Playwright

Node

- Node.js is JavaScript runtime.
- This example use program written in JavaScript.
- “npm” is a package manager commands used by Node.js

Download >> https:node.js.org/en

Playwright

- Playwright is a framework for Web Testing and Automation.
- It allows testing Chromium, Firefox and WebKit with a single API.
- Playwright is built to enable cross-browser web automation that is ever-green, capable, reliable and fast.

Pros

- Cross-Browser Support
- Multi-Language Support
- Interactive Test Generation

Source: https://github.com/microsoft/playwright

Playwright : Install (1/2)

```
$ mkdir yourProjectName
```

```
$ cd “/yourProjectDirectory”
```

```
$ npm init playwright@latest
```

Playwright : Install (2/2)

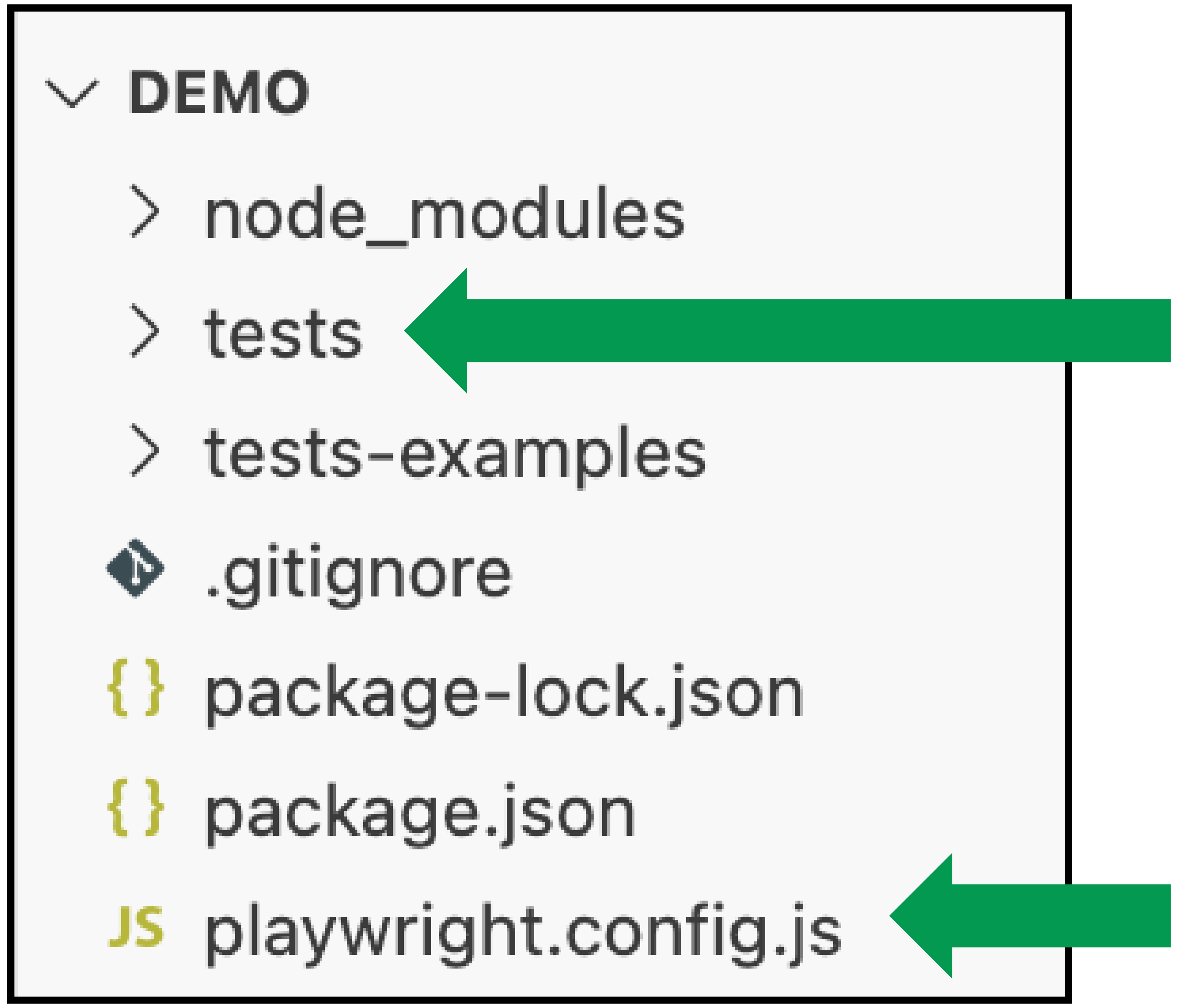
```
Do you want to use TypeScript or JavaScript: JavaScript
```

```
Where to put your end-to-end tests? : tests
```

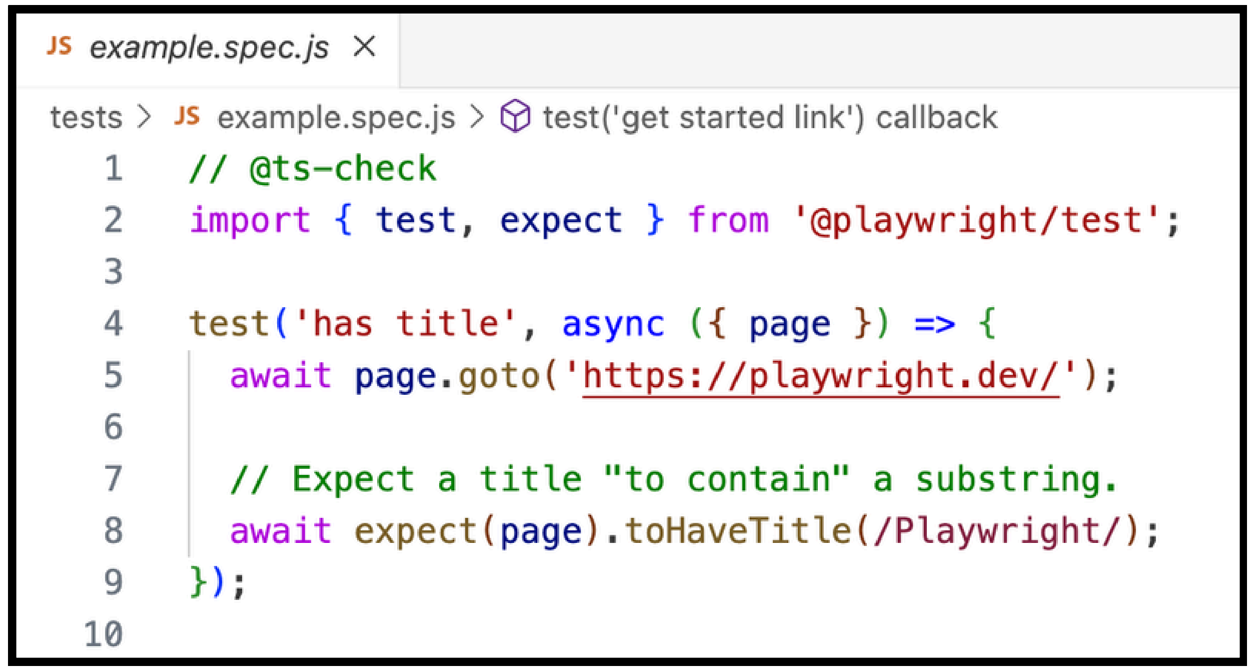
```
Add a GitHub Actions workflow? (y/N) : false
```

```
Install Playwright browsers? (Y/n): true
```

