



WebDriver BiDi Protocol

The future of cross-browser automation

New York USA

London UK

Munich Germany

Zug Switzerland

About me

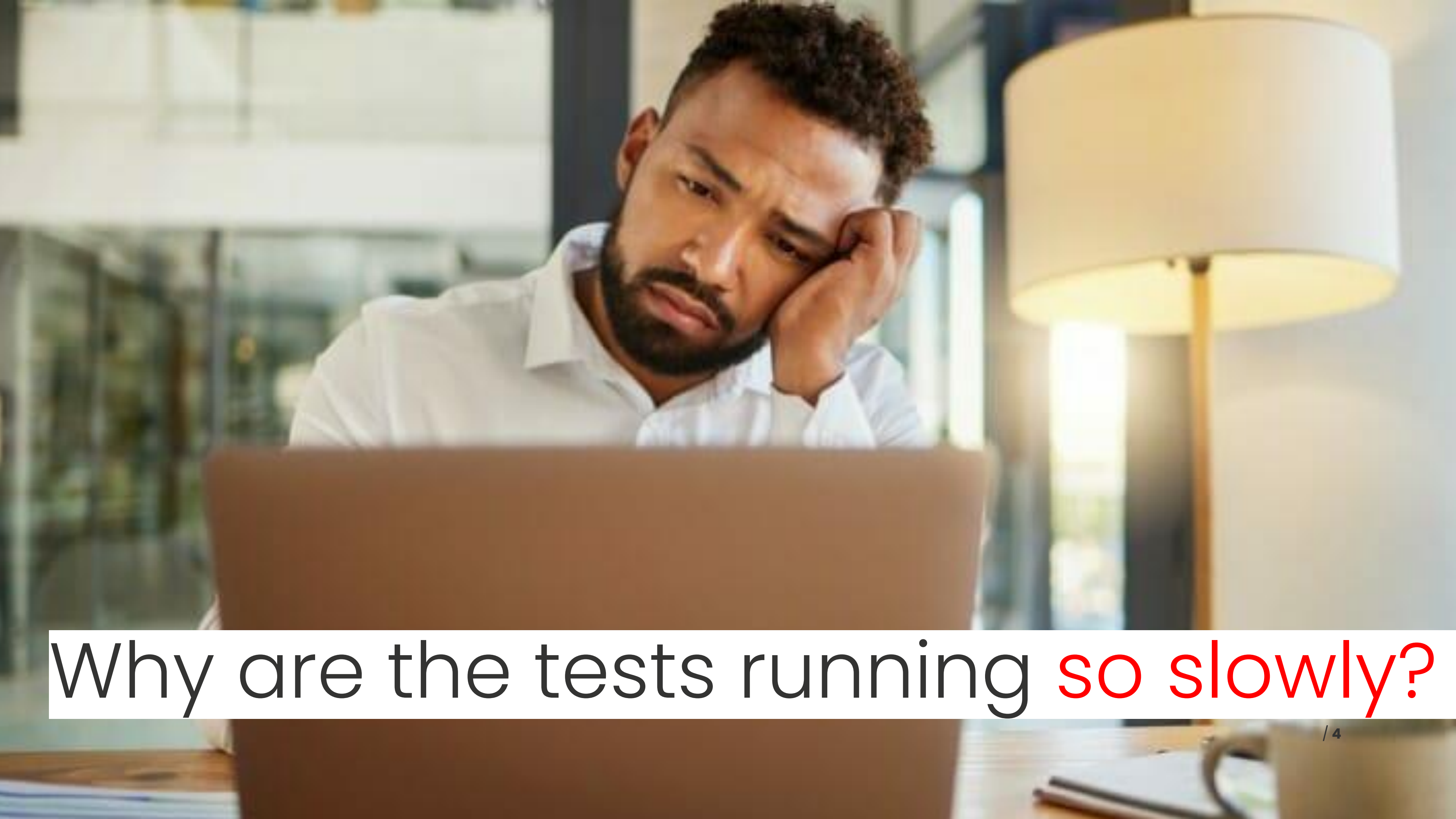


- **Mohammad Monfared**
- Test Automation Architect / Team Lead
- +12 Years in IT
- +8 Years in QA/QAA (Py/JS)
- Mentor (top 10 in QA | top 1% in Engineering – adplist)
- YouTuber / Blogger (~12K on LinkedIn)
- Cypress ambassador
- <https://monfared.io>



Agenda

- WebDriver Specification
- Web Platform Tests
- The history of test automation
- WebDriver Ecosystem
- ChromeDevTools
- Categorize test automation tools
- WebDriver Classic vs ChromeDevTools
- WebDriver BiDi Specification
- WebDriver Bidi Protocol
- Features of WebDriver BiDi
- Demo
- Q&A



Why are the tests running so slowly?

W3C Specifications (Standards)

TABLE OF CONTENTS	
	Abstract
	Status of This Document
1.	Design
1.1	Compatibility
1.2	Simplicity
1.3	Extensions
2.	Conformance
3.	Terminology
4.	Interface
5.	Nodes
6.	Protocol
6.1	Algorithms
6.2	Commands
6.3	Processing model
6.4	Routing requests
6.5	Endpoints
6.6	Errors
6.7	Extensions
7.	Capabilities
7.1	Proxy
7.2	Processing capabilities
8.	Sessions

WebDriver

W3C Working Draft 14 November 2023



▼ More details about this document

This version:

<https://www.w3.org/TR/2023/WD-webdriver2-20231114/>

Latest published version:

<https://www.w3.org/TR/webdriver2/>

Latest editor's draft:

<https://w3c.github.io/webdriver/>

History:

<https://www.w3.org/standards/history/webdriver2/>

[Commit history](#)

Test suite:

<https://wpt.live/webdriver/>

Implementation report:

<https://wpt.fyi/results/webdriver>

Editors:

[Simon Stewart](#) (Apple)

[David Burns](#) (BrowserStack)

Feedback:

[GitHub w3c/webdriver](#) (pull requests, new issue, open issues)

