Unit Testing

WikiRace

Wikiracers

Version 2 - March 10, 2015

Unit Tests:

For this part, our group did not use any specific tool for testing. We decided to proceed that way because after some research we didn't find anything suitable for us and anything that is really integrated with Android to use, as there exists for other platforms. Our app is simple and a manual test of each module in isolation is not hard to do because the number of classes and activities are about 10. In addition, it was more comfortable and less time consuming to not use another tool, given the time constraints and availability issues that we had. Also, we could not dedicate all the time that we would like into this project. We are already using Github, Jira and having meetings, so adding a new tool will require learning it and then using it among all of us. Since each of us coded a different part, it would end up being really difficult to merge.

For testing we did three types of test:

- Playtesting of the game: this consists on playing the game to detect errors on the game loop as well as interface problems. This playtesting was done in an emulator, in an Android smartphone and in an Android tablet. That way we could have more efficient tests.

- Manual test: this tests were done by the programmer independently each time a new functionality was added. In addition, once the whole module was finished it was tested and the pushed into GitHub. This tests consist on debugging and checking correct output.

- Blackbox testing with equivalence classes: This is done when it is required an input to the whole system. The test cases that we proposed and check are:

Test case 1: Register fields

Test case 1.1: Username check

- Restriction: cannot be empty and has to be unique.
- Equivalence classes:

- Empty field: “”. Expected output: error msg pops up.

- Not empty field: any string with any character (including symbols).

    Expected output: can be submitted.

- Value already in database. Expected output: error msg pops up.

● Result: Success.

Test case 1.2: Name check

● Restriction: it is optional, not unique.
● Equivalence classes:

- Empty field: “”. Expected output: can be submitted.

- Not empty field: any string with any character (including symbols).

    Expected output: can be submitted.

- Value already in database. Expected output: Can be submitted.

● Result: Success.

Test case 1.3: Email check.

● Restriction: cannot be empty, it has to be of type email ([A-Za-z1-9]+@[A-Za-z1-9]+.[A-Za-z]+), unique.

● Equivalence classes:

- Empty field or value of not type email.

    Expected output: error message pops up.

- Type string: any string that is not an email regular expression.

    Expected output: cannot be submitted, error message pops up

- Type email: any string with any character (including symbols).

    Expected output: can be submitted.

- Value already in database.

    Expected output: can not be submitted.

● Result: Success.

Test case 1.4: Phone number check

- Restriction: it is optional, not unique, no specific format. The system only allows to use numbers.
- Equivalence classes:

  - Any input: empty or any character or string.

    Expected output: can be submitted.

  - Value already in database.

    Expected output: can be submitted.

- Result: Success.

Test case 1.5: Password check

- Restriction: cannot be empty, any length, not unique.
- Equivalence classes:

  - Empty field: “”.

    Expected output: can not be submitted.

  - Not empty field: any string with any character (including symbols).

    Expected output: can be submitted.

  - Value already in database.

    Expected output: Can be submitted.

- Result: Success.

Test case 2: Log in fields

Test case 2.1: Check credentials

- Restrictions: there are no restrictions for the format of the values in the input fields.
- Equivalence classes:

  - Invalid credentials: any value or empty value.

    Expected output: can not be submitted.

  - Valid credentials: pair of username and password in database.

    Expected output: can be submitted.

- Result: Success.

Test case 3: Add friends

Test case 3.1: Search input

- Restriction: Only adds friendship if the inputted string is a user.
- Equivalence classes:

  - Empty String: Expected output: user not found.

  - Non-empty String (user in database):

    Expected Output: friendship added to database.

  - Non-empty String (user not in database):

    Expected Output: user not found.

- Result: Success.