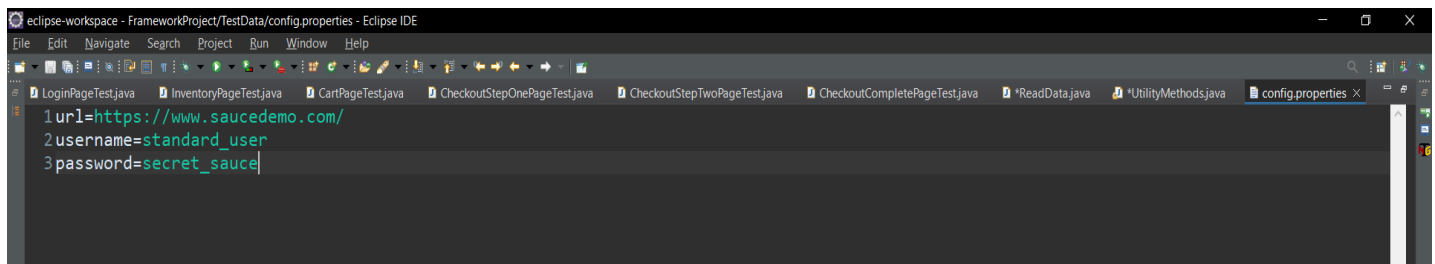


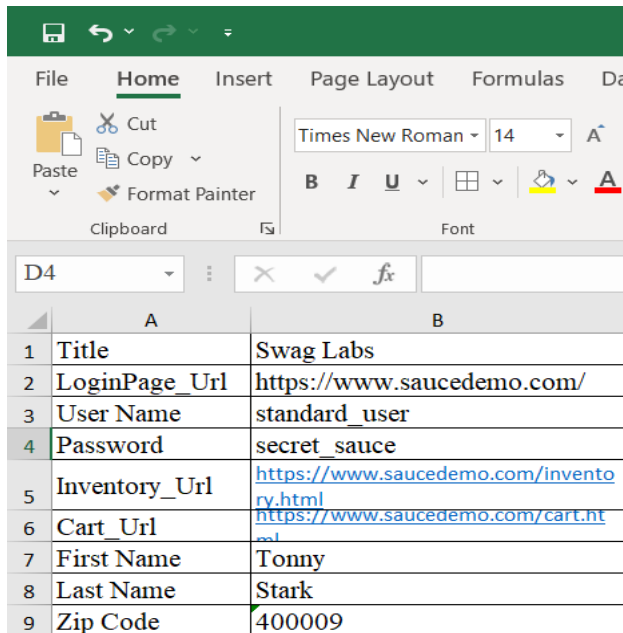
Situ

Property File:



```
1 url=https://www.saucedemo.com/
2 username=standard_user
3 password=secret_sauce
```

Excel File:

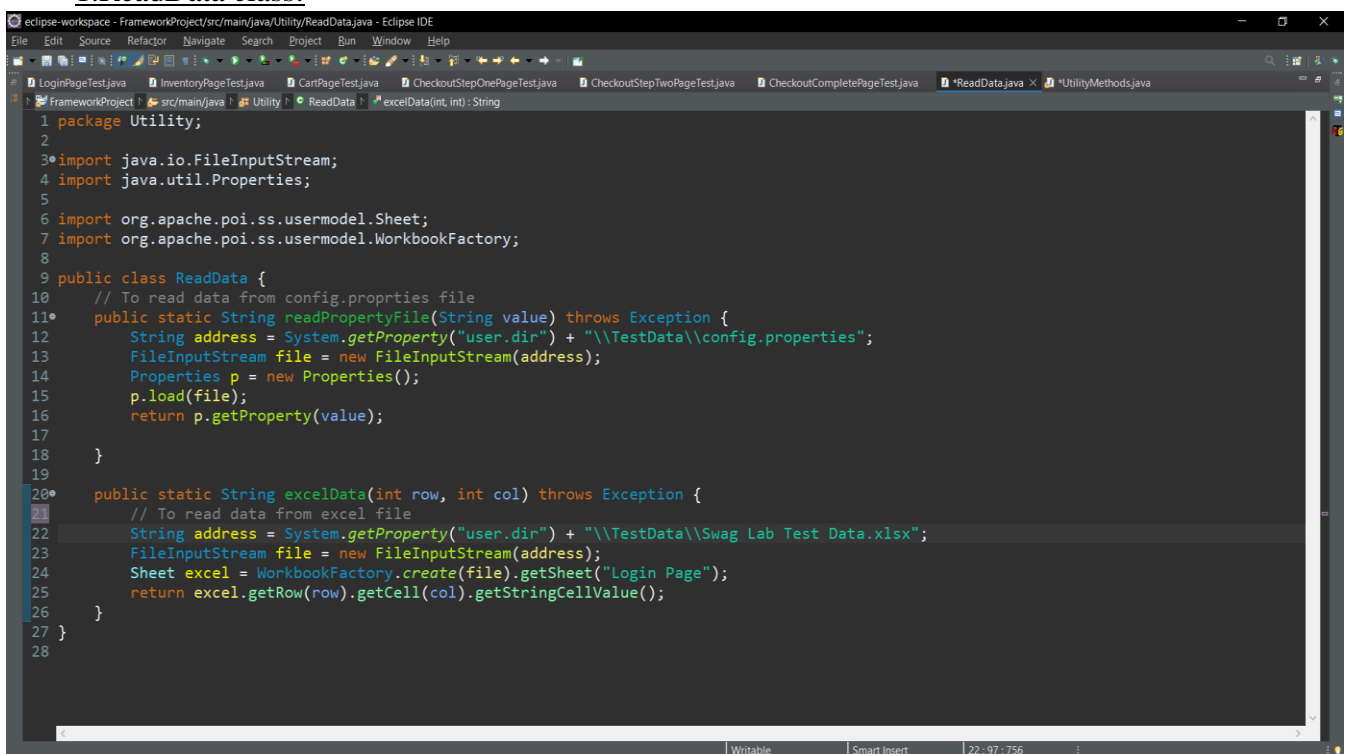


	A	B
1	Title	Swag Labs
2	LoginPage_Url	https://www.saucedemo.com/
3	User Name	standard_user
4	Password	secret_sauce
5	Inventory_Url	https://www.saucedemo.com/inventory.html
6	Cart_Url	https://www.saucedemo.com/cart.html
7	First Name	Tonny
8	Last Name	Stark
9	Zip Code	400009

10		
11	Checkout Title	CHECKOUT: YOUR INFORMATION
12		
13	Error Masssage of checkout one page	Error: First Name is required
14		
15	check out page two url	https://www.saucedemo.com/checkout-step-two.html
16		
17	checkout overview Title	CHECKOUT: OVERVIEW
18	check out complete title	CHECKOUT: COMPLETE!
19	Thank You Text	THANK YOU FOR YOUR ORDER
20	checkout page complete URL	https://www.saucedemo.com/checkout-complete.html

Utility Package:

1.ReadData class:



```
1 package Utility;
2
3 import java.io.FileInputStream;
4 import java.util.Properties;
5
6 import org.apache.poi.ss.usermodel.Sheet;
7 import org.apache.poi.ss.usermodel.WorkbookFactory;
8
9 public class ReadData {
10     // To read data from config.proprties file
11     public static String readPropertyFile(String value) throws Exception {
12         String address = System.getProperty("user.dir") + "\\TestData\\config.properties";
13         FileInputStream file = new FileInputStream(address);
14         Properties p = new Properties();
15         p.load(file);
16         return p.getProperty(value);
17     }
18
19
20     public static String excelData(int row, int col) throws Exception {
21         // To read data from excel file
22         String address = System.getProperty("user.dir") + "\\TestData\\Swag Lab Test Data.xlsx";
23         FileInputStream file = new FileInputStream(address);
24         Sheet excel = WorkbookFactory.create(file).getSheet("Login Page");
25         return excel.getRow(row).getCell(col).getStringCellValue();
26     }
27 }
28
```

Testers don't make software; they make it better.

