Locators

**Introduction**

* Locators are essential to Playwright's auto-waiting and retry-ability.
* They represent a method to find elements on a page at any moment.

**Quick Guide**

* **Recommended Built-in Locators:**
  + **page.getByRole()**: Locate by explicit and implicit accessibility attributes.
  + **page.getByText()**: Locate by text content.
  + **page.getByLabel()**: Locate form control by the associated label's text.
  + **page.getByPlaceholder()**: Locate input by placeholder text.
  + **page.getByAltText()**: Locate an element, usually an image, by its text alternative.
  + **page.getByTitle()**: Locate an element by its title attribute.
  + **page.getByTestId()**: Locate an element based on its data-testid attribute (configurable).

**Example**

js

Copy code

await page.getByLabel('User Name').fill('John');

await page.getByLabel('Password').fill('secret-password');

await page.getByRole('button', { name: 'Sign in' }).click();

await expect(page.getByText('Welcome, John!')).toBeVisible();

**Locating Elements**

* **Resilient Tests**: Prioritize user-facing attributes like page.getByRole().
* **DOM Example**:
  + <button>Sign in</button>: Use page.getByRole('button', { name: 'Sign in' }).click();.

**Usage Notes**

* Use the code generator to create and edit locators.
* Locators dynamically find up-to-date DOM elements before every action.
* **Example of Dynamic DOM Elements**:

js

Copy code

const locator = page.getByRole('button', { name: 'Sign in' });

await locator.hover();

await locator.click();

**Chaining Locators**

* Locators are available in Locator and FrameLocator classes, enabling chaining.
* **Example**:

js

Copy code

const locator = page.frameLocator('#my-frame').getByRole('button', { name: 'Sign in' });

await locator.click();

**Locate by Role**

* **page.getByRole()** reflects how users and assistive technology perceive elements.
* Role locators recognize buttons, checkboxes, headings, links, etc., based on W3C specifications.
* **Example**:

js

Copy code

await page.getByRole('heading', { name: 'Sign up' }).toBeVisible();

await page.getByRole('checkbox', { name: 'Subscribe' }).check();

await page.getByRole('button', { name: /submit/i }).click();

**Locate by Label**

* Convenient for interacting with form controls.
* **Example**:

js

Copy code

await page.getByLabel('Password').fill('secret');

**Locate by Placeholder**

* Useful when form elements lack labels but have placeholder texts.
* **Example**:

js

Copy code

await page.getByPlaceholder('name@example.com').fill('playwright@microsoft.com');

**Locate by Text**

* Find an element by matching text content (substring, exact string, or regex).
* **Example**:

js

Copy code

await expect(page.getByText('Welcome, John')).toBeVisible();

await expect(page.getByText('Welcome, John', { exact: true })).toBeVisible();

**Locate by Alt Text**

* Useful for locating images or elements with alt attributes.
* **Example**:

js

Copy code

await page.getByAltText('playwright logo').click();

**Locate by Title**

* Use when an element has a matching title attribute.
* **Example**:

js

Copy code

await expect(page.getByTitle('Issues count')).toHaveText('25 issues');

**Locate by Test ID**

* The most resilient way to locate elements (QA-defined explicit test ids).
* **Example**:

js

Copy code

await page.getByTestId('directions').click();

* Test id attribute is configurable via testIdAttribute in test config.

**Locate by CSS or XPath**

* Playwright supports both CSS and XPath selectors.
* **Example**:

js

Copy code

await page.locator('css=button').click();

await page.locator('xpath=//button').click();

**When to Avoid CSS/XPath**

* Avoid long CSS or XPath chains as they are prone to breaking with DOM changes.

**Locate in Shadow DOM**

* Locators work by default in Shadow DOM except for XPath and closed-mode shadow roots.
* **Example**:

js

Copy code

await page.getByText('Details').click();

**Filtering Locators**

* Locators can be filtered by text or descendants.
* **Example**:

js

Copy code

await page.getByRole('listitem').filter({ hasText: 'Product 2' }).getByRole('button', { name: 'Add to cart' }).click();

**Locator Operators**

* **Matching inside a Locator**: Chain locators like page.getByText() or locator.getByRole().
* **Example**:

js

Copy code

const product = page.getByRole('listitem').filter({ hasText: 'Product 2' });

await product.getByRole('button', { name: 'Add to cart' }).click();

await expect(product).toHaveCount(1);

* **Matching Simultaneously**: Use locator.and() to combine locators.
* **Example**:

js

Copy code

const button = page.getByRole('button').and(page.getByTitle('Subscribe'));

* **Matching Alternatives**: Use locator.or() when targeting multiple potential elements.