DataArt

WebDriver BiDi Protocol

The future of cross-browser automation

New York USA

London UK

Munich Germany

Zug Switzerland

About me

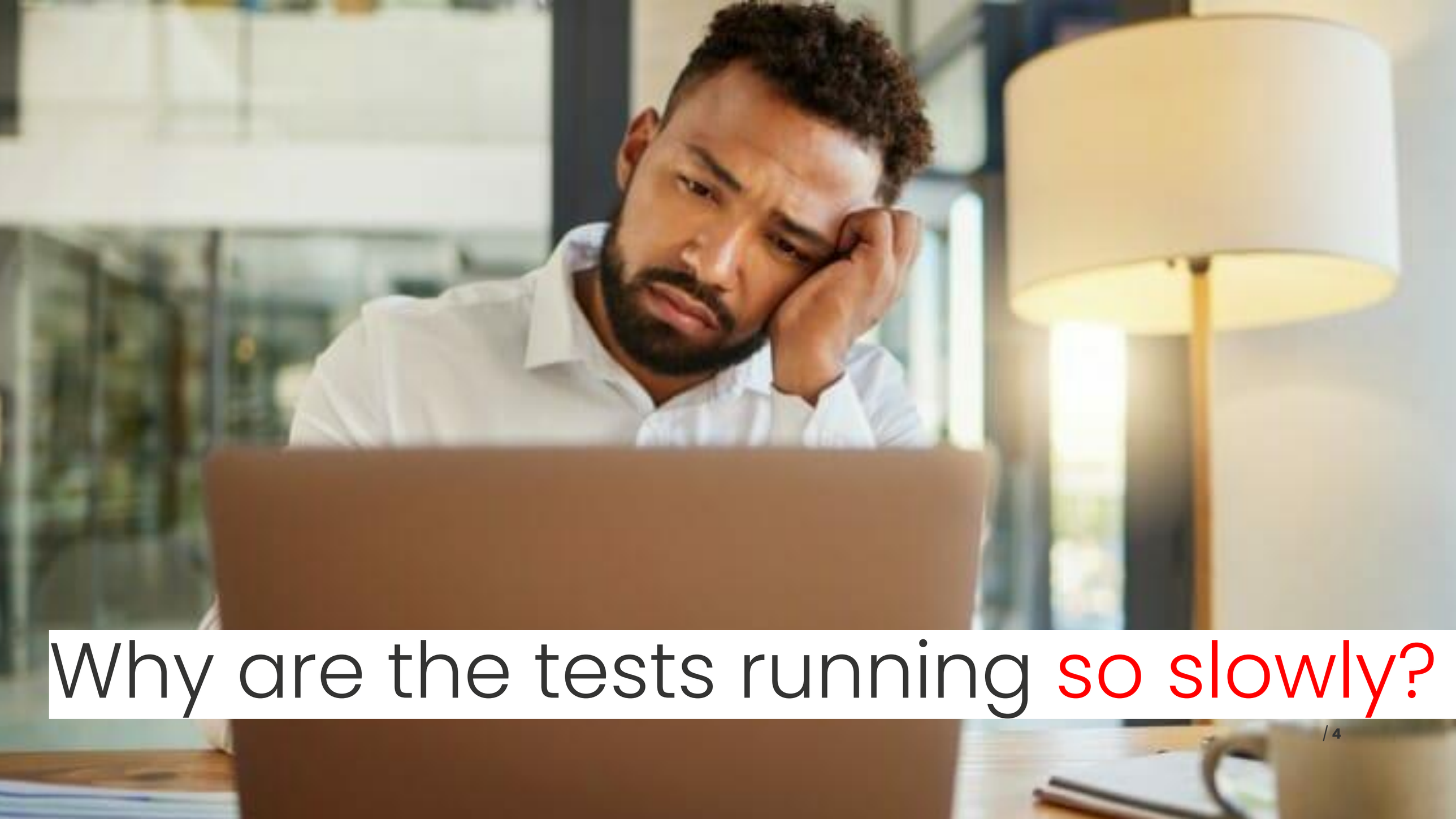
DataArt

- **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

DataArt

- 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)

DataArt

TABLE OF CONTENTS

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)