Channel

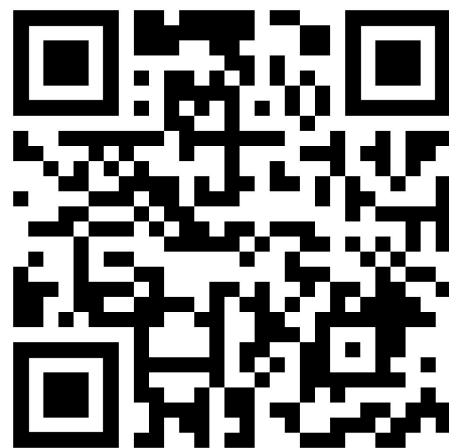
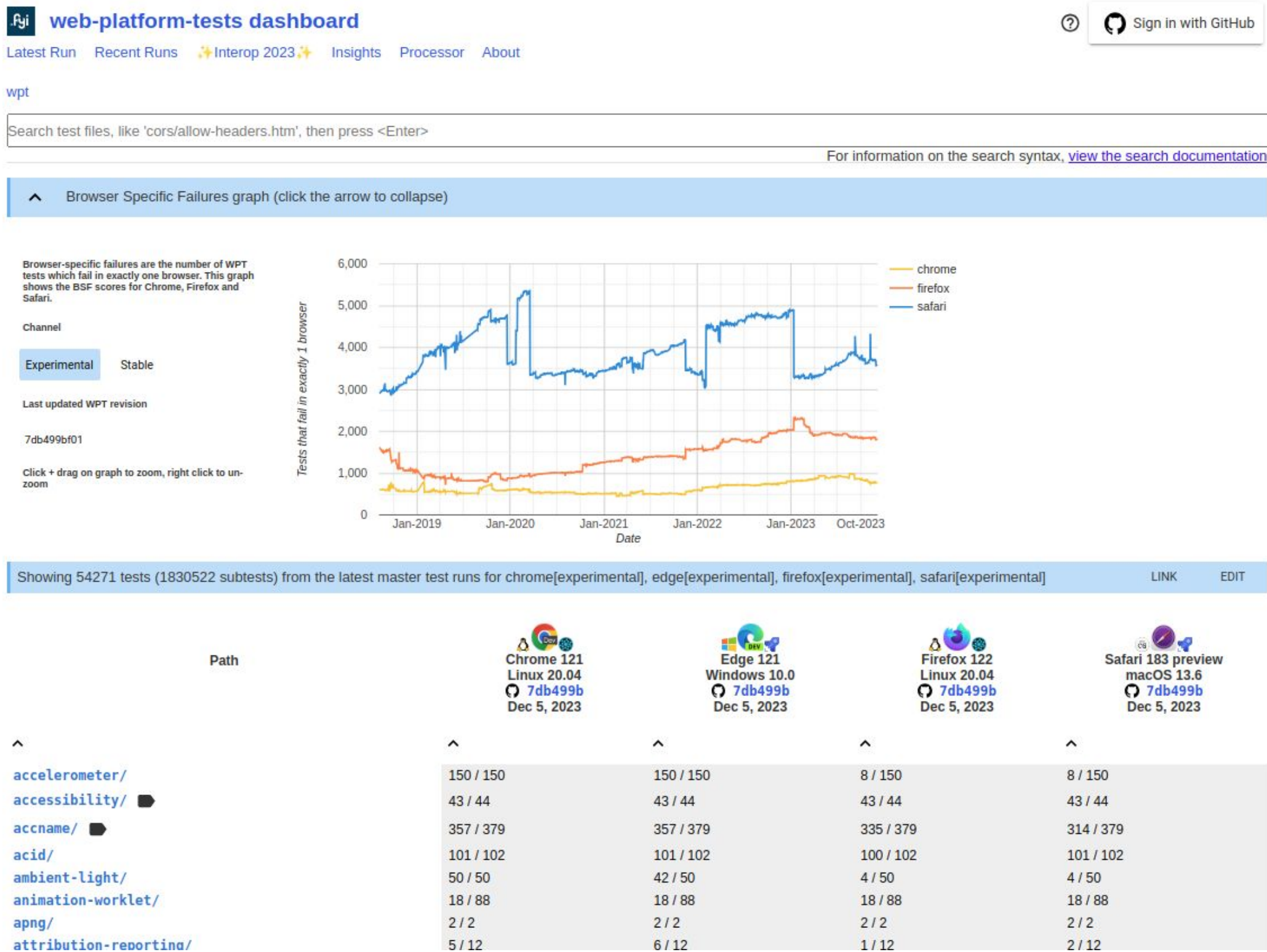
[#webdriver on irc.w3.org](#)

Copyright © 2023 World Wide Web Consortium. W3C® [liability](#), [trademark](#) and [permissive document license](#) rules apply.



[WebDriver Classic](#)

