İZMİR UNIVERSITY OF ECONOMICS

SE 318 – SOFTWARE VERIFICATION AND VALIDATION

HANGMAN-EYE

EMRE ÖZDEMİR

YİGİT CAN DÜNDAR

EMRE BAYGIN

UNIT TEST DOCUMENT

Version *1.0*

05/17/2016

VERSION HISTORY

Development of the unit test cases are done by NUnit and distribution of the test cases are done by GitHub, up to the final point of approval, was controlled and tracked by development team.

Version #	Implemented By	Revision Date	Approved By	Approval Date	Reason
1.0	*EMRE OZDEMİR*	*05/05/2016*	*EMRE BAYGIN*	*06/05/2016*	Initial Version
1.1	*EMRE OZDEMİR*	*13/05/2016*	*EMRE BAYGIN*	*14/05/2016*	Added a new help button
1.2	*EMRE OZDEMİR*	*14/05/2016*	*EMRE BAYGIN*	*15/05/2016*	Added a new reveal letter button
1.3	*EMRE OZDEMİR*	*14/05/2016*	*EMRE BAYGIN*	*15/05/2016*	Improved Hangman Image set

TABLE OF CONTENTS

1 INTRODUCTION

1.1 PURPOSE OF THE TEST CASE DOCUMENT

The purpose of the unit testing document is, to determine the program's current state of testing phases and verify the requirements. This document tailored to documentation of the unit test cases.

The Test Case document documents the functional requirements of the Unit Test Document test case. The intended audience is the project manager, project team, and testing team. Some portions of this document may on occasion be shared with the client/user and other stakeholder whose input/approval into the testing process is needed.

1.2 CONSTRAINTS

The constraints of the Test Cases performed are 20 test cases. Programming language is C# and unit test framework is NUnit. Integrated development environment (IDE) is Visual Studio.

2 UNIT TEST FRAMEWORK: *NUNIT*

NUnit is an open source unit-testing framework for all .net languages. Initially ported from Junit.

3 TEST CASES

Test Case 1	
Test Definition	
Given the selected words length, testing if the Reveal Letter Hint enables and disables correctly. This case will be tested with "word.Length < 5".	
Expected Value	
labeRevealLetter.Enabled = false	
Actual Value	
labeRevealLetter.Enabled = false	
Result of Test Case	***Successful***

Test Case 2	
Test Definition	
Given the selected words length, testing if the Reveal Letter Hint enables and disables correctly. This case will be tested with "word.Length == 5".	
Expected Value	
labeRevealLetter.Enabled = true	
Actual Value	
labeRevealLetter.Enabled = true	
Result of Test Case	***Successful***

Test Case 3	
Test Definition	
Given the selected words length, testing if the Reveal Letter Hint enables and disables correctly. This case will be tested with "word.Length > 5".	
Expected Value	
labeRevealLetter.Enabled = true	
Actual Value	
labeRevealLetter.Enabled = true	
Result of Test Case	***Successful***

Test Case 4	
Test Definition	
Given the selected words length, testing if the maxLength of the textBoxGuess sets correctly or not. This case will be tested with "word.Length ==3".	
Expected Value	
textBoxGuess.maxLength = 3	
Actual Value	
textBoxGuess.maxLength = 3	
Result of Test Case	***Successful***

Test Case 5	
Test Definition	
Testing if the guessed letter is added to the guessed list or not. Case tested with input "a".	
Expected Value	
richTextBoxGuessedLetters.Text.Contains("a") = true	
Actual Value	
richTextBoxGuessedLetters.Text.Contains("a") = true	
Result of Test Case	***Successful***

Test Case 6	
Test Definition	
Testing if the picture index sets correctly after a failed letter guess attempt. Tested with a failed guess attempt.	
Expected Value	
picIndex = 1	
Actual Value	
picIndex = 1	
Result of Test Case	***Successful***

Test Case 7	
Test Definition	
Testing if the picture index sets correctly after a successful letter guess attempt. Tested with a successful guess attempt.	
Expected Value	
picIndex = 0	
Actual Value	
picIndex = 0	
Result of Test Case	***Successful***

