Standard Keyword Signature

```
await web.keywordName(selector?, value?, failureHandling?, timeout?);
```

Parameters

- **selector** → XPath / CSS / Playwright selector
- **value / text / data** → Input value or expected value
- **failureHandling (optional)**
    - `STOP_ON_FAILURE` (default)
    - `CONTINUE_ON_FAILURE`
    - `OPTIONAL`
- **timeout (optional)**
    - Overrides global timeout (milliseconds)

🔹 Browser / Navigation Keywords

navigate(url)

Navigates to the specified URL.

```
await web.navigate('https://example.com');
```

---

refresh()

Refreshes the current page.

```
await web.refresh();
```

---

back()

Navigates back in browser history.

```
await web.back();
```

---

forward()

Navigates forward in browser history.

```
await web.forward();
```

---

switchToWindowIndex(index)

Switches focus to a browser window by index (0-based).

```
await web.switchToWindowIndex(1);
```

---

switchToDefaultContent()

Switches focus back to the main page from iframe.

```
await web.switchToDefaultContent();
```

---

switchToFrame(selectorOrName)

Switches focus to iframe using selector or frame name.

```
await web.switchToFrame('//iframe[@id="frame1"]');
```

🔹 Element Interaction Keywords

click(selector)

Clicks an element.

```
await web.click('//button[@type="submit"]');
```

---

doubleClick(selector)

Double-clicks an element.

```
await web.doubleClick('//div[@id="box"]');
```

---

rightClick(selector)

Right-clicks an element.

```
await web.rightClick('//div[@id="menu"]');
```

---

setText(selector, text)

Sets text into an input field.

```
await web.setText('//input[@name="username"]', 'Admin');
```

---

clearText(selector)

Clears text from an input field safely.

```
await web.clearText('//input[@name="username"]');
```

---

uploadFile(selector, filePath)

Uploads a file to an input[type=file].

```
await web.uploadFile('//input[@type="file"]', 'files/test.pdf');
```

---

hover(selector)

Hovers mouse over an element.

```
await web.hover('//div[@class="menu"]');
```

---

check(selector)

Checks a checkbox only if not already checked.

```
await web.check('//input[@type="checkbox"]');
```

---

uncheck(selector)

Unchecks a checkbox only if already checked.

```
await web.uncheck('//input[@type="checkbox"]');
```

---

dragAndDrop(sourceSelector, targetSelector)

Drags source element and drops it on target.

```
await web.dragAndDrop('//div[@id="src"]', '//div[@id="dest"]');
```

🔹 Get Info Keywords

getElementAttribute(selector, attributeName)

Returns the value of an element attribute.

```
const value = await web.getElementAttribute('//input', 'value');
```

---

getText(selector)

Returns visible text of an element.

```
const text = await web.getText('//h1');
```

---

getUrl()

Returns the current page URL.

```
const url = await web.getUrl();
```

---

getTitle()

Returns page title.

```
const title = await web.getTitle();
```

---

getWindowIndex()

Returns index of current browser window.

```
const index = await web.getWindowIndex();
```

🔹 Verification Keywords

verifyElementVisible(selector)

Verifies element is visible.

```
await web.verifyElementVisible('//button');
```

---

verifyEqual(expected, actual)

Verifies two values are equal.

```
await web.verifyEqual('Admin', username);
```

---

verifyMatch(expected, actual)

Verifies values match (string or regex).

```
await web.verifyMatch(/Admin/, username);
```

---

assertEqual(expected, actual)

Hard assertion for equality.

```
await web.assertEqual('Success', message);
```

---

verifyContains(actual, expected)

Verifies actual contains expected value.

```
await web.verifyContains(message, 'Success');
```

---

verifyTextInElements(selector, expectedText)

Verifies text exists in any matched elements.

```
await web.verifyTextInElements('//li', 'Dashboard');
```

---

verifyElementPresent(selector)

Verifies element exists in DOM.

```
await web.verifyElementPresent('//div[@id="main"]');
```

---

verifyElementChecked(selector)

Verifies checkbox is checked.

```
await web.verifyElementChecked('//input[@type="checkbox"]');
```

---

verifyTextPresent(text)

Verifies text exists anywhere on the page.

```
await web.verifyTextPresent('Welcome');
```

---

verifyElementText(selector, expectedText)

Verifies element text matches expected value.

```
await web.verifyElementText('//h1', 'Dashboard');
```

---

verifyElementDisabled(selector)

Verifies element is disabled.

```
await web.verifyElementDisabled('//button[@disabled]');
```

---

verifyElementEnabled(selector)

Verifies element is enabled.

```
await web.verifyElementEnabled('//button');
```

🔹 Wait Keywords

waitForElementVisible(selector)

Waits until the element becomes visible.

```
await web.waitForElementVisible('//button');
```

---

wait(ms)

Static wait (milliseconds).

```
await web.wait(3000);
```

---

waitForElementPresent(selector)

Waits until element is attached to DOM.

```
await web.waitForElementPresent('//div');
```

---

waitForPageLoad()

Waits until page load completes.

```
await web.waitForPageLoad();
```

🔹 Alert Keywords

acceptAlert()

Accepts browser alert.

```
await web.acceptAlert();
```

---

dismissAlert()

Dismisses browser alert.

```
await web.dismissAlert();
```

---

getAlertText()

Returns alert text and dismisses alert.

```
const alertText = await web.getAlertText();
```

🔹 Excel Keywords

readFromExcel(path, sheetName, rowNo, colNo)

Reads data from Excel cell.

```
const username = await web.readFromExcel(
  'DataFiles/usercredentials.xlsx',
  'LoginData',
  2,
  1
);
```

---

writeToExcel(path, sheetName, rowNo, colNo, value)

Writes data to Excel cell.

```
await web.writeToExcel(
  'DataFiles/usercredentials.xlsx',
  'LoginData',
  2,
  3,
  'PASS'
);
```