Playwright Automation Mastery

The Comprehensive Basic to Advanced Guide

Authored by Abhishesh Mishra

1. □ Installation & Setup

Getting Started

```
npm init playwright@latest
npm install -D @playwright/test
npx playwright install
npx playwright install chromium firefox webkit
```

2. □ Test Execution Commands

CLI Usage

```
npx playwright test                        # Run all tests
npx playwright test --headed               # Run in headed mode
npx playwright test --ui                   # Launch UI Mode
npx playwright test --debug                # Launch Debugger
npx playwright test --project=chromium # Run specific browser
npx playwright test tests/test.spec.ts # Run specific file
```

3. □ Browser & Context Management

Browser Instances

```
const browser = await chromium.launch();
const browserHeaded = await chromium.launch({ headless: false });
await browser.close();
```

□ Browser Context

```
const context = await browser.newContext();
const authContext = await browser.newContext({ storageState: 'state
    .json' });
await context.storageState({ path: 'state.json' });
```

## 4. 	 Page Handling & Navigation

Basic Navigation

```
const page = await context.newPage();
await page.goto('https://example.com');
await page.reload();
await page.goBack();
await page.goForward();
await page.close();
```

## 5. 	 Multiple Tabs / Windows

Handling New Pages

```
const [newPage] = await Promise.all([
  context.waitForEvent('page'),
  page.click('a[target=_blank]')
]);
```

## 6. 	 Locators (All Types)

Finding Elements

```
page.locator('css-selector');
page.locator('//xpath-selector');
page.getByText('Login');
page.getByRole('button', { name: 'Submit' });
page.getByLabel('Email');
page.getByPlaceholder('Enter Search Query');
page.getByTestId('submit-btn');
```

## 7. 	 User Interactions

Basic Actions

```
await locator.click();
await locator.fill('Standard text input');
await locator.type('Slow typing simulation');
await locator.clear();
await locator.check();
await locator.uncheck();
await locator.selectOption('Value_1');
```

8. □ Advanced Mouse & Keyboard

Precision Interactions

```
await page.dblclick('selector');
await page.hover('selector');
await page.dragAndDrop('#source', '#target');
await page.mouse.move(100, 200);

// Keyboard
await page.press('#input', 'Enter');
await page.keyboard.type('Hello World');
await page.keyboard.press('Control+A');
await page.keyboard.press('Backspace');
```

9. □ Waiting & Synchronization

Essential Waits

```
await page.waitForSelector('.loading-complete');
await page.waitForLoadState('networkidle');
await page.waitForURL('**/dashboard');
await page.waitForTimeout(3000);
```

10. □ Assertions (Expect)

Validation

```
await expect(page).toHaveURL('https://site.com/home');
await expect(page).toHaveTitle('Welcome');
await expect(locator).toBeVisible();
await expect(locator).toBeHidden();
await expect(locator).toHaveText('Success');
await expect(locator).toContainText('Part of string');
await expect(locator).toBeEnabled();
```

11. □ Frames, iFrames & Dialogs

Embedded Content

```
const frame = page.frame({ name: 'mainFrame' });
await frame.locator('#btn').click();

page.on('dialog', dialog => dialog.accept());
page.on('dialog', dialog => dialog.dismiss());
```

12. □ File Operations

Uploads & Downloads

```
await page.setInputFiles('#upload', 'resume.pdf');

const downloadPromise = page.waitForEvent('download');
await page.getByText('Download File').click();
const download = await downloadPromise;
await download.saveAs('local_file.pdf');
```

13. □ Authentication & Storage

Cookies & State

```
await context.addCookies([{ name: 'session', value: '123' }]);
const cookies = await context.cookies();
await context.clearCookies();
await context.storageState({ path: 'auth.json' });
```

14. □ API Testing & Mocking

Network Level

```
const response = await request.get('/api/v1/users');
expect(response.status()).toBe(200);

await page.route('**/api/**', route => {
  route.fulfill({ status: 200, body: JSON.stringify({ data: 'mocked
    ' }) });
});

await page.route('**/*.{png,jpg,jpeg}', route => route.abort());
```

15. □ Test Hooks & Configuration

Lifecycle

```
test.beforeAll(async () => { });
test.beforeEach(async ({ page }) => { });
test.afterEach(async () => { });

test.describe.configure({ mode: 'parallel', retries: 2 });
```

16. □ Debugging & Reporting

Insights

```
npx playwright test --trace on
npx playwright show-trace trace.zip
npx playwright show-report
```

17. □ PRO Interview Commands

Expert Features

- □ **test.only():** Isolates a single test for execution.
- □ **test.skip():** Skips a test based on conditions.
- □ **test.fail():** Explicitly marks a test as ”expected to fail”.
- □ **test.slow():** Increases the timeout specifically by 3x.
- □ **cross-env ENV=qa:** Use environment variables for targeting.

End of Professional Playwright Cheatsheet