Playwright : Project Structure



DEMO
- node_modules
- tests
- tests-examples
- .gitignore
- package-lock.json
- package.json
- playwright.config.js



example.spec.js

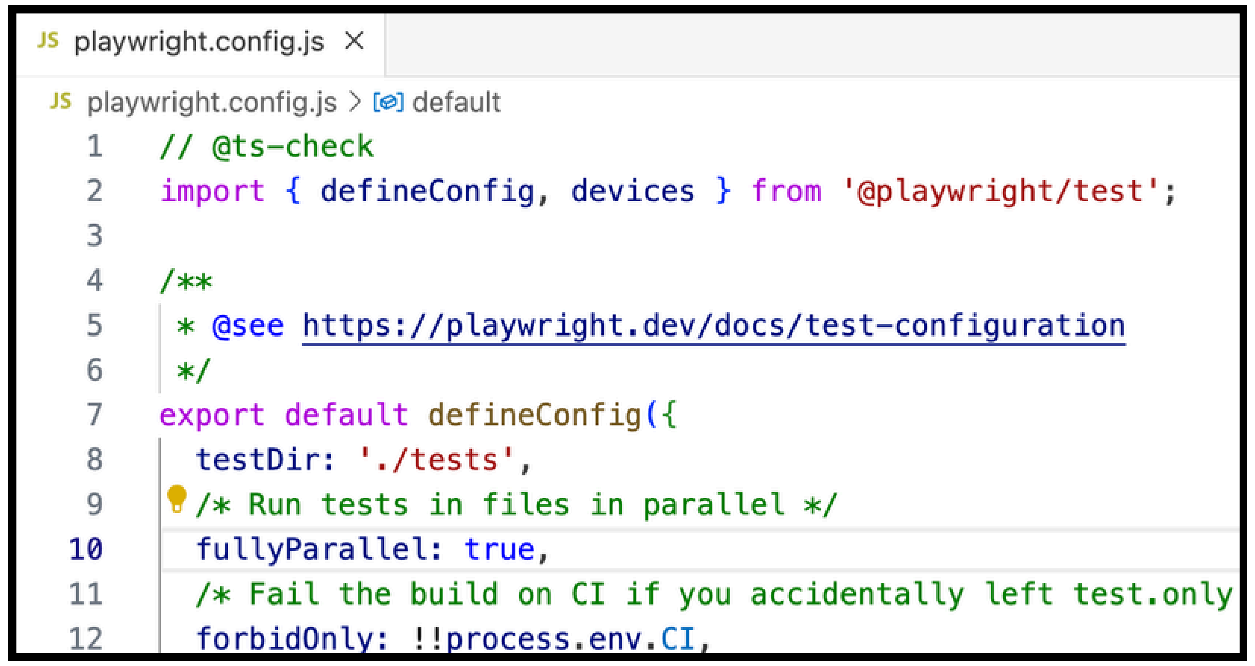
tests > example.spec.js > test('get started link') callback

```js
// @ts-check
import { test, expect } from '@playwright/test';

test('has title', async ({ page }) => {
  await page.goto('https://playwright.dev/');

  // Expect a title "to contain" a substring.
  await expect(page).toHaveTitle(/Playwright/);
});