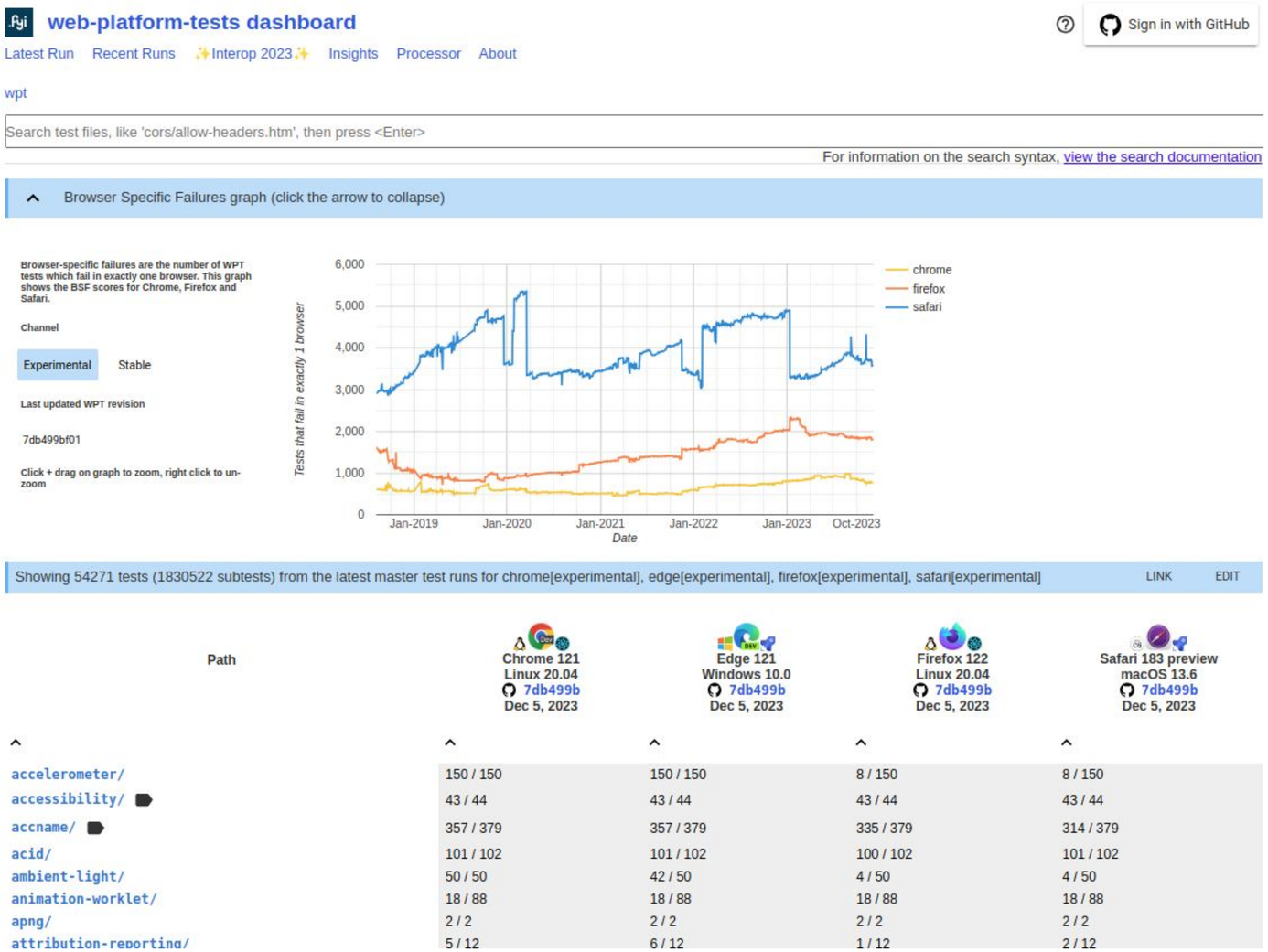
Feedback:
GitHub w3c/webdriver (pull requests, new issue, open issues)

