CERTIFICATION SYLLABUS

Certified Playwright Automation Specialist

Exam Code: CST-PLAY™



A complete preparation guide covering exam objectives, core competencies, domain weightage, and topics to prepare.

©2026 Certaining, Inc. All rights reserved.

Table of Contents

1. **Exam Name & Code** 1
2. **Table of Contents** 2
3. **Objective of the Certification** 3
4. **Core Competencies** 4-6
   - *Playwright Fundamentals & Architecture*
   - *Locators & Element Interaction*
   - *Test Framework Design & Fixtures*
   - *Test Execution Management*
   - *Cross-Browser Testing, Contexts & Network Control*
   - *CI/CD Integration & DevOps Practices*
   - *Reporting, Tracing & Debugging*
   - *Best Practices, Maintenance & Scalability*
5. **Who Should Take This Certification** 7
6. **Exam Information** 7
7. **Examination Domain & Weightage** 8
8. **Topics to Prepare** 9-10
   - *Playwright Fundamentals & Architecture*
   - *Locators & Element Interaction*
   - *Test Framework Design & Fixtures*
   - *Test Execution Management*
   - *Cross-Browser Testing, Contexts & Network Control*
   - *CI/CD Integration & DevOps Practices*
   - *Reporting, Tracing & Debugging*
   - *Best Practices, Maintenance & Scalability*

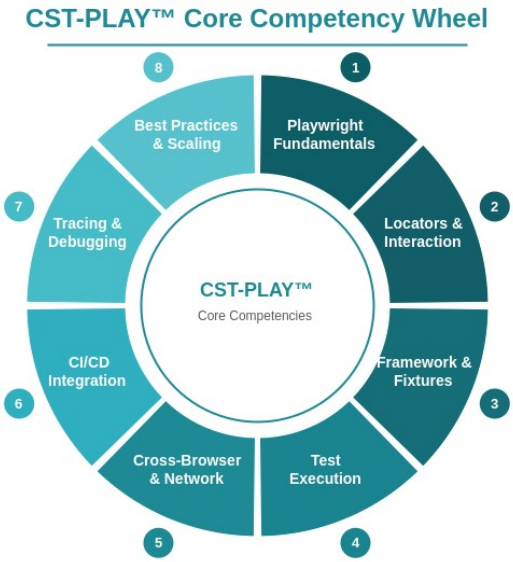
©2026 Certaining, Inc. All rights reserved.

Objective of the Certification

- **Build a strong foundation in Playwright architecture —** understanding how Playwright controls Chromium, Firefox, and WebKit through a single API, and how it differs from driver-based tools.
- **Develop proficiency in locators and element interaction —** reliably identifying and interacting with elements using role-based, text, label, and CSS/XPath locators.
- **Establish competency in test framework design —** building maintainable suites using the Playwright Test runner, fixtures, and the Page Object Model (POM).
- **Build skills in test execution management —** configuring projects, retries, tagging, data-driven tests, and parallel execution.
- **Develop cross-browser and multi-context testing capability —** running tests across browsers, devices, and isolated browser contexts, including network interception and mocking.
- **Introduce CI/CD integration practices —** running Playwright suites in pipelines (e.g., GitHub Actions, Jenkins, Azure DevOps) using headless mode, sharding, and containers.
- **Build competency in reporting, tracing, and debugging —** using HTML reports, Trace Viewer, Inspector, screenshots, and video to diagnose failures efficiently.
- **Establish best practices for stability and scalability —** eliminating flaky tests, handling asynchronous behavior, and keeping suites fast and maintainable as applications grow.
- **Prepare candidates to contribute effectively to automation teams —** equipping them to build, extend, and maintain real-world end-to-end test suites.
- **Create a foundation for progression toward advanced automation and SDET roles —** such as framework architecture, visual and performance testing, or automation leadership.

©2026 Certaining, Inc. All rights reserved.

Core Competencies — CST-PLAY™ (Playwright Automation Specialist)



©2026 Certaining, Inc. All rights reserved.

1. Playwright Fundamentals & Architecture

- Explain Playwright's architecture, including browsers, browser contexts, and pages.
- Understand how Playwright's auto-waiting and single cross-browser API differ from traditional automation tools.
- Set up and configure a Playwright project (installation, browsers, configuration file, language bindings).

2. Locators & Element Interaction

- Apply locator strategies (role, text, label, test ID, CSS, XPath) and choose the most resilient option.
- Rely on auto-waiting and actionability checks instead of manual sleeps, and use web-first assertions.
- Interact with complex UI elements: dropdowns, file uploads, dialogs, frames, multiple tabs/pages, and keyboard/mouse actions.

3. Test Framework Design & Fixtures

- Structure tests with the Playwright Test runner (test hooks, annotations, grouping).
- Create and use fixtures for reusable setup, authentication state, and test data.
- Implement the Page Object Model to separate test logic from page structure.

4. Test Execution Management

- Configure projects, retries, timeouts, and tags to control which tests run and how.
- Implement data-driven tests using external sources (JSON, CSV) and parameterization.
- Configure parallel execution and sharding to reduce overall suite runtime.

5. Cross-Browser Testing, Contexts & Network Control

- Execute tests across Chromium, Firefox, and WebKit, plus mobile device emulation.
- Use isolated browser contexts to simulate multiple users and sessions in a single test.
- Intercept, mock, and modify network requests, and test APIs using Playwright's request context.