Web Platform Tests

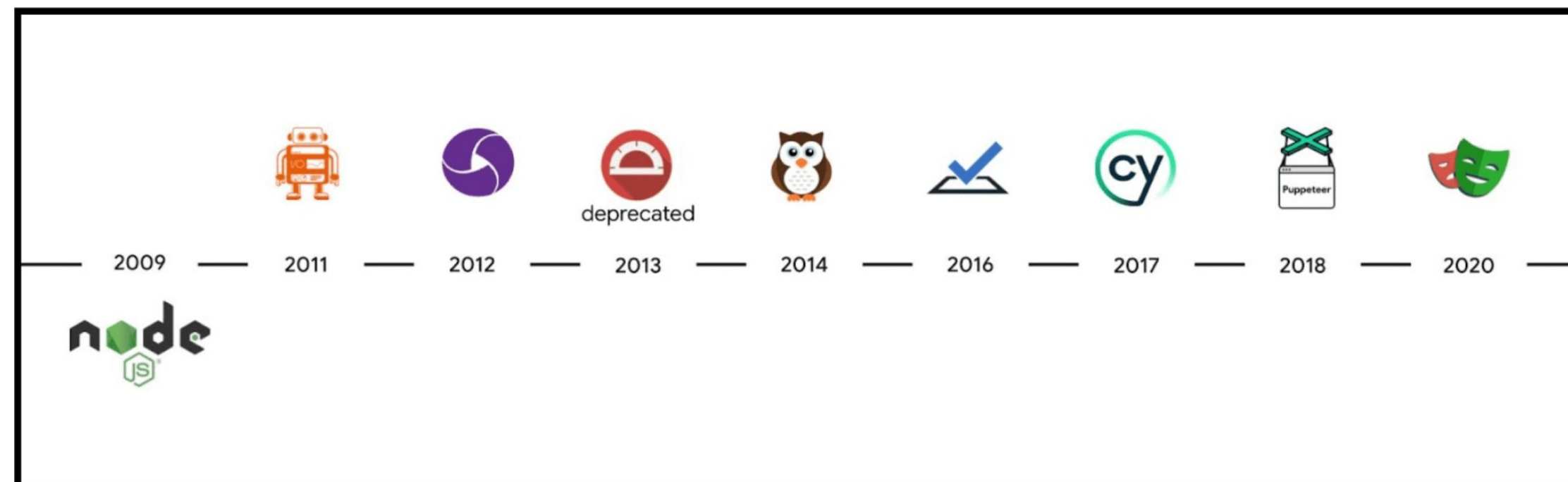
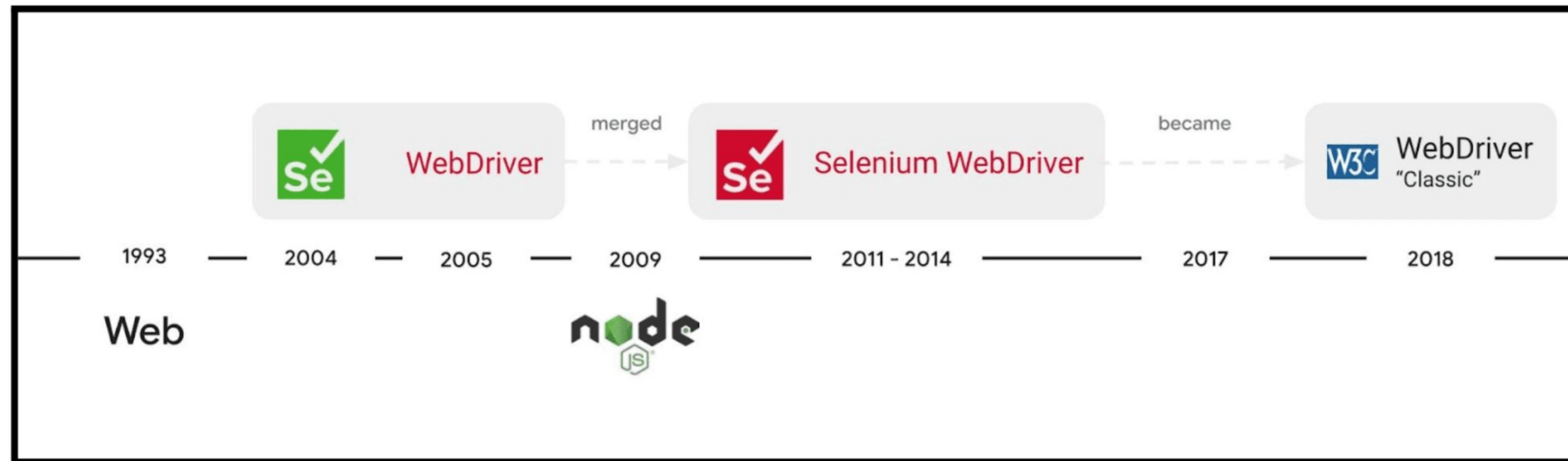


[Web Platform Tests](#)

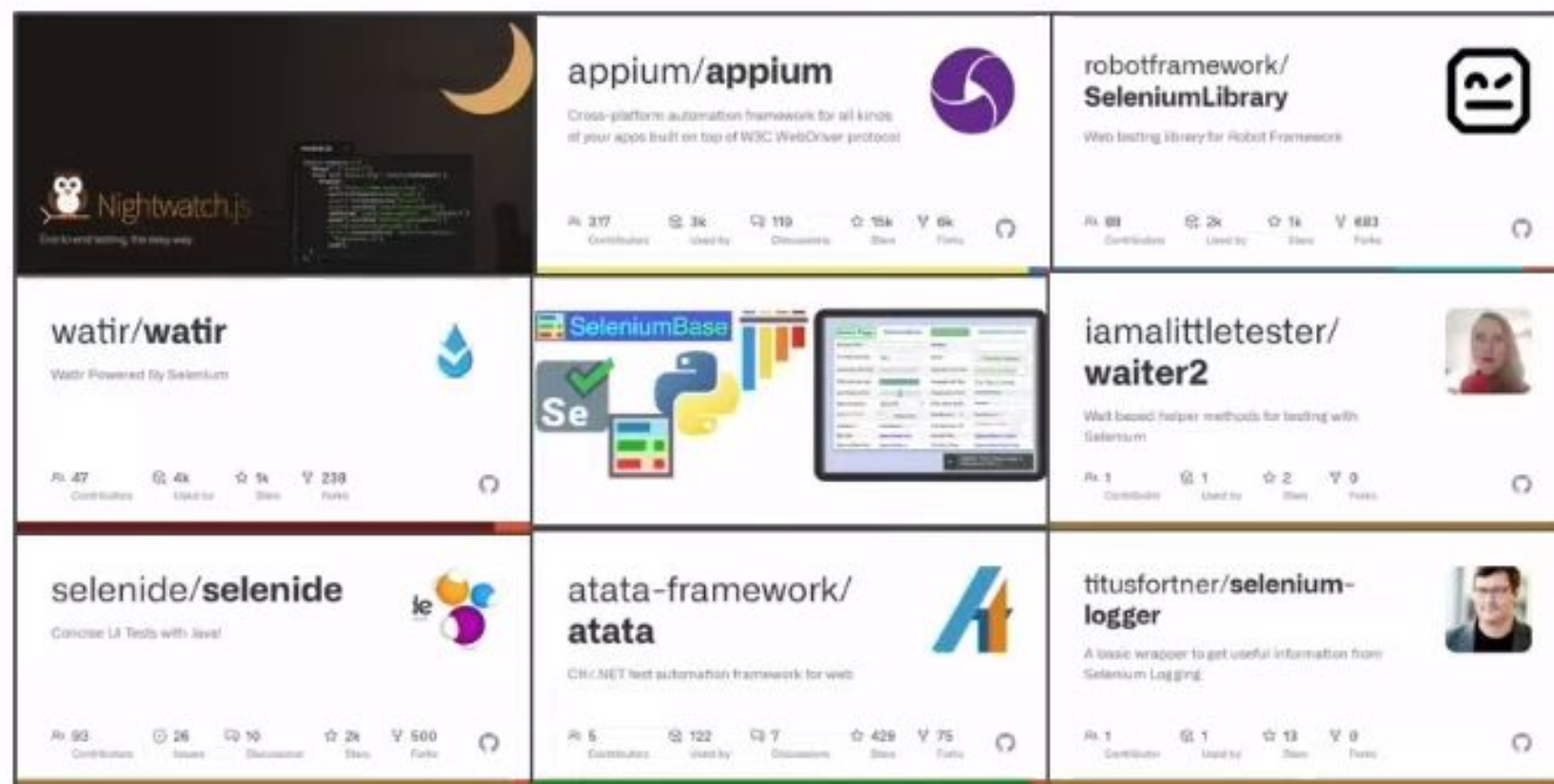


[Web Platform Tests Dashboard](#)

