement a way of autosaving all the code, written by the user. As soon as the user switches phases, the written code will be saved in the default path in a corresponding file.

Code highlighting:

In the end, the group implemented a way, so that the written code gets highlighted, depending on which keywords the user used. For example, if the user writes the word „public", it gets highlighted, like it would in IDEs like Eclipse or Intellij.

2. Project:

After clarifying the open questions, it is necessary to explain the purpose and goal of this project. On top of that, it is crucial to describe the repository and its structure.

Project description:

To begin with, it is important to clarify the general goal of this project, as well as its purpose and function.

The goal of this project is to write an application with a GUI where students would be able to learn how to program on test-driven basis. Therefore, the students would get together in groups of 3 to 5 people and work on it for 4 weeks. The application should be build in gradle and support travis, two tools which should help the students in their project. On top of that, the project should contain a manual and a report, which describes the whole project.

Now, the general idea of this project is going to be explained. Like mentioned before, the application should help the user understand and learn how to program in a test-driven way. Therefore, first, the user should be able to choose between different exercises. Here, the teacher sends the student a XML-file, which works like a catalog containing different exercises. From this catalog, the student can choose an exercise and start working on it.

In total, there are 3 phases which form a cycle:

First, the user needs to write exactly one test that fails. He is not allowed to write multiple failing tests or else he will not be able to continue. Second, the user needs to write code in order to pass that one failing test. After passing the test, the user gets into the third phase, where the user is allowed to improve his written code. When this code also passes the tests, the user is allowed to write the next failing test and so on, until the whole program succeeds.

On top of all that, the students could decide for two out of three possible features, which the application should have. The first one is called Babysteps and it limits the time, the user has to finish one phase. The second one is called Tracking, which should show a graph of how much time the user spent in every single phase. The final feature is called acceptance testing. Hereby, the user should write a test before going into the first phase. This test should stay unfulfilled until a feature has been implemented completely. It can be seen as a final test for an implemented feature.

For this project, the group decided to implement the features Babysteps and Tracking.

To sum it up, the group should write an application with which the first semester students would learn how Test-Driven-Development works.

Repository description:

After explaining the general idea of this project, it is crucial to describe the groups' repository and its structure.

To start, the repository is called "programmierpraktikum-abschlussprojekt-null". It mainly contains two folders and several files, which are going to be described.

To start, the folder called "Project 7" contains all relevant files that are necessary to execute the program. This includes Gradle- , Java- , Git- and XML-files. In this folder, you can find several other folders and files that are necessary to run the project. The only files that are not included in this folder, are files that are necessary for the project, but not relevant for the execution of the final application.

The repository includes a travis-file as well, which is used to check for compile-errors and notification purposes.

On top of all that, there is a folder that contains all protocols which have been documented during the process of this project.

A text-file which contains the MIT-license is also included in this repository.

In the end, there is the README-file, the user manual and the report.

To sum it up, the repository is divided into general files and files that are necessary for the program. The ones that are required to run the final program are located in "Project 7" whereas the general ones are located directly in the repository. Now, the files inside the folder "Project 7" need to be analyzed, because they are the core-files of the application.

3. Classes

Now, that the open questions have been clarified and the repository has been explained, it is necessary to talk about all the different classes, which are located in the "Project 7" folder.

To begin with, the folder "Project 7", can also be divided into sub-packages and files. The first file that needs explanation is called ".classpath" which handles errors, bug fixes and moves resources.

Next, the file ".gitignore" ignores certain files from the storage, which have been selected by the group.

The next two files are handling the building process for gradle. The two main files are called ".project" and "build.gradle". The first file integrates gradle in eclipse and the second one builds it.

The final file is called "default.xml" which contains example-exercises for this project. It is used as a testing file for the application.

Now, only three folders are left. To start, the "Storage"-folder contains only one class, namely "Catalog.xml". Later on, the teacher should put their exercises in this file.

The folder ".settings" merges the gui into the xmlParser and the "src/main" folder has all the different imports.

The "main" folder can also be divided into "java" and "gui". In "gui", there are all the resources located that are necessary to show the different scenes. It includes, icons, styles and different fxml-sheets which represent the different scenes.

On the other hand, the “java”-folder contains all java-files. To start, the “Main” file is necessary as a starting position for the program. It also initializes the GUI. Next, the files can be divided into four categories: “gui”, “models”, services” and “xmlParser”.

First, the “gui” folder is going to be described. The purpose of this folder is, to control and change the GUI of the program. It consists of controllers and views. There are different controllers and views for every part of the program. Hereby, the controller “controls” the corresponding fxml-file, which is located in the “resources/gui” folder. This includes the three phases the program is running through, as well as the overall menu. The controllers for the phases Red to Blue can be found in the sub-package called “cycle”.

Second, the “models” folder needs to be explained. Here, all the classes are located, that get created via the XML-Parser. Every exercise that gets created contains a class-, config-, test-, and trackingData-file.

Third, the “services” folder is going to be illustrated. In this folder, all the implemented features are located. To begin with, the BabystepsService.java file should implement the feature “Babysteps” into the program. Next, CompileService.java combines both, CompilerResult and TestResult, which has been given to the group. In the end, there is also a StorageService.java file which should handle storing the input, done by the student.

Fourth, the “xmlParser” folder contains all classes relevant for the XML-Parser.

To sum it up, the whole project has been divided into several minor packages in order to keep a good overview of all the different files which are necessary to make the program run. To close, the whole structure of the project and the functions of the different classes have been explained briefly.

4. Critical Appraisal:

In this Chapter, the group would like to address some points that make this project different from any other project.

To begin with, the group would like to address all the possible differences compared to other groups' projects. To start, the group had to decide for two out of three features. The group decided to go for Babysteps and Tracking. There is a high possibility, that other groups do not have the same features as we have.

Another aspect that could be different to other groups projects, could be the overall design. The group decided to have two different bars. One bar is shown on the left side of the screen. This one shows the general overview of all the exercises and options that user opened. The other bar is on the right side of the screen. This is the area where the user should write his code.

Next, the group would like to address the menu-system. The menu can be displayed by dragging in a XML-file into the drag and drop scene, at the start of the project. This feature is probably unique, compared to other projects.

Now, the group would like to mention aspects that always stay the same.

To begin with, every group had to decide for two out of three features. This means that, compared to other groups, at least one feature is the same. The only thing that is different, is the way it is implemented.

Moreover, the three phases, the user has to go through, stay the same. This is how TDD normally works. Therefore, every group needs to implement these steps in some way.

To sum it up, there are many features and aspects that make our application unique. However, there are several aspects that will never change, no matter which group is working on it.

In the end, we would like to thank our tutors for supporting us in the past four weeks and giving us advice when we needed it. The group enjoyed working on this project and hopes that this joy is represented in our application.