

The screenshot shows the IntelliJ IDEA interface with the project 'hiSelenium' open. The 'activity2.java' file is the active editor. The code implements a Selenium test for a login form:

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import io.github.bonigarcia.wdm.WebDriverManager;

public class activity2 {
    public static void main(String[] args) {
        // Setup the Firefox driver(GeckoDriver)
        WebDriverManager.firefoxdriver().setup();

        // Create a new instance of the Firefox driver
        WebDriver driver = new FirefoxDriver();

        // Open the browser
        driver.get("https://v1.training-support.net/selenium/login-form");
        WebElement username = driver.findElement(By.id("username"));
        username.sendKeys(...keysToSend: "bhavana");
        WebElement password = driver.findElement(By.id("password"));
        password.sendKeys(...keysToSend: "lesli");
        WebElement button = driver.findElement(By.id("login"));
        button.click();

        driver.quit();
    }
}
```

This screenshot is identical to the one above, showing the IntelliJ IDEA interface with the 'activity2.java' file active. The code for the Selenium test remains the same:

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import io.github.bonigarcia.wdm.WebDriverManager;

public class activity2 {
    public static void main(String[] args) {
        // Setup the Firefox driver(GeckoDriver)
        WebDriverManager.firefoxdriver().setup();

        // Create a new instance of the Firefox driver
        WebDriver driver = new FirefoxDriver();

        // Open the browser
        driver.get("https://v1.training-support.net/selenium/login-form");
        WebElement username = driver.findElement(By.id("username"));
        username.sendKeys(...keysToSend: "bhavana");
        WebElement password = driver.findElement(By.id("password"));
        password.sendKeys(...keysToSend: "lesli");
        WebElement button = driver.findElement(By.id("login"));
        button.click();

        driver.quit();
    }
}
```

The screenshot shows the IntelliJ IDEA interface with the project 'hiSelenium' open. The code editor displays the file 'activity4.java'. The code uses Selenium WebDriver to open a Firefox browser, navigate to a specific URL, and perform assertions on page titles and header elements. The code editor has tabs for test11.java, activity1.java, activity2.java, activity4.java (which is the active tab), activity9.java, and test7.java.

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.By;
import io.github.bonigarcia.wdm.WebDriverManager;

public class activity4 {
    public static void main(String[] args) {
        // Setup the Firefox driver(beckoDriver)
        WebDriverManager.firefoxdriver().setup();

        // Create a new instance of the Firefox driver
        WebDriver driver = new FirefoxDriver();

        // Open the browser
        driver.get("https://v1.training-support.net/selenium/target-practice");

        // Perform testing and assertions
        String title = driver.getTitle(); // Get the title of the webpage

        // Print the title to the console
        System.out.println("Title of the webpage: " + title);
        //finding 3rd header
        WebElement header3 = driver.findElement(By.xpath("//*[@id='third-header']"));
        //print the header3
        System.out.println("third header is" + header3.getText());
        //finding 5th header
        WebElement header5 = driver.findElement(By.xpath("//html/body/div/div[2]/div/div[5]"));
        //finding the color of the 5th header
    }
}
```

The screenshot shows the IntelliJ IDEA interface with the project 'hiSelenium' open. The code editor displays the file 'activity4.java'. The code uses Selenium WebDriver to interact with a web application, specifically targeting buttons and their properties like color and class. It prints the text of buttons and performs a click action on a button with the ID 'login'. The code editor has tabs for test11.java, activity1.java, activity2.java, activity4.java (which is the active tab), activity9.java, and test7.java.

```
public class activity4 {
    public static void main(String[] args) {
        String color = header5.getCssValue("color");
        // Print the color value
        System.out.println("Color of the text: " + color);

        //find violet button
        WebElement buttonviolet = driver.findElement(By.cssSelector(".violet"));
        // Get the value of the 'class' attribute
        String ss=buttonviolet.getAttribute("name");
        // Print the classes
        System.out.println("Classes of the button: " + ss);

        //find the grey button
        WebElement buttongrey = driver.findElement(By.cssSelector(".ui.grey.button"));

        //getting the text
        String buttonText = buttongrey.getText();

        //printing the text
        System.out.println("The Text of the button:" + buttonText);
        WebElement button =driver.findElement(By.id("login"));
        button.click();

        driver.quit();
    }
}
```

The screenshot shows the IntelliJ IDEA interface with the project 'hiSelenium' open. The code editor displays the file 'activity4.java'. The code uses Selenium WebDriver to interact with a Firefox browser. It sets up the Firefox driver, creates a new instance, opens a specific URL, and performs assertions on the page title and header colors.

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.By;
import io.github.bonigarcia.wdm.WebDriverManager;

public class activity4 {
    public static void main(String[] args) {
        // Setup the Firefox driver(beckoDriver)
        WebDriverManager.firefoxdriver().setup();

        // Create a new instance of the Firefox driver
        WebDriver driver = new FirefoxDriver();

        // Open the browser
        driver.get("https://v1.training-support.net/selenium/target-practice");

        // Perform testing and assertions
        String title = driver.getTitle(); // Get the title of the webpage

        // Print the title to the console
        System.out.println("Title of the webpage: " + title);
        //finding 3rd header
        WebElement header3 = driver.findElement(By.xpath("//*[@id='third-header']"));
        //print the header3
        System.out.println("third header is" + header3.getText());
        //finding 5th header
        WebElement header5 = driver.findElement(By.xpath("//html/body/div/div[2]/div/div[5]"));
        //finding the color of the 5th header
    }
}
```

The screenshot shows the IntelliJ IDEA interface with the project 'hiSelenium' open. The code editor displays the file 'activity4.java'. The code uses Selenium WebDriver to interact with a Firefox browser. It finds a violet button, gets its class attribute, prints the classes, finds a grey button, gets its text, prints the text, and finally clicks a login button.

```
public class activity4 {
    public static void main(String[] args) {
        //finding 5th header
        WebElement header5 = driver.findElement(By.xpath("//html/body/div/div[2]/div/div[5]"));
        //finding the color of the 5th header
        String color = header5.getCssValue("color");

        // Print the color value
        System.out.println("Color of the text: " + color);

        //find violet button
        WebElement buttonviolet = driver.findElement(By.cssSelector(".violet"));
        // Get the value of the 'class' attribute
        String ss=buttonviolet.getAttribute("class");
        // Print the classes
        System.out.println("Classes of the button: " + ss);

        //find the grey button
        WebElement buttongrey = driver.findElement(By.cssSelector(".ui.grey.button"));

        //getting the text
        String buttonText = buttongrey.getText();

        //printing the text
        System.out.println("The Text of the button:" + buttonText);

        //click the button
        WebElement button = driver.findElement(By.id("login"));
        button.click();
    }
}
```

The screenshot shows the IntelliJ IDEA interface with the project 'hiSelenium' open. The code editor displays the file 'activity6.java'. The code uses Selenium WebDriver to interact with a Firefox browser. It prints the title and page source of a specific URL and performs key presses ('A', 'a', 'c') using Actions.

```
import org.openqa.selenium.By;
import org.openqa.selenium.Keys;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import io.github.bonigarcia.wdm.WebDriverManager;
import org.openqa.selenium.interactions.Actions;

public class activity6 {
    public static void main(String[] args) {
        // Setup the Firefox driver(geckoDriver)
        WebDriverManager.firefoxdriver().setup();

        // Create a new instance of the Firefox driver
        WebDriver driver = new FirefoxDriver();

        // Open the browser
        driver.get("https://vi.training-support.net/selenium/input-events");
        String title = driver.getTitle();
        System.out.println("Training Support");
        String pageSource=driver.getPageSource();
        System.out.println("Page Source");
        System.out.println(pageSource);

        Actions actions=new Actions(driver);
        actions.sendKeys(Keys.SHIFT,"A");

        actions.sendKeys(Keys.CONTROL,"a");
    }
}
```

