Assignment HW6

**Cover Page**

Prepared for:

Dr. Mehra Borazjany

Trung Hieu Tran

Prepared by:

Alex Lundin

Daniel Neal

SE-4367.0U1-Testing

**Assignment Choice:**

N/A for this assignment

27 June, 2018

**Proof of Working Software**

GitHub link:

<https://github.com/AlexLundinEducational/SE-4367-Testing>

Branch Summary:

master – managed by Alex, only fully completed pulls allowed to make TA’s life easy. Master only contains assignment material once they reach completed status.

working – flexible branch for team, ideally, this material should build without causing technical debt during the project.

Commit for grading:

HW6\_Team\_4 Complete, Ready for Merge to master and Ready for Grading

Phase 1 Development

06-27-2018

Alex Lundin

Source folder and documents setup

06-28-2018

Alex Lundin

Double check files build correctly

https://autode.sk/2lFgjDx

Daniel Neal

Pulled github, tested files

Both

Brainstorm Base Choice

Reviewed slide material

Phase 2 Development

06-30-2018

Alex Lundin

Created IDM for Base choice

Documented all steps used to create model

Cited slide references

Documented reason for the Base Choice that will be used in the test setup

07-02-2018

Daniel

Created test set for Fact

07-03-2018

Alex

Created test set for Search Mode

Phase 3 Development

07-03-2018

Alex created simple test template

07-04-2018

Dan created test set

07-05-2018

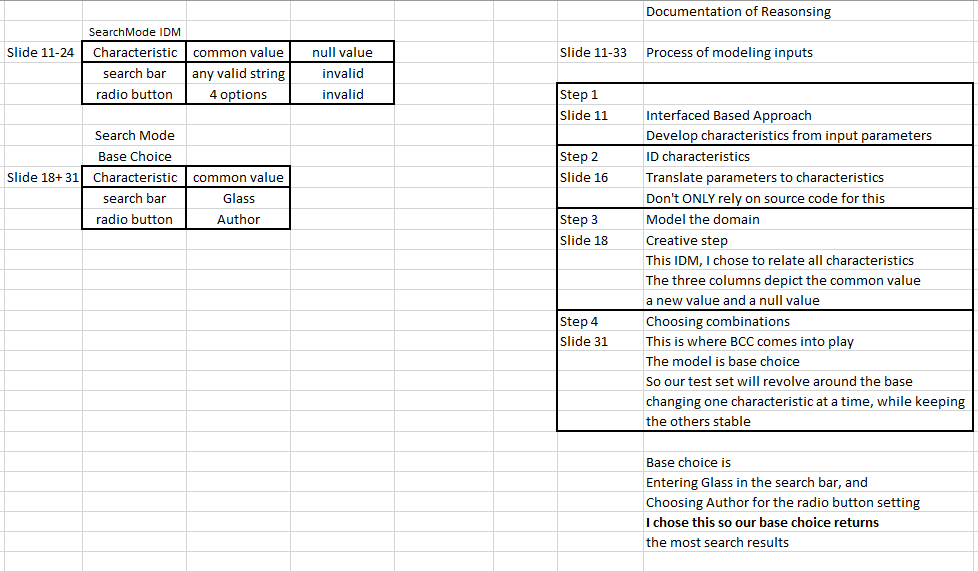
Alex finished set with assert fixture and xml research for finding the text in the page

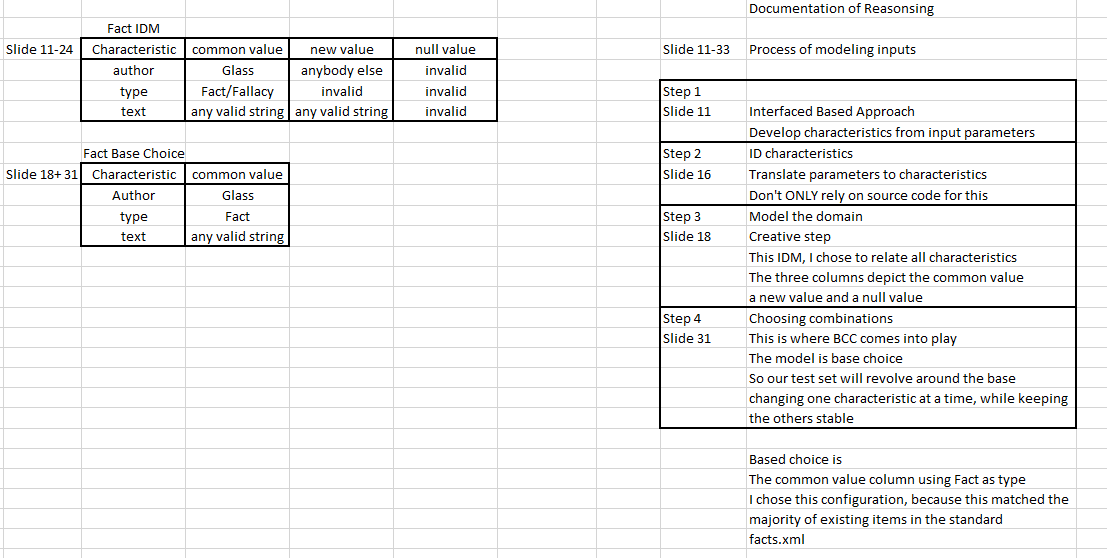
Proof of Functional Tests

https://autode.sk/2zbXhOT

1. input domain model

**SearchMode IDM**



**Fact IDM**

2. Your test designs, including the values and expected results, at the design abstraction level

D = search bar (5 total)

