API reference Classes Selectors

Selectors

Selectors can be used to install custom selector engines. See extensibility for more information.

Methods

register

Added in: v1.8

Selectors must be registered before creating the page.

Usage

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

Sync Async

```
from playwright.sync_api import sync_playwright, Playwright

def run(playwright: Playwright):
    tag_selector = """
      {
          // 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));
          }
      }"""

    # Register the engine. Selectors will be prefixed with "tag=".
    playwright.selectors.register("tag", tag_selector)
    browser = playwright.chromium.launch()
    page = browser.new_page()
    page.set_content('<div><button>Click me</button></div>')

    # Use the selector prefixed with its name.
    button = page.locator('tag=button')
    # Combine it with built-in locators.
    page.locator('tag=div').get_by_text('Click me').click()
    # Can use it in any methods supporting selectors.
    button_count = page.locator('tag=button').count()
    print(button_count)
    browser.close()

with sync_playwright() as playwright:
    run(playwright)
```

Arguments

- `name` str

  Name that is used in selectors as a prefix, e.g. `{name: 'foo'}` enables `foo=myselectorbody` selectors. May only contain `[a-zA-Z0-9_]` characters.

- `script` str *(optional)*

  Raw script content.

- `content_script` bool *(optional)*

  Whether to run this selector engine in isolated JavaScript environment. This environment has access to the same DOM, but not any JavaScript objects from the frame's scripts. Defaults to `false`. Note that running as a content script is not guaranteed when this engine is used together with other registered engines.

- `path` Union[str, pathlib.Path] *(optional)*

  Path to the JavaScript file. If `path` is a relative path, then it is resolved relative to the current working directory.

set_test_id_attribute

Added in: v1.27

Defines custom attribute name to be used in page.get_by_test_id(). `data-testid` is used by default.

Usage

```
selectors.set_test_id_attribute(attribute_name)
```

Arguments

- `attribute_name` str

  Test id attribute name.