**Software Testing and Inspection Project**

**David Kearney – 15169235**

|  |
| --- |
| **Task 2** |
| **Task Title**  Test-driven Development using JUnit |
| 1. **The log**   Fig 1.1, 1.2 and 1.3 were completed on 15/04/2017  Fig 1.4 was completed on 16/04/2017  Fig 1.5, 1.6 and 1.7 were completed on 18/04/2017 |
| 1. **Reflections**   Initially I expected JUnit testing to be very pedantic and unnecessary work. I presumed it would add unnecessary work to creating software. However after doing some simple tests I learned how useful it will be to me in the future.  I sometimes run into issues with large arrays and I can definitely see myself using the assertArrayEquals functions a lot in my programming in the future, I found it useful that it tells you if the lengths are not the same as this could instantly show you that the arrays are not being populated correctly.  Using JUnit has shown me that testing is not going to add unnecessary workload but actually save time and stress, while also resulting in better software.  Something I really enjoyed about using Junit was that it came included with my eclipse IDE and was very simple to use after looking at a tutorial.  Overall, I’ve discovered that looking manually at output manually for errors is simply not a sustainable method of testing for larger programs and unit testing is a great alternative. |
| **C. Submitted Material**    Fig 1.0 JUnitTestClass    Fig 1.1 Test class and test failure as 301 is not the expected result    Fig 1.2 Test Success    Fig 1.3 Added Thread.sleep(100) to make the method run slower    Fig 1.4 Added @Test(timeout=100) which makes the test fail if the program takes longer than 100ms to execute. As you can see the test failed despite achieving the desired result (300).    Fig 1.5 Here we see multiple assertions being made in the one test. The “Test” class will print any of the failures as a string in the “Failure Trace” window in the bottom right of eclipse. This test will fail if any of the assertions are false.    Fig 1.6 Now we see that the test fails as the assertions was looking for the String str to contain “abc” but instead it contains “testing” (this can be seen in the next image more clearly). The whole test case fails despite the other two assertions being true    Fig 1.7 I have added a testArray method here to learn about testing arrays. Both arrays are the same so this method didn’t return any failures. Had there been a mismatch in length the actual and expected length would be shown in the failure trace. If the lengths were the same but the values were different the failure trace would show us at what element they first differed, what was expected at that element and what was actually in that element. |