The history of test automation



WebDriver Ecosystem



Implementation



webdriverio/webdriverio



Next-gen browser and mobile automation test framework for Node.js

425 Contributors 43k Used by 202 Discussions 8k Stars 2k Forks

php-webdriver/php-webdriver



PHP client for Selenium/WebDriver protocol. Previously facebook/php-webdriver

86 Contributors 16k Used by 77 Discussions 5k Stars 848 Forks

Specification

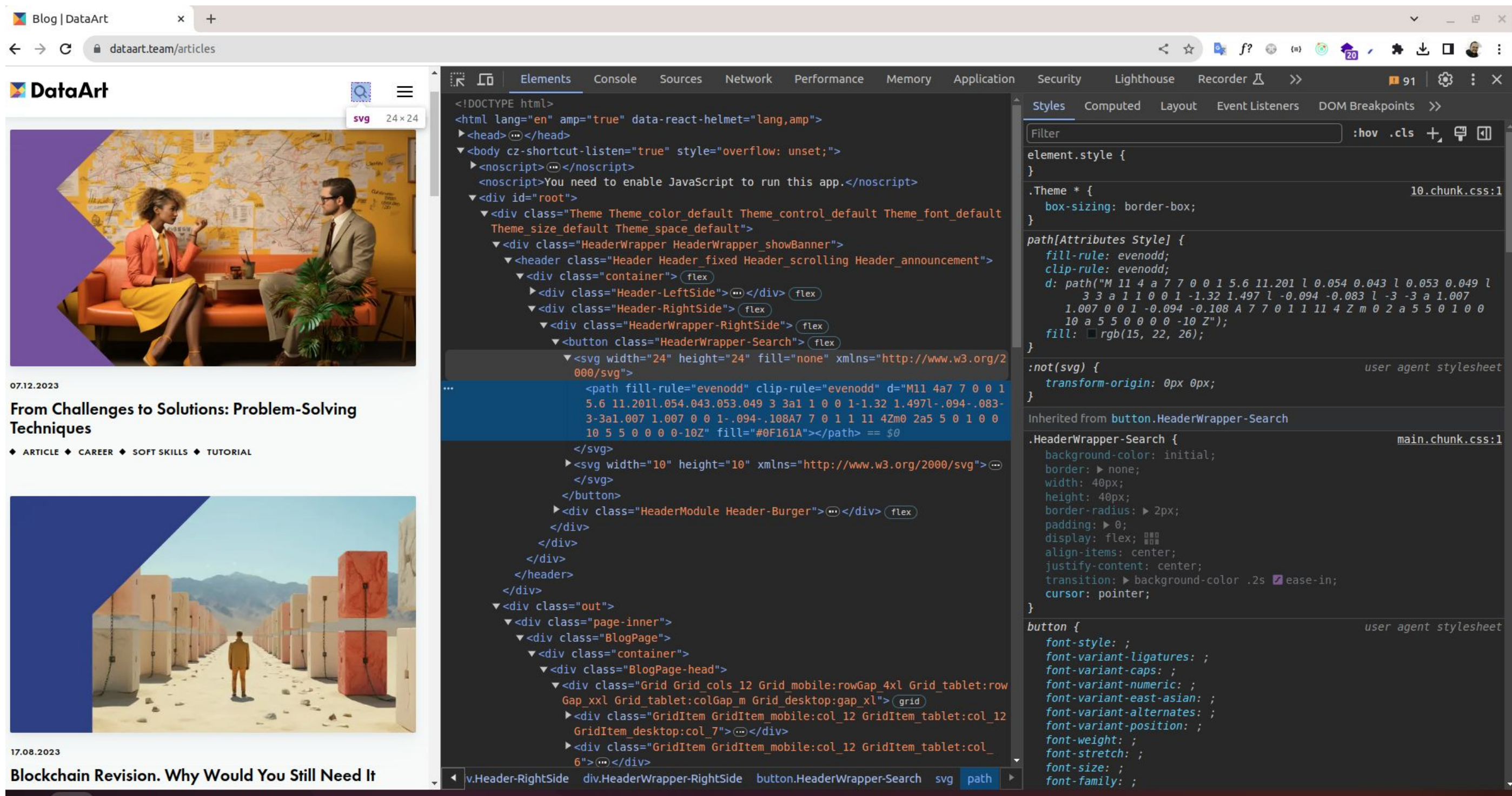
w3c/webdriver



Remote control interface that enables introspection and control of user agents.

63 Contributors 239 Issues 631 Stars 199 Forks

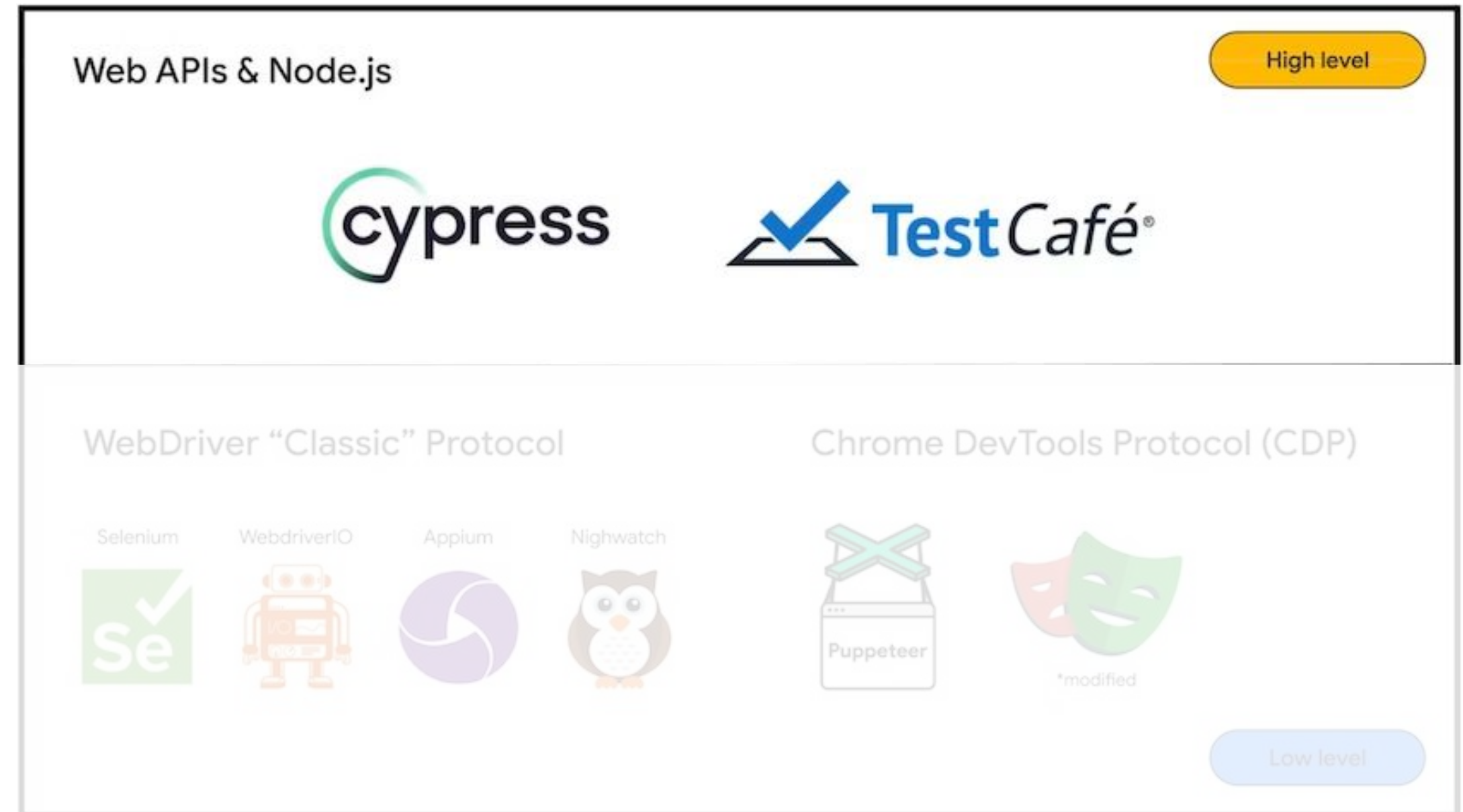
What is ChromeDevTools?



