End to End Testing

Ablauf

Warum End-to-End Testing ?

Warum End-to-End Testing ?
Was ist cypress.io ?

Warum End-to-End Testing ?

Was ist cypress.io ?

Wie läuft der Workshop ab ?

Warum End-to-End Testing ?

Was ist cypress.io ?

Wie läuft der Workshop ab ?

Lass mich mal machen !

Warum End-to-End Testing ?

Was ist cypress.io ?

Wie läuft der Workshop ab ?

Lass mich mal machen !

Warum End to End Testen ?

Warum
End to End
Testen ?

Warum

End to End

Testen ?

Unit Test

Funktioniert diese Komponente ?

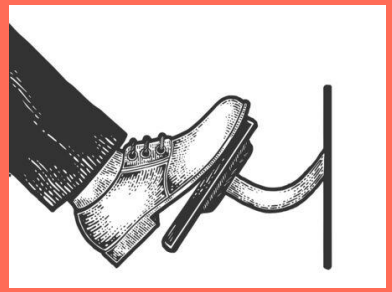
Unit Test

Input

Gaspedal betätigen

Test Scope

Gaspedalsensor erkennt Druckstärke



Output

Gaspedalsensor gibt Druckstärke aus

80%

Integration Test

Funktioniert das Zusammenspiel dieser Komponenten ?

Integration Test

Input

Gaspedal betätigen

Test Scope

Gaspedalsensor erkennt Druckstärke

↓

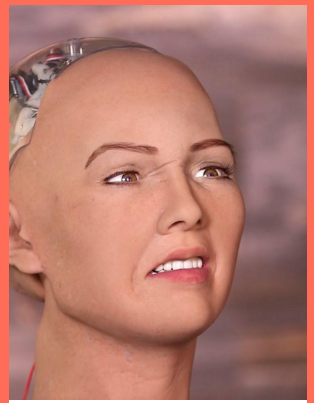
Gaspedalsensor gibt Druckstärke an Motorsteuereinheit weiter

↓

Motorsteuereinheit interpretiert Stärke und wählt Aktion

Output

Motorsteuereinheit wählt Aktion "Motor aufdrehen"



Full Power !

End to End Test

Funktioniert das Zusammenspiel aller Komponenten von User Input bis für User erkenntlichen Output ?

End to End Test

Input

Gaspedal betätigen

Test Scope

Gaspedalsensor erkennt Druckstärke

↓

Gaspedalsensor gibt Druckstärke an Motorsteuereinheit weiter

↓

⁉

↓

Kraft wird auf Räder übertragen

Output

Auto bewegt sich vorwärts

End to End Test

Input

Gaspedal betätigen

Test Scope

Gaspedalsensor erkennt Druckstärke

Gaspedalsensor gibt Druckstärke an Motorsteuereinheit weiter

Unit Test !?

