<ASSESSMENT>

TESTING GENERAL USE CASES AND INITIAL REFACTORING

VERSION <1.0>

Ιορδάνης-Ραφαήλ Φυδανάκης AM:420

Ευάγγελος Δημουλής AM:421

-

TABLE OF CONTENTS

INTRODUCTION

The objectives of this phase is to develop tests that involve general use cases of the Plutarch Parallel Lives tool and do some basic refactoring on the legacy code.

You can find the source code of the project in GitHub at:

https://github.com/JordanRaphael/l1-software-data-evolution

REFACTORING

REFACTORING

In order to test the GUI's main functionality we had to apply some basic refactoring to the code of the GUI class, or else we wouldn't be able to test the methods without adding/changing their code. That said, we extracted the Action Listeners from the GUI to another class called EventListenerHandler which is responsible for generating the Action Listeners that are being used by the Jitems of the GUI.

Also, we added the class JItemsCreator which is responsible for handling the creation and initialization of Jitems such as Jbutton, Jlabel, JscrollPane etc. Furthermore, we moved the business logic methods from GUI to a new class called BusinessLogic, in order to break the connection between the GUI and GlobalDataKeeper and reduce the overall coupling.

As a last step towards refactoring the legacy code, we generated a Factory class called TablesListenerFactory that is responsible for creating the listeners of the basic JvTables of the tool General Table and Zoom Table. The creation of this Factory class stems from the fact that the newly generated EventListenerHandler mentioned above increased too much in size. This stage of the refactoring is still on-going and not completed yet.

TESTING

TESTS

We developed tests for GUI’s *createProject*, *editProject*, *loadProject*, *zoomIn* and *zoomOut*. In each test we have a ground truth and we test that the methods have the expected results.

For the GlobalDataKeeper we developed a test to verify that the method *setData()* works properly. To achieve that, we had to create a file with the expected results of the *setData()* method. We then call the *importData()* which uses the *setData()* and other methods to insert the data to the GUI components. When this step is complete, we check if the imported data match perfectly the data that we wanted to import. To summarize the test we introduced, here is a bullet list:

<u>List of tests implemented so far:</u>

- testCreateProject
- testEditProject
- testLoadProject
- testZoomIn
- testZoomOut
- testSetData

NEXT STEPS

In the next phase we are going to develop even more tests for the GlobalDataKeeper and if there is time we will build tests for the GUI as well.

We are going to apply more refactoring to the GUI and BusinessLogic classes (e.g. the EventListenerHandler class will be replaced by a Factory class whose development is work in progress).

We will also apply the pattern *"Move the behavior closer to data"* by moving functionality that does some useful processing on data to the DataKeeper turning the DataKeeper to a *"Service Provider"* . As a last step, we will try to refactor the newly generated DataKeeper by extracting sub-systems that handle the processing.