Name: Babes Adrian

User name: ab224sh

Git link: https://github.com/AndiBabes/ab224sh\_1dv600

**Testing**

**Test Plan**

The objective is to test the code that was implemented the last iteration.

I will be testing UC1 and UC2 by writing and running dinamic manual test-cases. Automated unit tests will be implemented for the methods: “play”, “stillPlaying” and “getWord” from the class ‘Play’.

|  |  |  |
| --- | --- | --- |
| Task | Expected | Actual |
| Manual Test Cases | 1h | 45m |
| Running Manual Tests | 10m | 5m |
| Unit Tests | 1h | 1h |
| Code Inspection | 20m | 20m |
| Test Report | 20m | 10m |

**Manual test cases**

**TC1 Start Game Successfully**

UC1 Start Game

Scenario: game starts successfully

The main scenario of UC1 is tested when a user starts the game successfully.

Precondition: none

Test Steps:

1. Start the game
2. The system shows the main menu with the buttons: “Play Game”, “View Highscores”, “Quit Game”
3. Click on the “Play Game” button

Expectations:

1. The game itself should start; The system will ask for input
2. See UC2

**TC2.1 Win game**

UC2 Play game

Scenario: The user plays and wins the game

The main scenario of UC2 Play game is tested. In this iteration the user is required to guess the word “testing”, instead of a random one.

Precondition: the game is running

Test Steps:

1. Start the game (see UC1)
2. The system will show a representation of the word using underscores and ask for input
3. Input the letters ‘t’, ‘e’, ‘s’, ‘i’, ‘n’, ‘g’ by typing them one by one and then clicking the “Check Letter” button

Expectations:

1. The system decides that all letters have been guessed and thus the game is won
2. The Game Over menu is shown with the message “You have won” giving the option to play again or to return to the main menu

**TC2.2 Win game**

UC2 Play game

Scenario: The user plays and loses the game

The alternate scenario 7.1 of UC2 Play game is tested. In this iteration the user is required to guess the word “testing”, instead of a random one.

Precondition: the game is running

Test Steps:

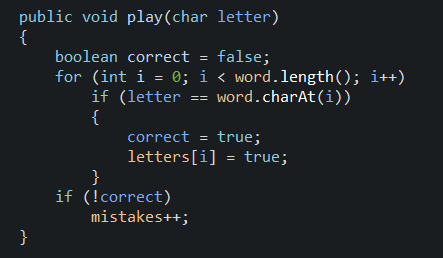
1. Start the game (see UC1)
2. The system will show a representation of the word using underscores and ask for input
3. Input the letter ‘a’ in the textbox and click on the “Check Letter” button for 6 times

Expectations:

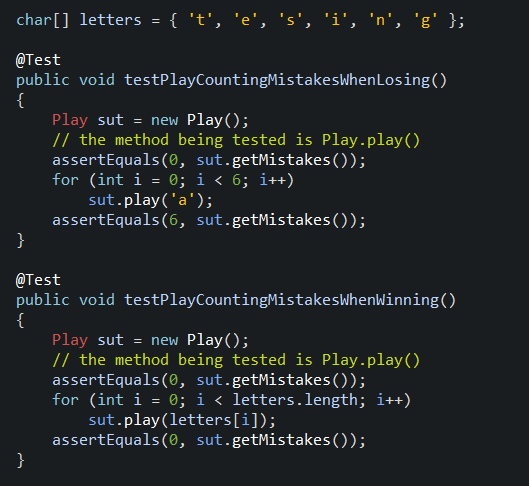
1. The system decides that too many mistakes have been made and the game is lost
2. The Game Over menu is shown with the message “You have lost” giving the option to play again or to return to the main menu

**Automated Tests**

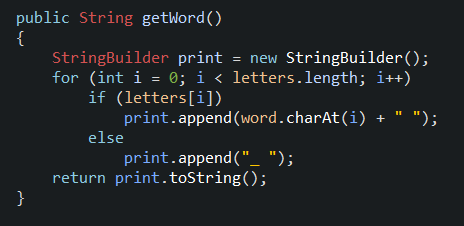
The first method being tested:

****

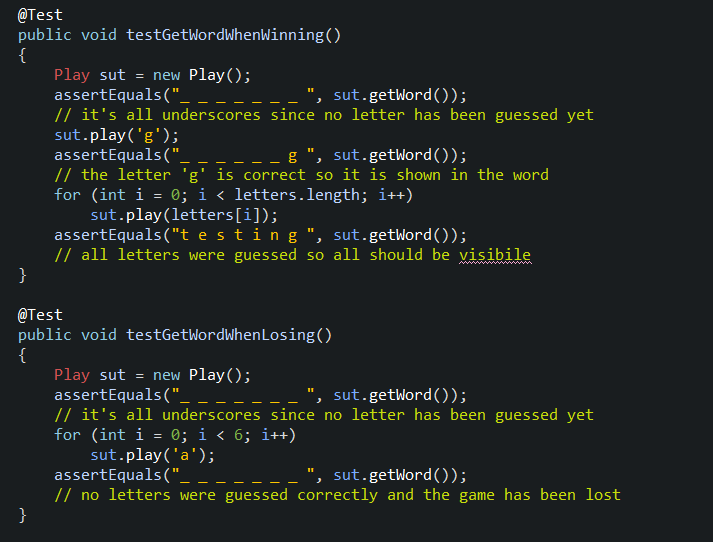
The test methods

****

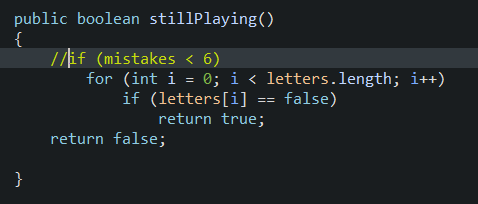
The second method being tested



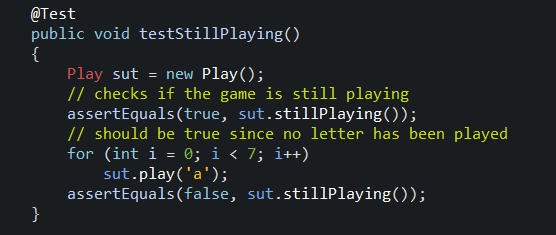
The test methods



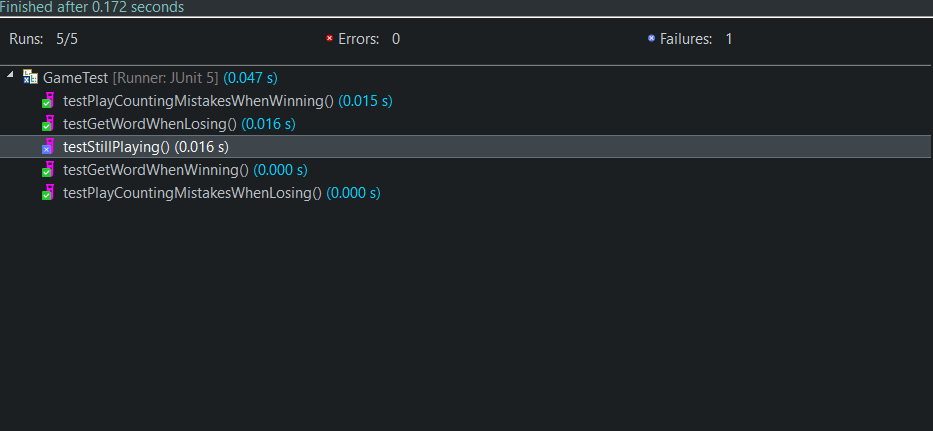
Third method being tested

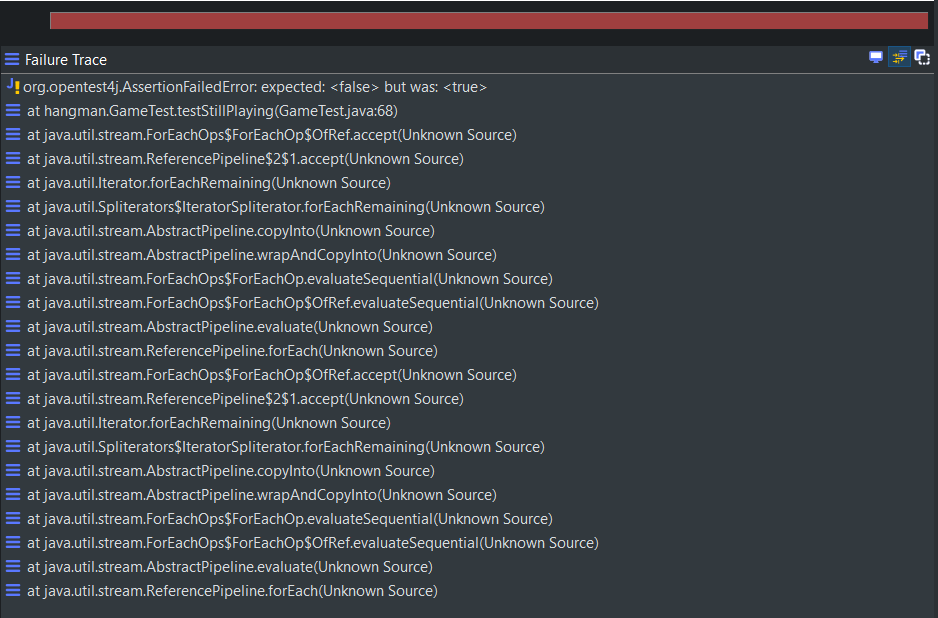


The test method



The test results





**Test Report**

Manual Tests

|  |  |  |
| --- | --- | --- |
| Test | UC1 | UC2 |
| TC1 | 1/OK | 0 |
| TC2.1 | 0 | 1/OK |
| TC2.2 | 0 | 1/OK |
| COVERAGE & SUCCESS | 1/OK | 2/OK |

**Reflection**

This assignment helped me familiarize myself more with software testing. It plays an important role in software development and helps with detecting and ultimately removing any sorts of bugs or problems that might appear in the program.

More importantly however, it made me realize that my way of writing code proved to be inefficient in this case. My program is written in such a way that makes writing automated unit tests. I will try to rethink my code and include as many methods and classes as I can.