Website: awesome-testing.com

Contact: https://awesome-testing.com/contact

Playwright + LLM + MCP for API and UI Testing

Hands-on training that combines Playwright, MCP server integrations, and AI tools to accelerate exploratory and automated testing.

Level: Advanced

Duration: 2 days

Mode: remote, onsite

Languages: PL, EN

Tooling: ChatGPT, Cursor, Copilot, Codex, Playwright, MCP, JavaScript, TypeScript, Playwright MCP, AI agents

Details: https://awesome-testing.com/workshops/playwright-llm-mcp

Key outcomes

- Connect AI tools with Playwright through MCP and tool calling
- Automate API and UI test flows with controlled AI assistance
- Evaluate productivity gains and reliability risks in AI-driven testing

Full program

Learning objective

This training shows how to connect a Model Context Protocol (MCP) server for Playwright with AI tooling (e.g., GitHub Copilot / Cursor) and modern LLM approaches such as tool calling (function calling) and Retrieval-Augmented Generation (RAG). The goal is to accelerate exploratory testing and the development of automated tests for web applications.

By the end of the course, participants will be able to:

- Explain what tool calling (function calling) and RAG are, and how they apply to software testing.
- Describe the role of MCP and its architecture in the Playwright ecosystem.
- Run LLM-assisted exploratory testing sessions.
- Generate and maintain Playwright tests (UI and API) with AI assistance.
- Assess when AI support improves productivity and when it increases risk.

Agenda

1) Modern LLM techniques

- LLM fundamentals (how tools like ChatGPT operate)
- Tool calling / tools: connecting the model to external capabilities
- Retrieval-Augmented Generation (RAG): improving answer quality using internal documentation
- AI agents: concepts, theory, and practical applications

2) MCP and Playwright

- MCP server architecture
- Installation, configuration, and integration with Playwright (JS/TS)
- API overview: accessibility snapshots, navigation, and element actions

3) MCP in hands-on testing

- Prompt-driven exploratory testing
- Generating test cases from user stories

4) Test automation with AI

- Using Copilot / Cursor for test automation
- Connecting Copilot / Cursor with MCP: a "prompt → code → run → fix" workflow
- Practical workshops:
  - Automating REST API checks with Playwright
  - Automating UI tests with MCP + Playwright
- Productivity review: what to write manually vs what to generate

5) Risks, ethics, and maintainability

- Model hallucinations vs test stability and quality
- Versioning prompts and ensuring reproducibility
- Cost analysis and return on investment
- MCP and Playwright roadmap: what to watch next

Prerequisites

Who is this for?

The course is intended for testers who have basic programming skills in JavaScript/TypeScript and understand Playwright fundamentals. Familiarity with Git is recommended.

What to prepare

- A laptop with Node.js (LTS), Git, and VS Code
- An AI coding tool configured (e.g., GitHub Copilot or Cursor)

A detailed setup checklist is provided after registration.

Teaching methods

Workshops (~65%) plus lecture-style explanation (~35%). Each practical block ends with a short retrospective and knowledge sharing. Participants leave with working code and a prompt history.

Training materials

- Slide deck (PDF)
- Git repository with branches for each module
- Prompt notebook (Markdown)
- A curated board with links and theory

Benefits

- Practical skill: controlling a browser via MCP, increasingly in demand.
- Faster automation development with AI support.
- Stronger exploratory testing with AI as an assistant.
- Better risk management: checklists and guidance on when not to trust LLM outputs.
- Knowledge sharing with peers: prompts, templates, and repository patterns.