Two groups of browser automation tools



High Level: Execute JavaScript within the browser using web APIs and Node.js. (e.g. **Cypress**)

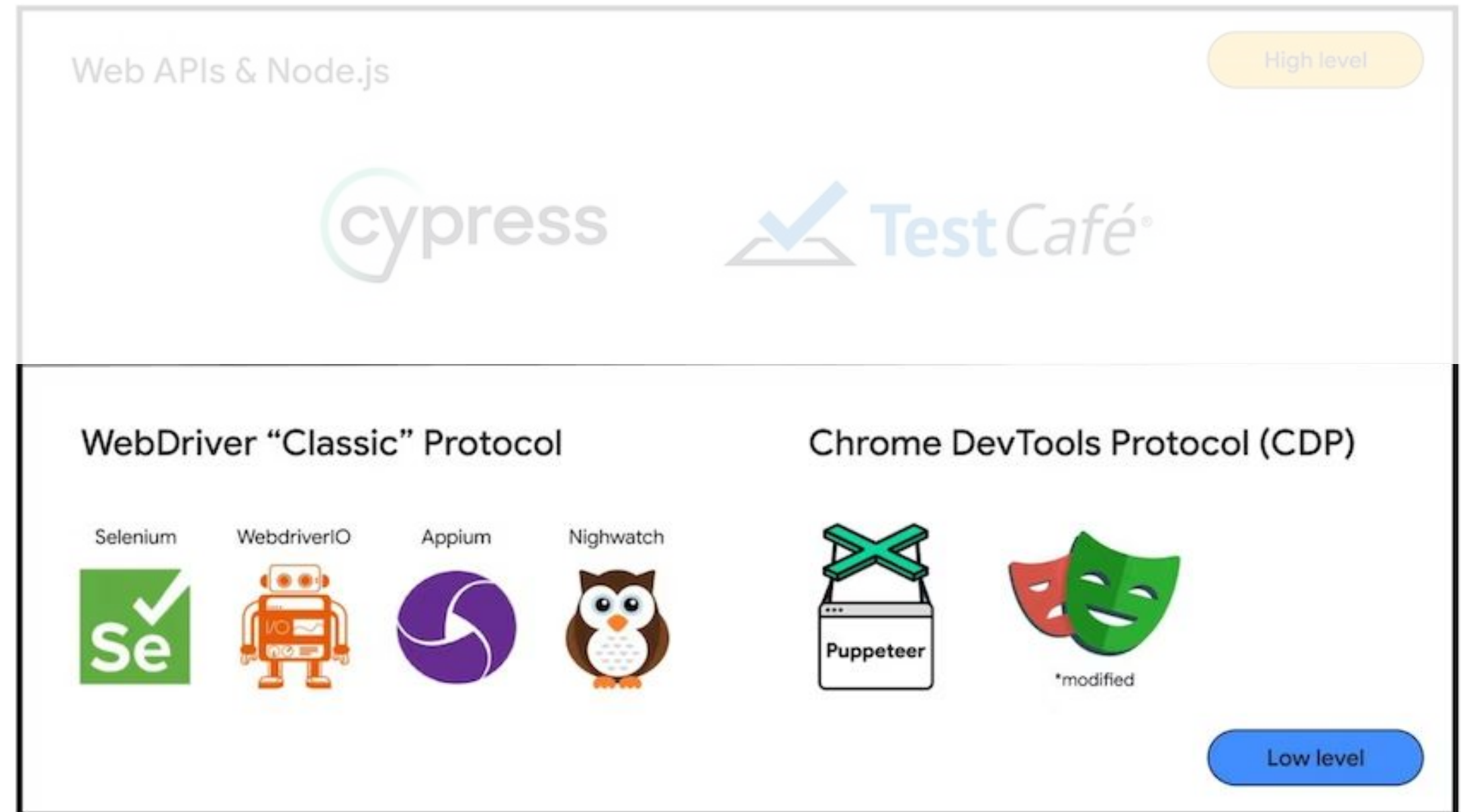


Two groups of browser automation tools

High Level: Execute JavaScript within the browser using web APIs and Node.js. (e.g. **Cypress**)

