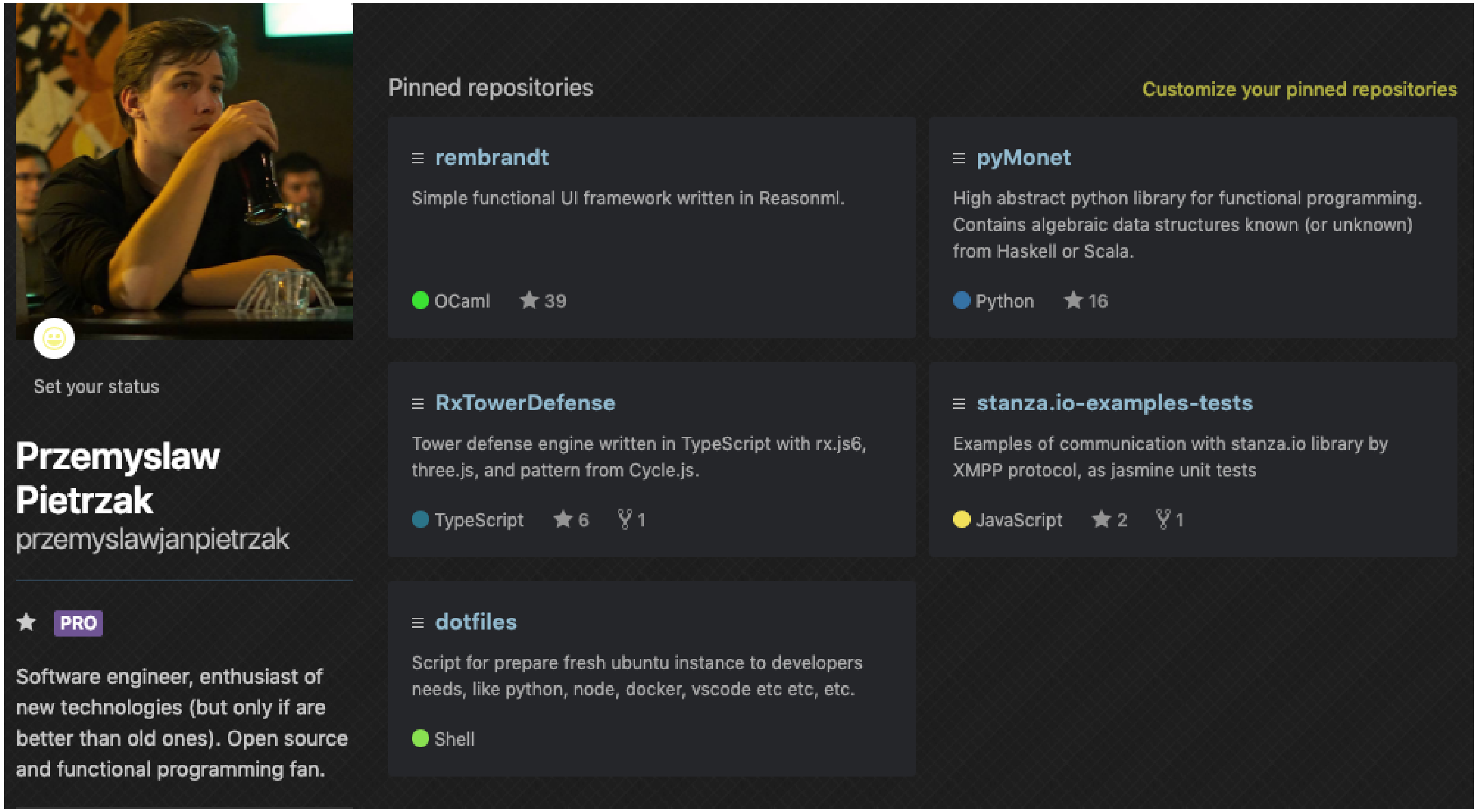
CYPRESS.IO & E2E TESTS

https://github.com/przemyslawjanpietrzak



Set your status

Przemyslaw Pietrzak
przemyslawjanpietrzak

PRO

Software engineer, enthusiast of new technologies (but only if are better than old ones). Open source and functional programming fan.

