Web automation testing

… for developers?

info.nl

What is Automation Testing?

Definition

Test automation is the use of special software to control the execution of tests and the comparison of actual outcomes with predicted outcomes. It’s done through an automation tools which nowadays exist for almost every platform.

What is Automation Testing?

What can be automated?

- Web UI testing
- Desktop UI testing
- CLI interface testing
- API testing
- Database testing
- System stability testing

What is Automation Testing?

What does it fit for?

- Smoke testing for production deploys
- Regression testing (majorly happy flows)
- Long run end-to-end workflows testing
- Performance and load testing
- Chaotic testing (micro-services)

What is Automation Testing?

What it doesn't fit for?

- Look and feel testing
- 100% coverage idea
- Replace manual QA
- Frequently changing/unstable application/modules

What is Automation Testing?

Automation vs. manual

Manual	Automation
Pros:	***Pros:***
• Easy to set up and start with	• Time saving
• Consistent with life testing	• Eliminate the human factor
• Suitable for look and feel check	• High reusability
• Suitable for exploratory testing	• Better ROI on long-term
• Lower initial expenses and investments	• Transparency
• Expanded adaptability	***Cons:***
Cons:	• Expensive
• More assets and time required	• Lack of input on usability and UI
• Not all things can be tried physically	• Tools' limitations and compatibility
• Low reusability	• Cannot replace human intellectual skills
• Load and performing testing is not possible	• Cannot be used as for exploratory testing
• Not suitable for large projects with short deadlines	• Basic programming knowledge required

What is Automation Testing?

Automation vs. manual

Manual + Automation = ❤

Cypress Overview

General overview

Cypress is a front end testing tool built for the modern web. It address the key pain points developers and QA engineers face when testing applications.

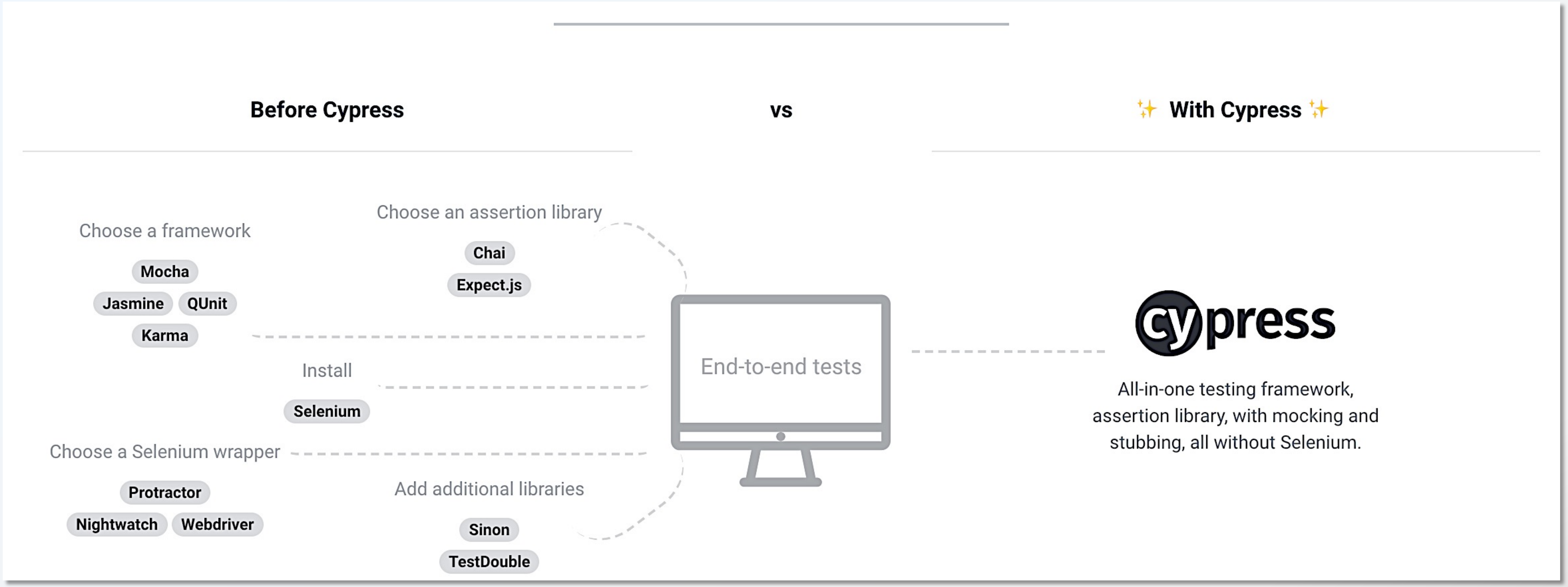
It’s open source, but… With subscription for Dashboard Service.