6. CI/CD Integration & DevOps Practices

- Run Playwright suites in CI/CD pipelines (e.g., GitHub Actions, Jenkins, Azure DevOps, GitLab CI).
- Configure headless execution, containers, artifacts, and sharded runs for pipeline efficiency.
- Understand how end-to-end automation fits into a broader DevOps/continuous testing strategy.

©2026 Certaining, Inc. All rights reserved.

7. Reporting, Tracing & Debugging

- Generate and interpret reports (HTML report, Allure, JUnit) and capture screenshots and video.
- Use Trace Viewer, Inspector, and codegen to investigate and reproduce failures.
- Debug common failures: locator ambiguity, timing issues, strict-mode violations, environment differences.

8. Best Practices, Maintenance & Scalability

- Apply best practices for readable, reusable, and maintainable test code.
- Reduce flakiness through stable locators, isolated test data, and independent tests.
- Plan for suite growth as application UI and test coverage expand.

©2026 Certaining, Inc. All rights reserved.

Who Should Take This Certification

- Manual QA testers transitioning into modern web test automation.
- SDETs building or maintaining end-to-end automation frameworks.
- QA Automation Engineers moving from Selenium or other tools to Playwright.
- Front-end and full-stack developers writing end-to-end tests as part of Agile/DevOps teams.
- Automation leads/architects evaluating or standardizing team automation tooling.
- Individuals who have completed CSTF™ (or hold equivalent foundational testing knowledge) and want to specialize in Playwright.

Prerequisite: None mandatory. Completion of CSTF™ (Software Testing Foundations) or equivalent practical testing experience is recommended.

Exam Information

Exam Name	Playwright Automation Specialist
Exam Code	CST-PLAY™
Exam Format	Multiple Choice Questions (MCQ) & Code-Based/Practical Scenario Questions
Duration	120 minutes
Number of Questions	75
Passing Score	70%
Mode of Exam	Test Center Based Testing / Online Testing
Language	English
Validity	Lifetime
Open Book	No

©2026 Certaining, Inc. All rights reserved.

Examination Domain & Weightage

Domain	Weightage
Playwright Fundamentals & Architecture	15%
Locators & Element Interaction	15%
Test Framework Design & Fixtures	20%
Test Execution Management	15%
Cross-Browser Testing, Contexts & Network Control	10%
CI/CD Integration & DevOps Practices	10%
Reporting, Tracing & Debugging	8%
Best Practices, Maintenance & Scalability	7%
Total	**100%**

©2026 Certaining, Inc. All rights reserved.

Topics to Prepare

The topics below represent the core areas candidates should focus on for each domain, reflecting the practical, hands-on depth expected at the CST-PLAY™ level.

1. Playwright Fundamentals & Architecture

- **Core Concepts:** Browser, browser context, and page, and how they relate.
- **Auto-Waiting:** How Playwright waits for elements to be actionable before acting.
- **Environment Setup:** Installing Playwright, browsers, and configuring playwright.config.

2. Locators & Element Interaction

- **Locator Types:** getByRole, getByText, getByLabel, getByTestId, CSS, and XPath.
- **Web-First Assertions:** Auto-retrying assertions such as toBeVisible and toHaveText.
- **Complex UI Handling:** Dropdowns, file uploads, dialogs, frames, multiple pages, and keyboard/mouse actions.

3. Test Framework Design & Fixtures

- **Playwright Test Runner:** Test structure, hooks, annotations, and grouping.
- **Fixtures:** Built-in and custom fixtures for setup, teardown, and shared state.
- **Page Object Model:** Structuring page classes to separate test logic from locators.
- **Authentication State:** Reusing saved login state across tests.

4. Test Execution Management

- **Configuration:** Projects, retries, timeouts, and tags/annotations.
- **Data-Driven Testing:** Parameterizing tests with JSON, CSV, or other data sources.
- **Parallel Execution & Sharding:** Running tests in parallel and splitting suites across machines.

5. Cross-Browser Testing, Contexts & Network Control

- **Multi-Browser Execution:** Running on Chromium, Firefox, and WebKit.
- **Device Emulation:** Mobile viewports, geolocation, and permissions.
- **Browser Contexts:** Isolated sessions for multi-user scenarios.
- **Network Handling:** Request interception, mocking, and API testing with the request context.

6. CI/CD Integration & DevOps Practices

- **Pipeline Integration:** Incorporating Playwright into GitHub Actions, Jenkins, Azure DevOps, or GitLab CI.

©2026 Certaining, Inc. All rights reserved.

- **Headless & Containers:** Running tests in headless mode and Docker containers.
- **Artifacts & Triggers:** Publishing reports and traces; running on commit, schedule, or pre-deployment gates.

7. Reporting, Tracing & Debugging

- **Reporting:** HTML, Allure, and JUnit reports.
- **Trace Viewer:** Recording and analyzing traces to diagnose failures.
- **Debug Tools:** Inspector, codegen, screenshots, and video capture.
- **Common Failures:** Strict-mode violations, timing issues, and environment-related errors.

8. Best Practices, Maintenance & Scalability

- **Code Quality:** Writing readable, reusable, and maintainable test code.
- **Flakiness Reduction:** Stable locators, isolated data, and independent tests.
- **Framework Evolution:** Planning for growth as the application UI and coverage expand.

©2026 Certaining, Inc. All rights reserved.