Contents

My Questions

Lecture 100

My Marks

it.only('Open.. - only this test

xit('Open - not this test

Links

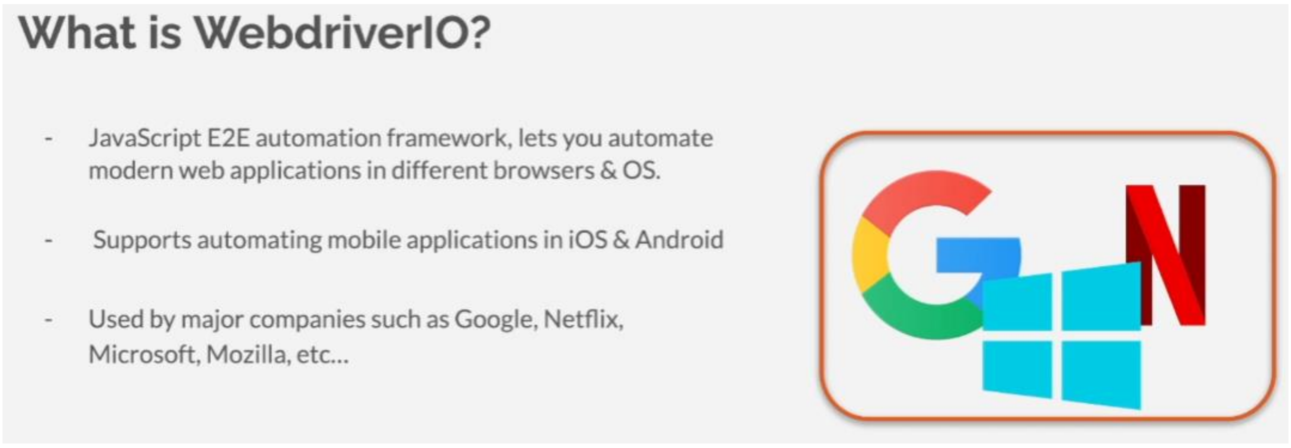
- https://webdriver.io/docs/api/expect-webdriverio/
- https://webdriver.io/docs/api/expect-webdriverio/#default-matchers
- https://jasmine.github.io/api/3.5/global.html#expect

see Lessons 35 & 36

Section 1

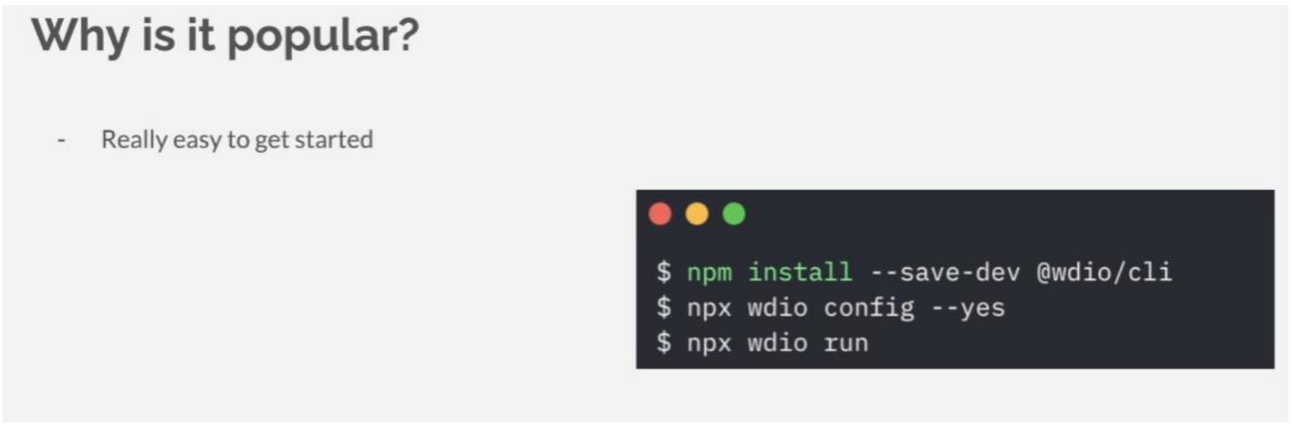
Lesson 3 What is WebdriverIO

- JS E2E automation framework



What is WebdriverIO?