Channel
#webdriver on irc.w3.org

WebDriver Classic

Web Platform Tests

DataArt

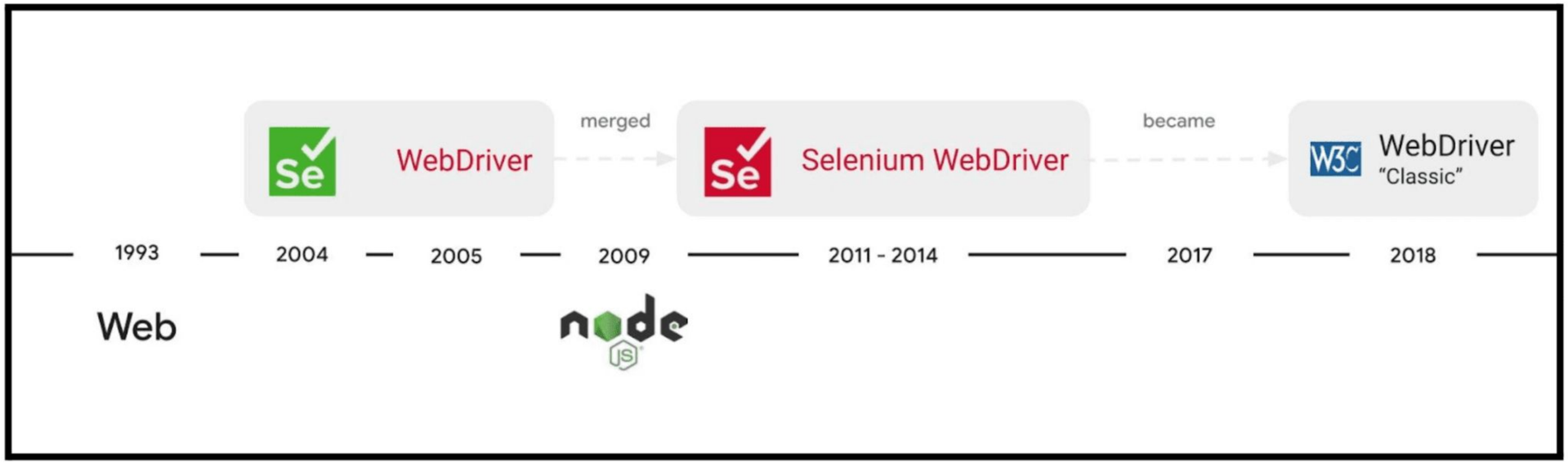


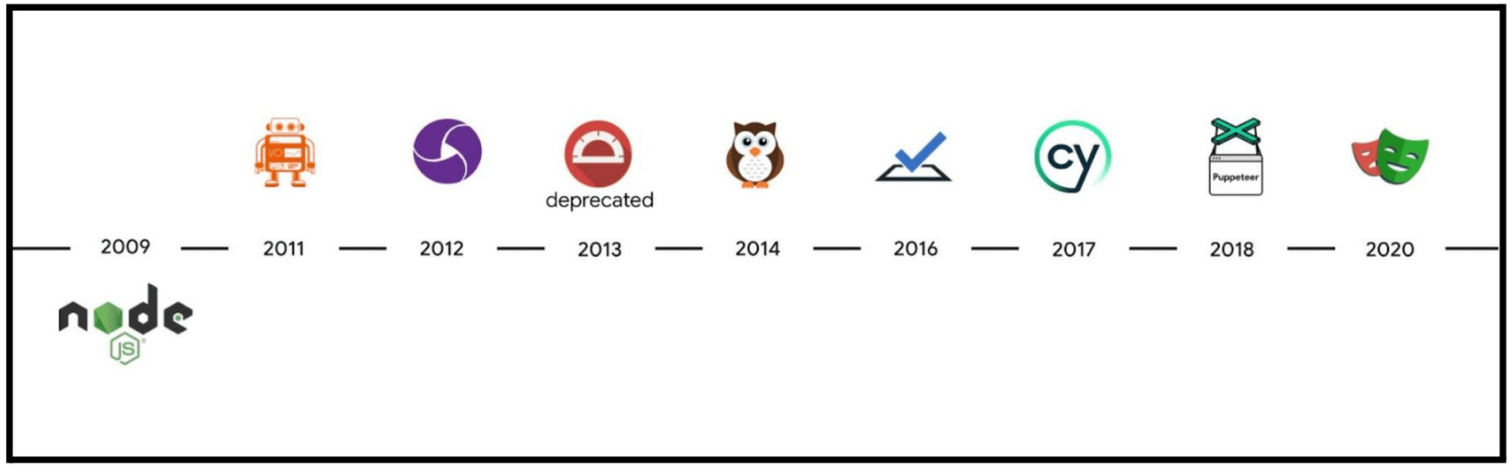
Web Platform Tests

Web Platform Tests Dashboard

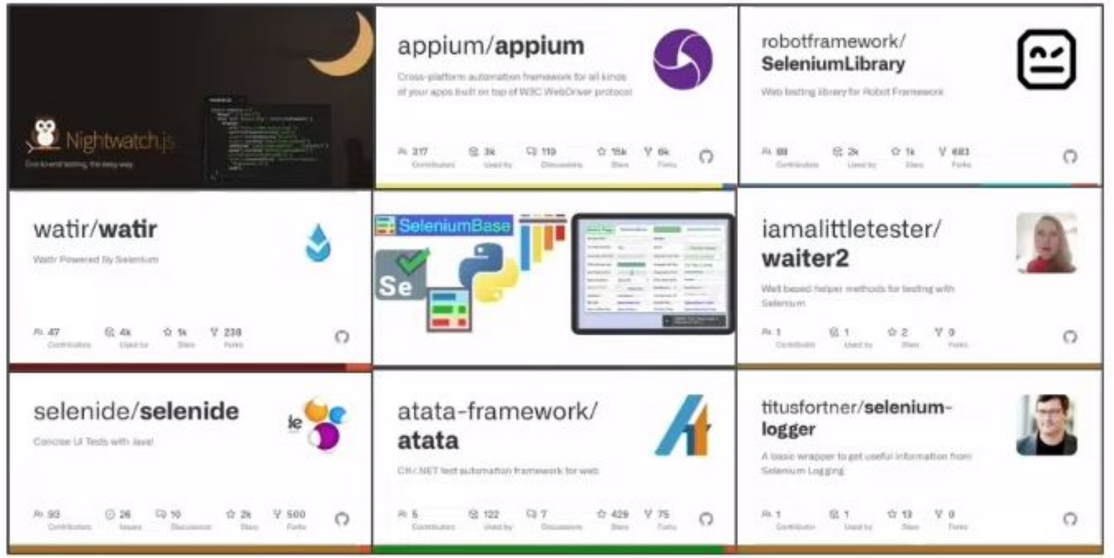
The history of test automation

DataArt





WebDriver Ecosystem



Serenity /JS

Laravel Dusk

PANTHER

Implementation

Selenium