The screenshot shows the IntelliJ IDEA interface with the project 'hiSelenium' open. The code editor displays the file 'activity6.java'. The code has been modified to include a click operation on a WebElement and to quit the driver at the end of the main method.

```
public class activity6 {
    public static void main(String[] args) {
        System.out.println("Training Support");
        String pageSource=driver.getPageSource();
        System.out.println("Page Source");
        System.out.println(pageSource);

        Actions actions=new Actions(driver);
        actions.sendKeys(Keys.SHIFT,"A");

        actions.sendKeys(Keys.CONTROL,"a");
        actions.sendKeys(Keys.CONTROL,"c");

        actions.build().perform();

        WebElement button=driver.findElement(By.id("target"));
        button.click();

        driver.quit();
    }
}
```

```
1 import org.openqa.selenium.*;
2 import org.openqa.selenium.firefox.FirefoxDriver;
3 import io.github.bonigarcia.wdm.WebDriverManager;
4 import org.openqa.selenium.interactions.Actions;
5 import org.openqa.selenium.By;
6 import org.openqa.selenium.WebDriver;
7 import org.openqa.selenium.WebElement;
8
9
10 public class activity777 {
11     public static void main(String[] args) {
12         WebDriverManager.firefoxdriver().setup();
13         WebDriver driver = new FirefoxDriver();
14         driver.get("https://v1.training-support.net/selenium/drag-drop");
15         WebElement x= driver.findElement(By.xpath("//*[@id='draggable']"));
16         WebElement yy= driver.findElement(By.xpath("//*[@id='droppable']"));
17         WebElement yyyy= driver.findElement(By.xpath("//*[@id='dropzone2']"));
18         new Actions(driver)
19             .sendKeys("s")
20             .keyDown(Keys.CONTROL)
21             .sendKeys("a")
22             .keyDown(Keys.CONTROL)
23             .sendKeys("c")
24             .dragAndDrop(x,yy)
25             .dragAndDrop(x,yyyy)
26             .perform();
27         if (yy.getText().contains("Dropped")){
28             System.out.println("Dropped on left");
29         }
30     }
31 }
```

```
10 public class activity777 {
11     public static void main(String[] args) {
14         driver.get("https://v1.training-support.net/selenium/drag-drop");
15         WebElement x= driver.findElement(By.xpath("//*[@id='draggable']"));
16         WebElement yy= driver.findElement(By.xpath("//*[@id='droppable']"));
17         WebElement yyyy= driver.findElement(By.xpath("//*[@id='dropzone2']"));
18         new Actions(driver)
19             .sendKeys("s")
20             .keyDown(Keys.CONTROL)
21             .sendKeys("a")
22             .keyDown(Keys.CONTROL)
23             .sendKeys("c")
24             .dragAndDrop(x,yy)
25             .dragAndDrop(x,yyyy)
26             .perform();
27         if (yy.getText().contains("Dropped")){
28             System.out.println("Dropped on left");
29         }
30         if (yyyy.getText().contains("Dropped")){
31             System.out.println("Dropped on right");
32         }
33         driver.quit();
34     }
35 }
```

The screenshot shows two instances of the IntelliJ IDEA IDE. Both instances have the same project structure and code content, indicating they are likely screenshots of the same session.

**Project Structure:**

- hiSelenium
- src
- main
- java
- packageName
- activity1
- activity2
- activity4
- activity5
- activity6
- activity7
- activity8
- activity9
- activity10
- activity11
- activity12
- activity13
- activity14
- activity15
- activity16
- activity17
- activity18
- activity19
- activity20
- activity21

**Code Content (activity10.java):**

```
1 import org.openqa.selenium.firefox.FirefoxDriver;
2 import org.openqa.selenium.By;
3 import org.openqa.selenium.WebDriver;
4 import org.openqa.selenium.WebElement;
5 import org.openqa.selenium.support.ui.WebDriverWait;
6 import java.time.Duration;
7 public class activity10 {
8     public static void main(String[] args) {
9         WebDriver driver = new FirefoxDriver();
10        // Create the Wait object
11        WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
12        // Open the page
13        driver.get("https://v1.training-support.net/selenium/dynamic-controls");
14        // Print the title of the page
15        System.out.println("Home page title: " + driver.getTitle());
16        // Find the checkbox input element
17        WebElement checkbox = driver.findElement(By.cssSelector("input[type='checkbox']"));
18        // Check if the checkbox is visible on the page
19        boolean isCheckboxVisible = checkbox.isDisplayed();
20        System.out.println("Is checkbox visible on the page? " + isCheckboxVisible);
21        // Find the "remove checkbox" button and click it
22        WebElement removeButton = driver.findElement(By.id("toggleCheckbox"));
23        removeButton.click();
24        // Check if the checkbox is visible again
25        boolean isCheckboxVisibleAfterClick = checkbox.isDisplayed();
26        System.out.println("Is checkbox visible after clicking 'Remove Checkbox' button? " + isCheckboxVisibleAfterClick);
27        // Close the browser
28        driver.quit();
29    }
}
```

**Code Content (activity11.java):**

```
1 import org.openqa.selenium.firefox.FirefoxDriver;
2 import org.openqa.selenium.By;
3 import org.openqa.selenium.WebDriver;
4 import org.openqa.selenium.WebElement;
5 import org.openqa.selenium.firefox.FirefoxDriver;
6 import org.openqa.selenium.support.ui.WebDriverWait;
7 import java.time.Duration;
8
9
10 public class activity11 {
11     public static void main(String[] args) {
12         WebDriver driver = new FirefoxDriver();
13         // Create the Wait object
14         WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
15         // Open the page
16         driver.get("https://v1.training-support.net/selenium/dynamic-controls");
17         // Print the title of the page
18         System.out.println("Home page title: " + driver.getTitle());
19         // Find the checkbox input element
20         WebElement checkbox = driver.findElement(By.cssSelector("input[type='checkbox']"));
21
22
23
24
25
26
27
28
29
}
```

The screenshot shows the IntelliJ IDEA interface with the project 'hiSelenium' open. The file 'activity11.java' is the active editor. The code implements a Selenium test for a checkbox. It prints the page title, finds the checkbox element, checks if it's selected, clicks it, and then checks again after the click.

```
11 public class activity11 {
12     public static void main(String[] args) {
13         // Print the title of the page
14         System.out.println("Home page title: " + driver.getTitle());
15
16         // Find the checkbox input element
17         WebElement checkbox = driver.findElement(By.cssSelector("input[type='checkbox']"));
18
19         // Check if the checkbox is selected and print the result
20         boolean isCheckboxSelected = checkbox.isSelected();
21         System.out.println("Is checkbox selected? " + isCheckboxSelected);
22
23         // Click the checkbox to select it
24         checkbox.click();
25
26         // Check if the checkbox is selected again
27         boolean isCheckboxSelectedAfterClick = checkbox.isSelected();
28         System.out.println("Is checkbox selected after clicking? " + isCheckboxSelectedAfterClick);
29
30         // Close the browser
31         driver.quit();
32     }
33 }
```

The screenshot shows the IntelliJ IDEA interface with the project 'hiSelenium' open. The file 'activity12.java' is the active editor. The code sets up a FirefoxDriver, creates a WebDriverWait object, opens a URL, prints the page title, and finds a text input field using Xpath.

```
1 import org.openqa.selenium.firefox.FirefoxDriver;
2 import org.openqa.selenium.By;
3 import org.openqa.selenium.WebDriver;
4 import org.openqa.selenium.WebElement;
5 import org.openqa.selenium.firefox.FirefoxDriver;
6 import org.openqa.selenium.support.ui.WebDriverWait;
7
8 import java.time.Duration;
9
10
11 public class activity12{
12     public static void main(String[] args) {
13
14         WebDriver driver = new FirefoxDriver();
15
16         // Create the Wait object
17
18         WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
19
20         // Open the page
21
22         driver.get("https://v1.training-support.net/selenium/dynamic-controls");
23
24         // Print the title of the page
25         System.out.println("Home page title: " + driver.getTitle());
26
27         // Find the text field
28         WebElement textField = driver.findElement(By.xpath("//[@id=\"$input-text\"]"));
29     }
29 }
```

The screenshot shows the IntelliJ IDEA interface with the project 'hiSelenium' open. The code editor displays 'activity12.java' which contains the following Java code:

```
11  public class activity12{
12      public static void main(String[] args) {
13          // Find the text field
14          WebElement textField = driver.findElement(By.xpath("//*[@id='input-text']"));
15
16          // Check if the text field is enabled and print it
17          boolean isTextFieldEnabled = textField.isEnabled();
18          System.out.println("Is Text Field Enabled: " + isTextFieldEnabled);
19
20          // Click the "Enable Input" button
21          WebElement enableButton = driver.findElement(By.xpath("//*[@id='toggleInput']"));
22          enableButton.click();
23
24          // Check if the text field is enabled again and print it
25          isTextFieldEnabled = textField.isEnabled();
26          System.out.println("Is Text Field Enabled after clicking Enable Input button: " + isTextFieldEnabled);
27
28          // Close the browser
29          driver.quit();
30      }
31  }
```

The status bar at the bottom indicates the file is 47:1, CRLF, UTF-8, 4 spaces, and the system status shows 85°F, Mostly cloudy, ENG IN, 22:40, and 02-05-2024.