```



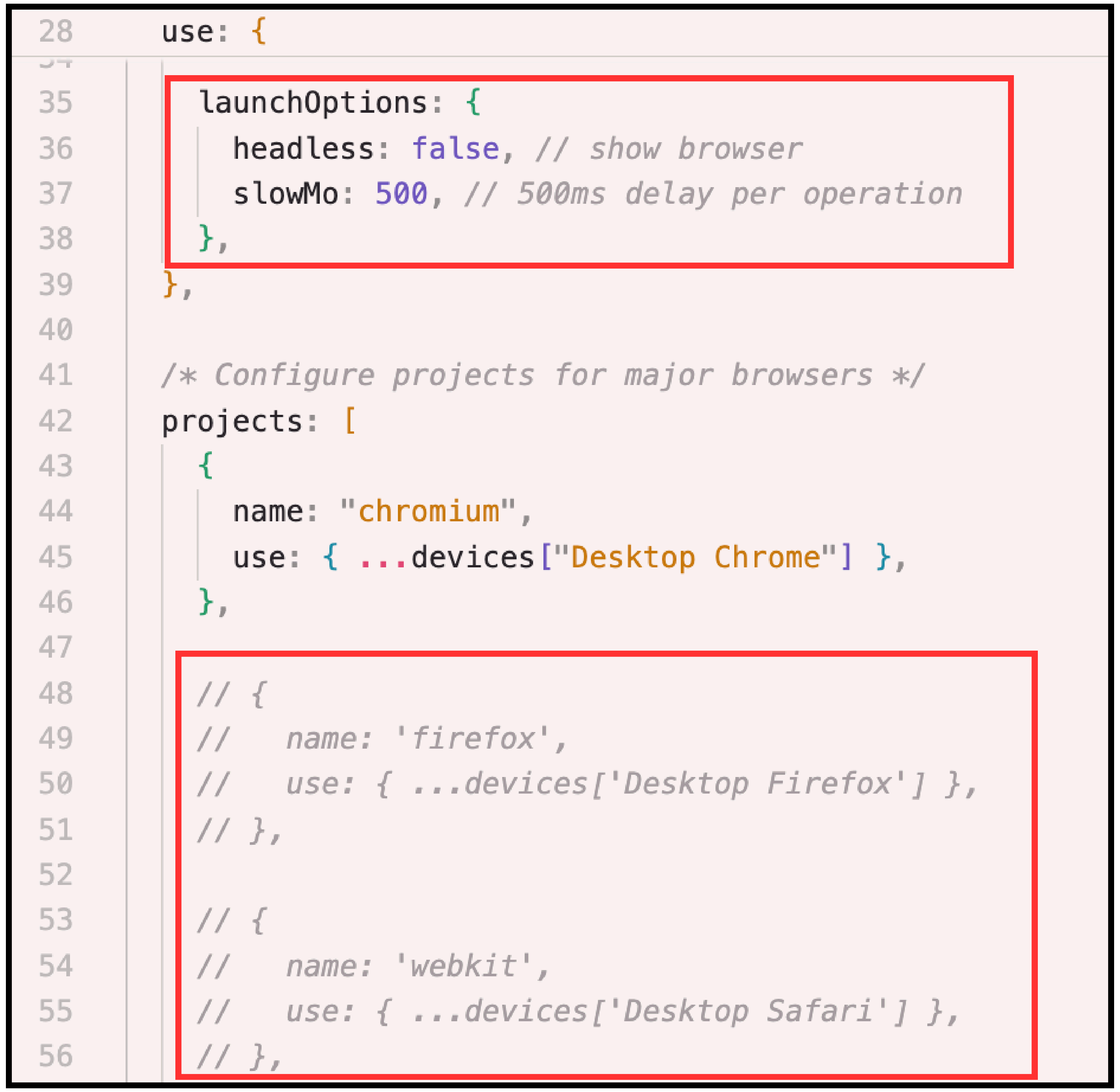
playwright.config.js

playwright.config.js > default

```js
// @ts-check
import { defineConfig, devices } from '@playwright/test';

/**
 * @see https://playwright.dev/docs/test-configuration
 */
export default defineConfig({
  testDir: './tests',
  /* Run tests in files in parallel */
  fullyParallel: true,
  /* Fail the build on CI if you accidentally left test.only
  forbidOnly: !!process.env.CI,
```

Playwright : Config



```
  use: {
    launchOptions: {
      headless: false, // show browser
      slowMo: 500, // 500ms delay per operation
    },
  },

  /* Configure projects for major browsers */
  projects: [
    {
      name: "chromium",
      use: { ...devices["Desktop Chrome"] },
    },

    // {
    //   name: 'firefox',
    //   use: { ...devices['Desktop Firefox'] },
    // },

    // {
    //   name: 'webkit',
    //   use: { ...devices['Desktop Safari'] },
    // },
```

HTML

5

HTML

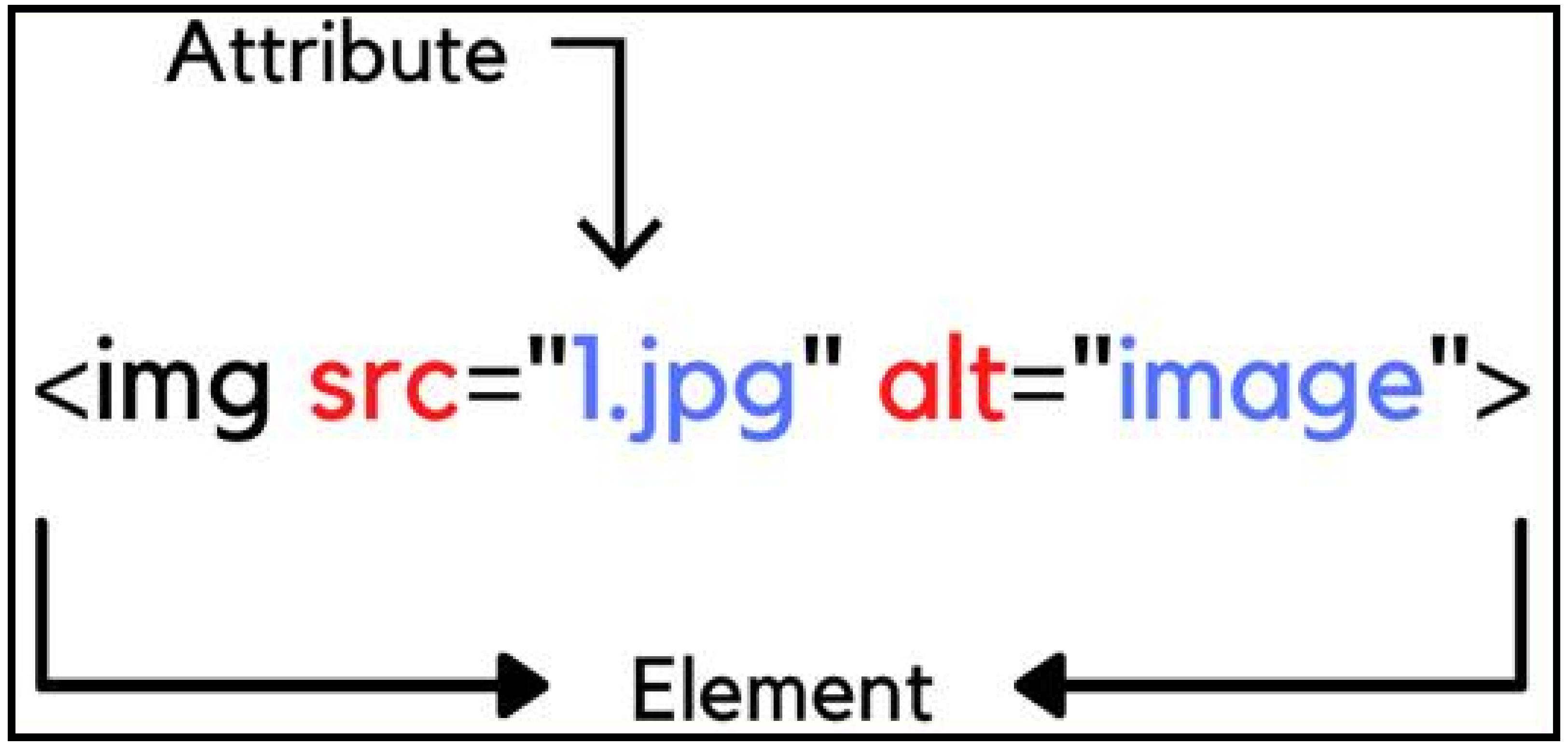
HTML (HyperText Markup Language) is the most basic building block of the Web.
It defines the meaning and structure of web content.

```
ass="container">
v class="row">
 <div class="col-md-6 col-lg-8"> <!-- __________ BEGIN N
     <nav id="nav" role="navigation">
         <ul>
             <li><a href="index.html">Home</a></li>
             <li><a href="home-events.html">Home Events
             <li><a href="multi-col-menu.html">Multiple
             <li class="has-children"> <a href="#" clas
                 <ul>
                     <li><a href="tall-button-header.htm
                     <li><a href="image-logo.html">Image
                     <li class="active"><a href="tall-lo
                 </ul>
             </li>
             <li class="has-children"> <a href="#">Carou
                 <ul>
                     <li><a href="variable-width-slider.h
```