Main characteristics:

- UI end-to-end automation testing framework
- Not based on Selenium
- Runs directly in the browser
- For QA engineers and developers
- Easy to debug tests

Cypress Overview

How it works



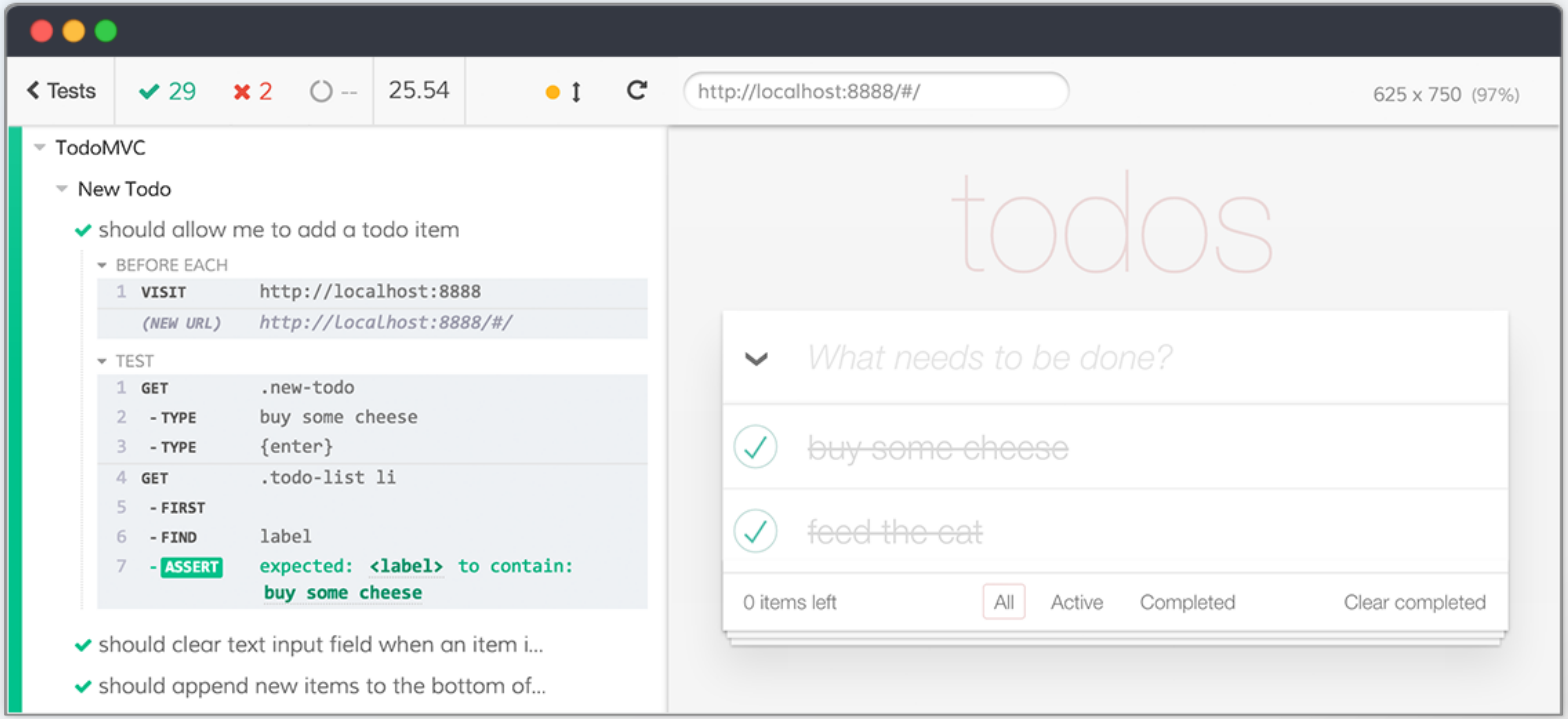
Cypress Overview

Features

- Time Travel
- Debuggability
- Automatic Waiting
- Spies, Stubs, and Clocks
- Network Traffic Control
- Screenshots and Videos

Cypress Overview

Example



TodoMVC

New Todo

✓ should allow me to add a todo item

BEFORE EACH

```
1  VISIT        http://localhost:8888
   (NEW URL)    http://localhost:8888/#/
```

TEST

```
1  GET          .new-todo
2   - TYPE      buy some cheese
3   - TYPE      {enter}
4  GET          .todo-list li
5   - FIRST
6   - FIND      label
7   - ASSERT    expected: <label> to contain:
                buy some cheese
```

✓ should clear text input field when an item i...

✓ should append new items to the bottom of...

