@Created By : Kushal Parikh | SDET
Checkout my topmate - **https://topmate.io/kushalparikh11**

How to Create Your First Script in Playwright: Step-by-Step Guide

Playwright is a powerful, modern testing library developed by Microsoft, which enables developers to automate browsers with a robust API and cross-browser support. Whether you are automating user interactions or testing web applications, Playwright provides a reliable and efficient solution. In this guide, we'll walk you through creating your first automated script in Playwright using JavaScript.

#Prerequisites

Before starting, ensure that you have the following installed:

- Node.js: Ensure Node.js is installed on your system. You can download it from nodejs.org.

- Package Manager: npm (included with Node.js) or yarn.

- Text Editor or IDE: Use an IDE like Visual Studio Code for a smooth coding experience.

Step 1: Set Up Your Project

1. Create a New Project Folder:

   Open your terminal and create a new project directory:

   mkdir playwright-demo

   cd playwright-demo

2. Initialize Node.js Project:

   Initialize a new Node.js project by running:

   npm init -y

3. Install Playwright:

@Created By : Kushal Parikh | SDET

Checkout my topmate - https://topmate.io/kushalparikh11

Install Playwright as a dependency:

npm init playwright@latest

Playwright will download the browsers needed as well as create the following files.

playwright.config.ts
package.json
package-lock.json
tests/
  example.spec.ts
tests-examples/
  demo-todo-app.spec.ts

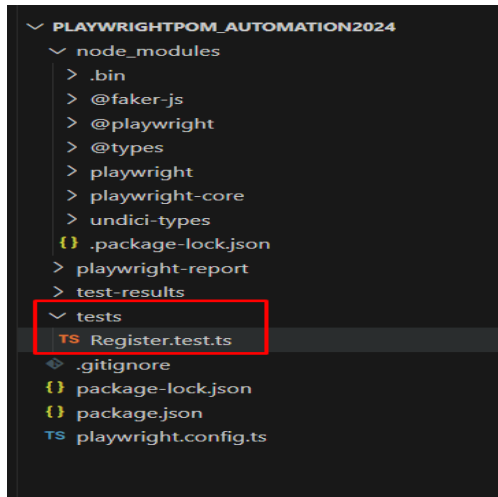4. Install Faker.js:

To generate random data, install Faker.js:

npm install @faker-js/faker

Step 2: Create Your First Test Script

Create a file called `RegisterTest.js` in your project folder under tests folder and add the following code:


```typescript
import { chromium, test } from "@playwright/test";
import { faker } from '@faker-js/faker';

// Variables to store random data and page context
let randomFirstName, randomLastName, randomEmail, randomPhoneNumber;
let browser, context, page;

// Set up the browser before all tests
test.beforeAll(async () => {
    browser = await chromium.launch({ headless: false });
    context = await browser.newContext();
    page = await context.newPage();
});

// Generate random data before each test
test.beforeEach(async () => {
    randomFirstName = faker.name.firstName();
    randomLastName = faker.name.lastName();
    randomEmail = faker.internet.email();
    randomPhoneNumber = faker.phone.number();
});

// Close the browser after all tests
test.afterAll(async () => {
    await browser.close();
});

// Define the test
test("Register a new account", async () => {
    await page.goto("https://ecommerce-playground.lambdatest.io/");
    await page.hover("//a[@role='button']//span[@class='title'][normalize-space()='My account']");
    await page.click("//a//span[normalize-space()='Register']");
```

```javascript
    // Fill in the form with random data
    await page.fill("#input-firstname", randomFirstName);
    await page.fill("#input-lastname", randomLastName);
    await page.fill("#input-email", randomEmail);
    await page.fill("#input-telephone", randomPhoneNumber);
    await page.fill("#input-password", "Test@123");
    await page.fill("#input-confirm", "Test@123");
    await page.getByText('Yes').click();
    await page.getByText('I have read and agree to the').click();
    await page.getByRole('button', { name: 'Continue' }).click();

    // Validate account creation
    const validationMSG = await page.locator("//h1[normalize-space()='Your Account Has Been
Created!']").textContent();

    if (validationMSG && validationMSG.includes("Created")) {
        console.log("Account creation successful!");
    } else {
        console.error("Account creation failed!");
    }
});
```

Step 3: Run the Script

To run your Playwright test, execute the following command in your terminal:

npx playwright test registerTest.js

This command will launch a browser, navigate to the specified URL, perform the automated interactions, and log the outcome.

Step 4: Understand the Code

- Imports:

  - `chromium` and `test` are imported from Playwright to create browser instances and define test cases.

  - `faker` is imported to generate dynamic data for the form fields.

- Hooks:

  - `test.beforeAll()` initializes the browser and page context.

 - `test.beforeEach()` generates fresh random data before each test run.

 - `test.afterAll()` ensures the browser is closed after tests complete.

- Test Steps:

  - Navigate to the registration page.

  - Fill out the form with generated data.

  - Validate the success message to confirm account creation.

Tips for Writing Playwright Tests

- Use Selectors Effectively: Use robust and maintainable selectors (e.g., `getByRole`, `getByText`).

- Run in Headless Mode: Set `headless: true` for faster, non-UI runs in CI environments.

- Use Assertions: Integrate assertions (`expect`) for clear test validations.

With these steps, you've successfully created and run your first Playwright test script. Playwright's simplicity and rich feature set make it an excellent choice for automating end-to-end tests. Expand on this script by adding more test cases, handling edge scenarios, or exploring browser contexts for deeper automation.