2.UtilityMethods class:

```
eclipse-workspace - FrameworkProject/src/main/java/Utility/UtilityMethods.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
FrameworkProject src/main/java Utility UtilityMethods
1 package Utility;
2
3 import java.time.Duration;
4 import java.util.ArrayList;
5 import java.util.List;
6
7 import org.openqa.selenium.WebDriver;
8 import org.openqa.selenium.WebElement;
9 import org.openqa.selenium.support.ui.ExpectedConditions;
10 import org.openqa.selenium.support.ui.Select;
11 import org.openqa.selenium.support.ui.WebDriverWait;
12
13 public class UtilityMethods {
14
15     public static void selectClass(WebElement dropDown, String visibleValue) {
16         // To handle Dropdown with select tagname
17         Select dropdown = new Select(dropDown);
18         dropdown.selectByValue(visibleValue);
19     }
20
21     public static List<Double> sortingInteger(List<WebElement> prices) {
22         // To sorting all Products price into Ascending order
23
24         List<WebElement> listOfAllPrices = prices; // To store all the price WebElements
25         List<Double> listOfDoubleTypePrices = new ArrayList<Double>();
26         for (WebElement ele : listOfAllPrices) {
27             // convert WebElements into double type values and remove $ sign
28             listOfDoubleTypePrices.add(Double.valueOf(ele.getText().replace("$", "")));
29         }
30         return listOfDoubleTypePrices;
31     }
32
33     public static List<String> sortingString(List<WebElement> allTexts) {
34         // To sorting all Products Name into Alphabetical order
35
36         List<WebElement> listOfAllStringTexts = allTexts;
37         List<String> list = new ArrayList<String>();
38         for (WebElement ele : listOfAllStringTexts) {
39             list.add(ele.getText());
40         }
41         return list;
42     }
43
44     public static void expliciteWait(WebDriver driver, WebElement ele) {
45         // To provide time upto 10 sec to element to visible
46         WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
47         wait.until(ExpectedConditions.visibilityOf(ele));
48     }
49 }
50
```

```
eclipse-workspace - FrameworkProject/src/main/java/Base/TestBase.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
TestBase.java
FrameworkProject src/main/java Base TestBase Initialization0: void
1 package Base;
2
3 import java.time.Duration;
4
5 import org.openqa.selenium.WebDriver;
6 import org.openqa.selenium.chrome.ChromeDriver;
7
8 import Pages.CartPage;
9 import Pages.CheckoutStepOnePage;
10 import Pages.CheckoutStepTwoPage;
11 import Pages.InventoryPage;
12 import Pages.LoginPage;
13 import Utility.ReadData;
14 import io.github.bonigarcia.wdm.WebDriverManager;
15
16 public class TestBase {
17     static public WebDriver driver;
18
19     public void initialization() throws Exception {
20         // TO open browser and enter URL
21         WebDriverManager.chromedriver().setup();
22         driver = new ChromeDriver();
23         driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(3));
24         driver.manage().window().maximize();
25         String url = ReadData.readPropertyFile("url");
26         driver.get(url);
27     }
28 }

```

TestBase Class

Testers don't make software; they make it better.

Situ

```
28
29• public void login() throws Exception {
30    // To Login to application
31    LoginPage loginPage = new LoginPage();
32    loginPage.getUserNameTextbox().sendKeys(ReadData.readPropertyFile("username"));
33    loginPage.getPasswordTextbox().sendKeys(ReadData.readPropertyFile("password"));
34    loginPage.getLoginBtn().click();
35 }
36
37• public void inventory() throws Exception {
38    // To navigate upto inventory page
39    login();
40 }
41
42• public void cart() throws Exception {
43    // To navigate upto cart page
44    login();
45    InventoryPage inventoryPage = new InventoryPage();
46    inventoryPage.getCartLink().click();
47 }
48
49• public void checkoutPageOne() throws Exception {
50    // To navigate upto checkOut page one
51    cart();
52    CartPage cartPage = new CartPage();
53    cartPage.getCheckoutBtn().click();
54 }
55
56• public void checkoutPageTwo() throws Exception {
57    // To navigate upto checkout page two
58    checkoutPageOne();
59    CheckoutStepOnePage checkoutStepOnePage = new CheckoutStepOnePage();
60    checkoutStepOnePage.getFirstNameTextbox().sendKeys(ReadData.excelData(6, 1));
61    checkoutStepOnePage.getLastNameTextbox().sendKeys(ReadData.excelData(7, 1));
62    checkoutStepOnePage.getPostalCodeTextbox().sendKeys(ReadData.excelData(8, 1));
63    checkoutStepOnePage.getContinueBtn().click();
64 }
65
66• public void checkoutComplete() throws Exception {
67    // To navigate upto checkout page complete
68    checkoutPageTwo();
69    CheckoutStepTwoPage checkoutStepTwoPage = new CheckoutStepTwoPage();
70    checkoutStepTwoPage.getFinishBtn();
71 }
72
73 }
74
```

LoginPage Class:

```
eclipse-workspace - FrameworkProject/src/main/java/Pages/LoginPage.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
src/main/java/Pages/LoginPage.java
1 package Pages;
2
3 import org.openqa.selenium.WebElement;
4 import org.openqa.selenium.support.FindBy;
5 import org.openqa.selenium.support.PageFactory;
6
7 import Base.TestBase;
8
9 public class LoginPage extends TestBase {
10
11     @FindBy(xpath = "//div[@class='login_logo']")
12     private WebElement loginLogo;
13     @FindBy(xpath = "//div[@class='bot_column']")
14     private WebElement botLogo;
15     @FindBy(id = "user-name")
16     private WebElement userNameTextbox;
17     @FindBy(id = "password")
18     private WebElement passwordTextbox;
19     @FindBy(name = "login-button")
20     private WebElement loginBtn;
21     @FindBy(id = "logout_sidebar_link")
22     private WebElement logoutBtn;
23
24     public LoginPage() {
25         // Constructor to initialize driver with instance variable
26         PageFactory.initElements(driver, this);
27     }
28 }
```

Testers don't make software; they make it better.