WebDriver Overview

Selenium Projects

- Selenium WebDriver
- Selenium Remote Control
- Selenium Grid
- Selenium IDE

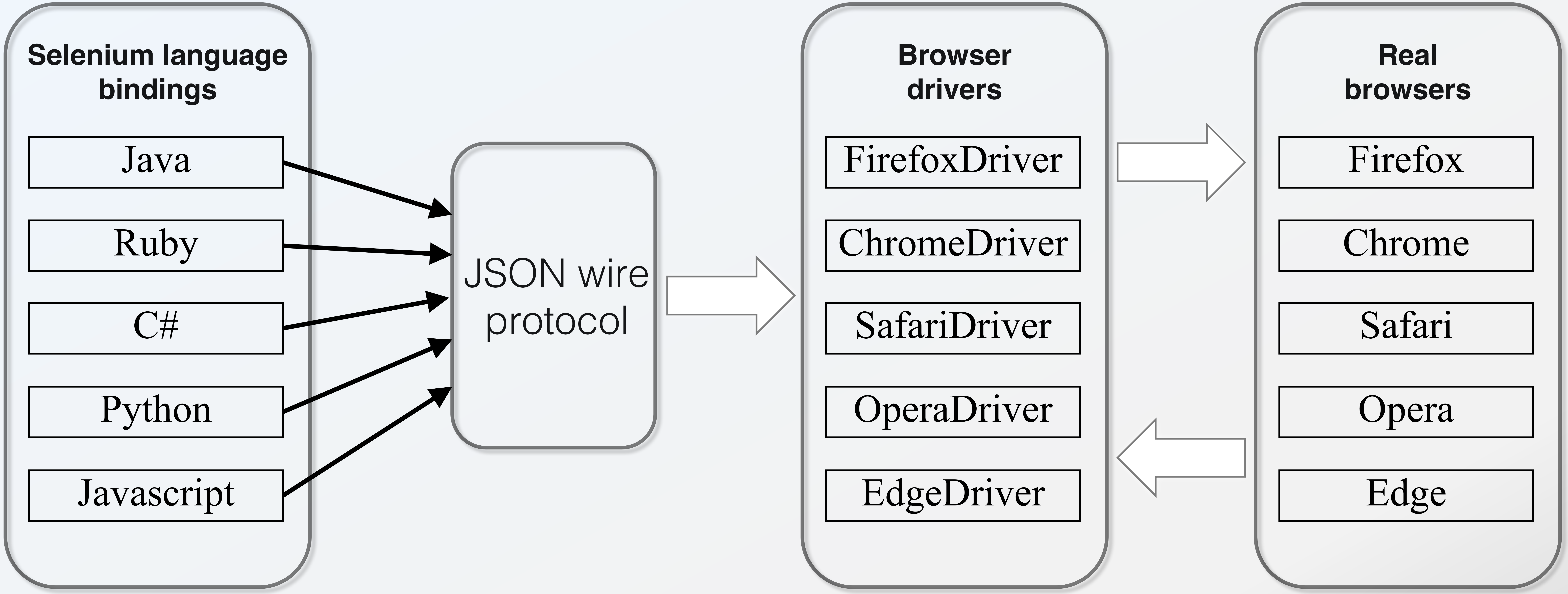
WebDriver Overview

General overview

Selenium WebDriver is a browser automation tool built for the modern web. Driving a browser natively as a user would either locally or on a remote machine using the Selenium Server it accepts commands and sends them to a browser. It is implemented through a browser-specific driver. It controls the browser by directly communicating with it.

WebDriver Overview

How it works



WebDriver Overview

Features

- Easy cross-browser testing
- Support different languages
- Easy framework development
- Parallel testing

Cypress vs. WebDriver

Supported programming languages

WebDriver	Cypress
• Java • C# • Ruby • Python • Javascript Perl, PHP, Haskell, Objective-C, R, Dart, Tcl, Elixir	• Javascript

Cypress vs. WebDriver

Supported selectors

WebDriver	Cypress
• Id • Name • Class name • Tag name • CSS • XPath • Link text • Partial link text	• jQuery style

Cypress vs. WebDriver

Supported browsers

WebDriver	Cypress
• Chrome • Firefox • Safari • Opera • Edge • IE	• Chrome • Firefox (coming soon…)

Automation Design Tips

- **Use Page Object pattern**
  Page Object is a Design Pattern which has become popular in test automation for enhancing test maintenance and reducing code duplication. A page object is an object-oriented class that serves as an interface to a page of your AUT. The tests then use the methods of this page object class whenever they need to interact with the UI of that page.

- **Use custom attributes on elements**
  A good practice is to use custom attributes for HTML elements location. Used attribute can be like: ***at-label="<component-name>"***.

Demo

Q&A

Thanks!