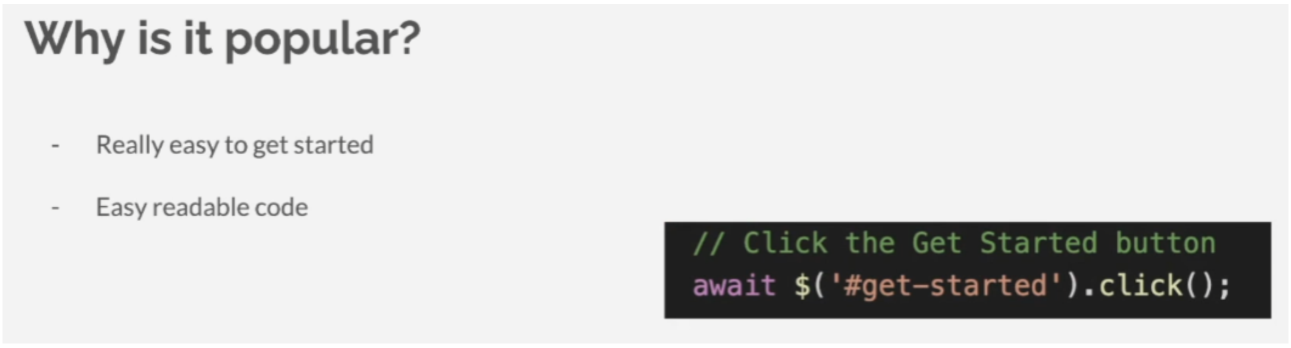
- JavaScript E2E automation framework, lets you automate modern web applications in different browsers & OS.
- Supports automating mobile applications in iOS & Android
- Used by major companies such as Google, Netflix, Microsoft, Mozilla, etc...



Why is it popular?

- Really easy to get started

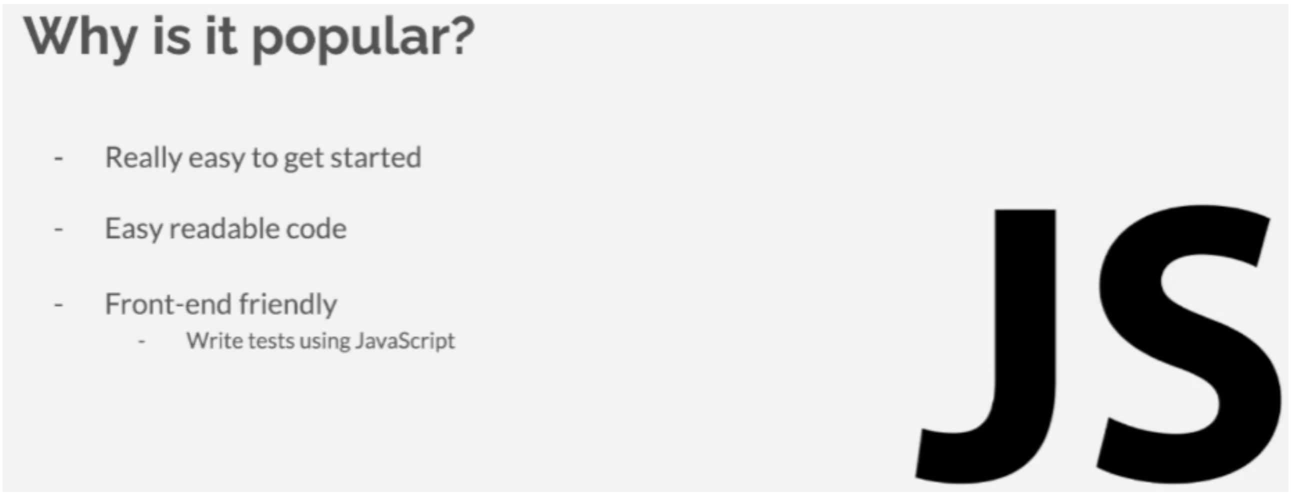
```
$ npm install --save-dev @wdio/cli
$ npx wdio config --yes
$ npx wdio run
```



Why is it popular?

