Selenium vs. Cypress vs. Playwright

Automation framework assessment table:

	Cypress	**Selenium**	**Playwright**
Framework Type	End-to-end testing framework for web test automation [1]	Library [2]	End-to-end testing framework for web test automation [48]
License	Open Source [3]	Open Source [4]	Open Source [49]
Ideal for	simple and large scale and complex websites. Testing in mobile native environment. [5]	simple and large scale and complex websites [6]	simple and large scale and complex websites. [50] Testing in mobile emulated environment. [51]
Learning curve	Faster than Selenium [7]	Steep learning curve [8]	Faster than Selenium [52]
Popularity (Compare package download counts over time)	4 165 468 npm trends / 20 November 2022 [9]	2 652 431 npm trends / 20 November 2022 [10]	995 555 npm trends / 20 November 2022 [53]
Coding speed	Fast [11]	Normal [12]	Fast [54]
Performance tests capability	Yes, and can combine it with UI script (the API and the UI browser test have the same session) [13]	No [14]	Yes, and can combine it with UI script (the API and the UI browser test have the same session) [55]
Manipulate the DOM tree	Yes [15]	No [16]	No, but can use JS or TS workarounds. [56]
Can handle asynchronous data	Yes [17]	No [18]	Yes [57]
Waits	Integrated waits into the commands. No need support from the QA side. [19]	Implicit, Explicit and Fluent. Need support from the QA side. [20]	It auto-waits for all the relevant checks to pass and only then performs the requested action. [58]
Combine methods/commands by default	Yes [21]	Not at all [22]	No [59]
Working with iFrames	Yes (with Plugin), [23] but nested iFrames can't be handled. [24]	Yes [25]	Yes [60]
Working with tabs	No [26] (we can still get the URL where the user is redirected) [27]	Yes [28]	Yes [61]
Test Scope	Unit [43], security [44], integration [45], end-to-end testing [46]	end-to-end testing [47]	integration [62], end-to-end testing [63]

Language Support	JavaScript [41] and TypeScript [42]	C#, JavaScript, Java, Python, Ruby, Groovy, Perl, Scala and PHP [40]	JavaScript and TypeScript, Python, Java, .NET [65]
Support	The support can be automated completely. [31]	With a lot of manual work from the QA side. [32]	The support can be automated completely. [66]
Future-proof	There is good described and supporting documentation. [33] A lot of free and paid courses. [34] There is already a community with a lot of questions and answers into the Stack Overflow site. [35]	There is good written documentation. [36] There is a huge community on the internet. [37] Many free courses. [38] Selenium is compatible with all popular OOP languages. [39]	A new and growing technology [67] Supported documentation [68] Community with a questions and answers into the Stack Overflow site [69] A lot of free and paid courses in Internet

Recommendation:
To continue using Playwright because of the testing framework and readiness to start with Automation within a week. Playwright is asynchronous by nature, so this will help a lot to automate the system because the FE of the system is based on React. React has an asynchronous nature. The BE of the system is written on JS, Playwright is supporting JS (and TS). The speed of the coding is really faster than using Selenium. The framework is ready to be used.

References (from the comparison table):

[1] - https://www.cypress.io/blog/2019/02/05/modern-frontend-testing-with-cypress/#header

[2] – https://www.selenium.dev/documentation/webdriver/_print/#requirements-by-language

[4] - https://docs.cypress.io/faq/questions/general-questions-faq#Is-Cypress-free-and-open-source

[5] - https://en.wikipedia.org/wiki/Selenium_(software)

- the opinion is based on the QAs experience.

[6] - the opinion is based on the QAs' experience.

[7] - the opinion is based on the QAs' experience.

[8] - the opinion is based on the QAs' experience.

[9] - https://www.npmtrends.com/cypress-vs-selenium-webdriver

[10] - https://www.npmtrends.com/cypress-vs-selenium-webdriver

[11] - the opinion is based on the QAs experience.

[12] - the opinion is based on the QAs experience.

