Selenium Page Scrolling

In Selenium with Java, **page scrolling** is often required to interact with elements that are not initially in view. Selenium itself doesn't have native methods to scroll, so JavaScript execution via `JavascriptExecutor` is commonly used.

🔸 Techniques for Page Scrolling in Selenium Java

✅ 1. Scroll by Pixels

Scrolls the page by a specific number of pixels vertically (positive for down, negative for up).

```
java

import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class ScrollByPixels {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.get("https://example.com");

        JavascriptExecutor js = (JavascriptExecutor) driver;

        // Scroll down by 500 pixels
        js.executeScript("window.scrollBy(0, 500)");

        // Scroll up by 500 pixels
        js.executeScript("window.scrollBy(0, -500)");

        driver.quit();
    }
}
```

🔍 **Explanation**:

- `window.scrollBy(x, y)` scrolls relative to the current position.
- `(0, 500)` means no horizontal scroll, and scroll down 500 pixels.

✅ 2. Scroll to Element (Element in View)

Scrolls the page until a specific element is visible.

```
java

import org.openqa.selenium.*;
import org.openqa.selenium.chrome.ChromeDriver;

public class ScrollToElement {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.get("https://example.com");

        WebElement element = driver.findElement(By.id("targetElement"));
```

```
        JavascriptExecutor js = (JavascriptExecutor) driver;
        js.executeScript("arguments[0].scrollIntoView(true);", element);

        driver.quit();
    }
}
```

🔍 **Explanation**:

- **`scrollIntoView(true)`** scrolls the page so that the element is aligned to the top of the viewport.

✅ 3. Scroll to Bottom of Page

Scrolls the page to the bottom (useful for lazy loading or infinite scroll).

```
java

JavascriptExecutor js = (JavascriptExecutor) driver;
js.executeScript("window.scrollTo(0, document.body.scrollHeight);");
```

🔍 **Explanation**:

- **`document.body.scrollHeight`** returns the height of the entire page.
- Scrolling to this value brings the user to the bottom.

✅ 4. Scroll to Top of Page

```
java

JavascriptExecutor js = (JavascriptExecutor) driver;
js.executeScript("window.scrollTo(0, 0);");
```

🔍 **Explanation**:

- Scrolls to the top-left of the page.

✅ 5. Scroll Horizontally

```
java

js.executeScript("window.scrollBy(300, 0);");
```

🔍 **Explanation**:

- Scrolls 300 pixels right. Replace with **`-300`** for left scroll.

✅ 6. Infinite Scroll (Dynamic Load)

Used in pages where content loads as the user scrolls.

```
java

long lastHeight = (long) js.executeScript("return document.body.scrollHeight");

while (true) {
    js.executeScript("window.scrollTo(0, document.body.scrollHeight);");
    Thread.sleep(2000); // wait for page load

    long newHeight = (long) js.executeScript("return document.body.scrollHeight");
    if (newHeight == lastHeight) {
        break;
    }
    lastHeight = newHeight;
}
```

🔍 **Explanation**:

- Scrolls down, waits for new content to load, checks if scroll height changed.
- If height is unchanged, scrolling has reached the end.

📝 Notes

- Always use **`Thread.sleep()`** wisely — prefer **`WebDriverWait`** when possible.
- Scrolling may not work on some pages due to CSS or JavaScript restrictions.

Would you like a downloadable demo project or JUnit/TestNG version of these examples?