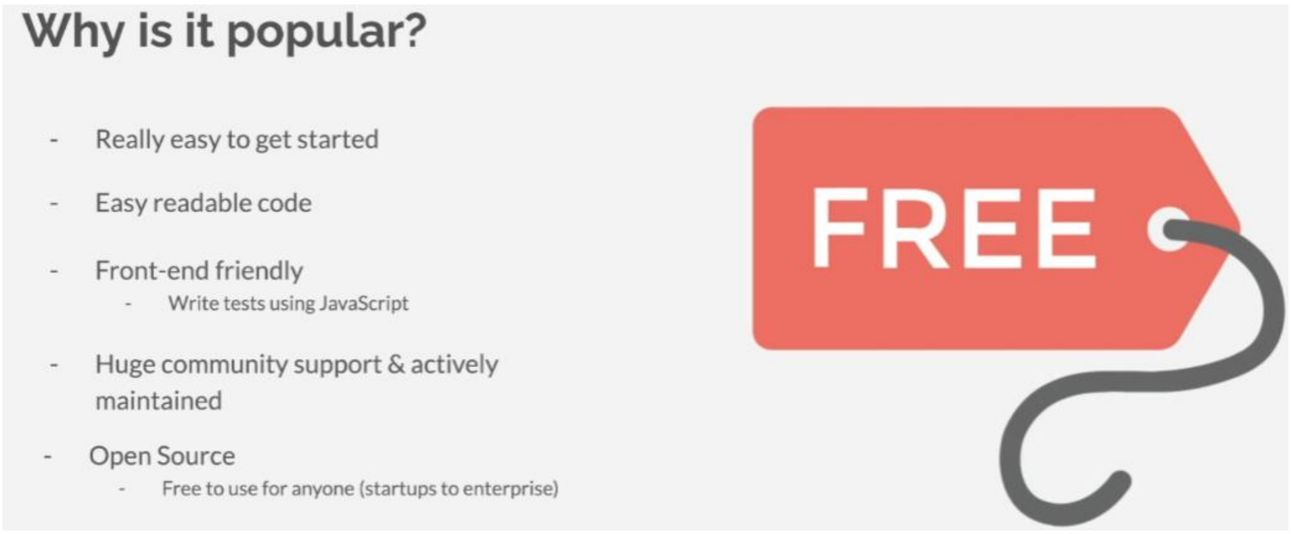
- Really easy to get started
- Easy readable code

```
// Click the Get Started button
await $('#get-started').click();
```



Why is it popular?

- Really easy to get started
- Easy readable code
- Front-end friendly
    - Write tests using JavaScript



Why is it popular?

- Really easy to get started
- Easy readable code
- Front-end friendly
    - Write tests using JavaScript
- Huge community support & actively maintained
- Open Source
    - Free to use for anyone (startups to enterprise)

Lesson 4 Getting Most out of the Course

Lesson 5 Discord Community

- https://discord.com/channels/1019379062013169734/@home

Lesson 8 Zipshare.com

- https://www.zipshare.com/

Section 2 Machine Setup

- Node
- VS Code installation
- Google Chrome
-

Section 3 WebdriverIO Setup & Installation

- codebase: https://github.com/dilpreetj/webdriverio-course

Lesson 13 WebdriverIO Project Setup

- npm init wdio ./path_to_new_project [npm init wdio ./wdio-course]
- this sigle command downloads the WebdriverIO CLI tool and runs a configuration wizard that helps to configure the test suite
- `To run your tests, execute:`
- `$ cd C:\coding\Udemy_WebdriverIO\wdio-course`
- `$ npm run wdio`
- `MUISTA PÄIVITTÄÄ CHROME-browser aloittaessasis (tarkista onko päivitystä)`

