Selenium WebDriver Syntax Pool

Navigate to Website

```
driver.get("https://www.example.com");
```

Get the Title of a Page

```
String title = driver.getTitle();
```

Get the URL of the Page

```
String currentUrl = driver.getCurrentUrl();
```

Navigate to Another Page

```
driver.navigate().to("https://www.example.com/another-page");
```

Capture the Page Header

```
String header = driver.findElement(By.tagName("h1")).getText();
 // //Adjust tag as necessary
```

Clear the Field

```
WebElement field = driver.findElement(By.id("inputFieldId")); //
Replace with actual ID
field.clear();
```

Add Input to the Field

```
WebElement field = driver.findElement(By.id("inputFieldId")); //
Replace with actual ID
field.sendKeys("Sample Input");
```

Button Click

```
WebElement button = driver.findElement(By.id("submitButtonId")); // Replace with actual ID
button.click( );
```

Capture and Return the Entered Text in a Field

```
String enteredText =driver.findElement(By.id("inputFieldId")).getAttribute("value");
```

Capture and Return All Items on a Page

```
List<WebElement> items = driver.findElements(By.className("item-class")); // Adjust class name as necessary
List<String> itemTexts = new ArrayList<>();
for (WebElement item : items)
{
        itemTexts.add(item.getText());
}
```

Return the Tag Name

```
String tagName = driver.findElement(By.id("elementId")).getTagName();
// Replace with actual ID
```

Click WebElements

```
WebElement element = driver.findElement(By.id("clickableElementId")); // Replace with actual ID
element.click();
```

Upload File

```
WebElement uploadField = driver.findElement(By.id("fileUploadId"));
// Replace with actual ID
uploadField.sendKeys("path/to/file.txt"); // Full path to the file
```

Capture Screenshots

```
File screenshot =
((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);
FileUtils.copyFile(screenshot, new File("screenshot.png"));
```

Handle Dropdown Elements

```
Select dropdown = new
select(driver.findElement(By.id("dropdownId")));
 // Replace with actual ID
dropdown.selectByVisibleText("Option Text"); // or
selectByValue("OptionValue")
```

Check Boxes

```
WebElement checkbox = driver.findElement(By.id("checkboxId"));
 // Replace with actual ID
if (!checkbox.isSelected())
{    checkbox.click(); }
```

Radio Button Selection

```
WebElement radioButton =
driver.findElement(By.id("radioButtonId"));
// Replace with actual ID
if (!radioButton.isSelected())
{
 radioButton.click();
}
```

Retrieve the Title Again

```
String newTitle = driver.getTitle();
```

Retrieve Data Using ArrayList

```
List<String> dataList = new ArrayList<>();
List<WebElement> dataElements
=driver.findElements(By.className("data-class"));
// Adjust class name as necessary
for (WebElement data : dataElements)
{
        dataList.add(data.getText( ) );
}
```

Retrieve All Elements of a Page

```
List<WebElement> allElements = driver.findElements(By.xpath("//*"));
      // Retrieves all elements on the page.
```

Storing Data to Web Element (Example: Storing Text)

```
String storedText = "Sample Text";
WebElement textField = driver.findElement(By.id("textFieldId"));
textField.sendKeys(storedText);
```

Find the Largest Number on a Page (Assuming numbers are in elements)

```
List<WebElement> numberElements =
driver.findElements(By.className("number-class"));
// Adjust class name as necessary.
double largestNumber = Double.MIN_VALUE; // Initialize to smallest
possible value.
```

```
for (WebElement number : numberElements)
 {
    double value = Double.parseDouble(number.getText());
    if (value > largestNumber)
          {
           largestNumber = value;
          }
}
```

Find the Smallest Value on a Page (Similar to Largest)

```
double smallestNumber = Double.MAX_VALUE;
 // Initialize to the largest possible value.

for (WebElement number : numberElements)
 {
    double value = Double.parseDouble(number.getText());
    if (value < smallestNumber)
          {
          smallestNumber = value;
          }
}
```