Situ

```
28
29* public String verifyTitle() {
30     // To get Title of WebPage
31     return driver.getTitle();
32 }
33
34* public WebElement getUserNameTextbox() {
35     // get method to return userName TextBox
36     return userNameTextbox;
37 }
38
39* public WebElement getPasswordTextbox() {
40     // get method to return password TextBox
41     return passwordTextbox;
42 }
43
44* public WebElement getLoginBtn() {
45     // get method to return login Button
46     return loginBtn;
47 }
48
49* public String verifyUrl() {
50     // To get current URL
51     return driver.getCurrentUrl();
52 }
53
54* public boolean verifyLoginLogo() {
55     // To verify logo is present
56     return loginLogo.isDisplayed();
57 }
58
59* public boolean verifyBotLogo() {
60     // To verify logo is present
61     return botLogo.isDisplayed();
62 }
63
64* public WebElement getLogoutBtn() {
65     // get method to return logout Button
66     return logoutBtn;
67 }
68 }
69
```

LoginPageTest Class:

```
eclipse-workspace - FrameworkProject/src/test/java/TestCases/LoginPageTest.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
FrameworkProject ▸ src/test/java ▸ TestCases ▸ LoginPageTest
1 package TestCases;
2
3*import org.testng.Assert;
4 import org.testng.Reporter;
5 import org.testng.annotations.AfterMethod;
6 import org.testng.annotations.BeforeMethod;
7 import org.testng.annotations.Test;
8
9 import Base.TestBase;
10 import Pages.LoginPage;
11 import Utility.ReadData;
12
13 Run All
14 public class LoginPageTest extends TestBase {
15     LoginPage loginPage;
16
17     @BeforeMethod
18     public void setup() throws Exception {
19         // To call initialization method and assign instance variable
20         initialization();
21         loginPage = new LoginPage();
22     }
23 }
```

Testers don't make software; they make it better.

Situ

```
22
23* @Test
    Run | Debug
24 public void verifyTitleTest() throws Exception {
25     // To verify Title of Login page
26     String actual = loginPage.verifyTitle();
27     String expected = ReadData.excelData(0, 1);
28     Assert.assertEquals(expected, actual);
29     Reporter.Log("Title of Login Page : " + actual);
30 }
31
32* @Test
    Run | Debug
33 public void verifyUrlTest() throws Exception {
34     // To verify URL of Login page
35     String actual = loginPage.verifyUrl();
36     String expected = ReadData.excelData(1, 1);
37     Assert.assertEquals(expected, actual);
38     Reporter.Log("Url of Login Page : " + actual);
39 }
40
41* @Test
    Run | Debug
42 public void verifyloginLogoTest() {
43     // To verify logo
44     boolean actual = loginPage.verifyLoginLogo();
45     Assert.assertTrue(actual);
46 }
47
48* @Test
    Run | Debug
49 public void verifyBotLogoTest() {
50     // To verify logo
51     boolean actual = loginPage.verifyBotLogo();
52     Assert.assertTrue(actual);
53 }
54
55* @Test(priority = 1)
    Run | Debug
56 public void verifyLoginTest() throws Exception {
57     // To verify Login to application
58     login();
59     boolean actual = loginPage.getLogoutBtn().isEnabled();
60     Assert.assertTrue(actual, "Login is not Successful");
61 }
62
63* @Test
    Run | Debug
64 public void verifyEnteredUsername() throws Exception {
65     // To verify UserName should be Entered in userName textbox
66     loginPage.getUserNameTextbox().sendKeys(ReadData.readPropertyFile("username"));
67     String actual = loginPage.getUserNameTextbox().getAttribute("value");
68     String expected = ReadData.readPropertyFile("username");
69     Assert.assertEquals(actual, expected);
70     Reporter.Log("Entered Username : " + actual);
71 }
72
73* @Test
    Run | Debug
74 public void verifyEnteredPassword() throws Exception {
75     // To verify password should be entered in password textbox
76     loginPage.getPasswordTextbox().sendKeys(ReadData.readPropertyFile("password"));
77     String actual = loginPage.getPasswordTextbox().getAttribute("value");
78     String expected = ReadData.readPropertyFile("password");
79     Assert.assertEquals(actual, expected);
80     Reporter.Log("Entered Password : " + actual);
81 }
82
83* @AfterMethod
84 public void browserClose() {
85     driver.close();
86 }
87
88 }
89
```

Testers don't make software; they make it better.

LoginPageTest Class Report:

Test results
1 suite

All suites

Default suite

Info

- C:\Users\HP\AppData\Local\Temp\testing-eclipse--77888813\testing-customsuite.xml
- 1 test
- 0 groups
- Times
- Reporter output
- Ignored methods
- Chronological view

Results

- 7 methods, 7 passed
- Passed methods (hide)
- verifyBotLogoTest
- verifyEnteredPassword
- verifyEnteredUsername
- verifyLoginTest
- verifyTitleTest
- verifyUrlTest
- verifyLoginLogoTest

Reporter output for Default suite

verifyEnteredPassword
Entered Password : secret_sauce

verifyTitleTest
Title of Login Page : Swag Labs

verifyEnteredUsername
Entered Username : standard_user

verifyUrlTest
Url of Login Page : https://www.saucedemo.com/

InventoryPage Class:

```

eclipse-workspace - FrameworkProject/src/main/java/Pages/InventoryPage.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
TestBase.java LoginPage.java LoginPageTest.java InventoryPage.java
FrameworkProject/src/main/java/Pages/InventoryPage.java
1 package Pages;
2
3 import java.util.Collections;
4 import java.util.List;
5
6 import org.openqa.selenium.WebElement;
7 import org.openqa.selenium.support.FindBy;
8 import org.openqa.selenium.support.PageFactory;
9
10 import Base.TestBase;
11 import Utility.UtilityMethods;
12
13 public class InventoryPage extends TestBase {
14     @FindBy(xpath = "//div[@class='peek']")
15     private WebElement peekLogo;
16     @FindBy(xpath = "//span[@class='title']")
17     private WebElement productText;
18     @FindBy(xpath = "//a[@class='shopping_cart_link']")
19     private WebElement cartLink;
20     @FindBy(xpath = "//div[contains(text(),'Backpack')]")
21     private WebElement backpack;
22     @FindBy(id = "add-to-cart-sauce-labs-backpack")
23     private WebElement addtocartBag;
24     @FindBy(className = "inventory_item_price")
25     private List<WebElement> prices;
26     @FindBy(className = "inventory_item_name")
27     private List<WebElement> names;
28     @FindBy(className = "product_sort_container")
29     private WebElement dropDown;
30
31     public InventoryPage() {
32         // Constructor to initialize driver with instance variable
33         PageFactory.initElements(driver, this);
34     }
35
36     public boolean verifyPeekLogo() {
37         // To verify logo is present
38         return peekLogo.isDisplayed();
39     }
40
41     public String verifyProductText() {
42         // To get Text of Product element
43         return productText.getText();
44     }

```

Situ

```
45
46 public WebElement getBagpack() {
47     // get method to return bagPack WebElement
48     return backpack;
49 }
50
51 public WebElement getAddtocartBag() {
52     // get method to return addToCartButton WebElement
53     return addtocartBag;
54 }
55
56 public WebElement getCartLink() {
57     // get method to return cartLink WebElement
58     return cartLink;
59 }
60
61 public List<Double> ascendingOrderPriceList() {
62     // To sort all products price into ascending order
63     List<Double> list = UtilityMethods.sortingInteger(prices);
64     Collections.sort(list);
65     return list;
66 }
67
68 public List<Double> actualPriceListAccordingToDropdownOptionSelected(String visibleValue) {
69     // To get all products prices from WebPage
70     UtilityMethods.selectClass(dropDown, visibleValue);
71     List<Double> list = UtilityMethods.sortingInteger(prices);
72     return list;
73 }
74
75 public List<String> alphabeticalOrderProductsNames() {
76     // To sort all products names into alphabetical order
77     List<String> list = UtilityMethods.sortingString(names);
78     Collections.sort(list);
79     return list;
80 }
81
82 public List<String> actualNamesAccordingToDropdownOptionSelected(String visibleValue) {
83     // To get all products names from WebPage
84     UtilityMethods.selectClass(dropDown, visibleValue);
85     List<String> list = UtilityMethods.sortingString(names);
86     return list;
87 }
88
89 }
90
```