|  |
| --- |
| **Task 3** |
| **Task Title**  **Test Automation using Selenium Tools** |
| **A. The log**  All work in this task was undertaken on 13/04/2017  Firstly I downloaded Firefox and installed Selenium IDE addon.  I recorded a simple test of ‘youtube.com’ by visiting the page and observing how Selenium IDE tracked my inputs and clicks.  Then I downloaded Eclipse IDE and added the Selenium JAR to my library.  I compared the HTML file generated by Selenium IDE with the java code I saved from my test.  I edited the Java code to output some information about the site (although it unfortunately didn’t run) |
| **B. Reflections**  I found that the Selenium IDE has a very good user interface. It was easy to use and didn’t take long to get accustomed to. This was an advantage over using eclipse which can be quite difficult to manage. I found the “Reference” section of Selenium IDE to be very helpful as it explained all of the commands thoroughly. The main limitation I found with Selenium IDE is that it can only be used with Firefox but overall I found Selenium IDE a pleasure to use.    In contrast, with Selenium Webdriver I ran into a lot of issues. I had some issues adding the correct JAR files to the library in eclipse. After this I had issues running the code with a ClassNotFoundExcepion and ultimately I was never able to run the code. |
| **C. Submitted Material**  Original Java Code:  package com.example.tests;  import java.util.regex.Pattern;  import java.util.concurrent.TimeUnit;  import org.junit.\*;  import static org.junit.Assert.\*;  import static org.hamcrest.CoreMatchers.\*;  import org.openqa.selenium.\*;  import org.openqa.selenium.firefox.FirefoxDriver;  import org.openqa.selenium.support.ui.Select;  public class Selenium {  private WebDriver driver;  private String baseUrl;  private boolean acceptNextAlert = true;  private StringBuffer verificationErrors = new StringBuffer();  @Before  public void setUp() throws Exception {  driver = new FirefoxDriver();  baseUrl = "https://www.youtube.com/";  driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);  }  @Test  public void testSelenium() throws Exception {  driver.get(baseUrl + "/");  driver.findElement(By.xpath("//li[@id='UC-9-kyTW8ZkZNDHQJ6FgpwQ-guide-item']/a/span/span[2]")).click();  driver.findElement(By.xpath("(//button[@type='button'])[2]")).click();  driver.findElement(By.id("Passwd-hidden")).clear();  driver.findElement(By.id("Passwd-hidden")).sendKeys("gmld4v1d");  driver.findElement(By.xpath("//ul[@id='browse-items-primary']/li[2]/div/div/div/div/div/h2/a[2]/span")).click();  }  @After  public void tearDown() throws Exception {  driver.quit();  String verificationErrorString = verificationErrors.toString();  if (!"".equals(verificationErrorString)) {  fail(verificationErrorString);  }  }  private boolean isElementPresent(By by) {  try {  driver.findElement(by);  return true;  } catch (NoSuchElementException e) {  return false;  }  }  private boolean isAlertPresent() {  try {  driver.switchTo().alert();  return true;  } catch (NoAlertPresentException e) {  return false;  }  }  private String closeAlertAndGetItsText() {  try {  Alert alert = driver.switchTo().alert();  String alertText = alert.getText();  if (acceptNextAlert) {  alert.accept();  } else {  alert.dismiss();  }  return alertText;  } finally {  acceptNextAlert = true;  }  }  }  Modified Java Code:  import java.util.regex.Pattern;  import java.util.concurrent.TimeUnit;  import org.junit.\*;  import static org.junit.Assert.\*;  import static org.hamcrest.CoreMatchers.\*;  import org.openqa.selenium.\*;  import org.openqa.selenium.firefox.FirefoxDriver;  import org.openqa.selenium.support.ui.Select;  public class Selenium {  private WebDriver driver;  private String baseUrl;  private boolean acceptNextAlert = true;  private StringBuffer verificationErrors = new StringBuffer();  @Before  public void setUp() throws Exception {  driver = new FirefoxDriver();  baseUrl = "https://www.youtube.com/";  driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);  }  @Test  public void testSelenium() throws Exception {      driver.get(baseUrl + "/");  // Prints expected URL and opened URL  System.out.println("Expected URL: " + baseUrl);  String URL = driver.getCurrentUrl();  System.out.println("URL Opened: " + URL);  driver.findElement(By.xpath("//li[@id='UC-9-kyTW8ZkZNDHQJ6FgpwQ-guide-item']/a/span/span[2]")).click();  driver.findElement(By.xpath("(//button[@type='button'])[2]")).click();  driver.findElement(By.id("Passwd-hidden")).clear();  driver.findElement(By.id("Passwd-hidden")).sendKeys("gmld4v1d");  driver.findElement(By.xpath("//ul[@id='browse-items-primary']/li[2]/div/div/div/div/div/h2/a[2]/span")).click();  // Print the title and title length  driver.get(baseUrl);  String actualTitle = driver.getTitle();  System.out.println("The title of the page is " + actualTitle);  System.out.println("The length of the title is " + actualTitle.length());    // Look for the youtube logo container and if it is present then the page has been opened correctly  if (!(driver.findElements(By.id("logo-container")).isEmpty()))  System.out.println("Correct Page Opened");  else System.out.println("Incorrect Page Opened");      }  @After  public void tearDown() throws Exception {  driver.quit();  String verificationErrorString = verificationErrors.toString();  if (!"".equals(verificationErrorString)) {  fail(verificationErrorString);  }  }  private boolean isElementPresent(By by) {  try {  driver.findElement(by);  return true;  } catch (NoSuchElementException e) {  return false;  }  }  private boolean isAlertPresent() {  try {  driver.switchTo().alert();  return true;  } catch (NoAlertPresentException e) {  return false;  }  }  private String closeAlertAndGetItsText() {  try {  Alert alert = driver.switchTo().alert();  String alertText = alert.getText();  if (acceptNextAlert) {  alert.accept();  } else {  alert.dismiss();  }  return alertText;  } finally {  acceptNextAlert = true;  }  }  } |