Kraft wird auf Räder übertragen

Output

Auto bewegt sich vorwärts

End to End Test

Input

Gaspedal betätigen

Test Scope

Gaspedalsensor erkennt Druckstärke

↓

Gaspedalsensor gibt Druckstärke an Motorsteuereinheit weiter

↓

Integration Test ⁉

↓

Kraft wird auf Räder übertragen

Output

Auto bewegt sich vorwärts

End to End Test

Input

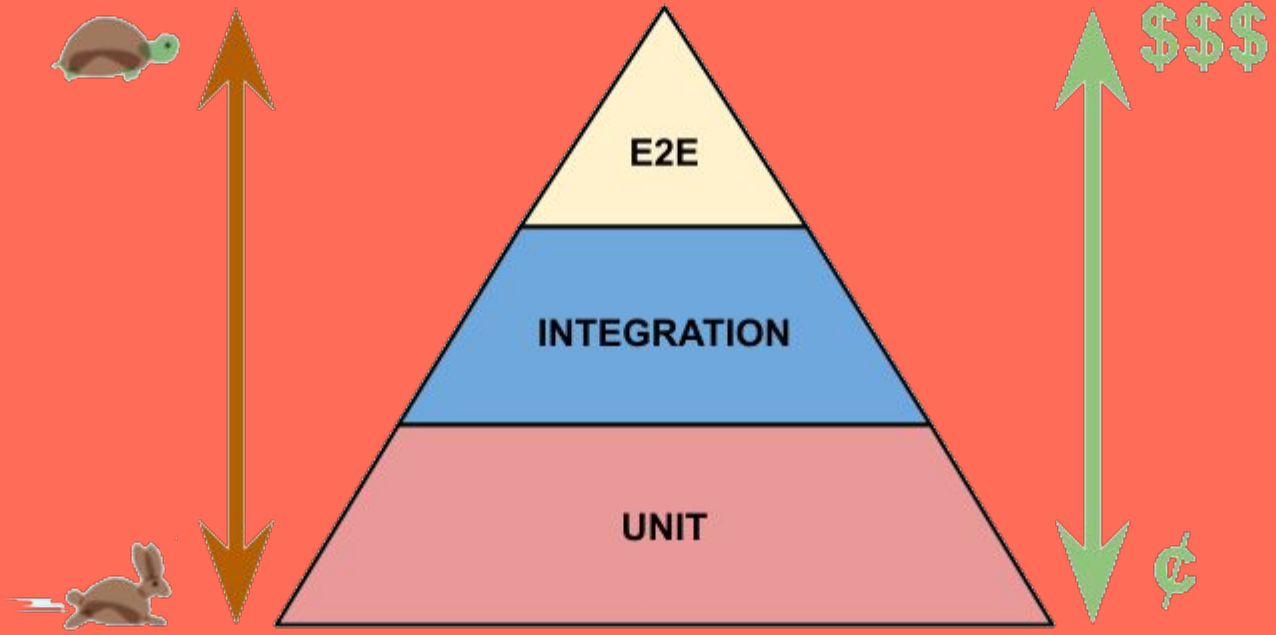
Gaspedal betätigen

Test Scope

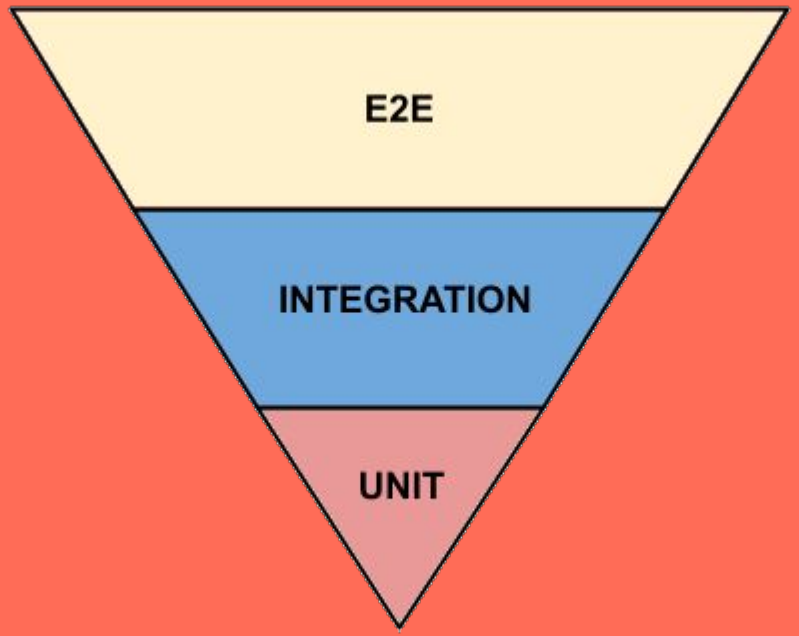
Ist dem End to End Test egal !