InventoryPageTest Class:

```
eclipse-workspace - FrameworkProject/src/test/java/TestCases/InventoryPageTest.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
TestBase.java LoginPageTest.java InventoryPageTest.java
FrameworkProject > src/test/java > TestCases > InventoryPageTest
1 package TestCases;
2
3 import java.util.Collections;
4 import java.util.List;
5
6 import org.testng.Assert;
7 import org.testng.Reporter;
8 import org.testng.annotations.AfterMethod;
9 import org.testng.annotations.BeforeMethod;
10 import org.testng.annotations.Test;
11
12 import Base.TestBase;
13 import Pages.InventoryPage;
14
15 Run All
16 public class InventoryPageTest extends TestBase {
17     InventoryPage inventoryPage;
18
19     @BeforeMethod
20     public void setup() throws Exception {
21         // To call initialization method and assign instance variable
22         initialization();
23         inventory();
24         inventoryPage = new InventoryPage();
25     }
26 }
```

Testers don't make software; they make it better.

```

25
26* @Test
    Run | Debug
27 public void verifyPeekLogoTest() {
28     // To verify logo is present
29     boolean actual = inventoryPage.verifyPeekLogo();
30     Assert.assertTrue(actual);
31 }
32
33* @Test
    Run | Debug
34 public void verifyproductTextTest() {
35     // To verify Product text spelling
36     String actual = inventoryPage.verifyProductText();
37     String expected = "PRODUCTS";
38     Assert.assertEquals(expected, actual);
39     Reporter.Log("Spelling of products text : " + actual);
40 }
41
42* @Test
    Run | Debug
43 public void verifycartItems() {
44     // To verify Item is added in cart
45     inventoryPage.getAddtocartBag().click();
46     inventoryPage.getCartLink().click();
47     boolean actual = inventoryPage.getBagpack().isDisplayed();
48     Assert.assertTrue(actual);
49 }
50
51* @Test
    Run | Debug
52 public void verifyAtoZSortOption() {
53     // To verify A to Z sorting option
54     List<String> actual = inventoryPage.actualNamesAccordingToDropdownOptionSelected("az");
55     List<String> expected = inventoryPage.alphabeticalOrderProductsNames();
56     Assert.assertEquals(actual, expected);
57     Reporter.Log("Order of Items : " + actual);
58 }
59
60* @Test
    Run | Debug
61 public void verifyZtoASortOption() {
62     // To verify Z to A sorting option
63     List<String> actual = inventoryPage.actualNamesAccordingToDropdownOptionSelected("za");
64     List<String> expected = inventoryPage.alphabeticalOrderProductsNames();
65     Collections.reverse(expected); // To change array list in reverse alphabetical order
66     Assert.assertEquals(actual, expected);
67     Reporter.Log("Order of Items : " + actual);
68 }
69
70* @Test
    Run | Debug
71 public void verifyLowToHighPriceSortOptionTest() {
72     // To verify Low to High price option
73     List<Double> actual = inventoryPage.actulPriceListAccordingToDropdownOptionSelected("lohi");
74     List<Double> expected = inventoryPage.ascendingOrderPriceList();
75     Assert.assertEquals(actual, expected);
76     Reporter.Log("Order of Items : " + actual);
77 }
78
79* @Test
    Run | Debug
80 public void verifyHighToLowPriceSortOptionTest() {
81     // To verify high to low price option
82     List<Double> actual = inventoryPage.actulPriceListAccordingToDropdownOptionSelected("hilo");
83     List<Double> expected = inventoryPage.ascendingOrderPriceList();
84     Collections.reverse(expected); // to change array list in Descending order
85     Assert.assertEquals(actual, expected);
86     Reporter.Log("Order of Items : " + actual);
87 }
88
89* @AfterMethod
90 public void closeBrowser() {
91     driver.close();
92 }
93
94 }
95

```


Inventory Report:

Test results
1 suite

All suites

Default suite

Info

- C:\Users\HP\AppData\Local\Temp\testing-eclipse--202259126\testing-customsuite.xml
- 1 test
- 0 groups
- Times
- Reporter output**
- Ignored methods
- Chronological view

Results

- 7 methods, 7 passed
- Passed methods (hide)
- verifyAtoZSortOption
- verifyHighToLowPriceSortOptionTest
- verifyLowToHighPriceSortOptionTest
- verifyPackLogoTest
- verifyZtoASortOption
- verifycartItems
- verifyproductTextTest

Reporter output for Default suite

verifyZtoASortOption
Order of Items : [Test.allTheThings() T-Shirt (Red), Sauce Labs Onesie, Sauce Labs Fleece Jacket, Sauce Labs Bolt T-Shirt, Sauce Labs Bike Light, Sauce Labs Backpack]

verifyHighToLowPriceSortOptionTest
Order of Items : [49.99, 29.99, 15.99, 15.99, 9.99, 7.99]

verifyLowToHighPriceSortOptionTest
Order of Items : [7.99, 9.99, 15.99, 15.99, 29.99, 49.99]

verifyAtoZSortOption
Order of Items : [Sauce Labs Backpack, Sauce Labs Bike Light, Sauce Labs Bolt T-Shirt, Sauce Labs Fleece Jacket, Sauce Labs Onesie, Test.allTheThings() T-Shirt (Red)]

verifyproductTextTest
Spelling of products text : PRODUCTS

CartPage Class:

```

eclipse-workspace - FrameworkProject/src/main/java/Pages/CartPage.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help

# TestBase.java # LoginPage.java # LoginPageTest.java # InventoryPageTest.java # *CartPage.java x
FrameworkProject src/main/java Pages CartPage CartPage0

1 package Pages;
2
3 *import org.openqa.selenium.WebElement;
4 import org.openqa.selenium.support.FindBy;
5 import org.openqa.selenium.support.PageFactory;
6
7 import Base.TestBase;
8
9 public class CartPage extends TestBase {
10 * @FindBy(xpath = "//a[@class='shopping_cart_link']")
11 public WebElement cart;
12 * @FindBy(xpath = "//span[@class='title']")
13 private WebElement yourCartText;
14 * @FindBy(id = "continue-shopping")
15 public WebElement continueShoppingBtn;
16 * @FindBy(id = "add-to-cart-sauce-labs-backpack")
17 private WebElement backPackAddToCartBtn;
18 * @FindBy(id = "add-to-cart-sauce-labs-fleece-jacket")
19 private WebElement jacketAddTOCartBtn;
20 * @FindBy(xpath = "//div[contains(text(),'Backpack')]")
21 public WebElement backPack;
22 * @FindBy(xpath = "//div[contains(text(),'Jacket')]")
23 public WebElement jacket;
24 * @FindBy(id = "remove-sauce-labs-backpack")
25 public WebElement backPackRemoveToCartBtn;
26 * @FindBy(id = "checkout")
27 private WebElement checkoutBtn;
28
29 * public CartPage() {
30 // To call initialization method
31 PageFactory.initElements(driver, this);
32 }
33
34 * public String verifyCartUrl() {
35 // To get current URL
36 return driver.getCurrentUrl();
37 }
38
39 * public String verifyYourCartText() {
40 // To verify Your Cart Text spelling
41 return yourCartText.getText();
42 }

```

Testers don't make software; they make it better.