Lesson 14 WebdriverIO Config Overview

Overview of wdio.conf.js

- baseURL,
- Specs,
- Cpabilities,
- Services,
- Framework,
- Reporter,
- Hooks

Section 4 Chrome 115+ Update Important

WebdriverIO v8.14.0+ Updates

- rm -rf node_modules package-lock.json
- npm i @wdio/cli@latest @wdio/local-runner@latest @wdio/mocha-framework@latest @wdio/spec-reporter@latest --save-dev
- in case updating older tests and conflicting with chrome browser drivers
- Chrome for testing: https://googlechromelabs.github.io/chrome-for-testing/

Section 5 Write Tests

Lesson 22 Write & Run first test

- Setup the folder structure
- Write a test for Home Page
  - Open URL & assert title (practice.automation.com)
- Run test: npx wdio

Lesson 23, 24 Fail test & Async

```
describe('Home', () => {
    it('Open URL & assert title', async () => {
        // Open URL
        await browser.url('https://practice.sdetunicorns.com/');

        // Assert title
        await expect(browser).toHaveTitle('Practice E-Fail-Commerce Site – SDET Unicorns');
        // await expect(browser).toHaveTitle('This is a random title');
    })
})
```

Section 6 Working with Elements

Lesson 27 Finding Elements ($)

Lesson 28 & 29 Exercise using Xpath

```
await $('//img[@alt="Practice E-Commerce Site"]').click();
```

Lesson 30 Find Element & Get text

Lesson 31 Finding Multiple Elements

- note this, more challenging!

Lesson 33 Exercise: Contact & Blog Page

Section 7 Assertions

Lesson 35 WebdriverIO Expect Assertions

Advantages

- Built-in-assertions (no external lib needed)
- Built-in wait & retry capabilities
- see: webdriver vs cypress

Matchers

- Browser Matchers
- Element Matchers

- Network Matchers
- Default Matchers (Jest/Jasmine)

https://webdriver.io/docs/api/expect-webdriverio/

Lesson 36 Jest/Jasmine Assertions

In addition to the expect-webdriverio matchers you can use builtin Jest's expect assertions or expect/expectAsync for Jasmine.

- https://webdriver.io/docs/api/expect-webdriverio/#default-matchers
- https://jasmine.github.io/api/3.5/global.html#expect

Section 8: Waits

Lesson 37 Pause command

- pause() - pause the execution for X amount of time

Lesson 38 Common Wait commands

- waitForDisplayed() – to check if element is displayed on the screen
- waitForClickable() - to check if element can be clickable
- waitForEnabled() -to check if the input field is enabled
- waitForExist() – to check if element is present in the DOM

Lesson 39 waitUntil command

- waitUntil – to check for particular condition

Lesson 9 Upload & iFrame Tests

Lesson 40 Simple upload tests

- https://the-internet.herokuapp.com/upload
- display: none (for Choose File -button)

Lesson 41 Upload a Hidden Element

- https://practice.sdetunicorns.com/cart/
- https://online2pdf.com/ seach: input[type=file] -> class=""
- https://www.online-image-editor.com/ seach: input[type=file]
  ~~`/* display: none;`~~
- or `display: block;`
- https://jpg2png.com/ seach: input[type=file] -> class="" or
  `/* visibility`~~`:`~~` hidden;`

Lesson 42 Working with iFrame

- https://practice.sdetunicorns.com/iframe-sample/
- see Questions I asked

Section 10 Debugging

- Using console.log, pront out the element(s) date
- Using browser.pause(), usually good to identify issues
- Using browser.debug(), all purpose debugging (increase mocha timeout)

Lesson 43 Console.log() & pause()

Lesson 44 Debug command

Section 11 Setup Framework

Lesson 46 Setup folder structure & Auto Completion

- setup folder structure: Tests, Pages, Data, Utils
- jsconfig.json file for autocompletion

Lesson 47 Babel setup

- setup Babel to use next-generation JS features
- https://webdriver.io/docs/babel/
- npm install --save-dev @babel/core @babel/cli @babel/preset-env @babel/register

Lesson 48 Linter

- https://www.npmjs.com/package/eslint-plugin-wdio
- `npm i eslint --save-dev`
- `npm install eslint-plugin-wdio --save-dev`