A browser automation framework and ecosystem

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

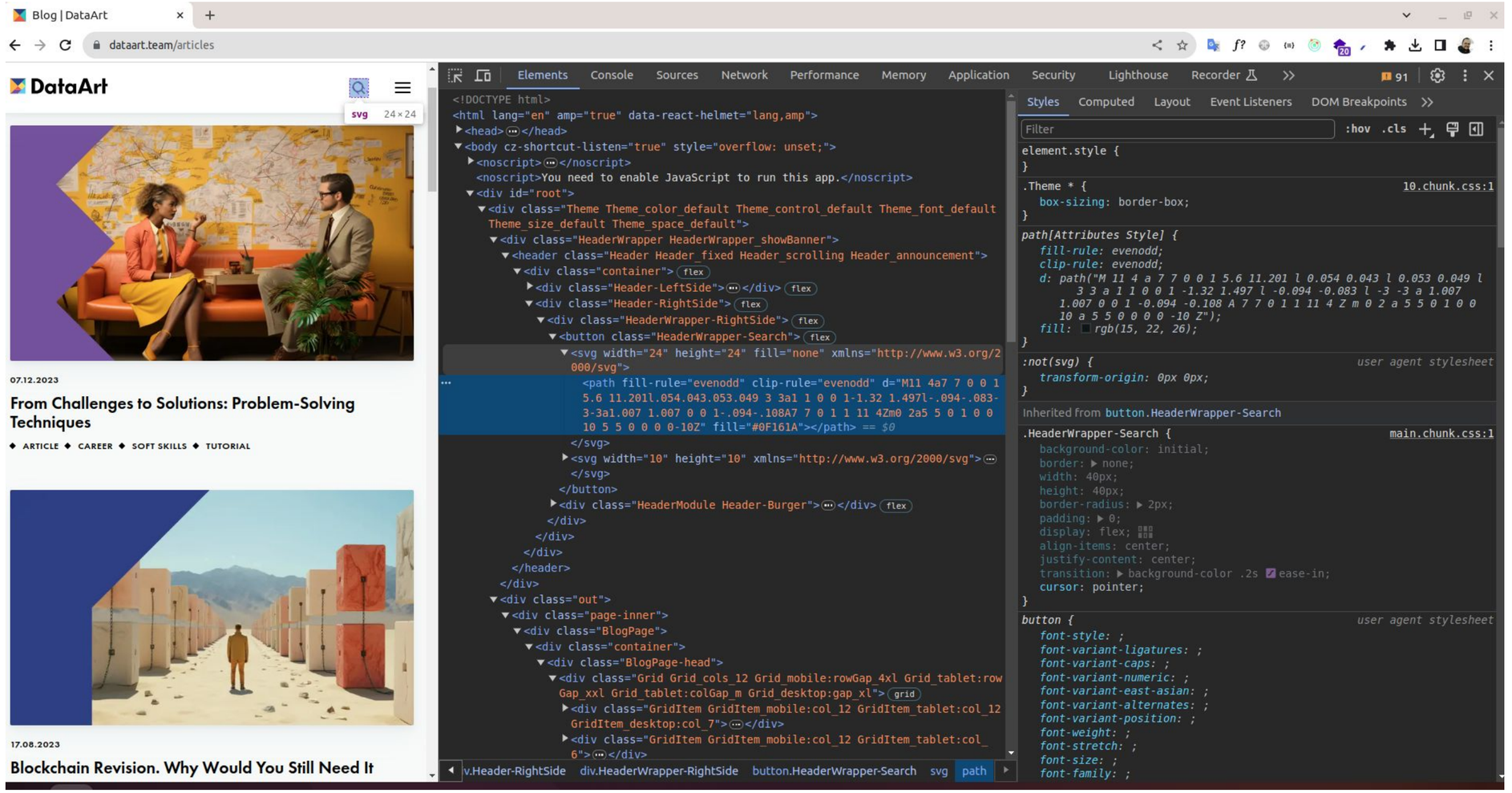
php-webdriver/**php-webdriver**
PHP client for Selenium/WebDriver protocol.
Previously facebook/php-webdriver

Specification

w3c/**webdriver**
Remote control interface that enables introspection and control of user agents.

What is ChromeDevTools?

