Playwright

Locators: Um Guia Completo

Preparado e traduzido por:

João Vitor de Souza
Tech Lead

Playwright: Dominando o Essencial

Dominando as Estratégias de Locators no Playwright: Um Guia Completo

Os Locators são cruciais no Playwright para interagir com elementos em uma página. Eles fornecem uma maneira de encontrar e interagir com elementos usando várias estratégias, garantindo que as ações sejam realizadas nos elementos corretos.

Guia Rápido dos Locators Integrados Recomendados:

- **page.getByRole()**: Encontra elementos por atributos de acessibilidade explícitos e implícitos.
- **page.getByText()**: Localiza elementos com base em seu conteúdo de texto.
- **page.getByLabel()**: Identifica controles de formulário pelo texto do rótulo associado.
- **page.getByPlaceholder()**: Encontra elementos de entrada pelo texto do placeholder.
- **page.getByAltText()**: Alveja elementos, tipicamente imagens, pelo texto alternativo.
- **page.getByTitle()**: Localiza elementos usando seu atributo de título.
- **page.getByTestId()**: Identifica elementos com base no atributo data-testid.

```
// Finds an input field by its associated label and fills it with the name
await page.getByLabel('Name').fill('Sonu Madheshiya');

// Locates an input field by its associated label and fills it with the password
await page.getByLabel('Password').fill('sonu1234');

// Finds an input element by its placeholder text and fills it with the value
await page.getByPlaceholder('Search...').fill('Playwright');

// Finds a button by its role and name, then clicks it
await page.getByRole('button', { name: 'Log In' }).click();

// Checks if the text 'Hello, Sonu!' is visible on the page
await expect(page.getByText('Hello, Sonu!')).toBeVisible();

// Targets an image by its alternative text and verifies it's visible
await expect(page.getByAltText('Playwright Logo')).toBeVisible();

// Locates an element by its title attribute and clicks it
await page.getByTitle('Settings').click();

// Identifies an element based on its data-testid attribute and verifies its visibility
await expect(page.getByTestId('submit-button')).toBeVisible();
```

Localizando Elementos

O Playwright fornece vários locators integrados para melhorar a confiabilidade dos testes. É recomendável usar atributos voltados para o usuário e contratos explícitos como:

1. Role Locators:

- **Método:** page.getByRole(role, { name: 'name' })
- **Descrição:** Encontra elementos com base em seu papel e nome acessível.
- **Exemplo:**

```
// Clicks a button with the role 'button' and name 'Submit'
await page.getByRole('button', { name: 'Submit' }).click();
```

2. Label Locators:

- **Método:** page.getByLabel('label text')
- **Descrição:** Localiza controles de formulário pelo texto do rótulo associado.
- **Exemplo:**

```
// Fills a text input field labeled 'Username'
await page.getByLabel('Username').fill('myUserName');
```

3. Placeholder Locators:

- **Método:** page.getByPlaceholder('placeholder text')
- **Descrição:** Encontra elementos de entrada com um atributo placeholder específico.
- **Exemplo:**

```
// Fills a phone number input with the placeholder 'Enter phone number'
await page.getByPlaceholder('Enter phone number').fill('123-456-7890');
```

4. Text Locators:

- **Método:** page.getByText(text, { exact: true })
- **Descrição:** Localiza elementos com base no texto que contêm, suportando correspondência exata ou expressões regulares.
- **Exemplo:**

```
// Asserts visibility of an element containing 'Hello, World!'
await expect(page.getByText('Hello, World!')).toBeVisible();

// Asserts visibility of an element with exact text 'Welcome to the site'
await expect(page.getByText('Welcome to the site', { exact: true })).toBeVisible()

// Asserts visibility of an element matching the regex 'user [0-9]+'
await expect(page.getByText(/user [0-9]+/i)).toBeVisible();
```

5. Alt Text Locators:

- **Método:** page.getByAltText('alt text')
- **Descrição:** Encontra elementos de imagem com base no atributo alt.
- **Exemplo:**

```
// Clicks an image with the alt text 'logo image'
await page.getByAltText('logo image').click();
```

6. Title Locators:

- **Método:** page.getByTitle('title text')
- **Descrição:** Localiza elementos pelo atributo de título.
- **Exemplo:**

```
// Asserts that an element with the title 'Profile Picture' has the text 'User Pro
await expect(page.getByTitle('Profile Picture')).toHaveText('User Profile');
```