The screenshot shows the IntelliJ IDEA interface with the project 'hiSelenium' open. The code editor displays 'activity13.java' which contains the following Java code:

```
1  import org.openqa.selenium.firefox.FirefoxDriver;
2  import org.openqa.selenium.By;
3  import org.openqa.selenium.WebDriver;
4  import org.openqa.selenium.WebElement;
5  import org.openqa.selenium.firefox.FirefoxDriver;
6  import org.openqa.selenium.support.ui.WebDriverWait;
7
8  import java.time.Duration;
9
10
11 public class activity13 {
12     public static void main(String[] args) {
13
14         WebDriver driver = new FirefoxDriver();
15
16         // Create the Wait object
17
18         WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
19
20         // Open the page
21
22         driver.get("https://v1.training-support.net/selenium/tables");
23
24         // Print the title of the page
25         System.out.println("Home page title: " + driver.getTitle());
26
27         // Find the number of rows and columns in the first table using XPath
28         WebElement firstTable = driver.findElement(By.xpath("//table[@id='sortableTable']"));
29         int rowCount = firstTable.findElements(By.xpath("//tr")).size();
```

The status bar at the bottom indicates the file is 38:10, CRLF, UTF-8, 4 spaces, and the system status shows 85°F, Mostly cloudy, ENG IN, 22:41, and 02-05-2024.

```
11     public class activity13 {
12         public static void main(String[] args) {
13             // Find the number of rows and columns in the first table using XPath
14             WebElement firstTable = driver.findElement(By.xpath("//table[@id='sortableTable']"));
15             int rowCount = firstTable.findElements(By.xpath("./tr")).size();
16             int columnCount = firstTable.findElements(By.xpath("./tr[1]/td")).size();
17             System.out.println("Number of rows in the first table: " + rowCount);
18             System.out.println("Number of columns in the first table: " + columnCount);
19
20             // Find and print all cell values in the third row of the table
21             WebElement thirdRow = firstTable.findElement(By.xpath("./tr[3]"));
22             for (WebElement cell : thirdRow.findElements(By.tagName("td"))) {
23                 System.out.println("Cell value: " + cell.getText());
24             }
25
26             // Find and print the cell value at the second row, second column
27             WebElement cellValue = firstTable.findElement(By.xpath("./tr[2]/td[2]"));
28             System.out.println("Cell value at second row, second column: " + cellValue.getText());
29
30             WebElement button=driver.findElement(By.id("target"));
31             button.click();
32
33             // Close the browser
34             driver.quit();
35         }
36     }
```

85°F Mostly cloudy

Current File Version control

Project hiSelenium C:\Users\SBhavana\IdeaProjects\hiSelenium

activity8.java activity10.java activity11.java activity12.java activity13.java activity14.java activity15.java activity16.java activity17.java activity18.java activity19.java activity20.java activity21.java

activity13.java

38:10 CRLF UTF-8 4 spaces

22:41 02-05-2024

```
1 import org.openqa.selenium.firefox.FirefoxDriver;
2 import org.openqa.selenium.By;
3 import org.openqa.selenium.WebDriver;
4 import org.openqa.selenium.WebElement;
5 import org.openqa.selenium.firefox.FirefoxDriver;
6 import org.openqa.selenium.support.ui.WebDriverWait;
7 import java.time.Duration;
8 public class activity14 {
9     public static void main(String[] args) {
10
11     WebDriver driver = new FirefoxDriver();
12     WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
13     driver.get("https://v1.training-support.net/selenium/tables");
14     System.out.println("Home page");
15
16     WebElement secondTable = driver.findElement(By.xpath("//table[@id='secondTable']"));
17     int rowCount = secondTable.findElements(By.xpath("./tr")).size();
18     int columnCount = secondTable.findElements(By.xpath("./tr[1]/td")).size();
19     System.out.println("Number of rows in the second table: " + rowCount);
20     System.out.println("Number of columns in the second table: " + columnCount);
21
22     // Find and print the cell value at the second row, second column
23     WebElement cellValueBeforeSorting = secondTable.findElement(By.xpath("./tr[2]/td[2]"));
24     System.out.println("Cell value at second row, second column before sorting: " + cellValueBeforeSorting.getText());
25
26     // Click the header of the first column to sort by name
27     WebElement headerName = secondTable.findElement(By.xpath("//thead/tr[1]/th"));
28     headerName.click();
29 }
```

85°F Mostly cloudy

Current File Version control

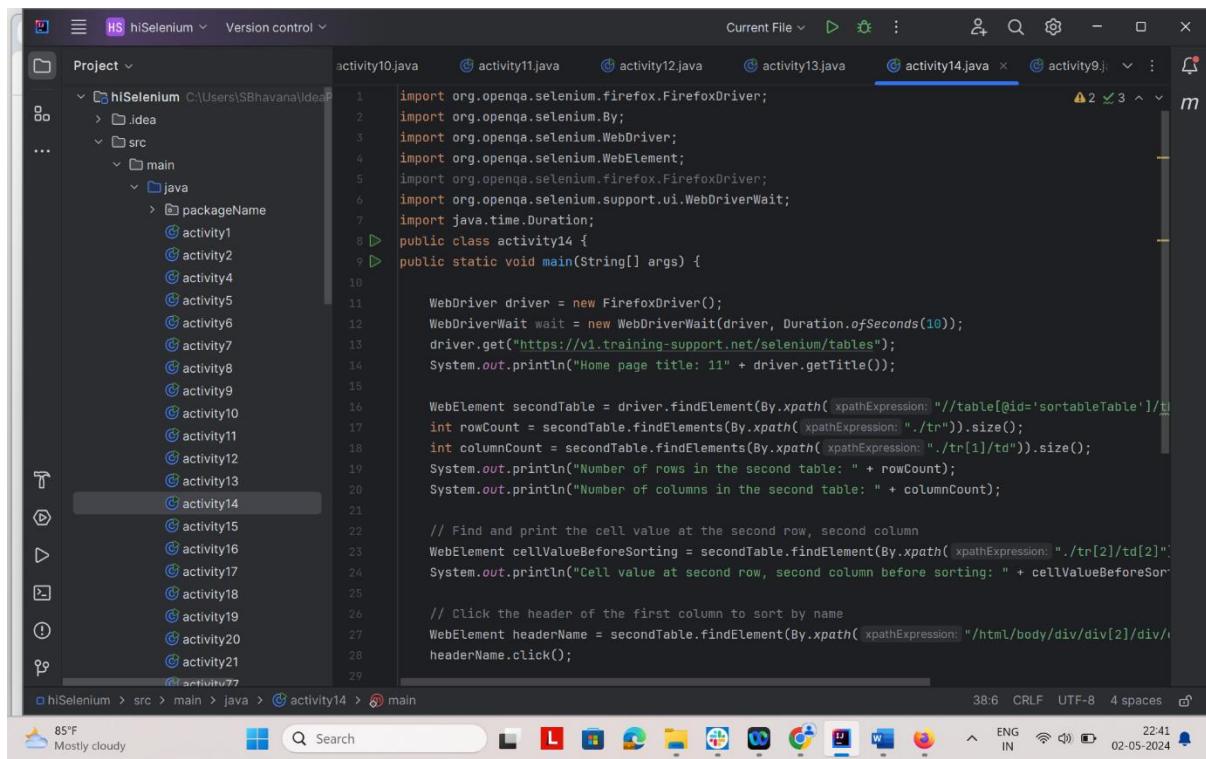
Project hiSelenium C:\Users\SBhavana\IdeaProjects\hiSelenium

activity10.java activity11.java activity12.java activity13.java activity14.java activity15.java activity16.java activity17.java activity18.java activity19.java activity20.java activity21.java