E = Radio button (4 total)

|  |  |
| --- | --- |
| **D1** | automation |
| **D2** | Glass |
| **D3** | Fact |
| **D4** | Fallacy |
| **D5** | NULL |

|  |  |
| --- | --- |
| **E1** | Text |
| **E2** | Author |
| **E3** | Type |
| **E4** | All |

BBC Total Tests (Bi -1 )

= 1 + (5-1) + (4-1)

= 8 Tests

**BASE D1, E1** D1, E1 D2, E1

D1, E2 D3, E1

D1, E3 D4, E1

D1, E4 D5, E1

Tests, parameters and results

Test Parameters Expected

SearchD1E1 D1, E1 "Test automation rarely is. Most testing activities cannot be automated.";

SearchD1E2 D1, E2 "Not Found!";

SearchD1E3 D1, E3 "Not Found!";

SearchD1E4 D1, E4 "Test automation rarely is. Most testing activities cannot be automated.";

SearchD2E1 D2, E1 "Not Found!";

SearchD3E1 D3, E1 "Quality is NOT: user satisfaction, meeting requirements, achieving cost/schedule, or reliability.";

SearchD4E1 D4, E1 "Not Found!";

SearchD5E1 D5, E1 null

3. Your automated tests

**This is a snippet of our code, we are not including every test in the report, just the main pieces.**

// SearchTest method

// This class contains the GUI test set for our second IDM

//Daniel Neal & Alex Lundin

//07-05-2018

//SE-4367-Testing

public class SearchTest implements StringConstants

{

WebDriver driver;

private static String expectedValue = null;

@Before

public void openFactsPage() //throws InterruptedException

{

System.setProperty("webdriver.chrome.driver", chromeExecutable);

driver = new ChromeDriver();

driver.get(thisServlet);

Assert.assertEquals("Software Engineering", driver.getTitle());

}

//this method automates the base case (D1,E1)

//searches "automation" with the "text" option selected

@Test

public void searchBase() throws InterruptedException

{

WebElement searchBox = driver.findElement(By.id("searchText"));

WebElement type = driver.findElement(By.id("text"));

type.click();

Thread.sleep(1000);

searchBox.sendKeys("automation");

Thread.sleep(1000);

searchBox.submit();

Thread.sleep(1000);

expectedValue = "Test automation rarely is. Most testing activities cannot be automated.";

}

@After

public void closePage() throws InterruptedException

{

assertFixture();

Thread.sleep(3000);

driver.close();

driver.quit();

}

// this method will pass or fail each test depending on if search String matching expected value

public void assertFixture() throws InterruptedException{

WebElement paragraph;

WebElement dt ;

// check for null first to avoid errors

if (expectedValue == null){

paragraph = findNotFoundInParagraphs();

Assert.assertNull(paragraph);

}else if (expectedValue.equals("Not Found!")){

paragraph = findNotFoundInParagraphs();

System.out.println(paragraph.getText());

Assert.assertEquals(paragraph.getText(), expectedValue);

}else if(expectedValue != null){

dt = findTextInDt();

System.out.println(dt.getText());

Assert.assertEquals(dt.getText(), expectedValue);

}

}

private static List<WebElement> elements = null;

// this method looks for the "Not Found!" string inside the body item

// there are 3 known to appear, the middle p is the one that contains "Not Found!"

// this method accesses it with get(1)

public WebElement findNotFoundInParagraphs(){

// top level element to look through

// in this case it's the body

elements = driver.findElements(By.cssSelector("body"));

for (WebElement element : elements) {

// use the p selector, because the "Not Found!" text is printed into a paragraph

List<WebElement> paragraphs = element.findElements(By.cssSelector("p"));

WebElement paragraph = paragraphs.get(1);

String myText = paragraph.getText();

if (myText.contains("Not Found!")) {

return paragraph;

}

}

return null;

}

// this method uses absolute path to find the first element of text in the Dt item

public WebElement findTextInDt(){

//html//body//table//tbody//tr//td//dl//dt

WebElement dt = driver.findElement(By.xpath("//html//body//table//tbody//tr//td[1]//dl//dt[1]"));

return dt;

}

4. A screen shot **and** printout of the results of running your tests

<https://autode.sk/2zbXhOT>