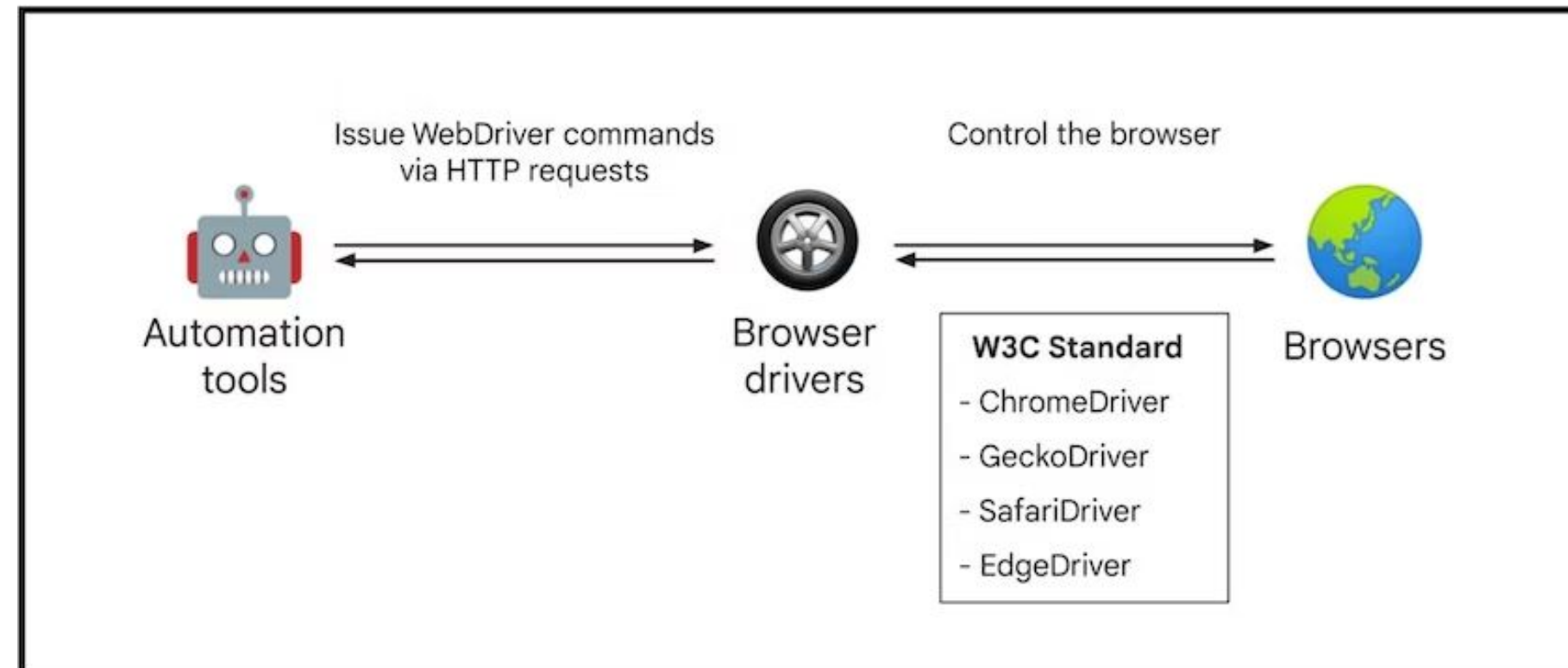
Low Level: Run tests outside of browser using either:

- **WebDriver classic** (e.g. **Selenium**)
- **ChromeDevTools** which is faster than WebDriver (e.g. **Playwright**)

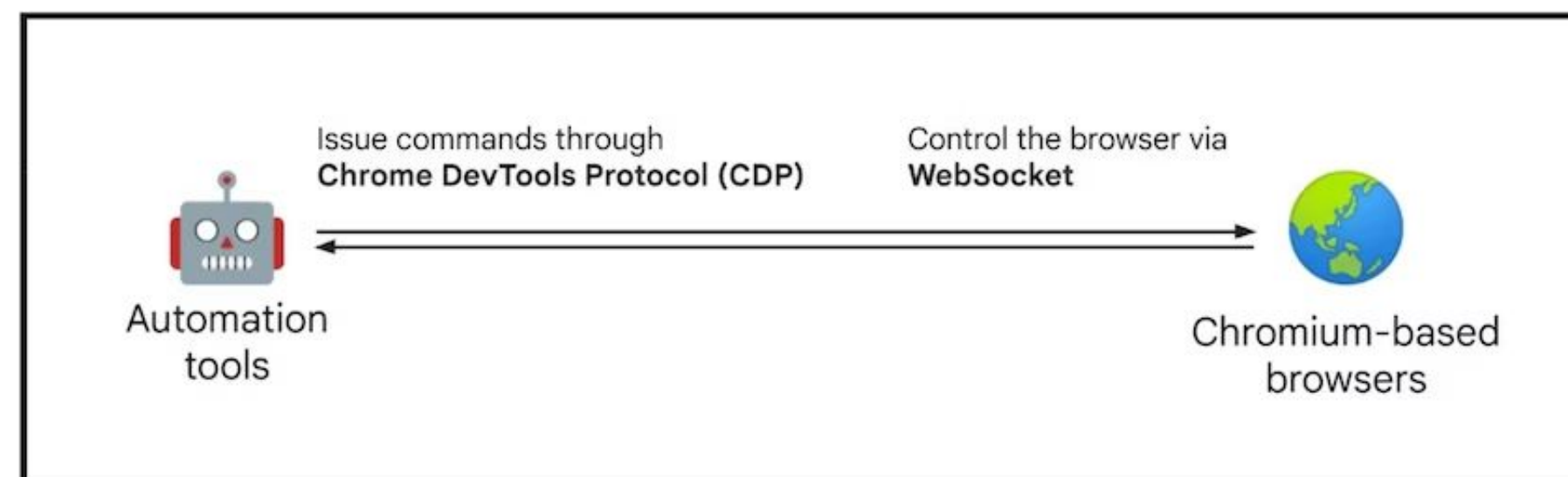


Low Levels: "WDC" vs "CDP"