7. Test ID Locators:

- **Método:** page.getByTestId('test-id')
- **Descrição:** Encontra elementos usando um atributo data-testid personalizado.
- **Exemplo:**

```
// Clicks an element with the test id 'submit-form'
await page.getByTestId('submit-form').click();
```

8. CSS or XPath Locators:

- **Método:** page.locator('selector') ou page.locator('xpath=//selector')
- **Descrição:** Encontra elementos usando seletores CSS ou XPath.
- **Exemplo:**

```
// Clicks a button using a CSS selector
await page.locator('css=.submit-button').click();

// Clicks a button using an XPath selector
await page.locator('xpath=//button[@id="submitBtn"]').click();
```

Localizando Elementos no Shadow DOM

O Playwright suporta a localização de elementos dentro do Shadow DOM por padrão, exceto para alguns casos:

- **XPath Locators:** Não funcionam com elementos dentro do Shadow DOM.
- **Closed-Mode Shadow Roots:** Não são suportados.

Exemplo de interação com elementos no Shadow DOM:

```
<x-details role="button" aria-expanded="true" aria-controls="inner-details">
  <div>Title</div>
  #shadow-root
    <div id="inner-details">Details</div>
</x-details>
```

```
// Click on the text 'Details'
await page.getByText('Details').click();

// Click on the <x-details> component
await page.locator('x-details', { hasText: 'Details' }).click();

// Ensure that <x-details> contains the text 'Details'
await expect(page.locator('x-details')).toContainText('Details');
```

Filtrando Locators:

1. Filtrar por Texto:

- Use locator.filter({ hasText: 'text' }) para encontrar elementos que contenham texto específico.
- Exemplo:

```
await page
    .getByRole('listitem')
    .filter({ hasText: 'Item B' })
    .getByRole('button', { name: 'Add to Bag' })
    .click();
```

2. Filtrar por Não Conter Texto:

- Use locator.filter({ hasNotText: 'text' }) para excluir elementos que contenham um texto específico.
- Exemplo:

```
await expect(page.getByRole('listitem').filter({ hasNotText: 'Sold' })).toHaveCount(5);
```

3. Filtrar por Filho/Descendente:

- Filtre elementos com base na presença de um elemento descendente.
- Exemplo:

```
await page
    .getByRole('listitem')
    .filter({ has: page.getByRole('heading', { name: 'Item B' }) })
    .getByRole('button', { name: 'Add to Bag' })
    .click();
```

4. Filtrar por Ausência de Filho/Descendente:

- Exclua elementos com base na ausência de um elemento descendente.
- Exemplo:

```
await expect(page
    .getByRole('listitem')
    .filter({ hasNot: page.getByText('Item B') }))
    .toHaveCount(1);
```

Operadores de Locators:

1. Correspondência Dentro de um Locator:

- Encadeie métodos para restringir buscas dentro de um locator.
- Exemplo:

```
const item = page.getByRole('listitem').filter({ hasText: 'Item B' });
await item.getByRole('button', { name: 'Add to Bag' }).click();
await expect(item).toHaveCount(1);
```

2. Correspondência de Dois Locators Simultaneamente:

- Use locator.and() para corresponder elementos que atendem aos critérios de ambos os locator.
- Exemplo:

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

3. Correspondência de Um dos Dois Locators Alternativos:

- Use locator.or() para corresponder a qualquer um dos locators alternativos.
- Exemplo:

```
const newEmail = page.getByRole('button', { name: 'Compose' });
const dialog = page.getByText('Confirm Security Settings');
await expect(newEmail.or(dialog).first()).toBeVisible();
if (await dialog.isVisible())
  await page.getByRole('button', { name: 'Dismiss' }).click();
await newEmail.click();
```

4. Correspondência Somente de Elementos Visíveis:

- Filtre elementos para interagir apenas com aqueles visíveis.
- Exemplo:

```
await page.locator('button').locator('visible=true').click();
```

Outros Casos de Uso:

1. Fazer Algo com Cada Elemento na Lista:

- Itere sobre itens de uma lista para realizar ações.
- Exemplo:

```
for (const row of await page.getByRole('listitem').all())
  console.log(await row.textContent());
```

- Iterar Usando um Loop For Regular:
- Exemplo:

```
const rows = page.getByRole('listitem');
const count = await rows.count();
for (let i = 0; i < count; ++i)
  console.log(await rows.nth(i).textContent());
```