Situ

```
43
44* public void addItemInCart() {
45    // To add Items in the cart
46    continueShopingBtn.click();
47    backPackAddToCartBtn.click();
48    jacketAddTOCartBtn.click();
49    cart.click();
50 }
51
52* public String verifyCountOfCart() {
53    continueShopingBtn.click();
54    backPackAddToCartBtn.click();
55    jacketAddTOCartBtn.click();
56    return cart.getText();
57 }
58
59* public WebElement getCheckoutBtn() {
60    return checkoutBtn;
61 }
62
63* public WebElement getCart() {
64    return cart;
65 }
66
67 }
68
```

CartPageTest Class:

```
eclipse-workspace - FrameworkProject/src/test/java/TestCases/CartPageTest.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
TestBase.java LoginPageTest.java LoginPageTest.java InventoryPageTest.java *CartPageTest.java CartPageTest.java
FrameworkProject > src/test/java > TestCases > CartPageTest > verifyCartUrlTest() : void
1 package TestCases;
2
3*import org.openqa.selenium.NoSuchElementException;
4 import org.testng.Assert;
5 import org.testng.annotations.AfterMethod;
6 import org.testng.annotations.BeforeMethod;
7 import org.testng.annotations.Test;
8
9 import Base.TestBase;
10 import Pages.CartPage;
11 import Utility.ReadData;
12
13 Run All
14 public class CartPageTest extends TestBase {
15     CartPage cartPage; // global variable
16
17* @BeforeMethod
18 public void setup() throws Exception {
19     // To call initialization method and login method from TestBase class
20     initialization();
21     cart();
22     cartPage = new CartPage();
23 }
24
25* @Test
26 Run | Debug
27 public void verifyCartUrlTest() throws Exception {
28     // To verify URL of cart page
29     String actual = cartPage.verifyCartUrl();
30     String expected = ReadData.excelData(5, 1);
31     Assert.assertEquals(actual, expected);
32 }
33
```

Testers don't make software; they make it better.

Situ

```
32* @Test
Run | Debug
33 public void verifyYourCartTextTest() {
34     // To verify Your cart Text spelling
35     String actual = cartPage.verifyYourCartText();
36     Assert.assertEquals("YOUR CART", actual);
37 }
38
39* @Test
Run | Debug
40 public void verifyItemsAddedInCartTest() {
41     // To verify Items added in cart
42     cartPage.addItemInCart();
43     boolean actual = cartPage.backPack.isDisplayed();
44     boolean expected = cartPage.jacket.isDisplayed();
45     Assert.assertEquals(actual, expected);
46 }
47
48* @Test
Run | Debug
49 public void verifyCountOfCartTest() {
50     // verify count of cart items
51     String actual = cartPage.verifyCountOfCart();
52     Assert.assertEquals(actual, "2");
53 }
54
55* @Test
Run | Debug
56 public void verifyItemsRemoverdFromCartTest() {
57     // To verify Item removed from Cart
58     cartPage.addItemInCart();
59     cartPage.backPackRemoveToCartBtn.click();
60     /* As we removed element so we get NoSuchElementException */
61     try {
62
63         cartPage.backPack.isDisplayed();
64         Assert.assertTrue(false);
65
66     } catch (NoSuchElementException e) {
67         /* if this exception occurs means element is removed from cart */
68         Assert.assertTrue(true);
69     }
70 }
71
72* @AfterMethod
73 public void closeBrowser() {
74     driver.close();
75 }
76 }
77
```

Cart Report:

Test results

1 suite

TestCases.CartPageTest
verifyCartUrlTest
verifyCountOfCartTest
verifyItemsAddedInCartTest
verifyItemsRemoverdFromCartTest
verifyYourCartTextTest

All suites

Default suite

Info

- C:\Users\HP\AppData\Local\Temp\testing-eclipse--636529323\Testing-customsuite.xml
- 1 test
- 0 groups
- Times
- Reporter output
- Ignored methods
- Chronological view

Results

- 5 methods, 5 passed
- Passed methods (hide)
- verifyCartUrlTest
- verifyCountOfCartTest
- verifyItemsAddedInCartTest
- verifyItemsRemoverdFromCartTest
- verifyYourCartTextTest

Testers don't make software; they make it better.

CheckoutStepOnePage Class:

```

1 package Pages;
2
3 import org.openqa.selenium.WebElement;
4 import org.openqa.selenium.support.FindBy;
5 import org.openqa.selenium.support.PageFactory;
6
7 import Base.TestBase;
8 import Utility.ReadData;
9
10 public class CheckoutStepOnePage extends TestBase {
11     CartPage cartPage;
12     InventoryPage inventoryPage;
13     @FindBy(className = "title")
14     private WebElement checkoutTitle;
15     @FindBy(id = "cancel")
16     private WebElement cancelBtn;
17     @FindBy(id = "continue")
18     private WebElement continueBtn;
19     @FindBy(id = "first-name")
20     private WebElement firstNameTextbox;
21     @FindBy(id = "last-name")
22     private WebElement lastNameTextbox;
23     @FindBy(id = "postal-code")
24     private WebElement postalCodeTextbox;
25     @FindBy(css = ".error-message-container.error")
26     private WebElement errorMessage;
27
28     public CheckoutStepOnePage() {
29         // constructor to initialize driver with instance variables
30         PageFactory.initElements(driver, this);
31         inventoryPage = new InventoryPage();
32         cartPage = new CartPage();
33     }
34
35     public String verifycheckoutTitleText() {
36         // To verify Title of checkOut page one
37         return checkoutTitle.getText();
38     }
39
40     public String verifyErrorMessage() {
41         // To verify Error Message
42         continueBtn.click();
43         return errorMessage.getText();
44     }
45
46     public String verifyEnteredFirstName() throws Exception {
47         // To enter first name in first name textBox
48         firstNameTextbox.sendKeys(ReadData.excelData(6, 1));
49         return firstNameTextbox.getAttribute("value");
50     }
51
52     public String verifyEnteredLastName() throws Exception {
53         // To enter last name in last name textBox
54         lastNameTextbox.sendKeys(ReadData.excelData(7, 1));
55         return lastNameTextbox.getAttribute("value");
56     }
57
58     public String verifyEnteredPostalCode() throws Exception {
59         // To enter postal code in postal code textBox
60         postalCodeTextbox.sendKeys(ReadData.excelData(8, 1));
61         return postalCodeTextbox.getAttribute("value");
62     }
63
64     public String verifyCheckoutFunctionality() throws Exception {
65         // To add items to cart and navigate to checkout
66         cancelBtn.click();
67         cartPage.continueShopingBtn.click();
68         inventoryPage.getAddtocartBag().click();
69         inventoryPage.getCartLink().click();
70         cartPage.getCheckoutBtn().click();
71         verifyEnteredFirstName(); // to enter first name
72         verifyEnteredLastName(); // to enter last name
73         verifyEnteredPostalCode(); // to enter postal code
74         continueBtn.click();
75         return driver.getCurrentUrl(); // verify this URL with check out page two URL
76     }

```

Situ

```
77
78* public WebElement getContinueBtn() {
79    // getMethod to return WebElement
80    return continueBtn;
81 }
82
83* public WebElement getFirstNameTextbox() {
84    // getMethod to return WebElement
85    return firstNameTextbox;
86 }
87
88* public WebElement getLastNameTextbox() {
89    // getMethod to return WebElement
90    return lastNameTextbox;
91 }
92
93* public WebElement getPostalCodeTextbox() {
94    // getMethod to return WebElement
95    return postalCodeTextbox;
96 }
97 }
98
```