WebDriver Classic



ChromeDevTools



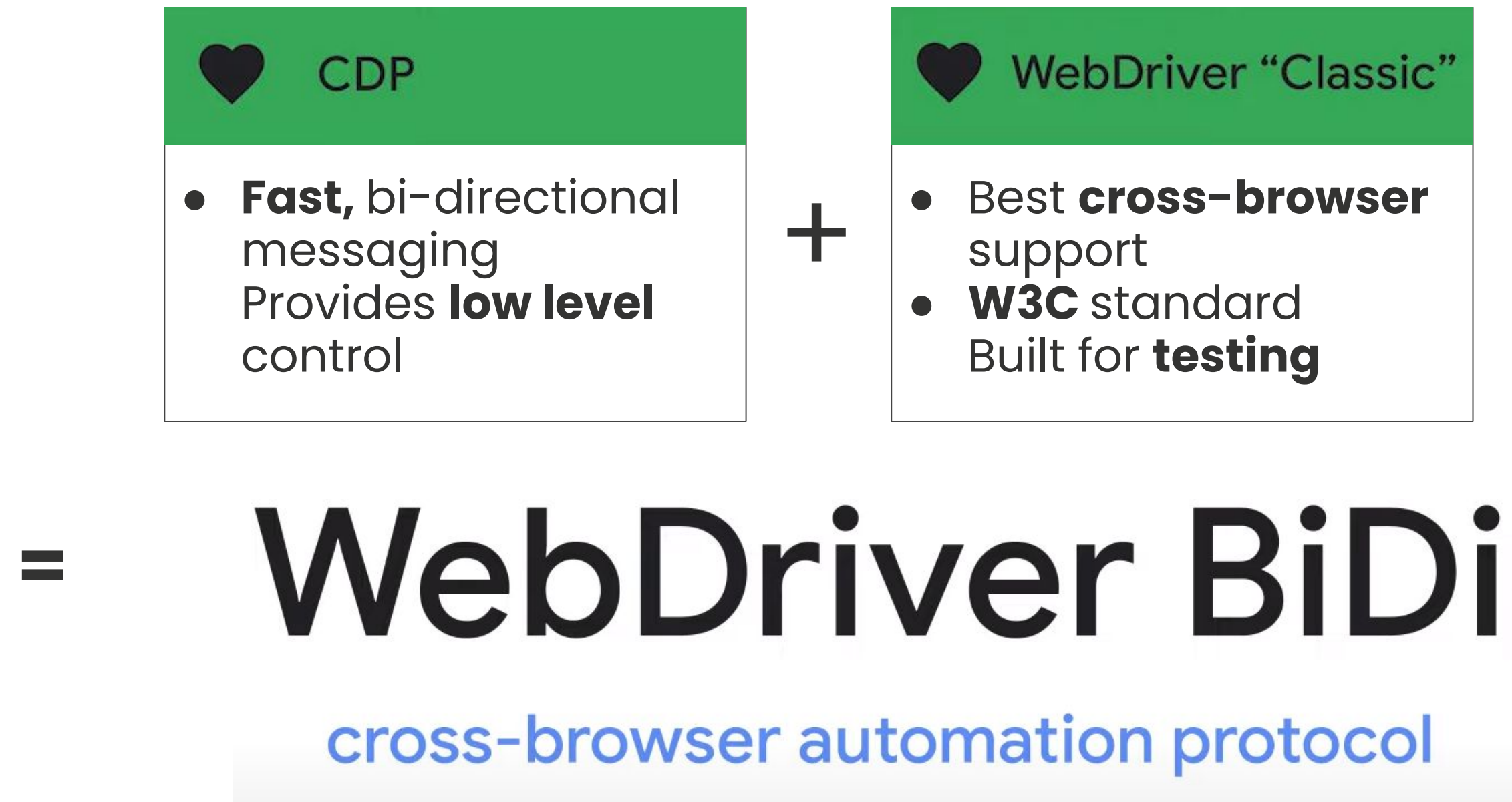
Low Levels: "WDC" vs "CDP"

WebDriver Classic

ChromeDevTools

Supported by All browsers (Standard Protocol)	Only Chromium-based browser
HTTP requests connection	Websocket connection (Faster)
Driver binaries needed	No driver binaries needed
Monodirectional (HTTP Polling needed)	BiDirectional (No polling needed)
No low-levels control (Monitor console, Intercept network, etc.)	low-levels control supported

What if we take the best of two and merge them
into something extraordinary?



W3C Specifications for BiDi

W3C Editor's Draft

TABLE OF CONTENTS

1

Introduction

2

Infrastructure

3

Protocol

3.1

Definition

3.2

Session

3.3

Modules

3.4

Commands

3.5

Errors

3.6

Events

4

Transport

4.1

Establishing a Connection

5

Sandboxed Script Execution

5.1

Sandbox Realms

5.2

Sandbox Proxy Objects

5.3

SandboxWindowProxy

6

Modules

6.1

The session Module

6.1.1

Definition

6.1.2

Types

6.1.2.1

The session.CapabilitiesRequest Type

6.1.2.2

The session.CapabilityRequest Type

6.1.2.3

The session.ProxyConfiguration Type

6.1.2.4

The session.SubscriptionRequest Type

6.1.3

Commands

6.1.3.1

The session.status Command

6.1.3.2

The session.new Command

6.1.3.3

The session.end Command

6.1.3.4

The session.subscribe Command

WebDriver BiDi

Editor's Draft, 5 December 2023



▼ More details about this document

This version:

<https://w3c.github.io/webdriver-bidi/>

Implementation Report:

<https://wpt.fyi/results/webdriver/tests/bidi>

Test Suite:

<https://github.com/web-platform-tests/wpt/tree/master/webdriver/tests/bidi>

Issue Tracking:

[GitHub](#)
[Inline In Spec](#)

Channel:

[#webdriver on irc.w3.org](#)

Wiki:

[W3C WebDriver Wiki](#)

Copyright © 2023 World Wide Web Consortium. W3C® liability, trademark and permissive document license rules apply.

Abstract

This document defines the BiDirectional WebDriver Protocol, a mechanism for remote control of user agents.

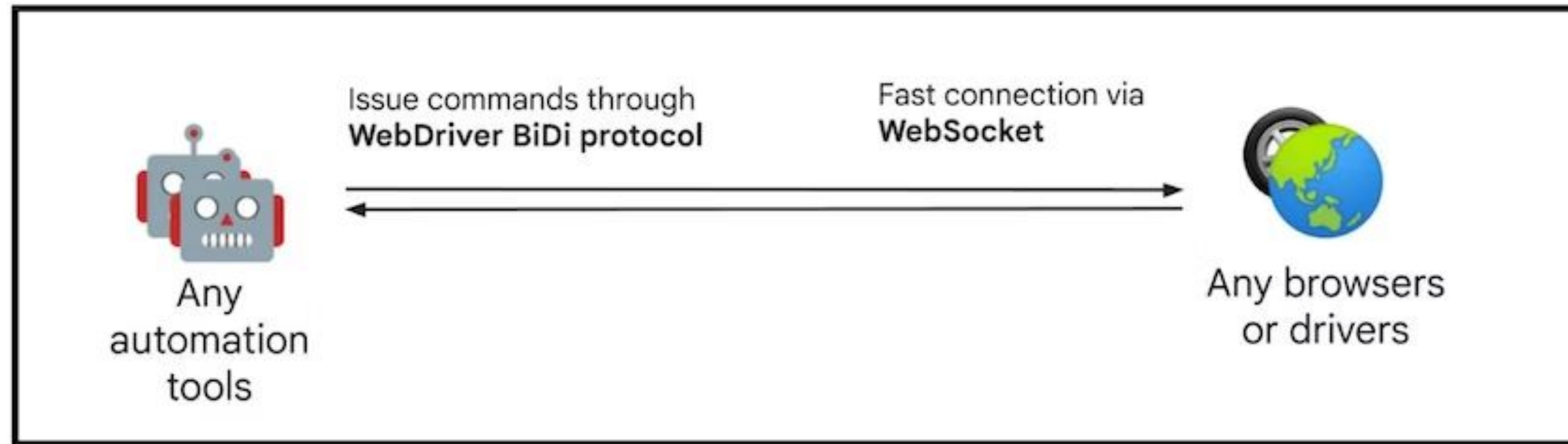
Status of this document

This is a public copy of the editors' draft. It is provided for discussion only and may change at any moment. Its publication here does not imply endorsement of its contents by W3C. Don't cite this document other than as work in progress.