https://developer.mozilla.org/en-US/docs/Web/HTML

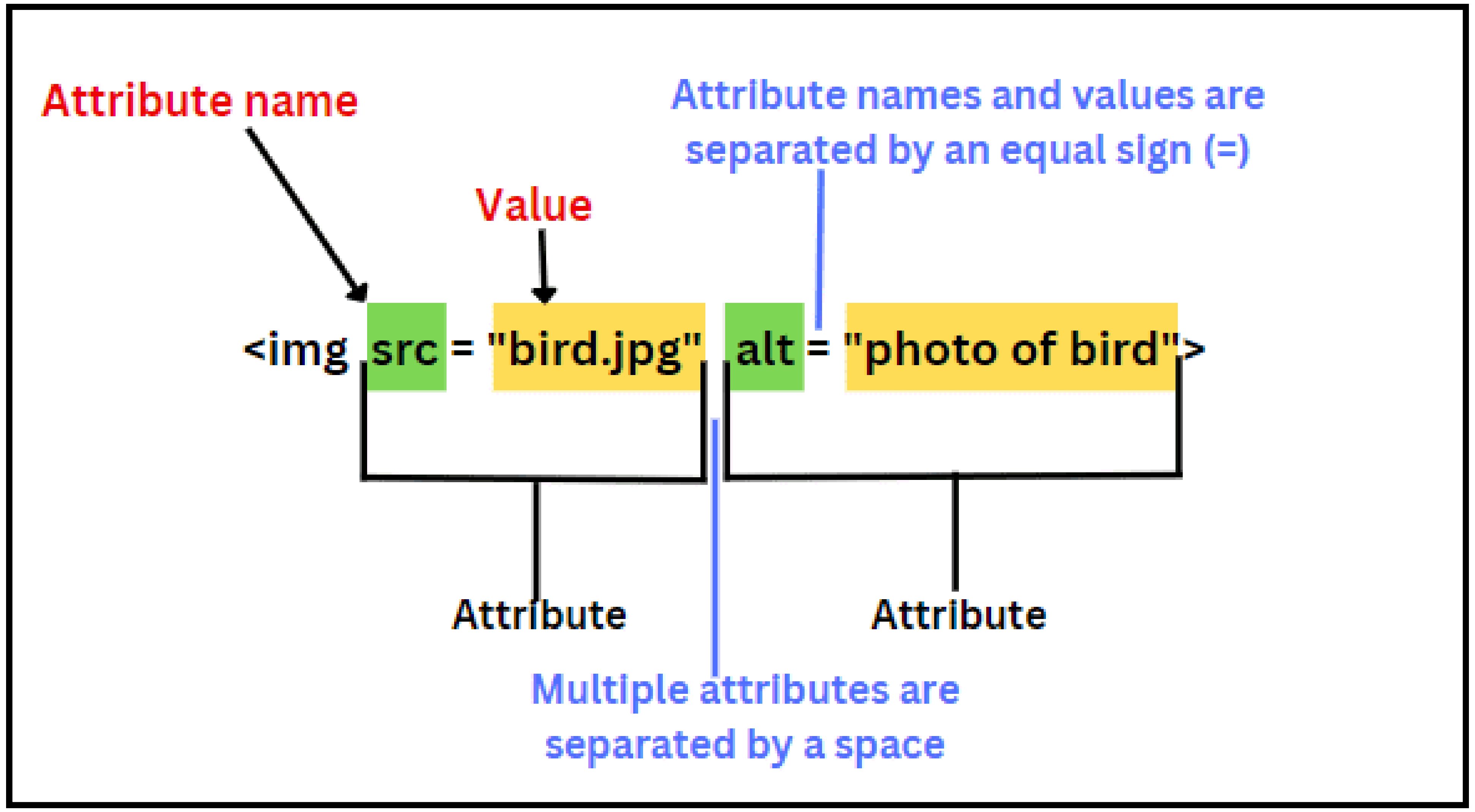
HTML : Elements

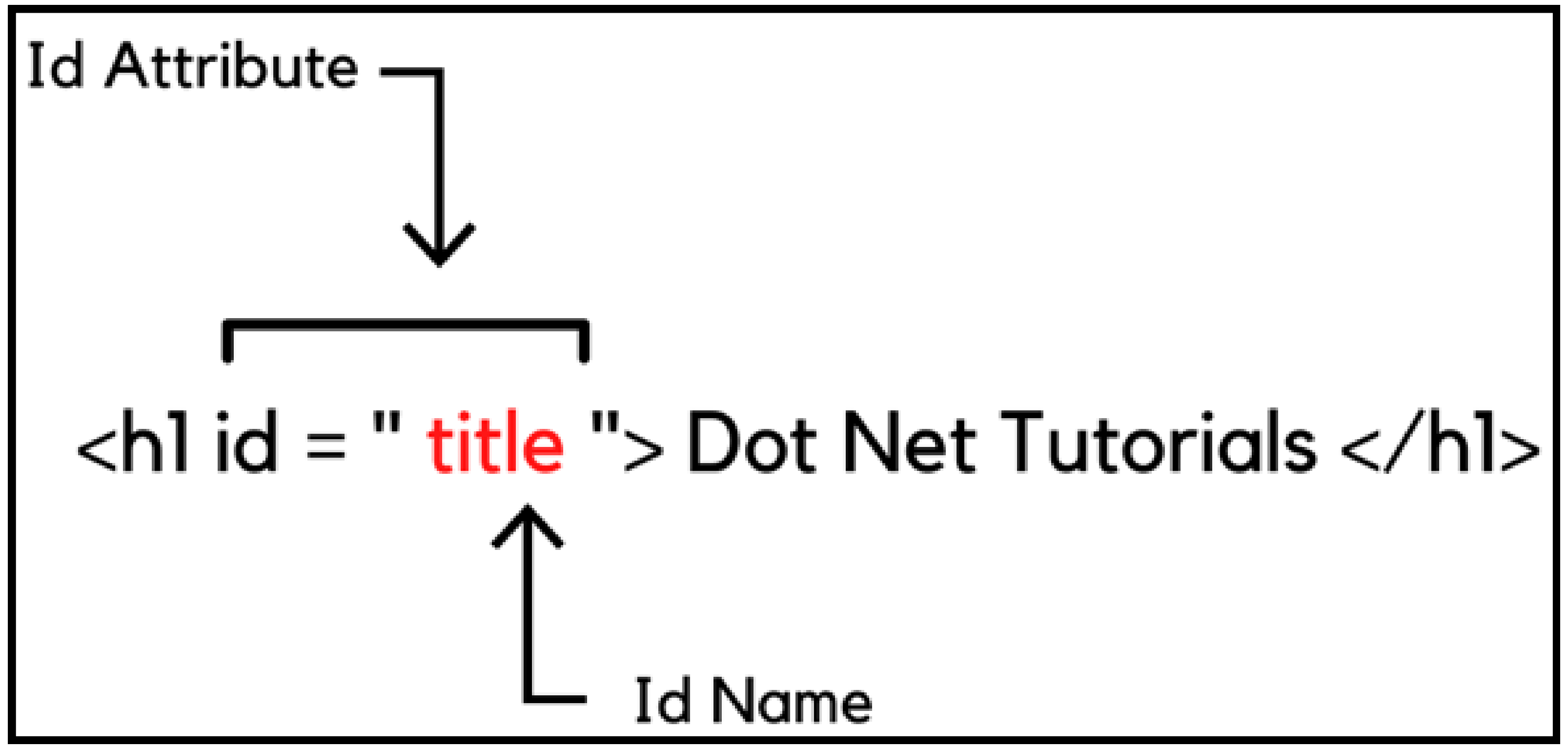
An HTML element is set off from other text in a document by "tags", which consist of