

How to Capture Screenshots in Playwright

Playwright provides an easy way to take screenshots during your test automation process. You can capture:

- A **particular section** of the page
- The **full page**
- A **specific element** on the page

All these can be done using the built-in `screenshot()` method from the `page` object.

1. Capture a Screenshot of the Current Page

Example: Take a screenshot right after opening the page

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

test('Page screenshot', async ({ page }) => {
  await page.goto('https://your-website.com');
  await page.screenshot({ path: 'HomePage.png' });
});
```

- The `path` option defines where the screenshot will be saved.
- If no folder is specified in the path, the screenshot will be saved in the **root of the project**.

Example with a folder path:

```
await page.screenshot({ path: 'tests/screenshots/HomePage.png' });
```

Overwriting Issue

If you run the test multiple times, the screenshot (`HomePage.png`) will be **overwritten** each time.

Solution: Add Timestamp to the Filename

You can use the `Date.now()` function to add a unique timestamp to each screenshot file.

```
await page.screenshot({ path: 'tests/screenshots/HomePage_' + Date.now() + '.png' });
```

Full Example with Timestamp

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

test('Page screenshot with timestamp', async ({ page }) => {
  await page.goto('https://your-website.com');
  await page.screenshot({
    path: 'tests/screenshots/HomePage_' + Date.now() + '.png'
  });
});
```

2. Full Page Screenshot in Playwright

Sometimes, you may want to capture the **entire webpage**, including content that's not visible unless you scroll — like the footer or long pages.

How to Capture a Full Page Screenshot

To do this, pass an additional option called `**fullPage: true**` in the `screenshot()` method. This tells Playwright to scroll and capture everything from top to bottom.

Example

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

test('Full page screenshot', async ({ page }) => {
  await page.goto('https://your-website.com');
  await page.screenshot({
    path: 'tests/screenshots/HomePageFull_' + Date.now() + '.png',
    fullPage: true
  });
});
```

Key Notes

- `fullPage: true` ensures that even off-screen content (like the footer) is included in the screenshot.
 - Using `Date.now()` prevents overwriting old screenshots by adding a unique timestamp.
-

3. Capture a Screenshot of a Specific Element in Playwright

If you want to capture a **particular element** (like a specific `div`, button, image, or section), you can use the `screenshot()` method on a **locator** instead of the entire page.

How It Works:

1. Use `page.locator()` to target the element.
 2. Call `.screenshot()` on that locator.
 3. Do **not** pass the `fullPage` option here — it's not needed when capturing a single element.
-

Example: Capture a Specific `<div>` Element

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

test('Element screenshot', async ({ page }) => {
  await page.goto('https://your-website.com');

  const element = page.locator('#product div div');
  await element.screenshot({
    path: 'tests/screenshots/ProductSection_' + Date.now() + '.png'
  });
});
```

Notes:

- Make sure the element is **visible** on the page before taking a screenshot.
 - Use CSS selectors (`#id`, `.class`, etc.) or XPath depending on your needs, but Playwright recommends CSS for better performance.
 - **Don't use `fullPage: true`**: when capturing an element — it's only for full-page screenshots.
-

Additional Configuration for Capturing Screenshots Automatically

If you want Playwright to automatically capture screenshots for all tests **without adding code in each test**, you can configure it in the **Playwright configuration file**. This will allow you to capture screenshots for all test executions, and they will be included in your report as well.

Steps to Automatically Capture Screenshots for All Tests

1. Open your `playwright.config.js` file.
2. Add the `screenshot: 'on'` option inside the `use` block.
3. After this, you **don't need to add the screenshot code** in each test. Playwright will automatically capture screenshots for every test.

Example Configuration

```
// playwright.config.js
module.exports = {
  use: {
    screenshot: 'on', // Automatically capture screenshots for every test
  },
};
```

Screenshots in Reports

- Once you've set this up, Playwright will save the screenshots in the **results folder**.
- Screenshots will also be included in the **test report**.

Viewing the Report and Screenshots

After test execution, you can view the report and the screenshots with this command:

```
npx playwright show-report
```

This will open the Playwright report in your browser, where you can see the screenshots for each test.

Key Notes:

- This configuration captures **screenshots for failed tests by default**.
- You can also customize the behavior (e.g., capturing screenshots for all tests, not just failures) by adjusting the configuration.

Customizing Screenshot Behavior in Playwright

By default, Playwright only captures screenshots for **failed tests**. However, you can easily customize the behavior to capture screenshots for **all tests**, or even customize when the screenshots are taken based on specific conditions.