[13] - https://example.cypress.io/cypress-api

[14] - https://www.selenium.dev/documentation/test_practices/discouraged/performance_testing/

[15] - https://docs.cypress.io/api/commands/invoke

[16] - https://www.selenium.dev/documentation/legacy/json_wire_protocol/

[17] - https://docs.cypress.io/guides/core-concepts/introduction-to-Cypress#Commands-Are-Asynchronous

[18] - https://www.selenium.dev/documentation/webdriver/waits/

[19] - https://docs.cypress.io/api/commands/wait#Nesting

[20] - https://www.selenium.dev/documentation/webdriver/waits/

[21] - https://docs.cypress.io/guides/core-concepts/introduction-to-cypress#Chains-of-Commands

[22] - https://www.selenium.dev/selenium/docs/api/py/webdriver/selenium.webdriver.common.action_chains.html

[23] - https://www.npmjs.com/package/cypress-iframe

[24] - https://github.com/cypress-io/cypress/issues/7437

[25] - https://www.selenium.dev/documentation/webdriver/browser/frames/

[26] - https://docs.cypress.io/guides/references/trade-offs#Multiple-tabs

[27] - https://docs.cypress.io/api/commands/window#No-Args

[28] - https://www.selenium.dev/documentation/webdriver/browser/windows/

[31] - https://github.com/cypress-io/cypress/pull/8751

[32] - https://www.selenium.dev/documentation/webdriver/getting_started/install_drivers/#1-driver-management-software

[33] - https://docs.cypress.io/api/commands/window

[34] - https://www.google.com/search?q=cypress+courses&oq=cypress+courses&aqs=chrome..69i57.1944j0j4&sourceid=chrome&ie=UTF-8

[35] - https://stackoverflow.com/questions/tagged/cypress

[36] - https://www.selenium.dev/documentation/

[37] - https://www.selenium.dev/documentation/webdriver/waits/

[38] - https://www.google.com/search?q=selenium+courses&oq=selenium+courses&aqs=chrome..69i57.5223j0j4&sourceid=chrome&ie=UTF-8

[39] - https://www.selenium.dev/documentation/legacy/selenium_1/#programming-your-test

[40] - https://en.wikipedia.org/wiki/Selenium_(software)

[41] - https://docs.cypress.io/guides/overview/why-cypress#Who-uses-Cypress

[42] - https://docs.cypress.io/guides/tooling/typescript-support

[43] - https://docs.cypress.io/examples/examples/recipes#Unit-Testing

[44] - https://docs.cypress.io/guides/guides/web-security#Limitations

[45] - https://docs.cypress.io/guides/guides/module-api

[46] - https://docs.cypress.io/examples/examples/workshop#End-to-end-Testing-with-Cypress-io

[47] - https://www.selenium.dev/documentation/webdriver/getting_started/first_script/

[48] - https://playwright.dev/docs/intro

[49] – https://github.com/microsoft/playwright/blob/main/LICENSE

[50] – https://playwright.dev/docs/intro

[51] – https://playwright.dev/docs/emulation

[52] – the opinion is based on the QAs' experience.

[53] – https://npmtrends.com/playwright

[54] – the opinion is based on the QAs' experience.

[55] – https://playwright.dev/docs/test-api-testing

[56] – https://github.com/microsoft/playwright/issues/4924

[57] – https://playwright.bootcss.com/python/docs/core-concepts#browser

[58] – https://playwright.bootcss.com/python/docs/actionability?_highlight=wait

[59] – Playwright uses normal behavior for creating methods.

[60] – https://playwright.bootcss.com/docs/api/class-frame#frameframe_element

[61] – https://playwright.dev/docs/api/class-browsercontext#browser-context-pages

[62] – https://playwright.dev/docs/test-api-testing

[63] – https://playwright.dev/docs/intro

[65] – https://playwright.dev/docs/languages

[66] – https://www.programsbuzz.com/article/how-update-playwright-version

[67] – https://npmtrends.com/playwright

[68] – https://playwright.dev/docs/intro

[69] – https://stackoverflow.com/questions/tagged/playwright