the element name surrounded by < and >.



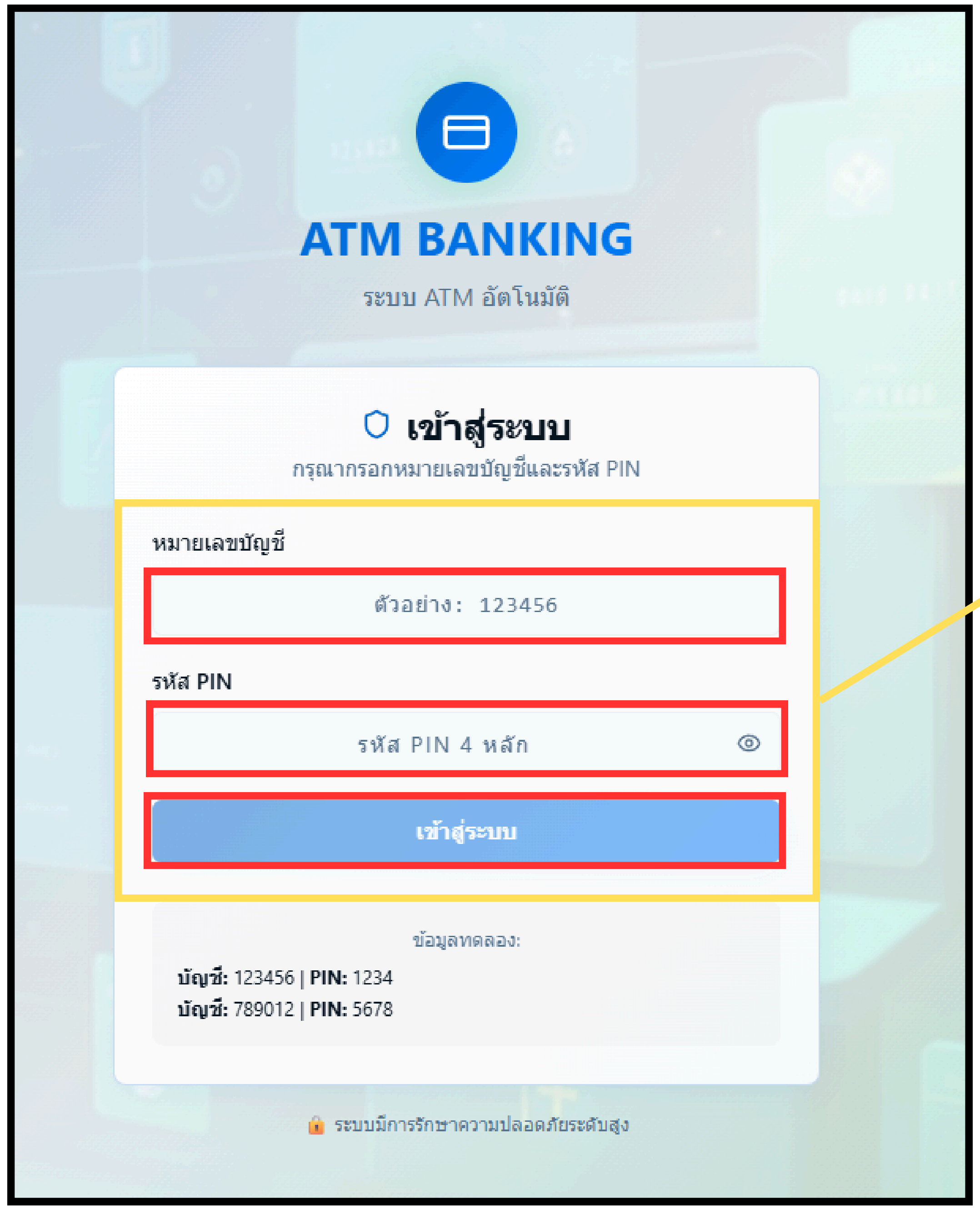
HTML : Attributes

Reference for all HTML attributes. Attributes are additional values that configure elements or adjust their behavior in various ways.