activity14.java

38:6 CRLF UTF-8 4 spaces

22:41 02-05-2024



```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.support.ui.WebDriverWait;
import java.time.Duration;

public class activity14 {
    public static void main(String[] args) {

        WebDriver driver = new FirefoxDriver();
        WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
        driver.get("https://v1.training-support.net/selenium/tables");
        System.out.println("Home page title: " + driver.getTitle());

        WebElement secondTable = driver.findElement(By.xpath("//table[@id='sortableTable']"));
        int rowCount = secondTable.findElements(By.xpath("./tr")).size();
        int columnCount = secondTable.findElements(By.xpath("./tr[1]/td")).size();
        System.out.println("Number of rows in the second table: " + rowCount);
        System.out.println("Number of columns in the second table: " + columnCount);

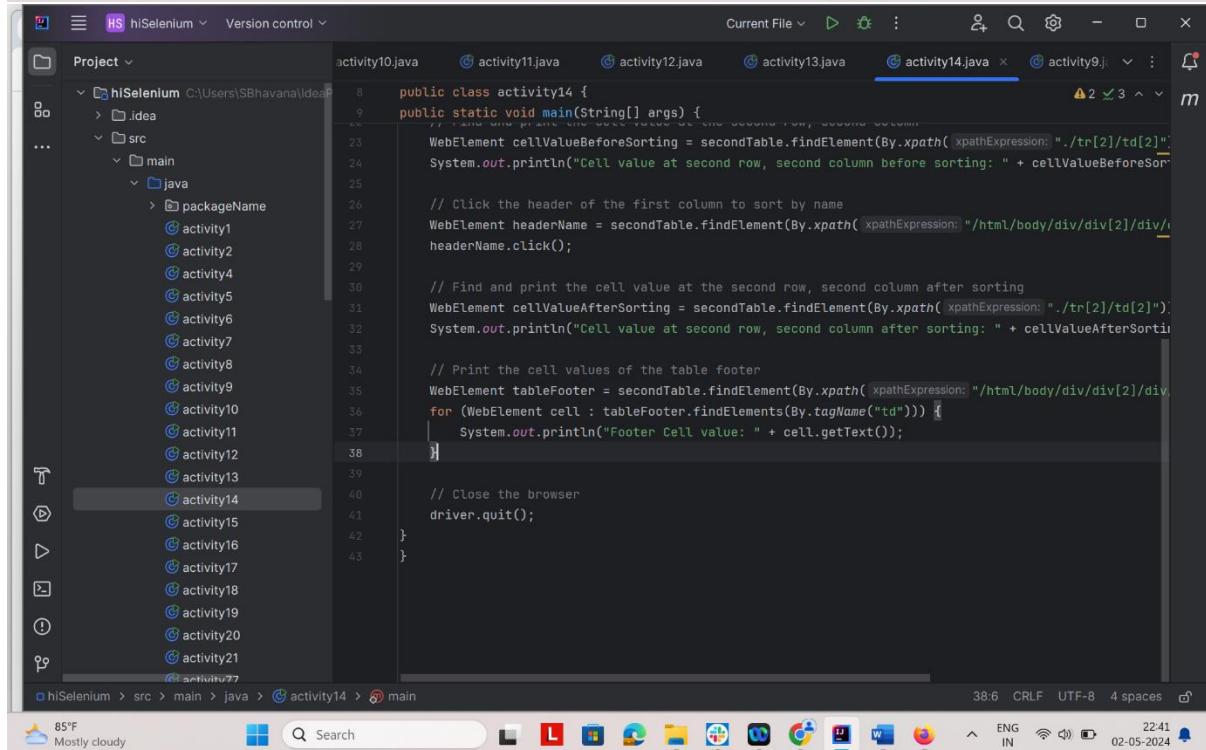
        // Find and print the cell value at the second row, second column
        WebElement cellValueBeforeSorting = secondTable.findElement(By.xpath("./tr[2]/td[2]"));
        System.out.println("Cell value at second row, second column before sorting: " + cellValueBeforeSorting.getText());

        // Click the header of the first column to sort by name
        WebElement headerName = secondTable.findElement(By.xpath("//html/body/div/div[2]/div/div[1]/table/thead/tr/th[1]"));
        headerName.click();

        // Find and print the cell value at the second row, second column after sorting
        WebElement cellValueAfterSorting = secondTable.findElement(By.xpath("./tr[2]/td[2]"));
        System.out.println("Cell value at second row, second column after sorting: " + cellValueAfterSorting.getText());

        // Print the cell values of the table footer
        WebElement tableFooter = secondTable.findElement(By.xpath("//html/body/div/div[2]/div/div[3]/table/tfoot/tr"));
        for (WebElement cell : tableFooter.findElements(By.tagName("td"))) {
            System.out.println("Footer Cell value: " + cell.getText());
        }

        // Close the browser
        driver.quit();
    }
}
```



```
public class activity14 {
    public static void main(String[] args) {

        WebElement cellValueBeforeSorting = secondTable.findElement(By.xpath("./tr[2]/td[2]"));
        System.out.println("Cell value at second row, second column before sorting: " + cellValueBeforeSorting.getText());

        // Click the header of the first column to sort by name
        WebElement headerName = secondTable.findElement(By.xpath("//html/body/div/div[2]/div/div[1]/table/thead/tr/th[1]"));
        headerName.click();

        // Find and print the cell value at the second row, second column after sorting
        WebElement cellValueAfterSorting = secondTable.findElement(By.xpath("./tr[2]/td[2]"));
        System.out.println("Cell value at second row, second column after sorting: " + cellValueAfterSorting.getText());

        // Print the cell values of the table footer
        WebElement tableFooter = secondTable.findElement(By.xpath("//html/body/div/div[2]/div/div[3]/table/tfoot/tr"));
        for (WebElement cell : tableFooter.findElements(By.tagName("td"))) {
            System.out.println("Footer Cell value: " + cell.getText());
        }

        // Close the browser
        driver.quit();
    }
}
```

The screenshot shows the IntelliJ IDEA interface with the project 'hiSelenium' open. The file 'activity15.java' is the active editor. The code implements a Selenium test for a dynamic attributes form. It includes imports for WebDriver, WebElement, and ExpectedConditions, and uses WebDriverWait to handle page load and element visibility.

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import java.time.Duration;

public class activity15 {
    public static void main(String[] args) {

        WebDriver driver = new FirefoxDriver();
        WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
        driver.get("https://v1.training-support.net/selenium/dynamic-attributes");
        System.out.println("Home page title: " + driver.getTitle());

        // Find username and password input fields and type credentials
        WebElement usernameInput = driver.findElement(By.xpath("//html/body/div[2]/div/div/username"));
        WebElement passwordInput = driver.findElement(By.xpath("//html/body/div[2]/div/div/div/password"));
        usernameInput.sendKeys("admin");
        passwordInput.sendKeys("password");

        // Click on the login button
        driver.findElement(By.xpath("//*[@id='dynamic-attributes-form']/div/button")).click();

        // Wait for the login message to appear
        WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
        WebElement loginMessage = wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("action")));
    }
}
```

This screenshot shows the same IntelliJ IDEA session with the code for 'activity15.java'. The code has been modified to include additional logic: after clicking the login button, it prints the login message to the console and then closes the browser. The code structure remains largely the same, with the addition of the print statement and the quit call.

```
public class activity15 {
    public static void main(String[] args) {

        // Find username and password input fields and type credentials
        WebElement usernameInput = driver.findElement(By.xpath("//html/body/div[2]/div/div/username"));
        WebElement passwordInput = driver.findElement(By.xpath("//html/body/div[2]/div/div/div/password"));
        usernameInput.sendKeys("admin");
        passwordInput.sendKeys("password");

        // Click on the login button
        driver.findElement(By.xpath("//*[@id='dynamic-attributes-form']/div/button")).click();

        // Wait for the login message to appear
        WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
        WebElement loginMessage = wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("action")));

        // Print the login message to the console
        System.out.println("Login message: " + loginMessage.getText());

        // Close the browser
        driver.quit();
    }
}
```

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import java.time.Duration;

public class activity16 {
    public static void main(String[] args) {

        WebDriver driver = new FirefoxDriver();
        WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
        driver.get("https://v1.training-support.net/selenium/dynamic-attributes");
        System.out.println("Home page title: " + driver.getTitle());

        // Find username and password input fields and type credentials
        WebElement usernameInput = driver.findElement(By.xpath("//html/body/div[2]/div/div/div[2]/div[1]/input"));
        WebElement usernameInput2 = driver.findElement(By.xpath("//html/body/div[2]/div/div/div[2]/div[1]/input"));
        WebElement passwordInput = driver.findElement(By.xpath("//html/body/div[2]/div/div/div[2]/div[1]/input"));
        WebElement passwordInput2 = driver.findElement(By.xpath("//html/body/div[2]/div/div/div[2]/div[1]/input"));
        WebElement confirmPassword = driver.findElement(By.xpath("//html/body/div[2]/div/div/div[2]/div[1]/input"));
        WebElement emailInput = driver.findElement(By.xpath("//html/body/div[2]/div/div/div[2]/div[1]/input"));

        usernameInput.sendKeys("admin");
        usernameInput2.sendKeys("S Bhavana");
        passwordInput.sendKeys("bhavana");
        passwordInput2.sendKeys("bhavana");
        confirmPassword.sendKeys("bhavana");
    }
}
```

```
public class activity16 {
    public static void main(String[] args) {

        passwordInput.sendKeys("bhavana");
        passwordInput2.sendKeys("bhavana");
        confirmPassword.sendKeys("bhavana");
        emailInput.sendKeys("bhavana.siddaraju18@gmail.com");

        // Click on the login button
        driver.findElement(By.xpath("//*[@id='dynamic-attributes-form']/div/button")).click();

        // Wait for the login message to appear
        WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
        WebElement loginMessage = wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("activity-login-msg")));

        // Print the login message to the console
        System.out.println("Login message: " + loginMessage.getText());

        WebElement button = driver.findElement(By.id("target"));
        button.click();

        // Close the browser
        driver.quit();
    }
}
```

The screenshot shows two instances of the IntelliJ IDEA IDE running side-by-side. Both instances have the same project structure and code editor open, displaying Java code for Selenium activities.

**Project Structure:**

- hiSelenium
- src
- main
- java
- packageName
- activity1
- activity2
- activity4
- activity5
- activity6
- activity7
- activity8
- activity9
- activity10
- activity11
- activity12
- activity13
- activity14
- activity15
- activity16
- activity17
- activity18
- activity19
- activity20
- activity21

**Code Editor (activity17.java):**

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.Select;
import org.openqa.selenium.support.ui.WebDriverWait;
import java.time.Duration;
import java.util.List;

public class activity17 {
    public static void main(String[] args) {

        WebDriver driver = new FirefoxDriver();
        WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
        driver.get("https://v1.training-support.net/selenium/selects");
        System.out.println("Home page title:" + driver.getTitle());

        // Find the single select dropdown
        WebElement singleSelect = driver.findElement(By.xpath("//*[@id='single-select']"));

        // Select the second option using visible text
        Select select = new Select(singleSelect);
        select.selectByVisibleText("Option 2");

        // Select the third option using index
        select.selectByIndex(2);

        // Select the fourth option using value
        select.selectByValue("4");

        // Get all the options and print them to the console
        List options = select.getOptions();
        System.out.println("Options:");
        for (WebElement option : options) {
            System.out.println(option.getText());
        }