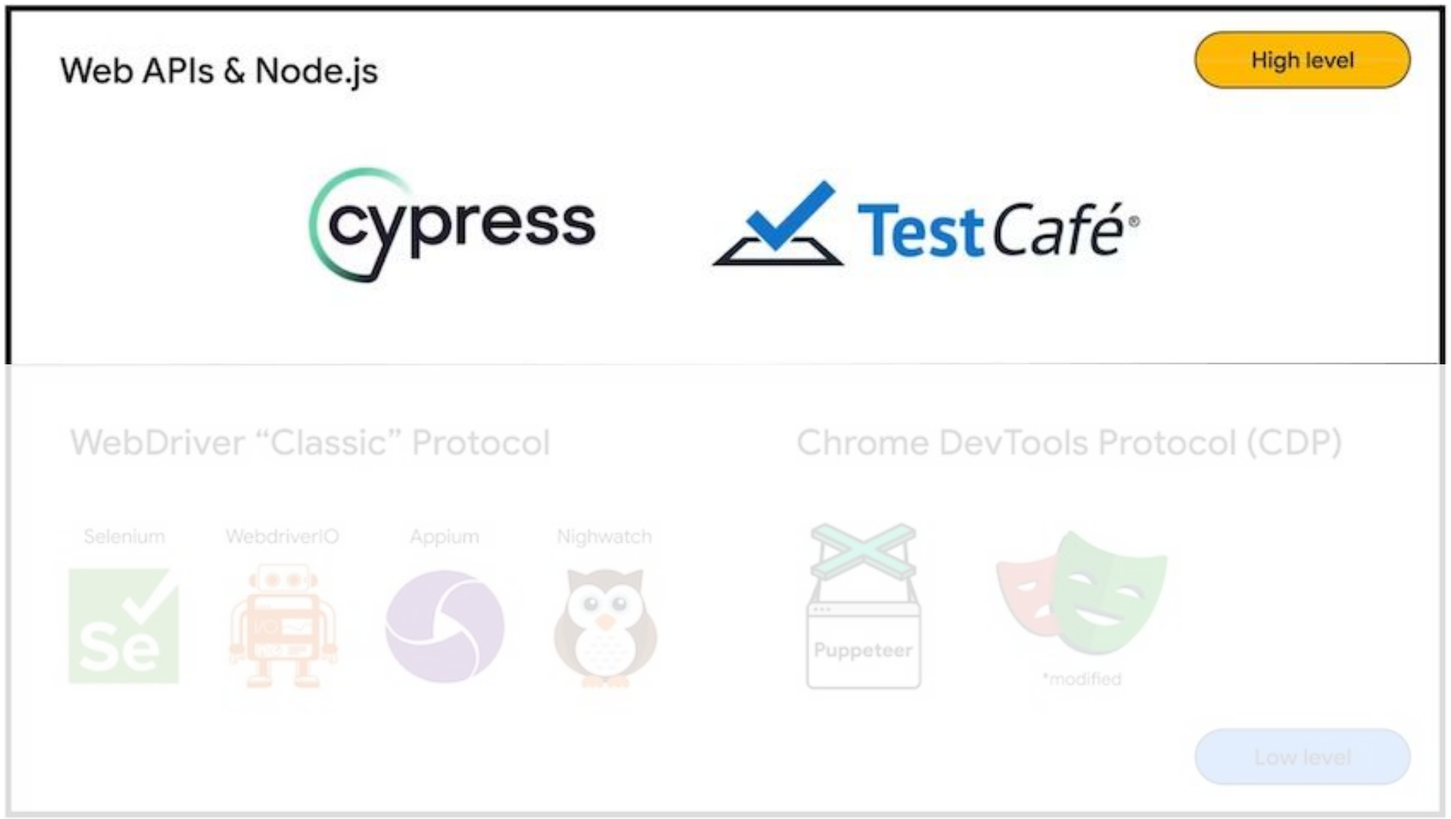
DataArt



Two groups of browser automation tools

DataArt

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



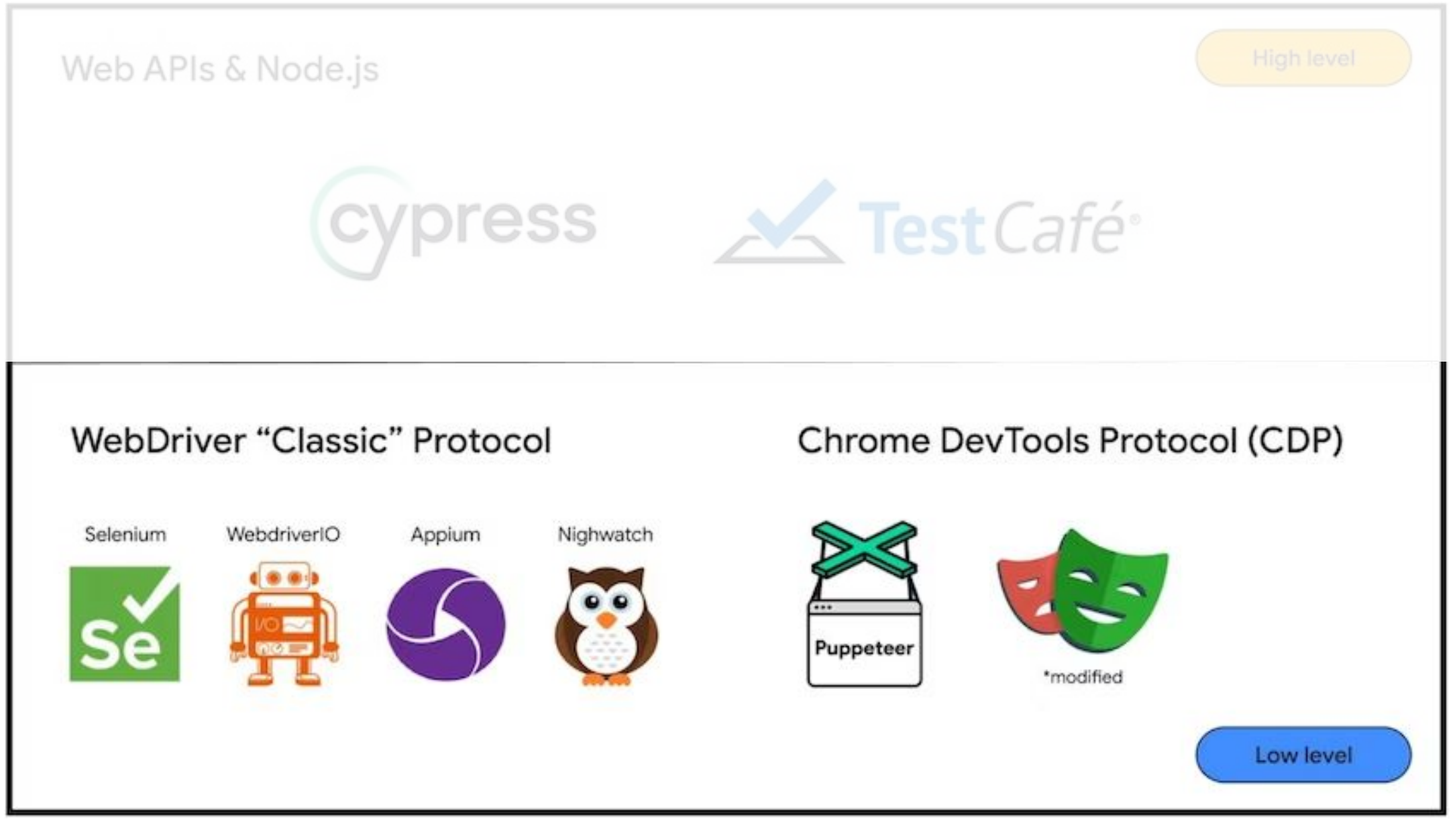
Two groups of browser automation tools

DataArt

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:

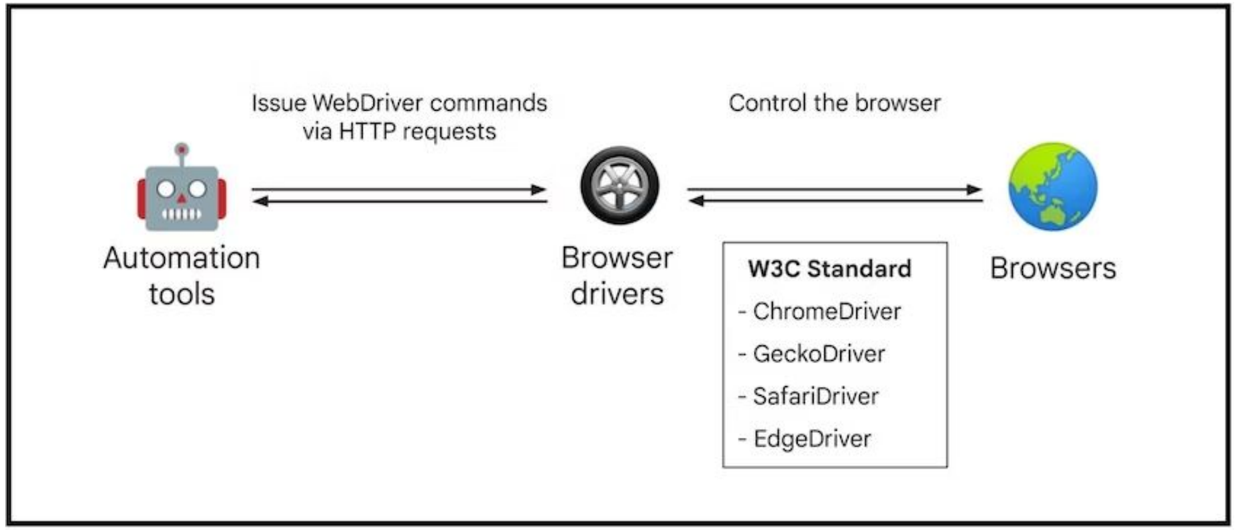
- **WebDriver classic** (e.g. **Selenium**)
- **ChromeDevTools** which is faster than WebDriver (e.g. **Playwright**)



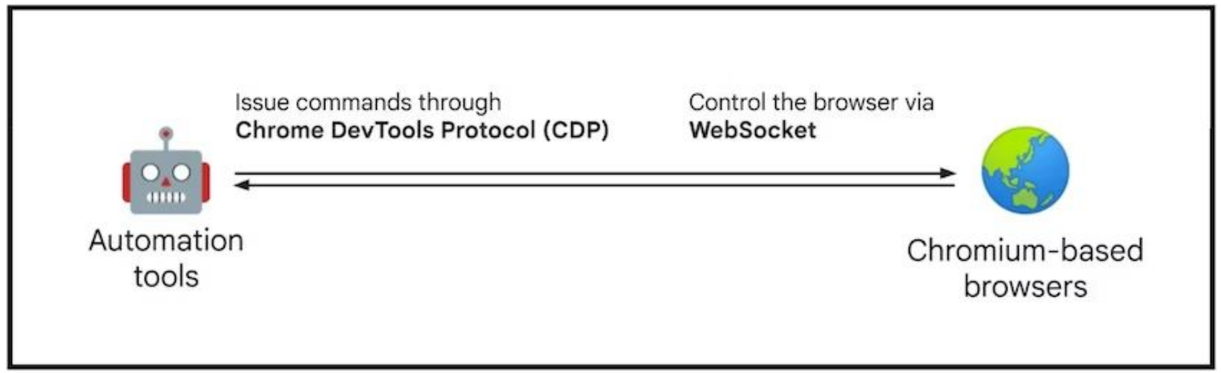
Low Levels: "WDC" vs "CDP"