Attribute name
Attribute names and values are separated by an equal sign (=)
Value
<img src = "bird.jpg" alt = "photo of bird">
Attribute
Attribute
Multiple attributes are separated by a space



HTML : HTML Form Elements


ATM BANKING

ระบบ ATM อัตโนมัติ

เข้าสู่ระบบ

กรุณากรอกหมายเลขบัญชีและรหัส PIN

หมายเลขบัญชี

ตัวอย่าง: 123456

รหัส PIN

รหัส PIN 4 หลัก

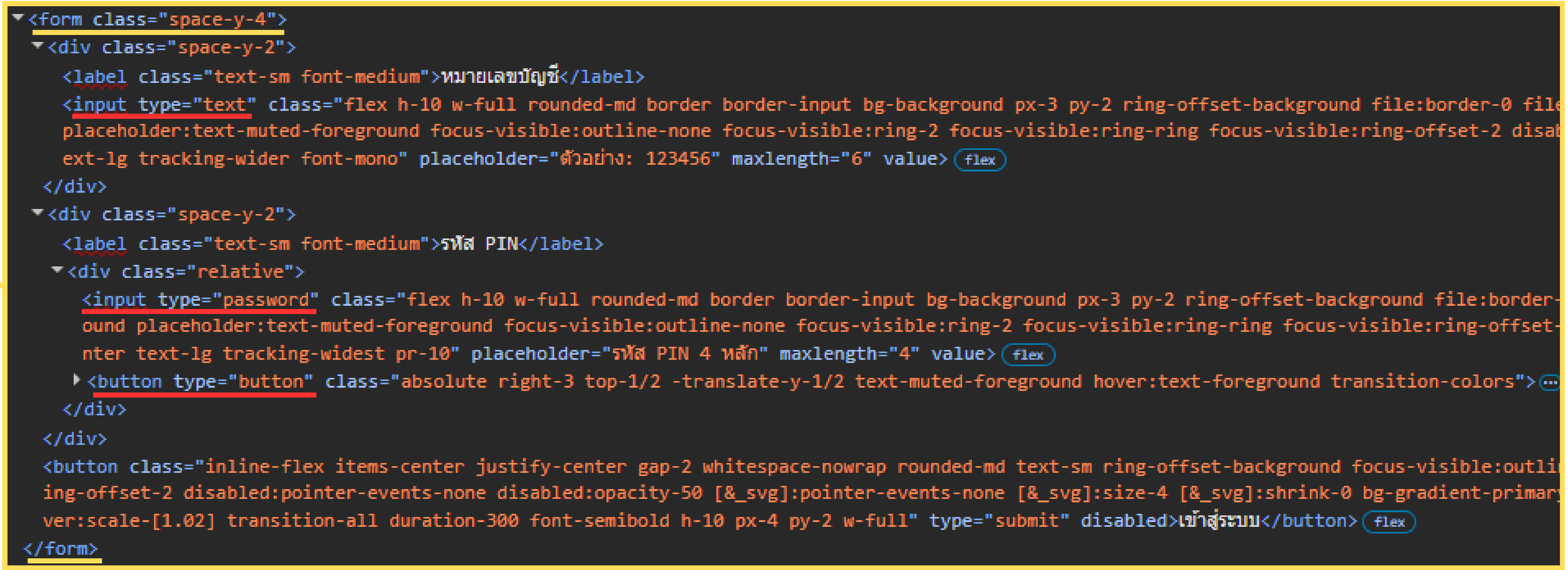
เข้าสู่ระบบ

ข้อมูลทดลอง:

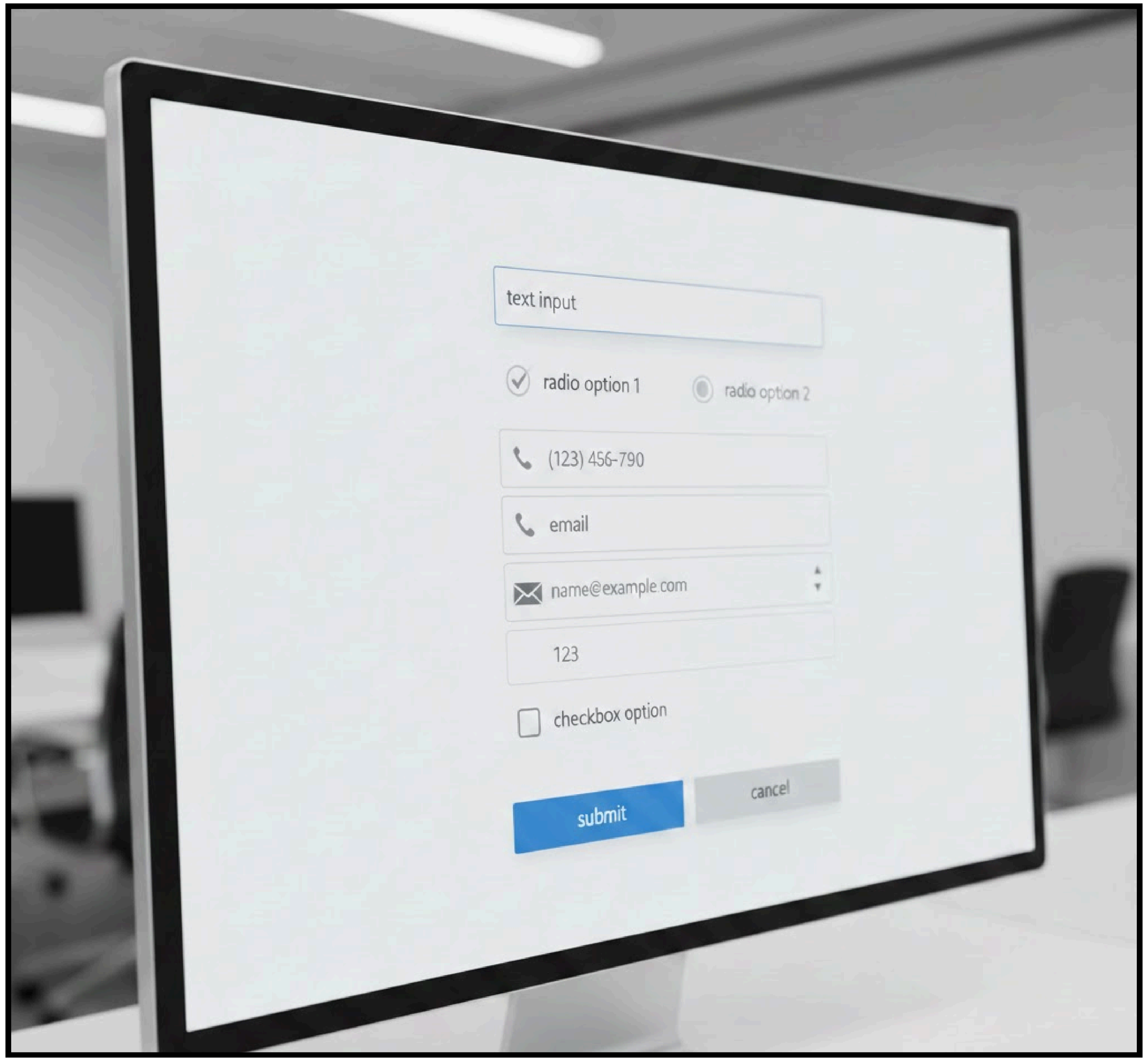
บัญชี: 123456 | **PIN:** 1234

บัญชี: 789012 | **PIN:** 5678

🔒 ระบบมีการรักษาความปลอดภัยระดับสูง


```
<form class="space-y-4">
  <div class="space-y-2">
    <label class="text-sm font-medium">หมายเลขบัญชี</label>
    <input type="text" class="flex h-10 w-full rounded-md border border-input bg-background px-3 py-2 ring-offset-background file:border-0 file
    placeholder:text-muted-foreground focus-visible:outline-none focus-visible:ring-2 focus-visible:ring-ring focus-visible:ring-offset-2 disab
    ext-lg tracking-wider font-mono" placeholder="ตัวอย่าง: 123456" maxlength="6" value> flex
  </div>
  <div class="space-y-2">
    <label class="text-sm font-medium">รหัส PIN</label>
    <div class="relative">
      <input type="password" class="flex h-10 w-full rounded-md border border-input bg-background px-3 py-2 ring-offset-background file:border-
      ound placeholder:text-muted-foreground focus-visible:outline-none focus-visible:ring-2 focus-visible:ring-ring focus-visible:ring-offset-
      nter text-lg tracking-widest pr-10" placeholder="รหัส PIN 4 หลัก" maxlength="4" value> flex
     <button type="button" class="absolute right-3 top-1/2 -translate-y-1/2 text-muted-foreground hover:text-foreground transition-colors">…
    </div>
  </div>
  <button class="inline-flex items-center justify-center gap-2 whitespace-nowrap rounded-md text-sm ring-offset-background focus-visible:outli
  ing-offset-2 disabled:pointer-events-none disabled:opacity-50 [&_svg]:pointer-events-none [&_svg]:size-4 [&_svg]:shrink-0 bg-gradient-primary
  ver:scale-[1.02] transition-all duration-300 font-semibold h-10 px-4 py-2 w-full" type="submit" disabled>เข้าสู่ระบบ</button> flex
</form>
```