1. Capture Screenshots for All Tests (Passed and Failed)

To capture screenshots for every test (both **passed** and **failed**), modify your `playwright.config.js` like this:

```
// playwright.config.js
module.exports = {
  use: [
    screenshot: 'only-on-failure', // Default behavior: capture
    screenshots only on failure
  ],
  projects: [
    {
      name: 'test-all-screenshots',
      use: [
        screenshot: 'on', // Capture screenshots for every test (even for
        passed tests)
      ],
    },
  ],
};
```

- `screenshot: 'on'` will capture a screenshot for **every test**, regardless of whether it passes or fails.
- You can apply this setting globally or per-project (as shown above).

2. Capture Screenshots Based on Test Results

You can also set Playwright to capture screenshots **only when a test fails** or under certain conditions (e.g., specific tests or suites). By doing this, screenshots are only taken when things go wrong, helping to keep the number of screenshots manageable.

```
// playwright.config.js
module.exports = {
  use: [
    screenshot: 'only-on-failure', // Only capture screenshots for failed
    tests
  ],
};
```

- `screenshot: 'only-on-failure'` is the default option. It ensures screenshots are only captured when a test fails, which helps reduce unnecessary screenshots in reports.

3. Advanced: Capture Screenshots Based on Custom Conditions

You can also dynamically capture screenshots by adding custom conditions inside your tests using hooks such as `beforeEach` and `afterEach`.

Example: Capture Screenshots in Specific Test Hooks

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

test.describe('Custom screenshot conditions', () => {
  test('Test with screenshot on failure', async ({ page }) => {
    await page.goto('https://your-website.com');
    // Some actions here
    // Screenshot will be captured automatically if the test fails
  });

  test.afterEach(async ({ page, testInfo }) => {
    if (testInfo.status === 'failed') {
      // Capture screenshot only if the test fails
      await page.screenshot({
        path: `results/screenshots/failed-test-${Date.now()}.png`,
      });
    }
  });
});
```

- Here, the screenshot is captured **only when a test fails**, even if `screenshot: 'on'` is globally set to capture for all tests.

4. Customize the Location of Screenshots

You can also customize the folder where screenshots are stored, using dynamic paths based on the test name, status, or timestamp. This helps in organizing the screenshots for easy reference.

Example: Save Screenshots in Custom Folders

```
// playwright.config.js
module.exports = {
  use: {
    screenshot: 'on', // Capture screenshots for all tests
    screenshotsPath: 'custom/screenshots-folder/', // Custom folder for
screenshots
  },
};
```

This will ensure screenshots for all tests are stored in the `custom/screenshots-folder/`.

Key Notes:

- `screenshot: 'on'` captures screenshots for all tests.
 - `screenshot: 'only-on-failure'` captures screenshots only when tests fail (default behavior).
 - Custom conditions (e.g., test status) can be added via hooks like `afterEach`.
 - You can organize screenshots with custom folder paths for better organization.
-

Playwright Screenshot Functions Overview

1. Capture Screenshot of the Page:

- Function: `page.screenshot()`
- Captures a screenshot of the entire page.

Example: `await page.screenshot({ path: 'HomePage.png' });`

2. Capture Full Page Screenshot:

- Function: `page.screenshot()`
- Use the `fullPage` option to capture the entire scrollable page.

Example: `await page.screenshot({ path: 'HomePageFull.png', fullPage: true });`

3. Capture Screenshot of a Specific Element:

- Function: `elementLocator.screenshot()`
- Capture a screenshot of a specific element (e.g., `div`, `button`).

Example: `await page.locator('#product').screenshot({ path: 'Product.png' });`

4. Automatically Capture Screenshots for All Tests:

- Configuration: `playwright.config.js`
- Add `screenshot: 'on'` to capture screenshots for **all** tests.

Example:

```
use: {  
  screenshot: 'on',  
}
```

5. Capture Screenshots Only on Test Failure:

- Configuration: `playwright.config.js`
- Default behavior to capture screenshots only on failures.

Example:

```
use: {
  screenshot: 'only-on-failure',
}
```

6. Advanced: Custom Screenshot Conditions (Using Hooks):

- Function: `test.afterEach()`
- Capture screenshots conditionally based on test status (e.g., only on failure).

Example:

```
test.afterEach(async ({ page, testInfo }) => {
  if (testInfo.status === 'failed') {
    await page.screenshot({ path: 'failed-test.png' });
  }
});
```

Key Functions & Properties:

- **Functions:**
 - `page.screenshot()` – Capture screenshot of the page or a specific element.
 - `elementLocator.screenshot()` – Capture screenshot of a specific element.
 - **Configuration:**
 - `playwright.config.js` – Global configuration for automatic screenshot capture.
 - **Options:**
 - `fullPage: true` – Capture entire page.
 - `screenshot: 'on'` – Capture screenshots for all tests.
 - `screenshot: 'only-on-failure'` – Capture screenshots only on test failure.
-