        // Close the browser
        driver.quit();
    }
}
```

**System Tray and Status Bar:**

- 85°F Mostly cloudy
- Search bar
- Taskbar icons (L, A, E, etc.)
- ENG IN 22:51 02-05-2024

The screenshot shows two instances of an IDE (IntelliJ IDEA) displaying Java code for a Selenium project named 'hiSelenium'. The code is located in the 'src/main/java' package and is part of the 'activity18' class.

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.support.ui.Select;
import org.openqa.selenium.support.ui.WebDriverWait;
import java.util.List;

public class activity18 {
    public static void main(String[] args) {

        WebDriver driver = new FirefoxDriver();
        WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
        driver.get("https://vi.training-support.net/selenium/selects");
        System.out.println("Home page title:" + driver.getTitle());
        // Get the title of the page and print it to the console
        String title = driver.getTitle();
        System.out.println("Page Title: " + title);

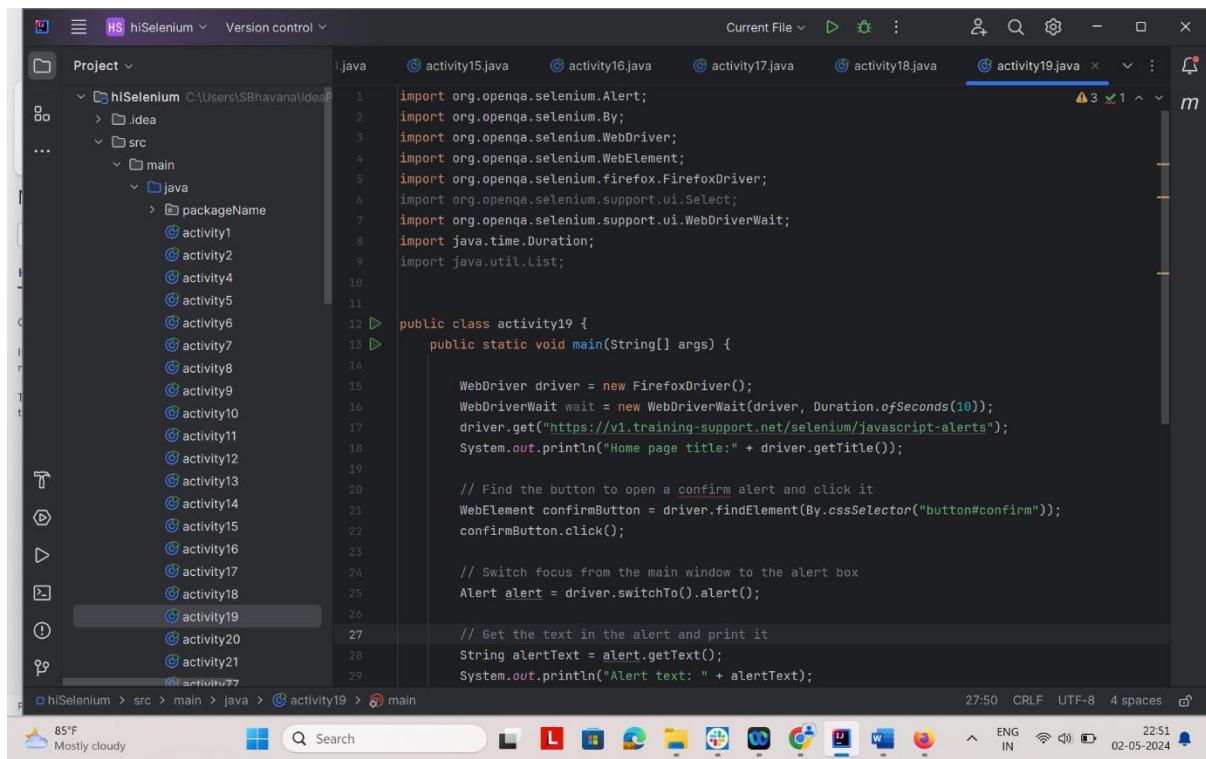
        // Locate the multi-select dropdown element
        WebElement multiSelect = driver.findElement(By.xpath("//*[@id='multi-select']"));

        // Initialize a Select object with the multi-select element
        Select select = new Select(multiSelect);

        // Select the "javascript" option using the visible text
        select.selectByVisibleText("Javascript");
    }
}
```

The code uses Selenium WebDriver to interact with a dropdown menu on a web page. It prints the page title and then selects the 'javascript' option from the dropdown.

The IDE interface includes a navigation bar with tabs for 'activity14.java', 'activity15.java', 'activity16.java', 'activity17.java', and 'activity18.java'. The status bar at the bottom shows the current file is 'activity18.java'. The system tray indicates the date is 02-05-2024 and the time is 22:51.



The screenshot shows the IntelliJ IDEA interface with the project 'hiSelenium' open. The code editor displays 'activity19.java' which contains Java code for interacting with a JavaScript alert using a Firefox driver. The code includes imports for Selenium packages, a main method that navigates to a URL containing a JavaScript alert, finds the confirm button, and prints its text.

```
import org.openqa.selenium.Alert;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.support.ui.Select;
import org.openqa.selenium.support.ui.WebDriverWait;
import java.time.Duration;
import java.util.List;

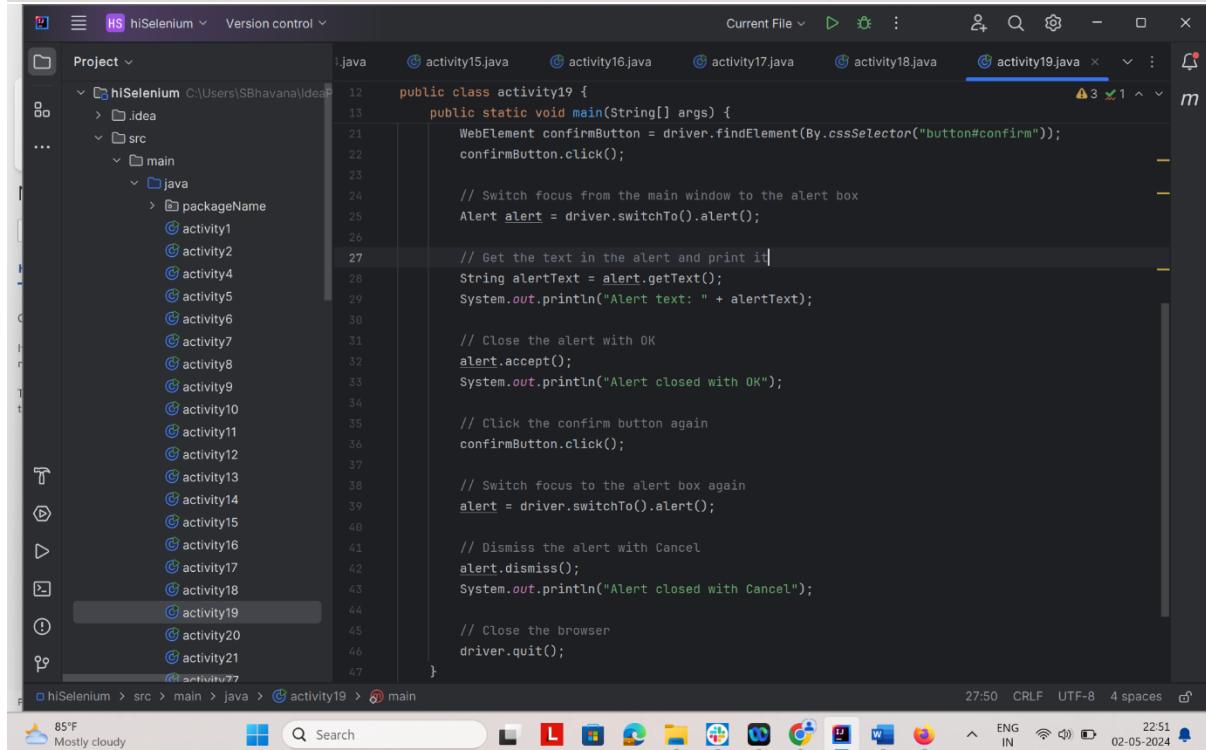
public class activity19 {
    public static void main(String[] args) {

        WebDriver driver = new FirefoxDriver();
        WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
        driver.get("https://v1.training-support.net/selenium/javascript-alerts");
        System.out.println("Home page title:" + driver.getTitle());

        // Find the button to open a confirm alert and click it
        WebElement confirmButton = driver.findElement(By.cssSelector("button#confirm"));
        confirmButton.click();

        // Switch focus from the main window to the alert box
        Alert alert = driver.switchTo().alert();

        // Get the text in the alert and print it
        String alertDialogText = alert.getText();
        System.out.println("Alert text: " + alertDialogText);
    }
}
```



