



Sonu Madheshiya

# Playwright: Mastering the Essentials - Part 6

Playwright Assertions : A Comprehensive Guide

***Prepared By:***

***Sonu Madheshiya***  
***Automation Tester***



<https://www.linkedin.com/in/sonumadheshiya/>

## Playwright : Mastering the Essentials - Part 6

### Mastering Playwright Assertions: A Comprehensive Guide

Playwright offers powerful assertion capabilities through the expect function, making it easy to validate your tests. To make an assertion, call expect(value) and select a matcher that aligns with your expectation. Generic matchers like toEqual, toContain, and toBeTruthy allow you to assert various conditions effectively.

Example:

```
expect(success).toBeTruthy();
```

### Web-Specific Async Matchers

Playwright also provides web-specific async matchers that wait for expected conditions to be met. Consider this example:

```
await expect(page.getByTestId('status')).toHaveText('Submitted');
```

In this case, Playwright will continuously re-fetch the element with the test ID of status and check its text until it matches "Submitted" or the timeout is reached.

By default, assertion timeouts are set to 5 seconds.

### Auto-Retrying Assertions

The following assertions will retry until they pass or the timeout is reached. Since these assertions are async, always use await.

Assertion	Description
await expect(locator).toBeAttached()	Element is attached
await expect(locator).toBeChecked()	Checkbox is checked
await expect(locator).toBeDisabled()	Element is disabled
await expect(locator).toBeEditable()	Element is editable
await expect(locator).toBeEmpty()	Container is empty
await expect(locator).toBeEnabled()	Element is enabled



<https://www.linkedin.com/in/sonumadheshiya/>

Assertion	Description
await expect(locator).toBeFocused()	Element is focused
await expect(locator).toBeHidden()	Element is not visible
await expect(locator).toBeInViewport()	Element intersects viewport
await expect(locator).toBeVisible()	Element is visible
await expect(locator).toContainText()	Element contains text
await expect(locator).toHaveAccessibleDescription()	Element has a matching accessible description
await expect(locator).toHaveAccessibleName()	Element has a matching accessible name
await expect(locator).toHaveAttribute()	Element has a DOM attribute
await expect(locator).toHaveClass()	Element has a class property
await expect(locator).toHaveCount()	List has an exact number of children
await expect(locator).toHaveCSS()	Element has a CSS property
await expect(locator).toHaveId()	Element has an ID
await expect(locator).toHaveJSPROPERTY()	Element has a JavaScript property
await expect(locator).toHaveRole()	Element has a specific ARIA role
await expect(locator).toHaveScreenshot()	Element has a screenshot
await expect(locator).toHaveText()	Element matches text
await expect(locator).toHaveValue()	Input has a value
await expect(locator).toHaveValues()	Select has options selected
await expect(page).toHaveScreenshot()	Page has a screenshot
await expect(page).toHaveTitle()	Page has a title
await expect(page).toHaveURL()	Page has a URL
await expect(response).toBeOK()	Response has an OK status

### Non-Retrying Assertions

These assertions test conditions without retrying. Be cautious using them in web tests, as they can cause flaky results if the page loads asynchronously.



<https://www.linkedin.com/in/sonumadheshiya/>

Prefer auto-retrying assertions, but for complex cases that need retrying, use expect.poll or expect.toPass.

Assertion	Description
expect(value).toBe()	Value is the same
expect(value).toBeCloseTo()	Number is approximately equal
expect(value).toBeDefined()	Value is not undefined
expect(value).toBeFalsy()	Value is falsy (e.g., false, 0, null)
expect(value).toBeGreaterThan()	Number is more than
expect(value).toBeGreaterThanOrEqual()	Number is more than or equal
expect(value).toBeInstanceOf()	Object is an instance of a class
expect(value).toBeLessThan()	Number is less than
expect(value).toBeLessThanOrEqual()	Number is less than or equal
expect(value).toBeNaN()	Value is NaN
expect(value).toBeNull()	Value is null
expect(value).toBeTruthy()	Value is truthy (e.g., not false, 0, null)
expect(value).toBeUndefined()	Value is undefined
expect(value).toContain()	String contains a substring
expect(value).toContainEqual()	Array or set contains a similar element
expect(value).toEqual()	Value is similar - deep equality and pattern matching
expect(value).toHaveLength()	Array or string has length
expect(value).toHaveProperty()	Object has a property
expect(value).toMatch()	String matches a regular expression
expect(value).toMatchObject()	Object contains specified properties
expect(value).toStrictEqual()	Value is similar, including property types
expect(value).toThrow()	Function throws an error
expect(value).any()	Matches any instance of a class/primitive
expect(value).anything()	Matches anything

Assertion	Description
expect(value).arrayContaining()	Array contains specific elements
expect(value).closeTo()	Number is approximately equal
expect(value).objectContaining()	Object contains specific properties
expect(value).stringContaining()	String contains a substring
expect(value).stringMatching()	String matches a regular expression

## Negating Matchers

Invert expectations using .not to check the opposite:

```
expect(value).not.toEqual(0);
await expect(locator).not.toContainText('some text');
```

## Soft Assertions

Soft assertions don't terminate test execution on failure but mark the test as failed. This allows you to continue testing even if some checks fail.

Example:

```
await expect.soft(page.getByTestId('status')).toHaveText('Success');
await expect.soft(page.getByTestId('eta')).toHaveText('1 day');
```

You can check for soft assertion failures at any point:

```
expect(test.info().errors).toHaveLength(0);
```

**Note:** Soft assertions only work with the Playwright test runner.

## Custom Expect Messages

You can add custom messages to your assertions for better context in reports:

```
await expect(page.getText('Name'), 'should be logged in').toBeVisible();
```

Custom messages are visible in both passing and failing asserts, making them invaluable for debugging.



<https://www.linkedin.com/in/sonumadheshiya/>

Soft assertions also support custom messages:

```
expect.soft(value, 'my soft assertion').toBe(56);
```

---

*Stay tuned with [Sonu Madheshiya](#) on LinkedIn for more interesting content on automation testing.*