See question about Linter - Lesson 48

- npx eslint ./test/specs/contact.js
- npx eslint wdio.conf.js
-

Section 12 Page Object Model

Lesson 49 What is Page Object Model

- popular design patter, helps us to reduce code duplication & improves test maintenance

Lesson 50

- Create page objects for home.js
  - move selectors to HomePage
  - create page object methods for Home.js

Lesson51 What is Page Components

- divides pages into multiple components to build complex page object structures

Lesson 52 Setup Page Component for Navigation component

- made code changes and updated to GitHub

Lesson 53 Exercise: Setup POM model for Contact & Blog pages

- Create page objects for contact.js and blog.js
-

Lesson 54 POM Exercise Solution

- see GitHub: https://github.com/lasse1900/Udemy_webdriverIO

Section 13 Optimizing test Framework

Lesson 55 Hooks

- Helps setup & teardown of the tests
- reduce dependencies
- create hooks in home.js
    - Before hook
    - After hook
    - BeforeEach hook
    - AfterEach hook
- Global hooks
- setting window size in before hook

```
-       beforeTest: async function (test, context) {
-           browser.setWindowSize(1000,1000);
-       },
```

Lesson 56 Updated faker-js library

Section 17 Browser Stack

Try mobile

- https://www.youtube.com/watch?v=qLZArtTme8M
- https://www.youtube.com/watch?v=XSHifNNWML4
-

Lesson 66 Browser Stack Setup (sign up)

- https://automate.browserstack.com/dashboard/v2/builds

Lesson 67 Integrate BrowserStack with WebdriverIO

- https://webdriver.io/docs/browserstack-service/
- in case testing a local website: `browserstackLocal: true`

Lesson 68 Run tests in BrowserStack

- giving creds on command line

lauri@DESKTOP-S2JLDIN MINGW64 /c/coding/Udemy_WebdriverIO/Udemy_webdriverIO (main)

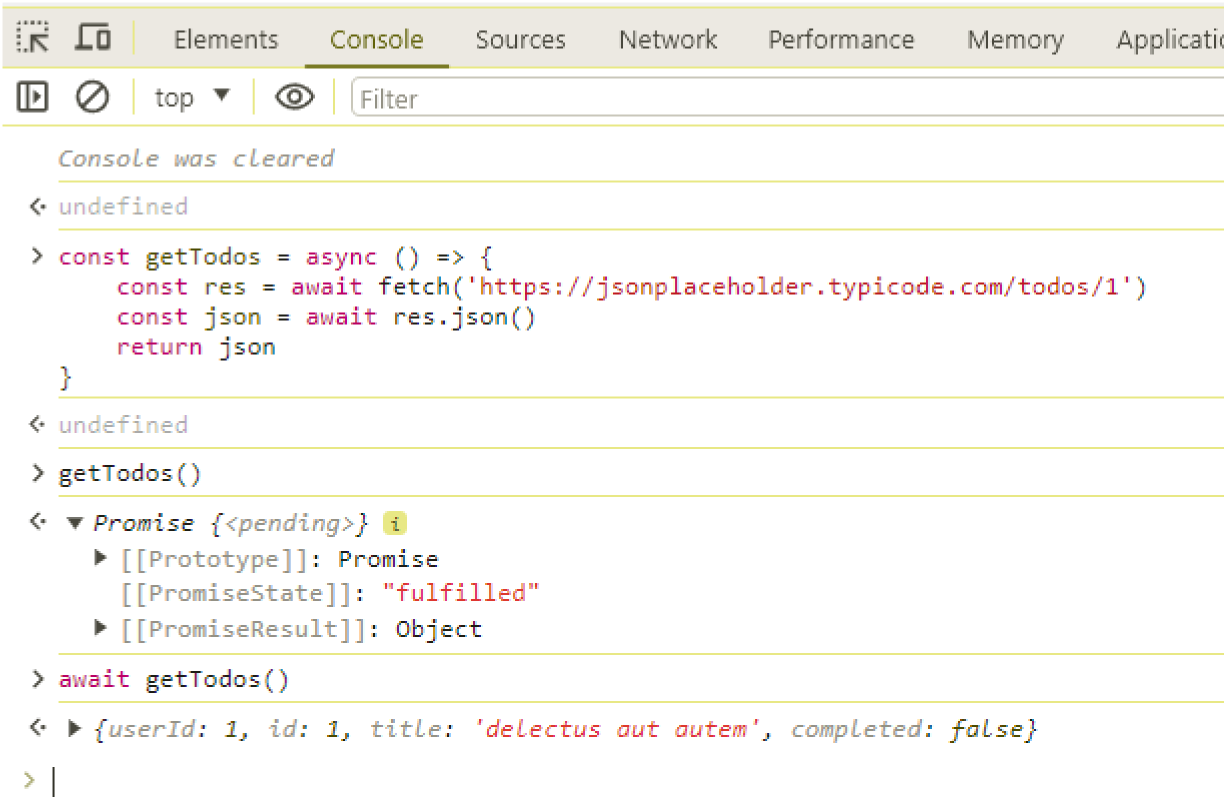
>$ BROWSERSTACK_USERNAME=@userKey BROWSERSTACK_ACCESS_KEY=@accessKey npx wdio --spec test/specs/contact.js