The second screenshot shows the same IntelliJ IDEA environment with the code editor displaying 'activity19.java'. This version of the code handles the alert differently by switching to the alert box, getting the text, accepting it, and then switching back to the main window to dismiss the alert with 'Cancel' before finally closing the browser.

```
public class activity19 {
    public static void main(String[] args) {

        WebElement confirmButton = driver.findElement(By.cssSelector("button#confirm"));
        confirmButton.click();

        // Switch focus from the main window to the alert box
        Alert alert = driver.switchTo().alert();

        // Get the text in the alert and print it
        String alertDialogText = alert.getText();
        System.out.println("Alert text: " + alertDialogText);

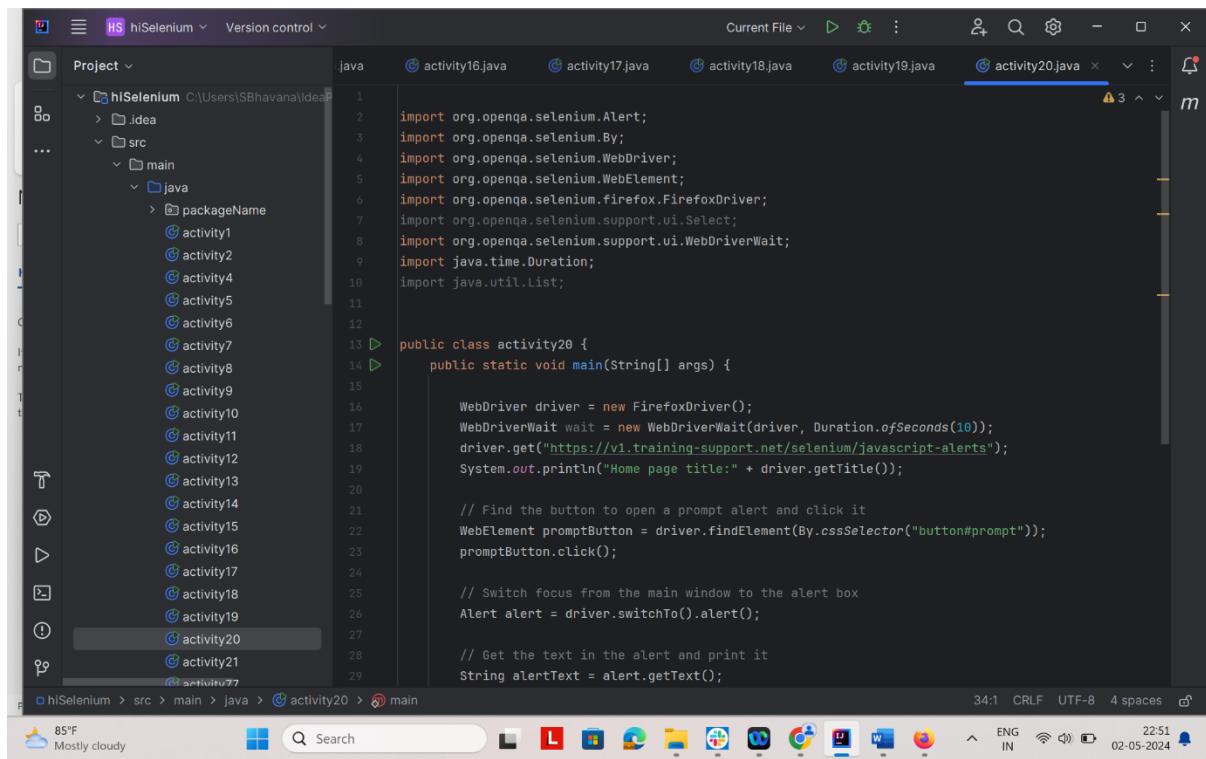
        // Close the alert with OK
        alert.accept();
        System.out.println("Alert closed with OK");

        // Click the confirm button again
        confirmButton.click();

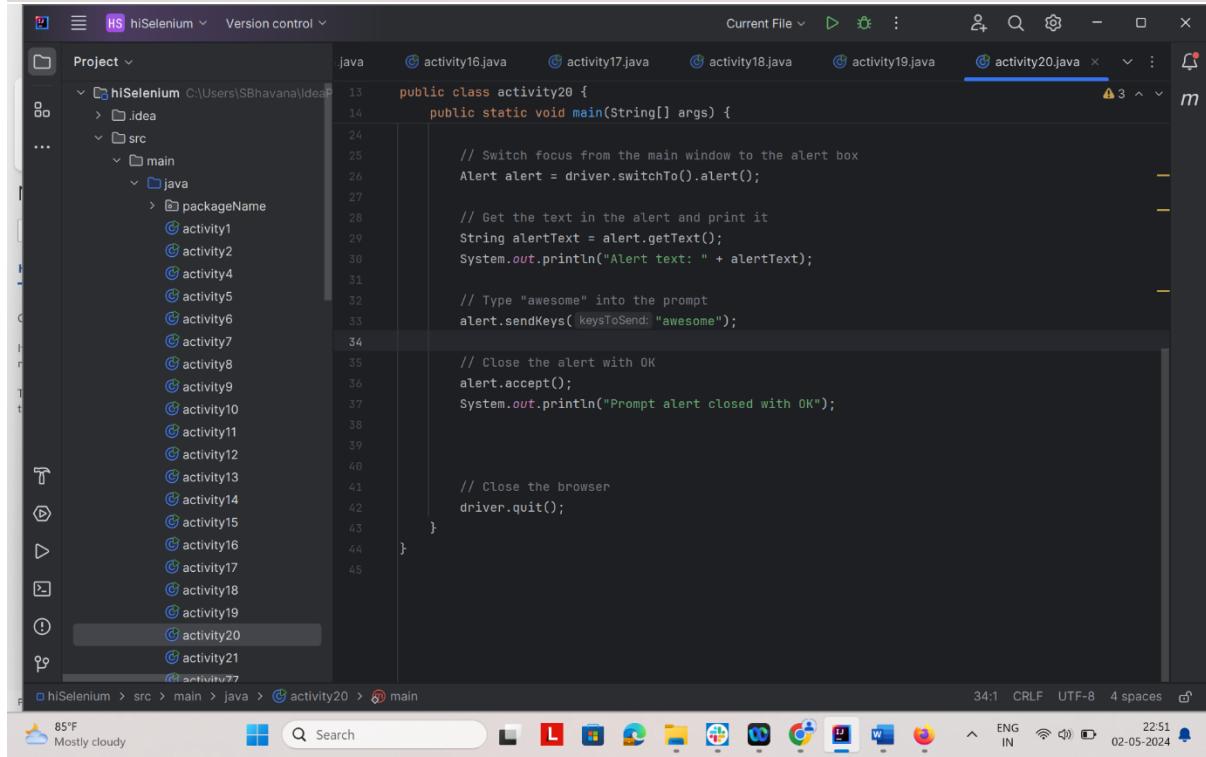
        // Switch focus to the alert box again
        alert = driver.switchTo().alert();

        // Dismiss the alert with Cancel
        alert.dismiss();
        System.out.println("Alert closed with Cancel");

        // Close the browser
        driver.quit();
    }
}
```



```
1 import org.openqa.selenium.Alert;
2 import org.openqa.selenium.By;
3 import org.openqa.selenium.WebDriver;
4 import org.openqa.selenium.WebElement;
5 import org.openqa.selenium.firefox.FirefoxDriver;
6 import org.openqa.selenium.support.ui.Select;
7 import org.openqa.selenium.support.ui.WebDriverWait;
8 import java.time.Duration;
9 import java.util.List;
10
11
12 public class activity20 {
13     public static void main(String[] args) {
14
15         WebDriver driver = new FirefoxDriver();
16         WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
17         driver.get("https://v1.training-support.net/selenium/javascript-alerts");
18         System.out.println("Home page title:" + driver.getTitle());
19
20         // Find the button to open a prompt alert and click it
21         WebElement promptButton = driver.findElement(By.cssSelector("button#prompt"));
22         promptButton.click();
23
24         // Switch focus from the main window to the alert box
25         Alert alert = driver.switchTo().alert();
26
27         // Get the text in the alert and print it
28         String alertText = alert.getText();
29         System.out.println("Alert text: " + alertText);
30
31         // Type "awesome" into the prompt
32         alert.sendKeys("awesome");
33
34         // Close the alert with OK
35         alert.accept();
36         System.out.println("Prompt alert closed with OK");
37
38
39         // Close the browser
40         driver.quit();
41     }
42
43 }
44
```



```
13 public class activity20 {
14     public static void main(String[] args) {
15
16         // Switch focus from the main window to the alert box
17         Alert alert = driver.switchTo().alert();
18
19         // Get the text in the alert and print it
20         String alertText = alert.getText();
21         System.out.println("Alert text: " + alertText);
22
23         // Type "awesome" into the prompt
24         alert.sendKeys("awesome");
25
26         // Close the alert with OK
27         alert.accept();
28         System.out.println("Prompt alert closed with OK");
29
30
31         // Close the browser
32         driver.quit();
33     }
34
35 }
```

The screenshot shows two instances of the IntelliJ IDEA IDE running side-by-side. Both instances have the same project structure and code editor open, displaying Java code for interacting with browser alerts.