Pinned repositories

Customize your pinned repositories

rembrandt
Simple functional UI framework written in Reasonml.
● OCaml ★ 39

pyMonet
High abstract python library for functional programming. Contains algebraic data structures known (or unknown) from Haskell or Scala.
● Python ★ 16

RxTowerDefense
Tower defense engine written in TypeScript with rx.js6, three.js, and pattern from Cycle.js.
● TypeScript ★ 6 1

stanza.io-examples-tests
Examples of communication with stanza.io library by XMPP protocol, as jasmine unit tests
● JavaScript ★ 2 1

dotfiles
Script for prepare fresh ubuntu instance to developers needs, like python, node, docker, vscode etc etc, etc.
● Shell

Table of Contents

cypress

* Open source (runner)

* Written in Node.js

* Based on Electron and Chromium

* Battery included paradigm

End-to-end testing is a methodology used to test whether the flow of an application is performing as designed from start to finish. The purpose of carrying out end-to-end tests is to identify system dependencies and to ensure that the right information is passed between various system components and systems.

Chapter I
Overview

Getting started

```
npm install cypress --save-dev
npx cypress init
```

Files

```
├── fixtures
│   └── recruitments.json
├── integration
│   ├── e2e
│   │   ├── recruitment.spec.js
│   ├── mocked
│   │   ├── recruitment.spec.js
├── plugins
│   └── index.js
├── screenshots
├── support
│   ├── commands.js
│   ├── index.js
│   ├── recruitment.js
└── utils.js
```

Test

```
it('saves recruitment when form filed', () => {

  cy
    .click('#newRecruitmentButton')

    .get('#name').type(name)
    .selectFirstFromInputDropdown('#supervisor')
    .click('#saveRecruitmentBottom')

    .get('#toast-container .toast-success').should('exist')
    .get('#supervisorsError').should('not.exist')
  ;
});
```

Command (page object)

```
Cypress.Commands.add('login', () => {
  cy
    .visit('localhost:4201/#/')
    .get('#inputEmail').type('admin')
    .get('#current-password').type('admin1')
    .get('#login').click();
});
```

Fixture

```json
{
  "data": {
    "id": 582,
    "name": "John Doe"
    "status": true,
    "position": 8
  },
  "status": 200
}
```

Demo #1

Chapter II

Test what/how?

What?

* Business logic

* Positive paths

* DB modifications

Why?

* E2E tests are costly
* Required much time to run
* Have to be supported well

How?

Assertion messages

Expect 5 to equal 4 ???

```
it('create new candidate', () => {
  cy
    .createCandidate(someData)
    .goToCandidatesList()
    .getCandidatesNumber().equal(candidates + 1, 'new candidate was NOT added to 
list')
  ;
});
```

Random data

```
it('Update candidate data', () => {
  const name = generateRandomString();
  const description = generateRandomString(40);
  cy
    .goToFirstCandidate()
    .editCandidate()

    .get('#name').type(name)
    .get('#description').type(description)
    .click('#saveRecruitmentBottom')

    .get('#toast-container .toast-success').should('exist')
    .get('.toast-error').should('not.exist')

   .get('#name').should('equal', name)
    .get('#description').should('equal', description)
  ;
});
```

Separate it

* Each test scenario must have his own data

* Prepare mocked database and reset it before test run

*? Mock all endpoints

Chapter III
HTTP spy

Prepare route

```
beforeEach(() => {
  cy.server({ delay: 1000 });
  cy.route('GET', 'candidates', 'fixture:candidates.json')
});
```

Demo #2

Prepare route

```
beforeEach(() => {
  cy.server({ delay: 1000 });
  cy.route({
    method: 'POST',
    url: '/api/candidates/42',
    response: { status: 200, data: {} },
    onRequest: ({ request }) => {
      lastRequest.body = request.body;
      wasCandidateCreated.done = true;
      },
    });
});
```

Assertion

```
it('create new candidate should send proper request', () => {
  cy
    .createCandidate(someData)

    .wrap(wasJobAdCreated).its('done').should('equal', true)
    .wrap(lastRequest).its('body.name').should('equal', name)
    .wrap(lastRequest).its('body.description').should('equal', description)
  ;
});
```

Chapter IV

CI

Docker

```
docker pull cypress
docker run --volume=~/code/project/:/src --network=host cypress -c bash "npx
cypress"
```

Run backend

```
(npm run backend & echo $! > backend.pid & (sleep 42 && npm run cypress))
kill $(echo backend.pid)
```

PROS	CONS
* Great debugger	* Only chrome
* Async assertion	* Only JavaScript
* Mock http	* Hard to parallel
* Battery included	

Thank you :*