3.Taking Screenshots on Test Failure

**Introduction**

Playwright offers the ability to capture screenshots automatically when a test fails. This feature is extremely useful for debugging and diagnosing issues by providing a visual representation of the page state at the point of failure.

* **Feature**: Automatically take a screenshot when a test fails.
  + Helps in identifying UI issues or unexpected changes in the application.
  + Can be configured globally or per test.

**Configuring Screenshots on Failure**

You can configure Playwright to capture screenshots when tests fail by setting the screenshot option in the Playwright configuration file (playwright.config.ts).

* **Global Configuration**: Enable screenshots on failure for all tests.

**Config Example**:

typescript

Copy code

import { defineConfig } from '@playwright/test';

export default defineConfig({

use: {

screenshot: 'only-on-failure',

},

});

* **Screenshot Options**:
  + off: Disables screenshot capture.
  + on: Takes screenshots for every test.
  + only-on-failure: Captures screenshots only when a test fails (recommended for debugging).

**Test-Level Configuration**

If you want to capture screenshots for specific tests only on failure, you can configure it at the test level using the use option inside the test definition.

* **Per-Test Configuration**: Specify the screenshot setting for individual tests.

**Test Example**:

typescript

Copy code

import { test } from '@playwright/test';

test.use({ screenshot: 'only-on-failure' });

test('example test', async ({ page }) => {

await page.goto('https://playwright.dev');

// Intentionally cause a failure

await expect(page).toHaveTitle('Wrong Title');

});

* **Use Case**: Capture a screenshot only when critical tests fail, ensuring a clear view of the page state at the time of failure.

**Where Screenshots Are Saved**

When a test fails and a screenshot is captured, Playwright saves the screenshot in the test-results folder by default. The folder is structured to include:

* **Directory Structure**:
  + Each test run creates a unique folder containing failure logs and screenshots.
  + Screenshots are named after the test file and the browser used.

**Example Folder Structure**:

markdown

Copy code

test-results/

└── example.spec.ts/

└── example-test-1-failure.png

* **Custom Screenshot Path**: You can customize where the screenshots are saved by configuring the outputDir option in the Playwright configuration.

**Config Example**:

typescript

Copy code

export default defineConfig({

use: {

screenshot: 'only-on-failure',

outputDir: 'screenshots', // Custom directory for screenshots

},

});

**Example: Capturing Screenshots on Failure**

Here’s an example test that demonstrates how Playwright captures a screenshot when the test fails.

**Test Example**:

typescript

Copy code

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

test('failing test with screenshot', async ({ page }) => {

await page.goto('https://playwright.dev');

// This assertion will fail

await expect(page).toHaveTitle('Nonexistent Title');

});

* **What Happens**:
  + Playwright will capture a screenshot of the page right before the failure.
  + The screenshot will be saved in the test-results folder (or custom path, if specified).

**Conclusion**

Capturing screenshots on test failure in Playwright is a powerful feature that aids in debugging and issue resolution. By configuring screenshots globally or per test, and managing where these screenshots are saved, you can have a visual record of failures, which significantly improves the ability to diagnose UI issues.

Would you like more details on screenshot customization or additional debugging techniques? Let me know!

4o

**You said:**