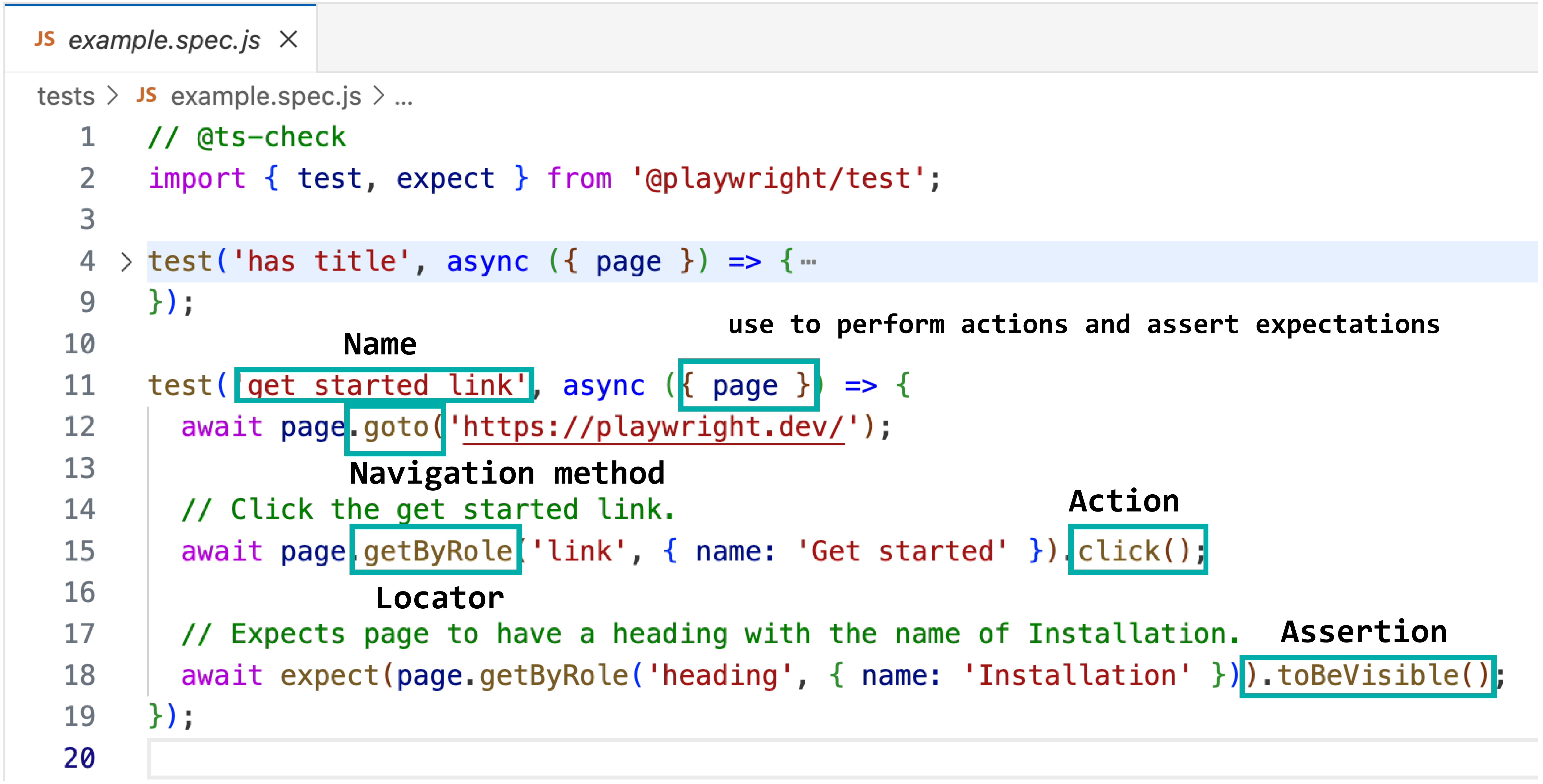
HTML : The HTML Input element



- checkbox
- radio
- tel
- text
- email
- number
- button

Test Script

Test Script : Basic Syntax


JS example.spec.js ×
tests > JS example.spec.js > ...
Name
use to perform actions and assert expectations
Navigation method
Action
Locator
Assertion

```
// @ts-check
import { test, expect } from '@playwright/test';

test('has title', async ({ page }) => {…
});

test('get started link', async ({ page }) => {
  await page.goto('https://playwright.dev/');

  // Click the get started link.
  await page.getByRole('link', { name: 'Get started' }).click();

  // Expects page to have a heading with the name of Installation.
  await expect(page.getByRole('heading', { name: 'Installation' })).toBeVisible();
});

```

Test Script : Locators

Locator	Description	Example
getByRole()	Locates an element based on its accessibility role (e.g., button, link, heading)	page.getByRole('button', { name: 'Login' }) – finds a button with visible text "Login"
getByTestId()	Locates an element using a data-testid attribute. Useful when developers add these attributes specifically for testing.	page.getByTestId('submit-button') – targets element with data-testid="submit-button"
locator(selector)	Locates an element using CSS selectors, such as #id, .class, [type="text"], or attribute selectors. Very flexible and works on any HTML attribute.	page.locator('#username') – selects element with id="username"

Test Script : Locator getByRole('Keyword')

Keywords	Html
page.getByRole('**heading**', { name: '**Sign up**' })	<**h1**>**Sign up**</**h1**>
page.getByRole('**checkbox**', { name: '**Subscribe**' })	<input type="**checkbox"** name="subscribe"> **Subscribe**
page.getByRole('**button**', { name: **/submit/**})	<**button** type="submit">**Submit**</**button**>
page.getByRole('**link**', { name: '**Learn more**' })	<**a** href="/learn-more">**Learn more**</**a**>

https://playwright.dev/docs/locators

Test Script : Actions

Action	**Example**
locator.**fill()**	await page.getByRole('textbox').**fill('Peter')**;
locator.**setChecked()**	await page.getByLabel('I agree to the terms above').**setCheck(true)**;
locator.**selectOption()**	await page.getByLabel('Choose a color').**selectOption('blue')**;
locator**.click()**	await page.getByRole('button').**click();**

https://playwright.dev/docs/input

Test Script : Assertions

Assertion	Description
await expect(locator).**toBeChecked()**	Checkbox is checked
await expect(locator).**toBeHidden()**	Element is not visible
await expect(locator).**toBeVisible()**	Element is visible
await expect(locator).**toContainText()**	Element contains text
await expect(page).**toHaveTitle()**	Page has a title

https://playwright.dev/docs/test-assertions

Test Execution

Test Execution : Command

```
$ npx playwright test
```

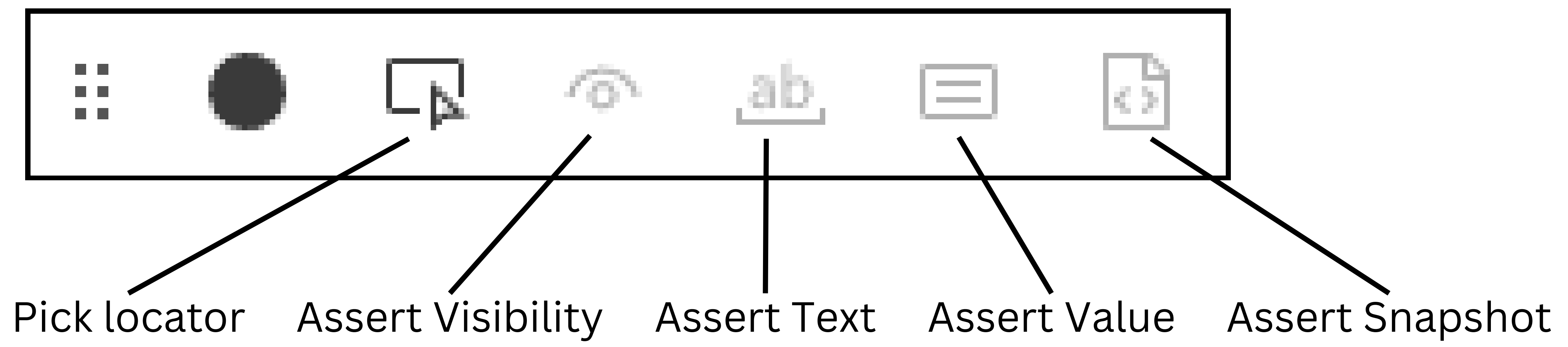
```
$ npx playwright test “filename.js”
```

```
$ npx playwright show-report
```

```
$ npx playwright test --ui
```

Test Execution : Code gen

```
$ npx playwright codegen
```



DEMO : Code gen

<u>ATM</u>

Test Execution : Parameterized

```
$ npm i csv-parse
```

```
$ npx playwright test --ui
```

DEMO : Parameterized

ATM