Test Case 8	
Test Definition	
Testing if the game lost event is triggered after 7 incorrect guess attempts.	
Expected Value	
textBoxGuess.BackColor = Color.red **textBoxGuess.ForeColor = Color.red**	
Actual Value	
textBoxGuess.BackColor = Color.red **textBoxGuess.ForeColor = Color.red**	
Result of Test Case	***Successful***

Test Case 9	
Test Definition	
Testing if the game won event is triggered after correct guess attempt(s).	
Expected Value	
textBoxGuess.BackColor = Color.GreenYellow **textBoxGuess.ForeColor = Color.GreenYellow**	
Actual Value	
textBoxGuess.BackColor = Color.GreenYellow **textBoxGuess.ForeColor = Color.GreenYellow**	
Result of Test Case	***Successful***

Test Case 10
Test Definition
Testing if the same incorrect letter guesses both increase the picIndex or not. First incorrect attempt should increase it. The second, same, attempt should not increase it. Tested with the same incorrect letters.
Expected Value
picIndex = 1
Actual Value
picIndex = 1
Result of Test Case ***Successful***

Test Case 11
Test Definition
Testing if the parts of the hidden word will count as correct guesses. Case tested with word substring from [0,2] and [2,word.Length-1].
Expected Value
textBoxGuess.BackColor = Color.GreenYellow
Actual Value
textBoxGuess.BackColor = Color.GreenYellow
Result of Test Case ***Successful***

Test Case 12
Test Definition
Testing if the parts of the hidden word will count as correct guesses. Case tested with word substring from [1,2],[0,1], [2,word.Length-1].
Expected Value
textBoxGuess.BackColor = Color.GreenYellow
Actual Value
textBoxGuess.BackColor = Color.GreenYellow
Result of Test Case ***Successful***

Test Case 13	
Test Definition	
Testing if the picIndex sets correctly after an incorrect partial word guess. Case tested with word=”abc” and guess word as “ed”.	
Expected Value	
picIndex = 1	
Actual Value	
picIndex = 1	
Result of Test Case	***Successful***

Test Case 14	
Test Definition	
Testing if the guessed word is added to the guessed list or not. Case tested with input “abc”.	
Expected Value	
isEnteredWord(“abc”) = true	
Actual Value	
isEnteredWord(“abc”) = true	
Result of Test Case	***Successful***

Test Case 15	
Test Definition	
Testing if the system will register a larger than maxLength word guess. Tested with word=”test”, maxLength=4 and guess=”abcde”.	
Expected Value	
picIndex = 0	
Actual Value	
picIndex = 0	
Result of Test Case	***Successful***

Test Case 16	
Test Definition	
Testing if the getRandomWord function will return a new word.	
Expected Value	
Word != initialWord	
Actual Value	
Word != initialWord	
Result of Test Case	***Successful***

Test Case 17	
Test Definition	
Testing if the picIndex sets correctly after a correct partial word guess. Case tested with word=”abc” and guess word as “ab”.	
Expected Value	
picIndex = 0	
Actual Value	
picIndex = 0	
Result of Test Case	***Successful***

Test Case 18	
Test Definition	
Testing if a whole word guess counts after partial guesses. Tested with word substring [1,3] and [0,word.Length-1].	
Expected Value	
textBoxGuess.BackColor = Color.GreenYellow	
Actual Value	
textBoxGuess.BackColor = Color.GreenYellow	
Result of Test Case	***Successful***

Test Case 19	
Test Definition	
Testing if the same incorrect word guesses both increase the picIndex or not. Tested with word = “test” and two sequential same word guesses “abc”.	
Expected Value	
picIndex = 1	
Actual Value	
picIndex = 1	
Result of Test Case	***Successful***

Test Case 20	
Test Definition	
Testing if non-letter inputs are registered correctly by the system or not. Tested with guess “2!3”.	
Expected Value	
isEnteredWord(“2!3”) = false	
Actual Value	
isEnteredWord(“2!3”) = false	
Result of Test Case	***Successful***

4 CONCLUSION

The constraints of the unit-testing (20 unit test) of HANGMAN-EYE have been achieved without failing a single test case. Unit testing phase of the HANGMAN-EYE is finished. Component testing phase is required for the following testing step.