CheckoutStepOnePageTest Class:

```
eclipse-workspace - FrameworkProject/src/test/java/TestCases/CheckoutStepOnePageTest.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
# TestBase.java LoginPage.java LoginPageTest.java InventoryPage.java InventoryPageTest.java *CartPage.java CartPageTest.java CheckoutStepOnePage.java CheckoutStepOnePageTest.java
1 package TestCases;
2
3* import org.testng.Assert;
4 import org.testng.Reporter;
5 import org.testng.annotations.AfterMethod;
6 import org.testng.annotations.BeforeMethod;
7 import org.testng.annotations.Test;
8
9 import Base.TestBase;
10 import Pages.CheckoutStepOnePage;
11 import Utility.ReadData;
12
13 Run All
14 public class CheckoutStepOnePageTest extends TestBase {
15     CheckoutStepOnePage checkOutPageOne;
16
17     @BeforeMethod
18     public void setup() throws Exception {
19         // To open browser enter URL and navigate to checkout page one
20         initialization();
21         checkOutPageOne();
22         checkOutPageOne = new CheckoutStepOnePage();
23     }
24
25     @Test
26     Run | Debug
27     public void verifyErrorMessageTest() throws Exception {
28         // To verify error message when click on continue without entering details
29         String actual = checkOutPageOne.verifyErrorMessage();
30         String expected = ReadData.excelData(12, 1);
31         Assert.assertEquals(actual, expected);
32         Reporter.log("Error message : " + actual);
33     }
34
35     @Test
36     Run | Debug
37     public void verifyCheckoutTitleTest() throws Exception {
38         //To verify Title of check out page one
39         String actual = checkOutPageOne.verifycheckoutTitleText();
40         String expected = ReadData.excelData(10, 1);
41         Assert.assertEquals(actual, expected);
42         Reporter.log("Title : " + actual);
43     }
44 }
```

Testers don't make software; they make it better.

Situ

```
41
42• @Test
    Run | Debug
43 public void verifyEnteredFirstNameTest() throws Exception {
44     // To verify firstName should be entered in firstName textBox
45     String actual = checkOutPageOne.verifyEnteredFirstName();
46     String expected = ReadData.excelData(6, 1);
47     Assert.assertEquals(actual, expected);
48     Reporter.Log("Entered First Name : " + actual);
49 }
50
51• @Test
    Run | Debug
52 public void verifyEnteredLastNameTest() throws Exception {
53     // To verify lastName should be entered in lastName textBox
54     String actual = checkOutPageOne.verifyEnteredLastName();
55     String expected = ReadData.excelData(7, 1);
56     Assert.assertEquals(actual, expected);
57     Reporter.Log("Entered Last Name : " + actual);
58 }
59
60• @Test
    Run | Debug
61 public void verifyEnteredPostalCodeTest() throws Exception {
62     // To verify postalCode should be entered in postalCode textBox
63     String actual = checkOutPageOne.verifyEnteredPostalCode();
64     String expected = ReadData.excelData(8, 1);
65     Assert.assertEquals(actual, expected);
66     Reporter.Log("Entered Postal Code : " + actual);
67 }
68
69• @Test
    Run | Debug
70 public void verifyCheckoutFunctionalityTest() throws Exception {
71     // To verify user should navigate to checkout page when click on continue button
72     // verify using URL
73     String actual = checkOutPageOne.verifyCheckoutFunctionality();
74     String expected = ReadData.excelData(14, 1);
75     Assert.assertEquals(actual, expected);
76 }
77
78• @AfterMethod
79 public void closeBrowser() {
80     driver.close();
81 }
82 }
83
```

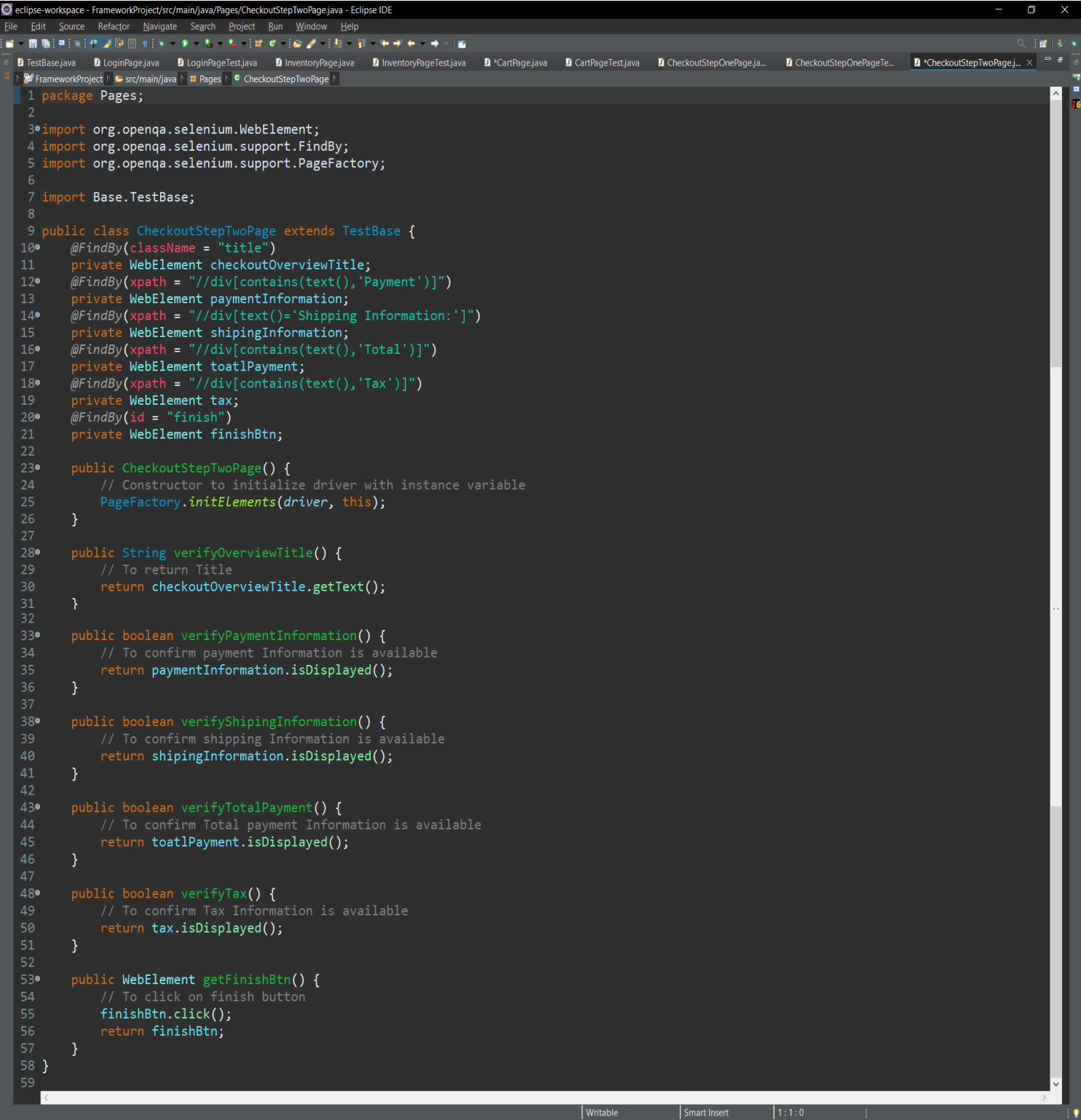
Checkout Page One Report:

The screenshot displays the 'Test results' window in an IDE. The top bar is blue and labeled 'Test results' with '1 suite' below it. The main area is divided into two panes. The left pane, titled 'All suites', shows a tree view for the 'Default suite' with a yellow background. It includes an 'Info' section with file paths and a 'Results' section showing '6 methods, 6 passed' and a list of test methods with green checkmarks. The right pane, titled 'Reporter output for Default suite', shows the output for five test methods: 'verifyCheckoutTitleTest' (Title: CHECKOUT: YOUR INFORMATION), 'verifyEnteredFirstNameTest' (Entered First Name: Tonny), 'verifyErrorMessageTest' (Error message: Error: First Name is required), 'verifyEnteredPostalCodeTest' (Entered Postal Code: 400009), and 'verifyEnteredLastNameTest' (Entered Last Name: Stark).