**Project Structure:**

- hiSelenium
- .idea
- src
  - main
    - java
      - packageName
        - activity1
        - activity2
        - activity4
        - activity5
        - activity6
        - activity7
        - activity8
        - activity9
        - activity10
        - activity11
        - activity12
        - activity13
        - activity14
        - activity15
        - activity16
        - activity17
        - activity18
        - activity19
        - activity20
        - activity21

**Code Editor (activity21.java):**

```
import org.openqa.selenium.Alert;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.support.ui.Select;
import org.openqa.selenium.support.ui.WebDriverWait;
import java.time.Duration;
import java.util.List;

public class activity21 {
    public static void main(String[] args) {

        WebDriver driver = new FirefoxDriver();
        WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
        driver.get("https://v1.training-support.net/selenium/tab-opened");
        System.out.println("Home page title:" + driver.getTitle());

        // Find the button to open a prompt alert and click it
        WebElement promptButton = driver.findElement(By.cssSelector("button#prompt"));
        promptButton.click();

        // Switch focus from the main window to the alert box
        Alert alert = driver.switchTo().alert();

        // Get the text in the alert and print it
        String alertText = alert.getText();
        System.out.println("Alert text: " + alertText);

        // Type "awesome" into the prompt
        alert.sendKeys("awesome");

        // Close the alert with OK
        alert.accept();
        System.out.println("Prompt alert closed with OK");

        // Close the browser
        driver.quit();
    }
}
```

**System Tray and Status Bar:**

  - 85°F Mostly cloudy
  - Search bar
  - Icons for various applications (L, A, E, etc.)
  - ENG IN 22:51 02-05-2024

## TestNG activities

```
import org.junit.Assert;
import org.junit.BeforeClass;
import org.junit.Test;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.testng.annotations.AfterClass;
import io.github.bonigarcia.wdm.WebDriverManager;

public class test1 {
    // Declare the WebDriver object
    static WebDriver driver; 7 usages

    @BeforeClass
    public static void beforeMethod() {
        // Set up the Firefox driver
        WebDriverManager.firefoxdriver().setup();
        //Create a new instance of the Firefox driver
        driver = new FirefoxDriver();

        //Open browser
        driver.get("https://v1.training-support.net");
    }

    @Test
}
```

```
public class test1 {
    @Test
    public void exampleTestCase() {
        // Check the title of the page
        String title = driver.getTitle();

        //Print the title of the page
        System.out.println("Page title is: " + title);

        //Assertion for page title
        Assert.assertEquals(expected: "Training Support", title);

        //Find the clickable link on the page and click it
        driver.findElement(By.id("about-link")).click();

        //Print title of new page
        System.out.println("New page title is: " + driver.getTitle());

        Assert.assertEquals(driver.getTitle(), actual: "About Training Support");
    }

    @AfterClass
    public void afterMethod() {
        //Close the browser
        driver.quit();
    }
}
```

```
import org.junit.Assert;
import org.junit.BeforeClass;
import org.junit.Test;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.testng.annotations.AfterClass;
import io.github.bonigarcia.wdm.WebDriverManager;
import org.testng.annotations.BeforeTest;

public class test3 {
    static WebDriver driver; 7 usages

    @BeforeClass
    public static void beforeClass() {
        // Set up the Firefox driver
        WebDriverManager.firefoxdriver().setup();
        driver = new FirefoxDriver();

        //Open browser
        driver.get("https://v1.training-support.net/selenium/login-form");
    }

    @Test
    public void loginTest() {
        //Find the username and password fields
        WebElement username = driver.findElement(By.id("username"));
        WebElement password = driver.findElement(By.id("password"));

        //Enter credentials
        username.sendKeys(...keysToSend: "admin");
        password.sendKeys(...keysToSend: "password");

        //Click login
        driver.findElement(By.xpath("//button[text()='Log in']")).click();

        //Read login message
        String loginMessage = driver.findElement(By.id("action-confirmation")).getText();
        Assert.assertEquals(expected: "Welcome Back, admin", loginMessage);
    }

    @AfterClass
    public void afterClass() {
        //Close browser
        driver.close();
    }
}
```

```
import org.openqa.selenium.*;
import org.openqa.selenium.firefox.FirefoxDriver;
import io.github.bonigarcia.wdm.WebDriverManager;
import org.testng.Assert;
import org.testng.annotations.AfterTest;
import org.testng.annotations.BeforeTest;
import org.testng.annotations.Test;

public class test5 {
    WebDriver driver; 8 usages
    //Include alwaysRun property on the @BeforeTest
    //to make sure the page opens
    @BeforeTest(alwaysRun = true)
    public void beforeMethod() {
        // Set up the Firefox driver
        WebDriverManager.firefoxdriver().setup();
        //Create a new instance of the Firefox driver
        driver = new FirefoxDriver();

        //Open the browser
        driver.get("https://v1.training-support.net/selenium/target-practice");
    }

    @Test (groups = {"HeaderTests", "ButtonTests"})
    public void pageTitleTest() {
        String title = driver.getTitle();
        System.out.println("Title is: " + title);
        Assert.assertEquals(title, expected: "Target Practice");
    }
}
```

```
public class test5 {
    public void pageTitleTest() {
        Assert.assertEquals(title, expected: "Target Practice");
    }

    @Test (dependsOnMethods = {"pageTitleTest"}, groups = {"HeaderTests"})
    public void HeaderTest1() {
        WebElement header3 = driver.findElement(By.cssSelector("h3#third-header"));
        Assert.assertEquals(header3.getText(), expected: "Third header");
    }

    @Test (dependsOnMethods = {"pageTitleTest"}, groups = {"HeaderTests"})
    public void HeaderTest2() {
        WebElement header5 = driver.findElement(By.cssSelector("h3#third-header"));
        Assert.assertEquals(header5.getCssValue( propertyName: "color"), expected: "rgb(251, 189, 8)");
    }

    @Test (dependsOnMethods = {"pageTitleTest"}, groups = {"ButtonTests"})
    public void ButtonTest1() {
        WebElement button1 = driver.findElement(By.cssSelector("button.olive"));
        Assert.assertEquals(button1.getText(), expected: "Olive");
    }

    @Test (dependsOnMethods = {"pageTitleTest"}, groups = {"ButtonTests"})
    public void ButtonTest2() {
        WebElement button2 = driver.findElement(By.cssSelector("button.brown"));
        Assert.assertEquals(button2.getCssValue( propertyName: "color"), expected: "rgb(255, 255, 255)");
    }
}

@AfterTest(alwaysRun = true)
```

```
public class test5 {
    public void beforeMethod() {
        WebElement header5 = driver.findElement(By.cssSelector("h3#third-header"));
        Assert.assertEquals(header5.getCssValue("color"), expected: "rgb(251, 189, 8)");
    }

    @Test (dependsOnMethods = {"pageTitleTest"}, groups = {"ButtonTests"})
    public void ButtonTest1() {
        WebElement button1 = driver.findElement(By.cssSelector("button.olive"));
        Assert.assertEquals(button1.getText(), expected: "Olive");
    }

    @Test (dependsOnMethods = {"pageTitleTest"}, groups = {"ButtonTests"})
    public void ButtonTest2() {
        WebElement button2 = driver.findElement(By.cssSelector("button.brown"));
        Assert.assertEquals(button2.getCssValue("color"), expected: "rgb(255, 255, 255"));
    }

    @AfterTest(alwaysRun = true)
    public void afterMethod() {
        //Close the browser
        driver.close();
    }
}
```

```
import io.github.bonigarcia.wdm.WebDriverManager;
import org.junit.BeforeClass;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.testng.annotations.*;

public class test6 {
    // Declare the WebDriver object
    static WebDriver driver; no usages

    @BeforeClass
    public static void beforeClass() {
        System.out.println("before class is executed");
    }

    @BeforeSuite
    public static void beforeSuite(){
        System.out.println("before suite is executed");
    }

    @AfterClass
    public static void afterClass(){
        System.out.println("After class is executed");
    }

    @AfterSuite
    public static void afterSuite(){
        System.out.println("After suite is executed");
    }
}
```

The screenshot displays two instances of a Java IDE interface, likely Eclipse or a similar environment, running on a Windows operating system. Both instances show the same project structure and codebase, with different files selected in each.

