TESTING

Hello

Kamil Richert

Senior Software Engineer at Atlassian

Kinds of tests

- **Unit tests**

  Jest, Karma

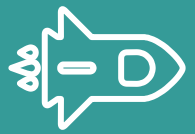
- **Integration tests**

  Cypress

- **End-to-end tests**

  Selenium

TASK



Let's start by writing simple tests in pure JS.

Complete tasks in the folder: *exercises*.

JEST

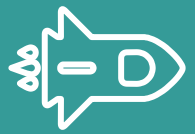
JEST

- Testing framework developed by Facebook
- Has its own test-runner → test launcher
- Has its own set of assertions (comparisons with which we verify test results)
- It gives us the ability to measure the coverage with tests
- Built-in mocks
- Handles exceptions (the intentional ones we want to test)
- It can be used to test frontend projects (using e.g. Angular, Vue or React) or projects based on Node.js

Useful links:

https://jestjs.io/docs/en/getting-started → the entire "Introduction" section covers the basics of working with Jest
https://create-react-app.dev/docs/running-tests/ → testing applications using create-react-app, where Jest is used as a test runner and the main library for testing.

TASK



Complete tasks in: *jest-exercises and jest-mock-exercises*

REACT TESTING LIBRARY

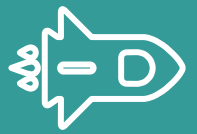
REACT TESTING LIBRARY

- It is not a standalone library / testing framework (as Jest)
- It gives us the opportunity during testing
  - render a React component
  - interaction with the component
  - reading values as from the DOM tree
- Built on the DOM Testing Library

Useful links:

https://testing-library.com/docs/react-testing-library/intro → the entire section of React Testing Library covers the basics of working with the library
https://testing-library.com/docs/dom-testing-library/api-queries → methods to build convenient DOM queries
https://testing-library.com/docs/dom-testing-library/api-events → fire events
https://create-react-app.dev/docs/running-tests/#react-testing-library → react-testing-library is the default in create-react-app

TASK



Complete tasks in: *react-testing-library-exercises*

CYPRESS

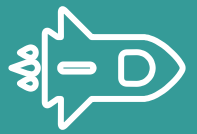
CYPRESS

- Comprehensive solution for e2e (end-to-end) testing
- Has its own runner test
- Independent of the application being tested (the technology used in the tested application is not important)
- Tests that "click" on the side - we describe these behaviors in JavaScript
- Cypress makes sure that the flow of our tests follows how the elements on the page change - we do not have to manually "wait" for the operation to be performed
- Own assertion library

Useful links:

https://docs.cypress.io/guides/getting-started/installing-cypress.html → the entire Getting Started section covers all the basics of working with Cypress
https://www.youtube.com/watch?v=LcGHiFnBh3Y → comprehensively Cypress basics in the form of a video

TASK



Complete tasks in: *cypress-exercises*

Thanks

You can find me:

https://www.linkedin.com/in/kamil-richert/

https://github.com/krichert