Test Method	Output
verifyCheckoutTitleTest	Title : CHECKOUT: YOUR INFORMATION
verifyEnteredFirstNameTest	Entered First Name : Tonny
verifyErrorMessageTest	Error message : Error: First Name is required
verifyEnteredPostalCodeTest	Entered Postal Code : 400009
verifyEnteredLastNameTest	Entered Last Name : Stark

Testers don't make software; they make it better.

CheckoutStepTwoPage Class:

A screenshot of the Eclipse IDE interface. The top menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. Below the menu is a toolbar with various icons. The project explorer on the left shows the file structure: FrameworkProject > src/main/java > Pages > CheckoutStepTwoPage. The editor window displays the code for CheckoutStepTwoPage.java. The code is as follows:

```
1 package Pages;
2
3 import org.openqa.selenium.WebElement;
4 import org.openqa.selenium.support.FindBy;
5 import org.openqa.selenium.support.PageFactory;
6
7 import Base.TestBase;
8
9 public class CheckoutStepTwoPage extends TestBase {
10     @FindBy(className = "title")
11     private WebElement checkoutOverviewTitle;
12     @FindBy(xpath = "//div[contains(text(),'Payment')]")
13     private WebElement paymentInformation;
14     @FindBy(xpath = "//div[text()='Shipping Information:']")
15     private WebElement shippingInformation;
16     @FindBy(xpath = "//div[contains(text(),'Total')]")
17     private WebElement totalPayment;
18     @FindBy(xpath = "//div[contains(text(),'Tax')]")
19     private WebElement tax;
20     @FindBy(id = "finish")
21     private WebElement finishBtn;
22
23     public CheckoutStepTwoPage() {
24         // Constructor to initialize driver with instance variable
25         PageFactory.initElements(driver, this);
26     }
27
28     public String verifyOverviewTitle() {
29         // To return Title
30         return checkoutOverviewTitle.getText();
31     }
32
33     public boolean verifyPaymentInformation() {
34         // To confirm payment Information is available
35         return paymentInformation.isDisplayed();
36     }
37
38     public boolean verifyShippingInformation() {
39         // To confirm shipping Information is available
40         return shippingInformation.isDisplayed();
41     }
42
43     public boolean verifyTotalPayment() {
44         // To confirm Total payment Information is available
45         return totalPayment.isDisplayed();
46     }
47
48     public boolean verifyTax() {
49         // To confirm Tax Information is available
50         return tax.isDisplayed();
51     }
52
53     public WebElement getFinishBtn() {
54         // To click on finish button
55         finishBtn.click();
56         return finishBtn;
57     }
58 }
59
```

The status bar at the bottom shows 'Writable', 'Smart Insert', and '1:1:0'.

CheckoutStepTwoPageTest Class:

```

1 package TestCases;
2
3 import org.testng.Assert;
4 import org.testng.annotations.AfterMethod;
5 import org.testng.annotations.BeforeMethod;
6 import org.testng.annotations.Test;
7
8 import Base.TestBase;
9 import Pages.CheckoutStepTwoPage;
10 import Utility.ReadData;
11
12 Run All
13 public class CheckoutStepTwoPageTest extends TestBase {
14     CheckoutStepTwoPage checkoutStepTwoPage;
15
16     @BeforeMethod
17     public void setup() throws Exception {
18         // To call initialize method and login method from TestBase class
19         initialization();
20         checkoutPageTwo();
21         checkoutStepTwoPage = new CheckoutStepTwoPage();
22     }
23
24     @Test
25     Run | Debug
26     public void verifyOverviewTitleTest() throws Exception {
27         // To confirm Title
28         String actual = checkoutStepTwoPage.verifyOverviewTitle();
29         String expected = ReadData.excelData(16, 1);
30         Assert.assertEquals(actual, expected);
31     }
32
33     @Test
34     Run | Debug
35     public void verifyPaymentInformationTest() {
36         // To conform Payment information
37         boolean actual = checkoutStepTwoPage.verifyPaymentInformation();
38         Assert.assertTrue(actual);
39     }
40
41     @Test
42     Run | Debug
43     public void verifyShipingInformationTest() {
44         // To conform shipping information
45         boolean actual = checkoutStepTwoPage.verifyShipingInformation();
46         Assert.assertTrue(actual);
47     }
48
49     @Test
50     Run | Debug
51     public void verifyTotalPaymentTest() {
52         // To confirm Total payment information
53         boolean actual = checkoutStepTwoPage.verifyTotalPayment();
54         Assert.assertTrue(actual);
55     }
56
57     @Test
58     Run | Debug
59     public void verifyTaxTest() {
60         // To confirm Tax information
61         boolean actual = checkoutStepTwoPage.verifyTax();
62         Assert.assertTrue(actual);
63     }
64 }

```


Situ

```

59
60 • @Test
    Run | Debug
61 public void verifyFinishButtonNavigateToLastPage() throws Exception {
62     // To verify after clicking on finish button navigate to last page using URL
63     checkoutStepTwoPage.getFinishBtn();
64     String actual = driver.getCurrentUrl();
65     String expected = ReadData.excelData(19, 1);
66     Assert.assertEquals(actual, expected);
67 }
68
69
70 • @AfterMethod
71 public void closeBrowser() {
72     driver.close();
73 }
74 }
75