Retrieve/Display Entered Values on a Page

```
String enteredValue =
driver.findElement(By.id("inputFieldId")).getAttribute("value"); //
Retrieve entered value.
System.out.println("Entered Value: " + enteredValue);
```

Page Navigation (Back and Forward)

```
driver.navigate().back();  // Go back to the previous page.
driver.navigate().forward();  // Go forward to the next page.
```

Handle Alerts

```
Alert alert = driver.switchTo().alert();  // Switch to alert.
alert.accept();  // Accepts the alert.
alert.dismiss();  // Dismisses the alert.
```

Implicit Waits

```
driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);  // Set implicit wait time.
```

Explicit Waits

```
WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(10));
wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("elementId")));  // Example of explicit wait.
```

Thread.sleep Method

```
Thread.sleep(2000);  // Pause execution for 2 seconds.
```

Testing the Title Using Assert (JUnit Example)

```
Assert.assertEquals(driver.getTitle(), "Expected Title");  // Check if title matches expected title.
```

Print Data of a Page Using ArrayList

```
List<String> printDataList = new ArrayList<>();
List<WebElement> elementsToPrint = driver.findElements(By.className("print-class"));

for (WebElement element : elementsToPrint) {
    printDataList.add(element.getText());
    System.out.println(element.getText());  // Print each element's text.
}
```

Getting the URL of Current Page Again

```
String currentPageUrl = driver.getCurrentUrl();
System.out.println(currentPageUrl);
```

Scroll Using JavaScriptExecutor

```
JavascriptExecutor jsExecutor = (JavascriptExecutor)driver;
jsExecutor.executeScript("window.scrollBy(0,250);");  // Scroll down by 250 pixels.
```

Maximize Browser Window

```
driver.manage().window().maximize();
```

Close the Browser

```
driver.close(); // Closes the current window.
```

Quit the Browser

```
driver.quit(); // Closes all browser windows and ends the WebDriver session.
```

Navigate Backward in Browser History

```
driver.navigate().back();
```

Navigate Forward in Browser History

```
driver.navigate().forward();
```

Refresh the Page

```
driver.navigate().refresh();
```

Switch to a Different Window

```
String originalWindow = driver.getWindowHandle();
        for (String windowHandle : driver.getWindowHandles()) {
            if (!originalWindow.equals(windowHandle)) {
                driver.switchTo().window(windowHandle);
                break;
            } }
```

Switch to a Frame

```
driver.switchTo().frame("frameNameOrId");
// Switch by name or ID or by index
driver.switchTo().frame(0); // Switch to the first frame
```

Switch Back from Frame to Default Content

```
driver.switchTo().defaultContent(); // Switch back to the main document
```

Drag and Drop Action

```
Actions actions = new Actions(driver);
WebElement source = driver.findElement(By.id("sourceElementId"));
WebElement target = driver.findElement(By.id("targetElementId"));
actions.dragAndDrop(source, target).perform();
```

Perform Mouse Hover Action

```
Actions actions = new Actions(driver);

WebElement elementToHover =
driver.findElement(By.id("hoverElementId"));

actions.moveToElement(elementToHover).perform();
```

Get Text from an Element

```
String text = driver.findElement(By.id("elementId")).getText();

// Replace with actual ID
```

Get Attribute Value from an Element

```
String value =
driver.findElement(By.id("elementId")).getAttribute("attributeName");

 // Replace with actual ID and attribute
```

Check if an Element is Displayed

```
boolean isDisplayed =
driver.findElement(By.id("elementId")).isDisplayed();
```

Check if an Element is Enabled

```
boolean isEnabled =
driver.findElement(By.id("elementId")).isEnabled();
```

Check if an Element is Selected (for checkboxes/radio buttons)

```
boolean isSelected =
driver.findElement(By.id("checkboxId")).isSelected();
 // Replace with actual ID
```

Handling Alerts (Accept/Dismiss)

```
Alert alert = driver.switchTo().alert(); // Switch to alert.
alert.accept(); // Accepts the alert.
alert.dismiss(); // Dismisses the alert.
```