Playwright Automation: Answering Your Most Common Questions

Doc by Aston Cook

Introduction

Over the past few months, I've received hundreds of questions about Playwright from manual testers and QA engineers looking to level up their automation skills. Today, I'm answering the most common ones to help you understand if Playwright is right for your automation journey.

Whether you're just starting with test automation or considering switching from Selenium, this Q&A will give you the clarity you need.

Q1: What exactly is Playwright, and why is everyone talking about it?

A: Playwright is a modern test automation framework created by Microsoft that lets you automate web applications across Chrome, Firefox, and Safari. It's gaining massive popularity because it's faster, more reliable, and easier to use than older tools like Selenium. The best part? It has built-in features that make writing stable tests much simpler.

Q2: I only know manual testing. Is Playwright too advanced for me?

A: Not at all! Playwright is actually one of the most beginner-friendly automation frameworks available. If you can write basic JavaScript or TypeScript, you can write Playwright tests. The syntax is clean and readable, and you don't need years of coding experience to get started. Many manual testers I've worked with were writing their first automated tests within a week.

Q3: Do I need to learn JavaScript first, or can I jump straight into Playwright?

A: You need basic JavaScript or TypeScript fundamentals, but you don't need to be an expert. Understanding variables, functions, loops, and basic async/await concepts will get you started. The good news? You can learn these basics in parallel with Playwright since you'll be using them immediately in real tests.

Q4: What makes Playwright better than Selenium?

A: Playwright has several key advantages:

- **Auto-waiting:** Playwright automatically waits for elements to be ready before interacting with them, eliminating most flaky tests
- **Speed:** Tests run significantly faster than Selenium
- **Modern features:** Built-in support for network interception, screenshots, videos, and mobile emulation
- **Multiple browsers:** One test runs across Chrome, Firefox, and Safari without changes
- **Better debugging:** Playwright Inspector makes troubleshooting much easier

Q5: Can I test APIs with Playwright, or is it just for UI testing?

A: Yes! Playwright has excellent built-in API testing capabilities. You can make API calls, validate responses, and even combine API and UI tests in the same framework. This is perfect for end-to-end scenarios where you need to set up data via API and then test the UI.

Q6: How long does it take to build a real automation framework with Playwright?

A: A basic framework can be built in a few hours, but a production-ready framework with proper page objects, test data management, reporting, and CI/CD integration typically takes 1-2 weeks of focused work. The key is starting with good structure from the beginning rather than retrofitting later.

Q7: What kind of applications can I test with Playwright?

A: Playwright can test virtually any web application:

- E-commerce sites
- SaaS platforms
- Admin dashboards
- Social media sites
- Banking applications
- Educational platforms
- Any modern web app

It handles single-page applications (SPAs) particularly well, which often cause issues in older frameworks.

Q8: Do I need to set up a lot of configuration before I can start testing?

A: Playwright has minimal setup requirements. After installing Node.js, you can have your first test running in about 5 minutes. The npm init playwright command sets up everything you need with sensible defaults. You can start testing immediately and optimize the configuration later.

Q9: Can I run Playwright tests in CI/CD pipelines?

A: Absolutely! Playwright is designed for CI/CD integration. It works seamlessly with:

- GitHub Actions

- Jenkins
- GitLab CI
- Azure DevOps
- CircleCI
- Any CI/CD platform

Tests can run in headless mode for faster execution, and Playwright generates reports that integrate perfectly with your pipelines.

Q10: How do I handle common testing scenarios like login, file uploads, and dropdowns?

A: Playwright makes these scenarios straightforward:

- **Login:** Store authentication state once and reuse it across tests
- **File uploads:** Simple file input handling with setInputFiles()
- **Dropdowns:** Easy selection with selectOption() method
- **Iframes:** Built-in iframe support without complex switching
- **Multiple tabs:** Native handling of new tabs and windows

Q11: What if my tests become flaky? How do I debug them?

A: Playwright has exceptional debugging tools:

- **Playwright Inspector:** Step through your test line by line
- **Trace Viewer:** Record and replay your test execution with full context
- **Screenshots:** Automatic screenshots on failure
- **Videos:** Record full test execution
- **Console logs:** Capture browser console output

These tools make finding and fixing issues much faster than traditional debugging methods.

Q12: Can Playwright help me build a portfolio to land a job?

A: Yes! A well-structured Playwright framework in your GitHub portfolio demonstrates:

- Modern automation skills employers want
- Ability to write clean, maintainable code
- Understanding of testing best practices
- Experience with CI/CD integration
- Problem-solving abilities

Many hiring managers specifically look for Playwright experience now since it's becoming an industry standard.

Q13: Is Playwright suitable for both small projects and enterprise applications?

A: Playwright scales beautifully from small projects to enterprise applications. You can start with simple tests and grow into a comprehensive framework with parallel execution, multiple environments, advanced reporting, and complex workflows. Major companies like Microsoft, VS Code, and Bing use Playwright in production.

Q14: What are the system requirements? Do I need a powerful computer?

A: Playwright runs on modest hardware:

- Windows, Mac, or Linux
- 8GB RAM minimum (16GB recommended)
- Modern processor
- Node.js 16+

You don't need a high-end development machine to write and run Playwright tests effectively.

Q15: Where should I start if I want to learn Playwright properly?

A: Start with these steps:

- Learn basic JavaScript/TypeScript fundamentals
- Install Playwright and run the example tests
- Practice on simple applications like TodoMVC
- Build a small framework with page objects
- Add reporting and CI/CD integration
- Test a real e-commerce application end-to-end

The key is hands-on practice with real applications, not just tutorials.

Key Takeaways

- Playwright is beginner-friendly and perfect for manual testers transitioning to automation
- It's faster, more reliable, and more modern than Selenium
- You can test both UI and APIs in the same framework
- Built-in debugging tools make troubleshooting simple
- Playwright skills are in high demand and impressive in portfolios
- You can start with minimal setup and scale as you learn

Ready to Get Started?

Playwright is the future of web test automation, and now is the perfect time to start learning. Whether you're transitioning from manual testing or upgrading from Selenium, Playwright will make you a more effective and valuable QA engineer.

Tomorrow, I'll be sharing something special for everyone who wants to master Playwright with real, production-ready projects. Stay tuned!

Created by Aston Cook