DataArt

WebDriver Classic



ChromeDevTools



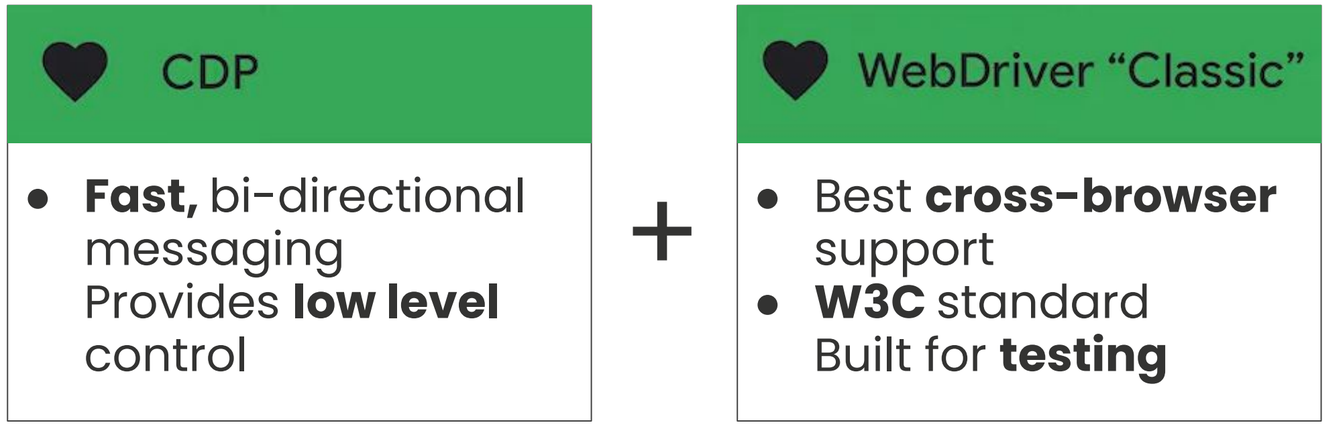
Low Levels: "WDC" vs "CDP"

DataArt

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?