Section 22 JavaScript Basics

```
const getTodos = async () => {
  const res = await fetch('https://jsonplaceholder.typicode.com/todos/1')
  const json = await res.json()
  return json
}

getTodos()

await getTodos()
```



```
Console was cleared
<· undefined
> const getTodos = async () => {
      const res = await fetch('https://jsonplaceholder.typicode.com/todos/1')
      const json = await res.json()
      return json
  }
<· undefined
> getTodos()
<· ▼ Promise {<pending>} i
    ▶ [[Prototype]]: Promise
      [[PromiseState]]: "fulfilled"
    ▶ [[PromiseResult]]: Object
> await getTodos()
<· ▶ {userId: 1, id: 1, title: 'delectus aut autem', completed: false}
> |
```

with [Shift + Enter] you can add a line to chrome console

Section 23 Quick HTML and CSS refresher

Lesson 97 HTML Overview

- https://replit.com/@automationbro/Introduction-to-HTML-CSS#index.html

Lesson 98 CSS Overview

Lesson 99 HTML DOM

Lesson 100 Custom CSS Selectors

- https://gist.github.com/magicznyleszek/809a69dd05e1d5f12d01

List of Selectors

- id (#id)
- Class (.class)
- Attribute (h1)
- Custom CSS (input[type="submit"])

Custom CSS Selector

- Attribute selectors
  - Exact, contains, begins with, etc..
- Contextual selectors
  - Descendant, child, sibling etc..
- Pseudo-class selectors
  - a:link, p:hover etc..
- Pseudo-class for siblings:
  - First-child, only-child, first-of-type etc..

p[class='paragraph'] a

- https://practice.sdetunicorns.com/
- #navigation>li>a:first-child
- #navigation>li>a:last-child
- [id="blog-link"]
- .main-menu .linkback a:last-child
- .main-menu .linkback a:nth-child(2)
- a[href="index.html"]
- a[href^="index"]
- a[href$="html"]
-

Lesson 101 How to use XPath

What is XPath

- XML path to navigate through HTML DOM
- Syntax
    - Xpath = //tagname[@Attribute='Value']
- Types of XPath
    - Absolute (never use)
    - /html/body//div[2]/div/[1]/div/h4[1]/b/html[1]/body[1]
- Relative
    - //tagname[@Attribute='Value']

List of Selectors

- Basic Xpath (//input[@type=’submit’]
- Contains (//*[contains(@href, ‘index’)])
- Or & And (//*[@href, ‘index.html’ and @role=’menuitem’])
- Starts with (//a[starts-with(@href,’index’)])
- Text (//a[text()=’About’])

Examples to HomePage:

//img[@src="https://cdn.pixabay.com/photo/2017/02/01/22/02/mountain-landscape-2031539_960_720.jpg"]

//*[contains(@href,'index')]

//*[@href='index.html' and @role='menu-item']

//a[starts-with(@href,'index')]

//a[text()='Backup']