Extensibility

Custom selector engines

Playwright supports custom selector engines, registered with selectors.register().

Selector engine should have the following properties:

- `query` function to query first element matching `selector` relative to the `root`.
- `queryAll` function to query all elements matching `selector` relative to the `root`.

By default the engine is run directly in the frame's JavaScript context and, for example, can call an application-defined function. To isolate the engine from any JavaScript in the frame, but leave access to the DOM, register the engine with `{contentScript: true}` option. Content script engine is safer because it is protected from any tampering with the global objects, for example altering `Node.prototype` methods. All built-in selector engines run as content scripts. Note that running as a content script is not guaranteed when the engine is used together with other custom engines.

Selectors must be registered before creating the page.

An example of registering selector engine that queries elements based on a tag name:

baseTest.ts

```
import { test as base } from '@playwright/test';

export { expect } from '@playwright/test';

// Must be a function that evaluates to a selector engine instance.
const createTagNameEngine = () => ({
  // Returns the first element matching given selector in the root's
subtree.
  query(root, selector) {
    return root.querySelector(selector);
  },

  // Returns all elements matching given selector in the root's subtree.
  queryAll(root, selector) {
    return Array.from(root.querySelectorAll(selector));
  }
});

export const test = base.extend<{}, { selectorRegistration: void }>({
```

```
  // Register selectors once per worker.
  selectorRegistration: [async ({ playwright }, use) => {
    // Register the engine. Selectors will be prefixed with "tag=".
    await playwright.selectors.register('tag', createTagNameEngine);
    await use();
  }, { scope: 'worker', auto: true }],
});
```

example.spec.ts

```
import { test, expect } from './baseTest';

test('selector engine test', async ({ page }) => {
  // Now we can use 'tag=' selectors.
  const button = page.locator('tag=button');
  await button.click();

  // We can combine it with built-in locators.
  await page.locator('tag=div').getByText('Click me').click();

  // We can use it in any methods supporting selectors.
  await expect(page.locator('tag=button')).toHaveCount(3);
});
```

Previous
Events

Next
Frames

Learn

- Getting started
- Playwright Training
- Learn Videos
- Feature Videos

Community

- Stack Overflow
- Discord
- Twitter
- LinkedIn

More

- GitHub
- YouTube
- Blog
- Ambassadors