**Project Structure:**

- hiSelenium
- src
- test
- java
- subtest9
- dataprovider
- suiteexample
- test1
- test3
- test5
- test6 (selected in both panes)
- test7
- test8
- test9
- test11
- target
- classes
- generated-sources
- generated-test-sources
- test-classes
- test-output
- Default Suite
- junitreports
- bullet\_point.png
- collapseall.gif
- emailable-report.html
- failed.png
- index.html
- inquiry-3.0-minis

**Code Editor 1 (test6.java):**

```
public class test6 {  
    public static void beforeClass() {  
        System.out.println("beforeClass is executed");  
    }  
    @Test  
    public static void test() {  
        System.out.println("Test is executed");  
    }  
    @BeforeTest  
    public static void beforeTest() {  
        System.out.println("BeforeTest is executed");  
    }  
    @AfterTest  
    public static void afterTest() {  
        System.out.println("AfterTest is executed");  
    }  
    @BeforeMethod  
    public static void beforeMethod() {  
        System.out.println("beforeMethod is executed");  
    }  
    @AfterMethod  
    public static void afterMethod() {  
        System.out.println("afterMethod is executed");  
    }  
}
```

**Code Editor 2 (test7.java):**

```
import org.openqa.selenium.*;  
import org.openqa.selenium.firefox.FirefoxDriver;  
import io.github.bonigarcia.wdm.WebDriverManager;  
import org.testng.annotations.*;  
  
public class test7{  
    WebDriver driver; 7 usages  
    @BeforeTest  
    public void beforeMethod() {  
        WebDriverManager.firefoxdriver().setup();  
        driver = new FirefoxDriver();  
        driver.get("https://v1.training-support.net/selenium/login-form");  
    }  
  
    @Test  
    @Parameters({"sUsername", "sPassword"})  
    public void test(String susername, String sPassword){  
        WebElement usernameField = driver.findElement(By.xpath("//*[@id='username']"));  
        usernameField.sendKeys(sUsername);  
        WebElement passwordField = driver.findElement(By.xpath("//*[@id='password']"));  
        passwordField.sendKeys(sPassword);  
        WebElement loginbutton = driver.findElement(By.cssSelector("button[type = 'submit']"));  
        loginbutton.click();  
    }  
    @AfterMethod  
    public void teardown() {  
        driver.quit();  
    }  
}
```

The screenshot shows a Java IDE interface with the project structure on the left and the code editor on the right. The code editor displays the file `test7.java`. The code implements a test class named `test7` with methods `test` and `teardown`.

```
public class test7{  
    @Test  
    public void test(String sUsername, String sPassword){  
        WebElement usernameField = driver.findElement(By.xpath("//*[@id='username']"));  
        usernameField.sendKeys(sUsername);  
        WebElement passwordField = driver.findElement(By.xpath("//*[@id='password']"));  
        passwordField.sendKeys(sPassword);  
        WebElement loginbutton = driver.findElement(By.cssSelector("button[type = 'submit']"));  
        loginbutton.click();  
    }  
    @AfterMethod  
    public void teardown(){  
        if (driver!= null){  
            driver.close();  
        }  
    }  
}
```

The screenshot shows a Java IDE interface with the project structure on the left and the code editor on the right. The code editor displays the file `test8.java`. The code implements a test class named `test8` with methods `beforeMethod`, `demo`, and `test`.

```
import org.openqa.selenium.*;  
import org.openqa.selenium.firefox.FirefoxDriver;  
import io.github.bonigarcia.wdm.WebDriverManager;  
import org.testng.annotations.*;  
  
public class test8{  
    WebDriver driver; 7 usages  
    @BeforeTest  
    public void beforeMethod(){  
        WebDriverManager.firefoxdriver().setup();  
        driver = new FirefoxDriver();  
        driver.get("https://v1.training-support.net/selenium/login-form");  
    }  
    @AfterMethod  
    public void demo(){  
        if (driver!= null){  
            driver.close();  
        }  
    }  
    @DataProvider(name = "authentication")  
    public Object[][] credentials(){  
        return new Object[][]{{"admin", "password"}};  
    }  
    @TestdataProvider = "authentication"  
    public void test(String Username, String Password){  
        WebElement usernameField = driver.findElement(By.xpath("//*[@id='username']"));  
        System.out.println("username of the user:" + usernameField);  
        usernameField.sendKeys(Username);  
        WebElement passwordField = driver.findElement(By.xpath("//*[@id='password']"));  
    }  
}
```

The screenshot shows a Java project structure in the left sidebar under the 'Project' tab. The 'test' package contains several files: test1.java, test3.java, test5.java, test6.java, test7.java, test8.java (which is currently selected), test9.java, and test11.java. The 'target' folder contains generated classes, sources, and test classes. The 'test-output' folder includes Default Suite, Junitreports, bullet\_point.png, collapseall.gif, emailable-report.html, failed.png, index.html, and a coverage report. The status bar at the bottom indicates the file is 0 minis long, the time is 16:29, and the encoding is CRLF UTF-8 4 spaces.

```
public class test8{  
    public void beforeMethod() {  
        driver.get("https://v1.training-support.net/selenium/login-form");  
    }  
    @AfterMethod  
    public void demo() {  
        if (driver!= null) {  
            driver.close();  
        }  
    }  
    @DataProvider(name = "authentication")  
    public Object[][] credentials() {  
        return new Object[][]{{"admin", "password"}};  
    }  
    @Test(dataProvider = "authentication")  
    public void test(String Username, String Password){  
        WebElement usernameField = driver.findElement(By.xpath("//*[@id='username']"));  
        System.out.println("username of the user:" + usernameField);  
        usernameField.sendKeys(Username);  
        WebElement passwordField = driver.findElement(By.xpath("//*[@id='password']"));  
        passwordField.sendKeys>Password);  
        WebElement loginbutton = driver.findElement(By.cssSelector("button[type = 'submit']"));  
        loginbutton.click();  
    }  
}
```

The screenshot shows a Java project structure in the left sidebar under the 'Project' tab. The 'test' package contains several files: test1.java, test3.java, test5.java, test6.java, test7.java, test8.java, test9.java (which is currently selected), and test11.java. The 'target' folder contains generated classes, sources, and test classes. The 'test-output' folder includes Default Suite, Junitreports, bullet\_point.png, collapseall.gif, emailable-report.html, failed.png, index.html, and a coverage report. The status bar at the bottom indicates the file is 0 minis long, the time is 16:29, and the encoding is CRLF UTF-8 4 spaces.

```
import io.github.bonigarcia.wdm.WebDriverManager;  
import org.openqa.selenium.By;  
import org.openqa.selenium.Keys;  
import org.openqa.selenium.WebDriver;  
import org.openqa.selenium.WebElement;  
import org.openqa.selenium.firefox.FirefoxDriver;  
import org.openqa.selenium.interactions.Actions;  
import org.testng.annotations.BeforeClass;  
import org.testng.annotations.DataProvider;  
import org.testng.annotations.Parameters;  
import org.testng.annotations.Test;  
import subtest9.dataprovider;  
  
import java.time.Duration;  
  
public class test9 {  
    WebDriver driver; 4 usages  
  
    @BeforeClass  
    public void beforeMethod() {  
        // Set up the Firefox driver  
        WebDriverManager.firefoxdriver().setup();  
        //Create a new instance of the Firefox driver  
        driver = new FirefoxDriver();  
  
        //Open browser  
        driver.get("https://v1.training-support.net/selenium/simple-form");  
    }  
}
```

```
public class test9 {
    public void test(String firstname, String lastname, String email, String number, String message) {
        WebElement fname=driver.findElement(By.xpath("//*[@id='firstName']"));
        new Actions(driver)
            .click(fname)
            .sendKeys(firstname)
            .sendKeys(Keys.TAB)
            .sendKeys(lastname)
            .sendKeys(Keys.TAB)
            .sendKeys(email)
            .sendKeys(Keys.TAB)
            .sendKeys(number)
            .sendKeys(Keys.TAB)
            .sendKeys(message)
            .sendKeys(Keys.TAB)
            .sendKeys(Keys.ENTER)
            .sendKeys(Keys.TAB)
            .perform();
    }
}
```

```
import org.openqa.selenium.Alert;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.testng.Reporter;
import org.testng.annotations.*;
import io.github.bonigarcia.wdm.WebDriverManager;

public class test11 {
    WebDriver driver; 10 usages

    @BeforeMethod
    public void beforeMethod() {
        WebDriverManager.firefoxdriver().setup();
        driver = new FirefoxDriver();
        driver.get("https://v1.training-support.net/selenium/javascript-alerts");
        driver.switchTo().defaultContent();
    }

    @Test
    public void simpleAlertTestCase() {
        driver.findElement(By.xpath("//*[@id='simple']")).click();
        Alert alert=driver.switchTo().alert();
        Reporter.log("Text in Simple Alert" + alert.getText());
        alert.accept();
    }
}
```

The screenshot shows a Java IDE interface with the following details:

- Project View:** On the left, the project structure is displayed under the "Project" tab. It includes a "test" package containing several test classes (e.g., test1, test2, test3, test4, test5, test6, test7, test8, test9, test11) and a "target" folder containing generated files like "classes", "generated-sources", "generated-test-sources", and "test-classes".
- Code Editor:** The main area shows the content of the file "test11.java". The code implements a class named "test11" with methods for handling confirm and prompt alerts.

```
public class test11 {
    @Test
    public void confirmAlertTestCase() {
        driver.findElement(By.xpath("//*[@id='confirm']")).click();
        Alert alert=driver.switchTo().alert();
        Reporter.log("Text in Confirm Alert"+alert.getText());
        alert.accept();
    }
    @Test
    public void promptAlertTestCase() {
        driver.findElement(By.xpath("//*[@id='prompt']")).click();
        Alert alert=driver.switchTo().alert();
        alert.sendKeys("HI");
        Reporter.log("Text in Prompt Alert"+alert.getText());
        alert.accept();
    }
    @AfterMethod
    public void afterMethod() {
        driver.quit();
    }
}
```

- Toolbars and Status Bar:** The top bar includes tabs for "st3.java", "test5.java", "test6.java", "test7.java", "test8.java", "test9.java", and "test11.java". The status bar at the bottom shows the file path ("hiSelenium > src > test > java > test11"), encoding ("CRLF"), character set ("UTF-8"), and other system information.
- System Tray:** The bottom right corner shows the Windows taskbar with icons for weather (85°F), search, and various system applications.