[WebDriver BiDi Specifications](#)

WebDriver BiDi Protocol



[WebDriver BiDi Project](#)

The **W3C** (World-Wide-Web Consortium) has introduced a new standard which is a revamped version of **Classic WebDriver** known as **WebDriver BiDi (BiDirectional)**, and expected to introduce a wide range of new functionalities and address several outstanding challenges. It is currently work in progress (but available experimentally).

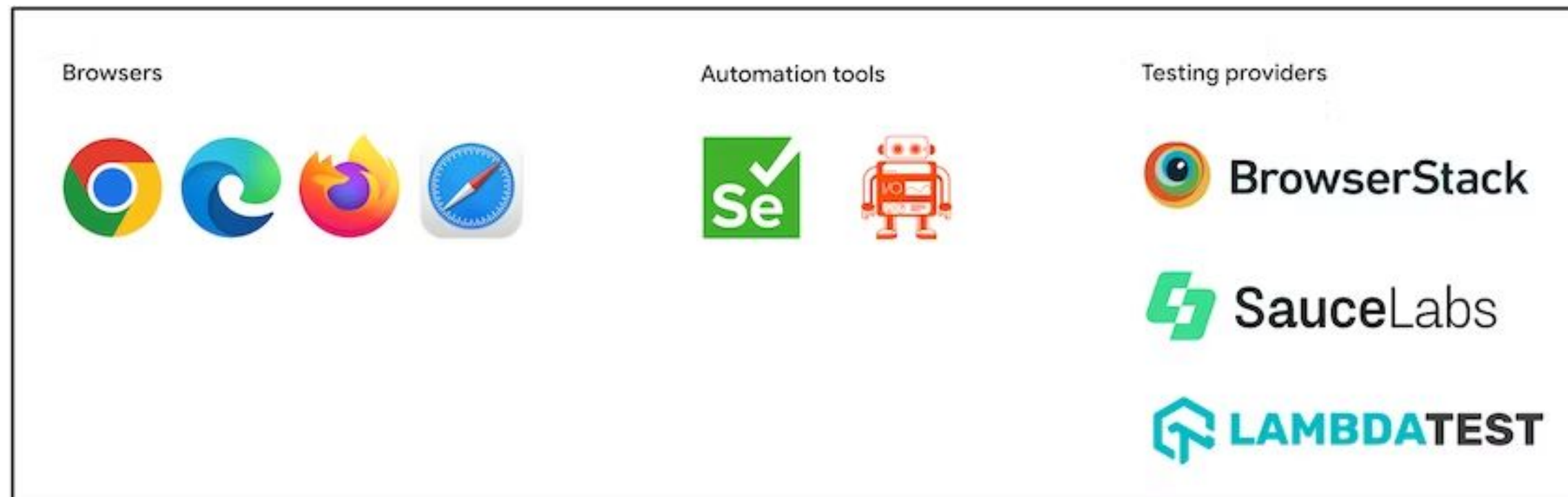
What challenges is WebDriver BiDi trying to solve? (Features)



- Listen for DOM events
- Monitor console logs
- Monitor JS error
- Mock backends and intercept network requests
- Record traffic
- Access to native devtools protocol
- Full-page screenshot
- Dynamic changes to iframe or documents
- Performance timings
- Notifying of new contexts
- Bootstrap scripts

etc.

Who is working on implementation of this standard?



[Project Roadmap](#)

The [WebDriver BiDi Working Group](#) comprises a diverse group of browser vendors, open-source browser automation projects, and companies offering browser automation solutions. This collaboration ensures a promising future for browser automation. Here's the roadmap link.

What will happen to the CDP?



Is WebDriver BiDi going to replace Chrome DevTools Protocol (CDP)?

No. Chromium-based browsers will continue to use CDP for debugging purposes, while WebDriver BiDi is the new specification to address the testing needs with a more ergonomic API.

Since Puppeteer is using CDP, does this mean Puppeteer will be deprecated?

No. However, WebDriver BiDi will enable Puppeteer to become a cross-browser automation tool.

When will we have access to it?



It is already out there as an experimental feature!
Selenium, **Puppeteer** and **WebDriverIO** have already introduced initial support for WebDriver BiDi.
WebDriver BiDi is interoperable with WebDriver Classic.
This means you can incrementally add BiDi support to your scripts starting today.

So let's do a demo! →

Monitor Console Messages (Selenium-JavaScript)

```
import * as assert from 'node:assert';
import { Builder, LogInspector } from 'selenium-webdriver';
import chrome from 'selenium-webdriver/chrome.js';

const driver = new Builder()
  .forBrowser('chrome')
  .setChromeOptions(new chrome.Options().enableBidi())
  .build();

const inspector = await LogInspector(driver);
await inspector.onConsoleEntry((entry) => {
  console.log(`Console message received: [${
    entry.type}][${entry.level}] ${entry.text}`);
});

await driver.get('https://www.selenium.dev/selenium/web/bidi/logEntryAdded.html');
await driver.findElement({ id: 'consoleLog' }).click();

await driver.quit();
```

Monitor Console Messages (Puppeteer)

```
import puppeteer from 'puppeteer';

const browser = await puppeteer.launch({
  protocol: 'webDriverBiDi',
  headless: 'new',
});

const context = await browser.createIncognitoBrowserContext();
const page = await context.newPage();

page.on('console', (message) => {
  console.log(`Console message received: [${
    message.type()
  }] ${message.text()}`);
});

await page.goto(`https://www.selenium.dev/selenium/web/bidi/logEntryAdded.html`);
await page.evaluate(() => {
  document.querySelector('#consoleLog').click();
});

await browser.close();
```



[Selenium Support](#)



[WebDriver IO support](#)



[WPT BiDi Suite](#)



[WPT BiDi Dashboard](#)

Questions & Answers