```

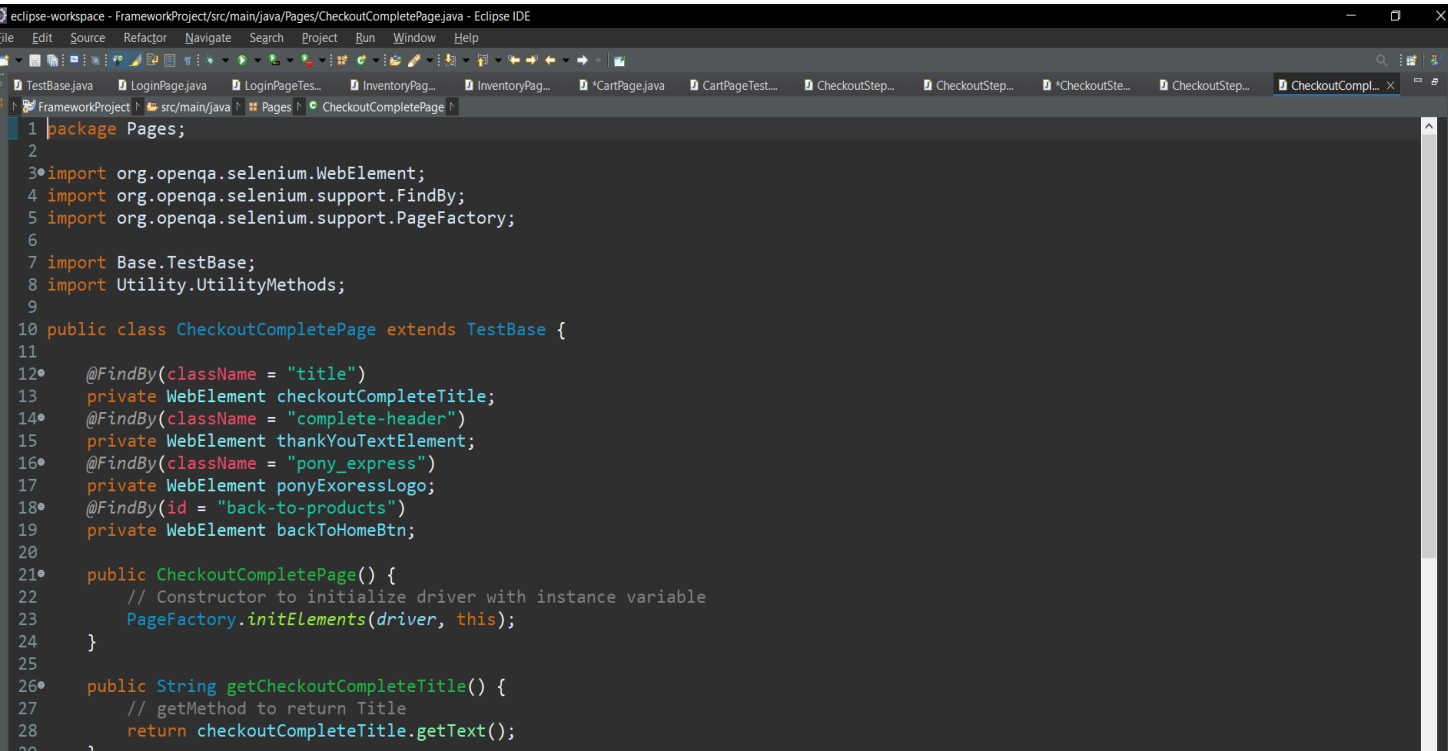
Checkout Step Two Report:

Test results

1 suite

All suites	TestCases.CheckoutStepTwoPageTest
Default suite Info <ul style="list-style-type: none"> C:\Users\HP\AppData\Local\Temp\testing-eclipse--498898131\testing-customsuite.xml 1 test 0 groups Times Reporter output Ignored methods Chronological view Results <ul style="list-style-type: none"> 6 methods, 6 passed Passed methods (hide) <ul style="list-style-type: none"> verifyFinishButtonNavigateToLastPage verifyOverviewTitleTest verifyPaymentInformationTest verifyShippingInformationTest verifyTaxTest verifyTotalPaymentTest 	<div style="background-color: #dcedc8; padding: 5px; margin-bottom: 5px;">✔ TestCases.CheckoutStepTwoPageTest</div> <pre> verifyFinishButtonNavigateToLastPage ----- verifyOverviewTitleTest ----- verifyPaymentInformationTest ----- verifyShippingInformationTest ----- verifyTaxTest ----- verifyTotalPaymentTest ----- </pre>

CheckoutStepCompletePage Class:



```

1 package Pages;
2
3 import org.openqa.selenium.WebElement;
4 import org.openqa.selenium.support.FindBy;
5 import org.openqa.selenium.support.PageFactory;
6
7 import Base.TestBase;
8 import Utility.UtilityMethods;
9
10 public class CheckoutCompletePage extends TestBase {
11
12     @FindBy(className = "title")
13     private WebElement checkoutCompleteTitle;
14     @FindBy(className = "complete-header")
15     private WebElement thankYouTextElement;
16     @FindBy(className = "pony_express")
17     private WebElement ponyExoressLogo;
18     @FindBy(id = "back-to-products")
19     private WebElement backToHomeBtn;
20
21     public CheckoutCompletePage() {
22         // Constructor to initialize driver with instance variable
23         PageFactory.initElements(driver, this);
24     }
25
26     public String getCheckoutCompleteTitle() {
27         // getMethod to return Title
28         return checkoutCompleteTitle.getText();
29     }

```

Testers don't make software; they make it better.

Situ

```
30
31• public String getThankYouTextElement() {
32    // getMthod to return Thank you text
33    return thankYouTextElement.getText();
34 }
35
36• public String verifyBackToHomeButtonFunctionality() {
37    // to click Back to button and get URL
38    backToHomeBtn.click();
39    return driver.getCurrentUrl();
40 }
41
42• public boolean verifyPonyExpressLogo() {
43    // To verify logo is present
44    UtilityMethods.expliciteWait(driver, ponyExoressLogo);
45    return ponyExoressLogo.isDisplayed();
46 }
47
48 }
49
```

CheckoutStepCompletePageTest Class:

```
eclipse-workspace - FrameworkProject/src/test/java/TestCases/CheckoutCompletePageTest.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help

1 package TestCases;
2
3•import org.testng.Assert;
4 import org.testng.Reporter;
5 import org.testng.annotations.AfterMethod;
6 import org.testng.annotations.BeforeMethod;
7 import org.testng.annotations.Test;
8
9 import Base.TestBase;
10 import Pages.CheckoutCompletePage;
11 import Utility.ReadData;
12
13 Run All
14 public class CheckoutCompletePageTest extends TestBase {
15     CheckoutCompletePage checkoutCompletePage;
16
17     @BeforeMethod
18     public void setup() throws Exception {
19         // To call initialize method and to navigate to last page
20         initialization();
21         checkoutComplete();
22         checkoutCompletePage = new CheckoutCompletePage();
23     }
24
25     @Test
26     Run | Debug
27     public void verifyPonyExpressLogoTest() {
28         // To verify logo
29         boolean actual = checkoutCompletePage.verifyPonyExpressLogo();
30         Assert.assertTrue(actual);
31     }
32
33     @Test
34     Run | Debug
35     public void verifyCheckoutCompleteTitleTest() throws Exception {
36         // To verify Title
37         String actual = checkoutCompletePage.getCheckoutCompleteTitle();
38         String expected = ReadData.excelData(17, 1);
39         Assert.assertEquals(actual, expected);
40     }
41 }
```

Testers don't make software; they make it better.

Situ

```
39* @Test
Run | Debug
40 public void verifyThankYouTextElementTest() throws Exception {
41     // To verify Thank You text
42     String actual = checkoutCompletePage.getThankYouTextElement();
43     String expected = ReadData.excelData(18, 1);
44     Assert.assertEquals(actual, expected);
45     Reporter.Log("Thank you text: " + actual);
46 }
47
48* @Test
Run | Debug
49 public void verifyBackToHomeButtonFunctionalityTest() throws Exception {
50     // To verify Back to home button navigate to inventory page
51     String actual = checkoutCompletePage.verifyBackToHomeButtonFunctionality();
52     String expected = ReadData.excelData(4, 1);
53     Assert.assertEquals(actual, expected);
54 }
55
56* @AfterMethod
57 public void closeBrowser() {
58     driver.close();
59 }
60 }
61
```

Checkout Step Two Page Report:

Test results
1 suite

All suites

Default suite

Info

- C:\Users\HP\AppData\Local Temp\testing-eclipse-1647191352\testing-customsuite.xml
- 1 test
- 0 groups
- Times
- Reporter output
- Ignored methods
- Chronological view

Results

- 4 methods, 4 passed
- Passed methods (hide)
- verifyBackToHomeButtonFunctionalityTest
- verifyCheckoutCompleteTitleTest
- verifyPonyExpressLogoTest
- verifyThankYouTextElementTest

Reporter output for Default suite

verifyThankYouTextElementTest
Thank you text: THANK YOU FOR YOUR ORDER

Testers don't make software; they make it better.