DataArt

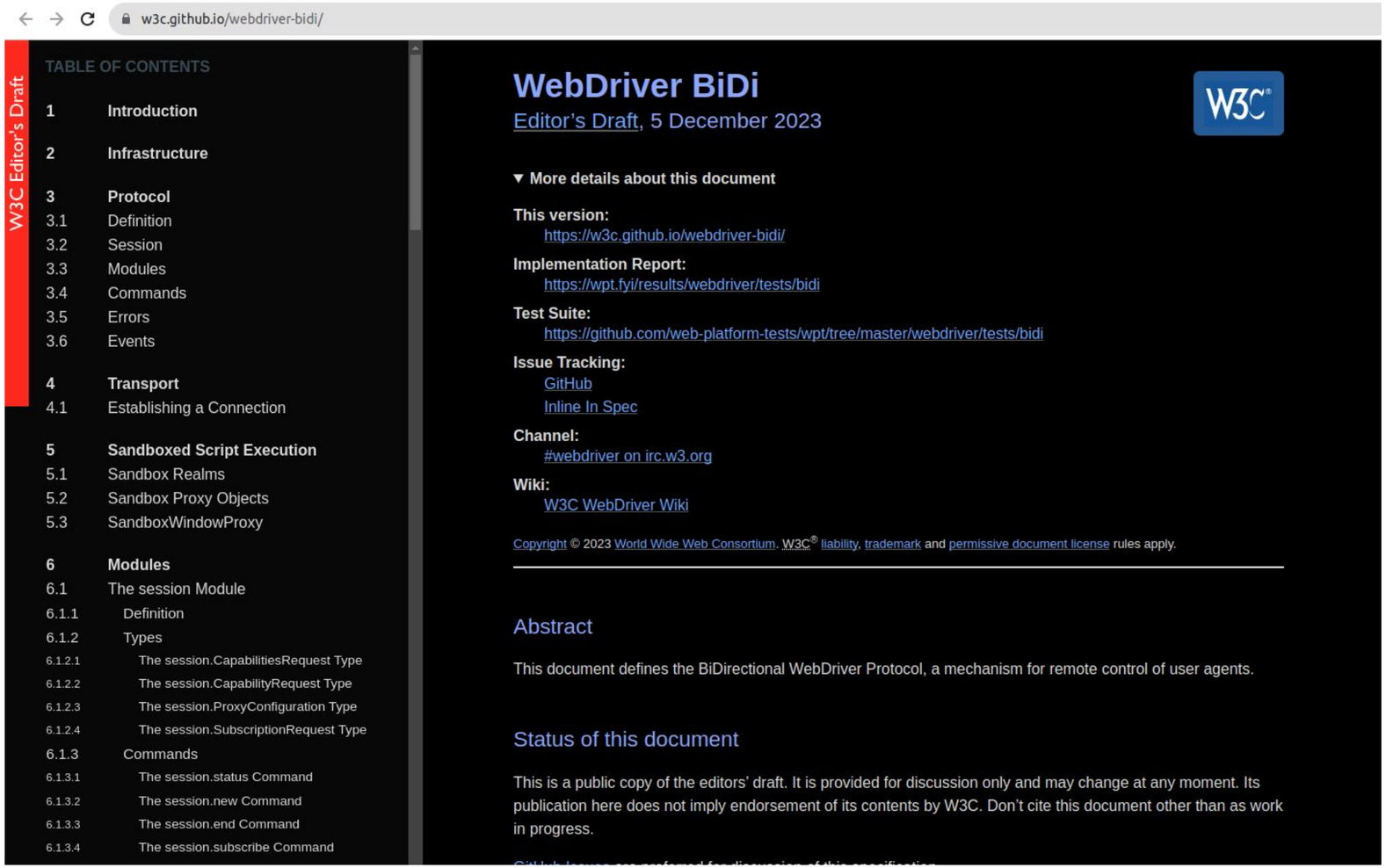


= WebDriver BiDi

cross-browser automation protocol

W3C Specifications for BiDi

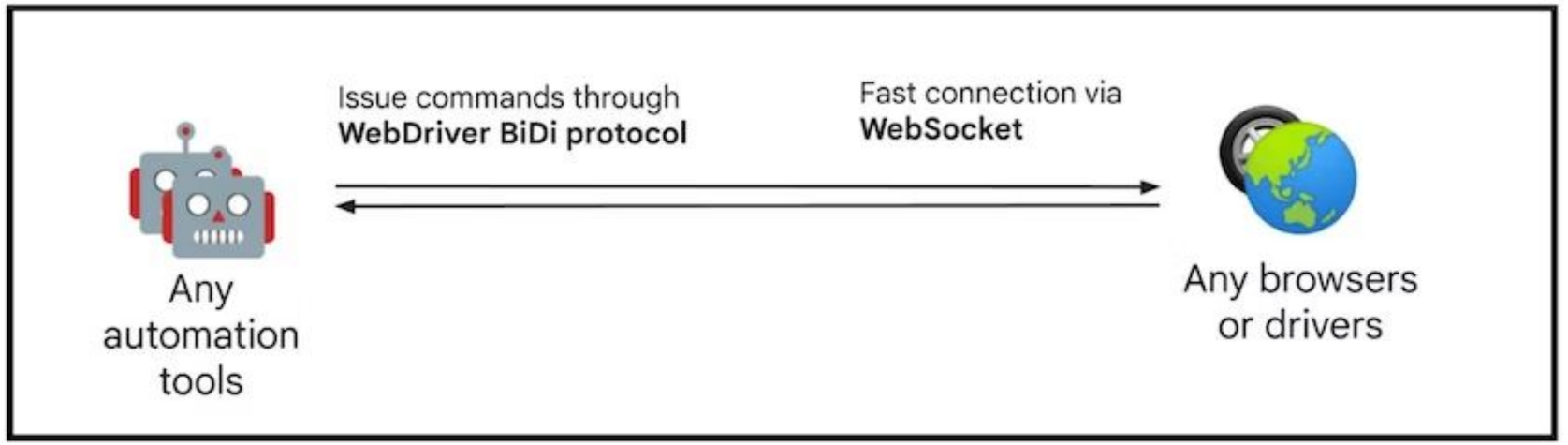
DataArt



WebDriver BiDi Specifications

WebDriver BiDi Protocol

DataArt



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)

DataArt

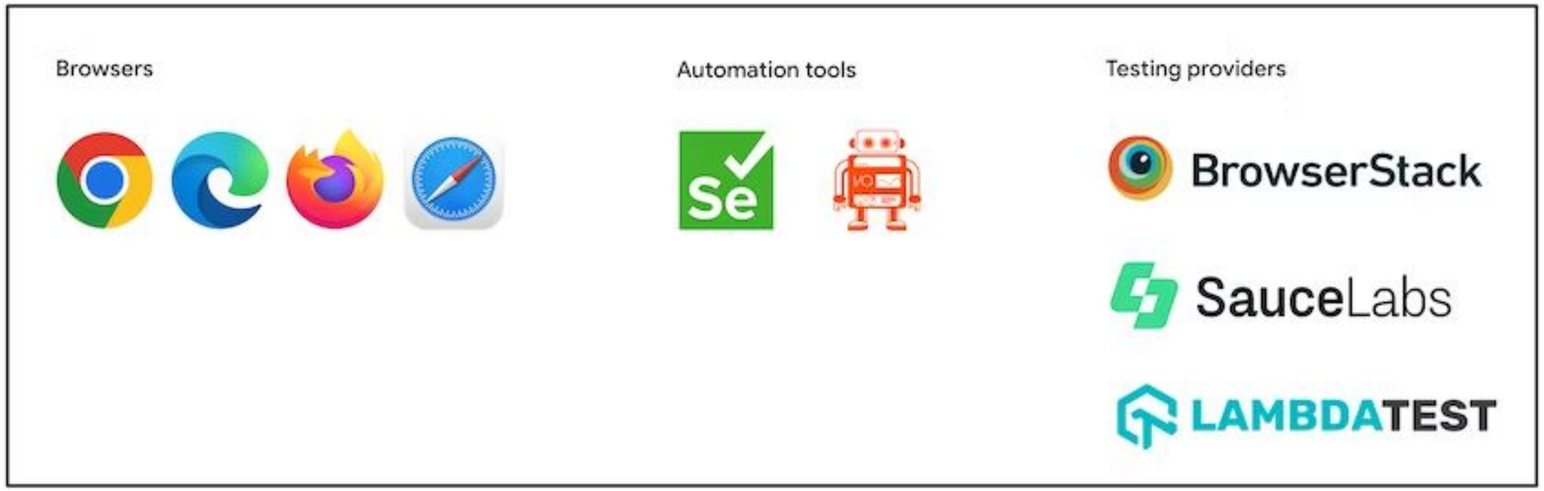
- Listen for DOM events
- Monitor console logs
- Monitor JS error
- Mock backends and intercept network requests
- Record traffic
- Access to native devtools protocol

etc.

- Full-page screenshot
- Dynamic changes to iframe or documents
- Performance timings
- Notifying of new contexts
- Bootstrap scripts

Who is working on implementation of this standard?

DataArt



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?

DataArt

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?

DataArt

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)

DataArt

```
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)

DataArt

```
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();
```

Docs / Status

DataArt

[Selenium Support]

[WebDriver IO support]

[WPT BiDi Suite]

[WPT BiDi Dashboard]

Questions & Answers