Isolation

Introduction

Tests written with Playwright execute in isolated clean-slate environments called browser contexts. This isolation model improves reproducibility and prevents cascading test failures.

What is Test Isolation?

Test Isolation is when each test is completely isolated from another test. Every test runs independently from any other test. This means that each test has its own local storage, session storage, cookies etc. Playwright achieves this using BrowserContexts which are equivalent to incognito-like profiles. They are fast and cheap to create and are completely isolated, even when running in a single browser. Playwright creates a context for each test, and provides a default Page in that context.

Why is Test Isolation Important?

- No failure carry-over. If one test fails it doesn't affect the other test.
- Easy to debug errors or flakiness, because you can run just a single test as many times as you'd like.
- Don't have to think about the order when running in parallel, sharding, etc.

Two Ways of Test Isolation

There are two different strategies when it comes to Test Isolation: start from scratch or cleanup in between. The problem with cleaning up in between tests is that it can be easy to forget to clean up and some things are impossible to clean up such as "visited links". State from one test can leak into the next test which could cause your test to fail and make debugging harder as the problem comes from another test. Starting from scratch means everything is new, so if the test fails you only have to look within that test to debug.

How Playwright Achieves Test Isolation

Playwright uses browser contexts to achieve Test Isolation. Each test has its own Browser Context. Running the test creates a new browser context each time. When using Playwright as a Test Runner, browser contexts are created by default. Otherwise, you can create browser contexts manually.

Sync **Async**

```
browser = playwright.chromium.launch()
context = browser.new_context()
page = context.new_page()
```

Browser contexts can also be used to emulate multi-page scenarios involving mobile devices, permissions, locale and color scheme. Check out our Emulation guide for more details.

Multiple Contexts in a Single Test

Playwright can create multiple browser contexts within a single scenario. This is useful when you want to test for multi-user functionality, like a chat.

Sync **Async**

```
from playwright.sync_api import sync_playwright, Playwright

def run(playwright: Playwright):
    # create a chromium browser instance
    chromium = playwright.chromium
    browser = chromium.launch()

    # create two isolated browser contexts
    user_context = browser.new_context()
    admin_context = browser.new_context()

    # create pages and interact with contexts independently

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