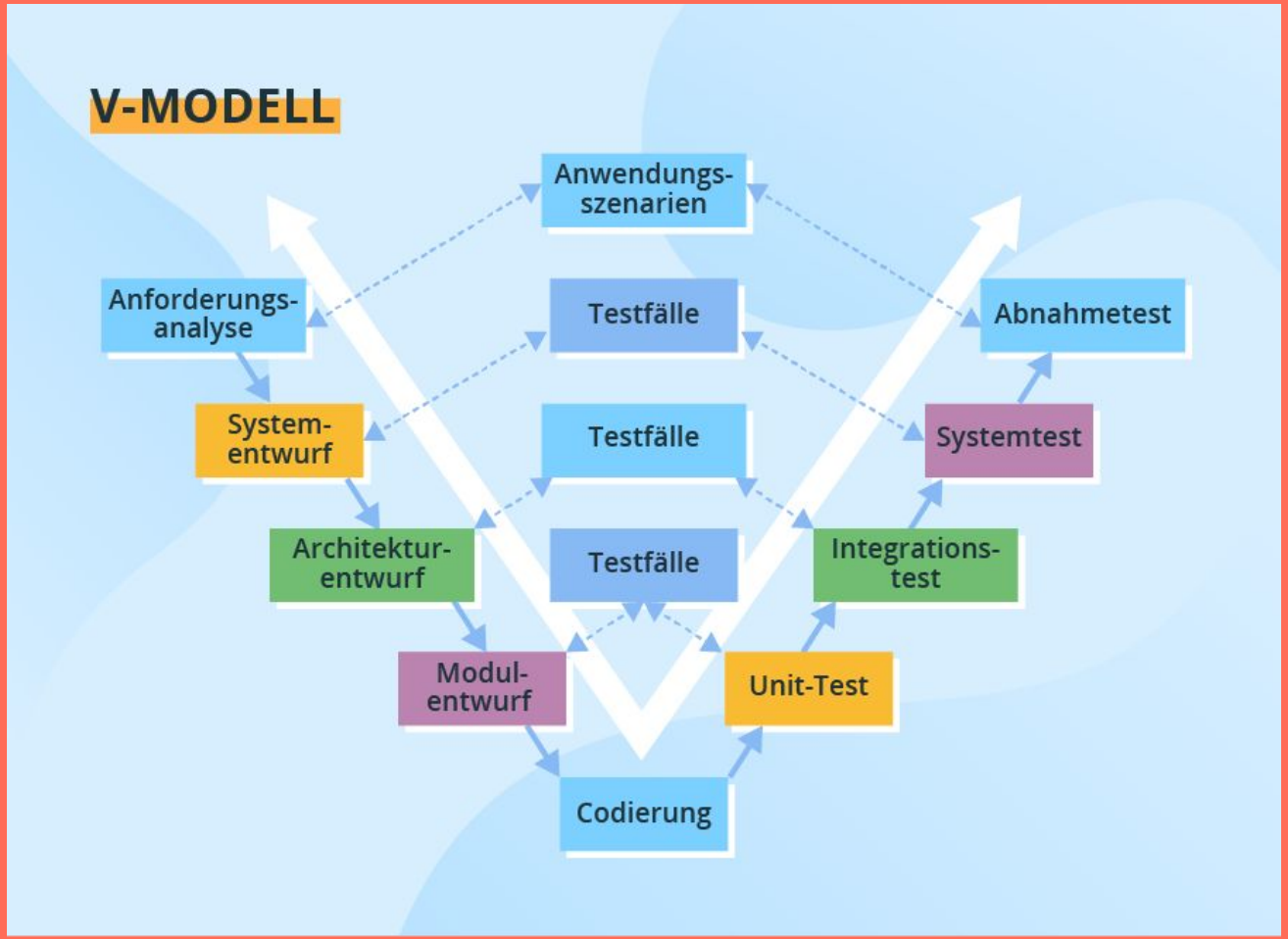
Output

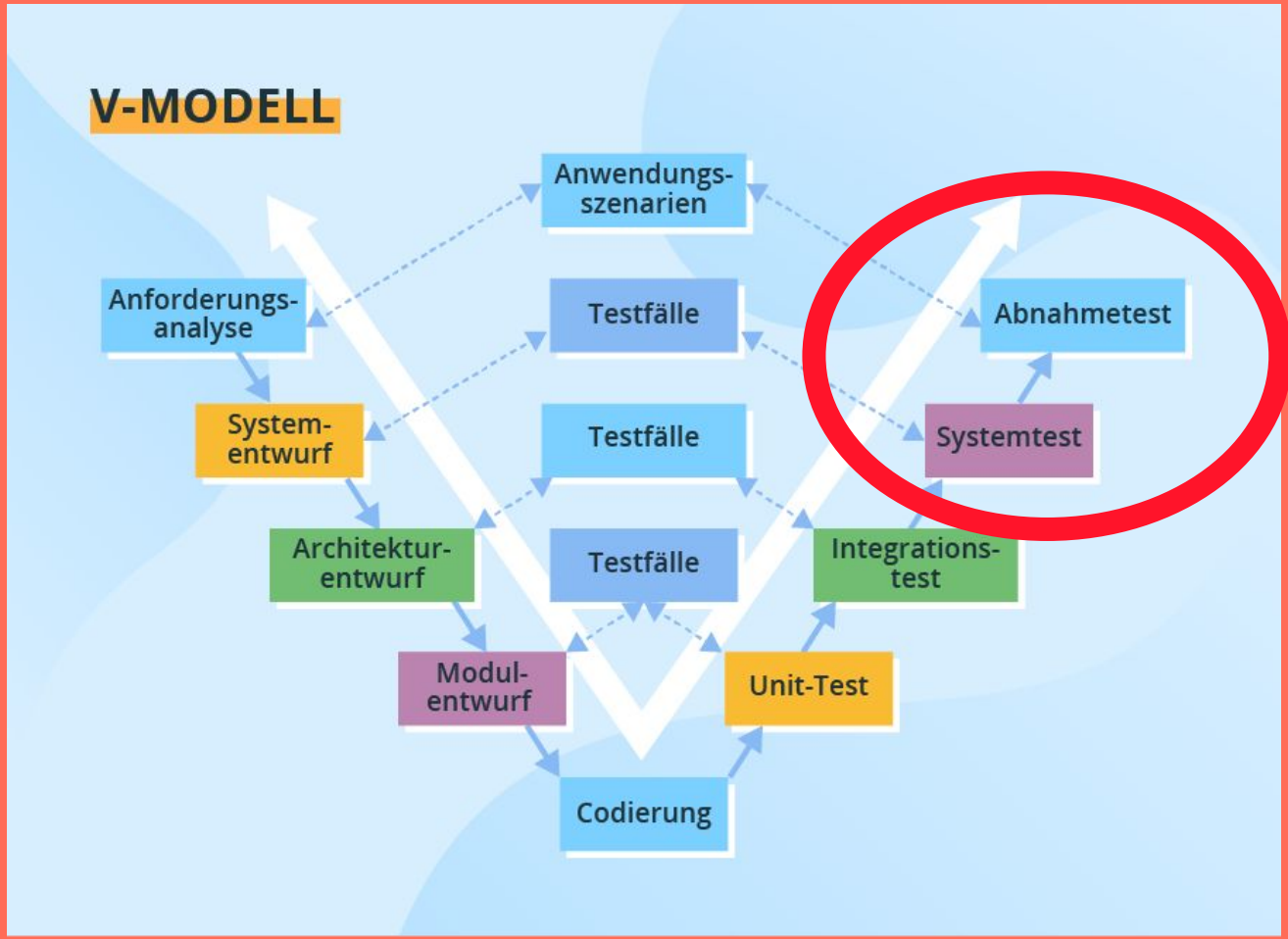
Auto bewegt sich vorwärts

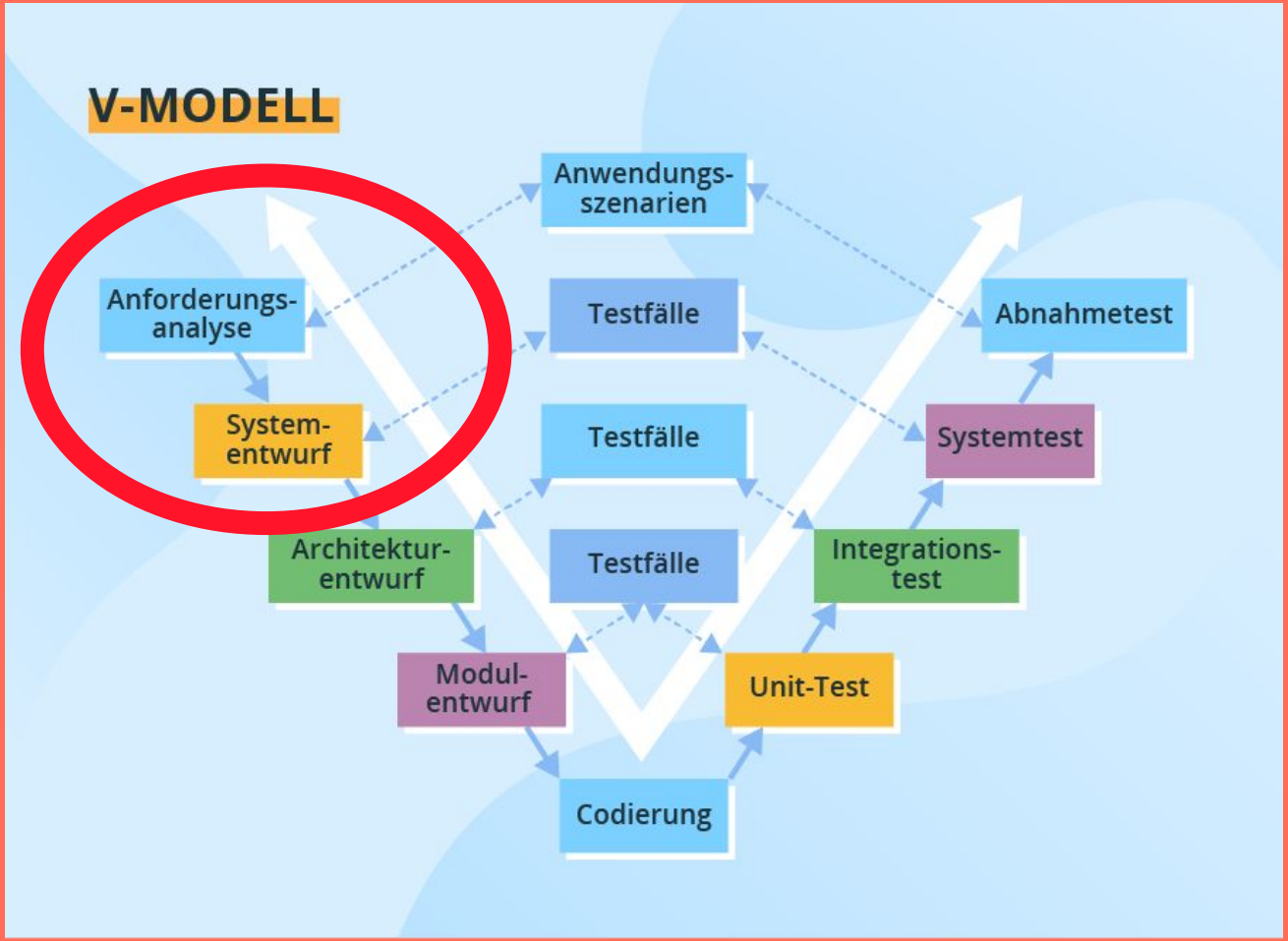

E2E
INTEGRATION
UNIT
$$$
¢

Wichtig
für Nutzer




V-MODELL
Anwendungs-
szenarien
Anforderungs-
analyse
Testfälle
Abnahmetest
System-
entwurf
Testfälle
Systemtest
Architektur-
entwurf
Testfälle
Integrations-
test
Modul-
entwurf
Unit-Test
Codierung


V-MODELL
Anwendungs-
szenarien
Anforderungs-
analyse
Testfälle
Abnahmetest
System-
entwurf
Testfälle
Systemtest
Architektur-
entwurf
Testfälle
Integrations-
test
Modul-
entwurf
Unit-Test
Codierung


V-MODELL
Anwendungs-
szenarien
Anforderungs-
analyse
Testfälle
Abnahmetest
System-
entwurf
Testfälle
Systemtest
Architektur-
entwurf
Testfälle
Integrations-
test
Modul-
entwurf
Unit-Test
Codierung

Warum End to End Testen ?

Fragen ?

Was ist cypress.io ?

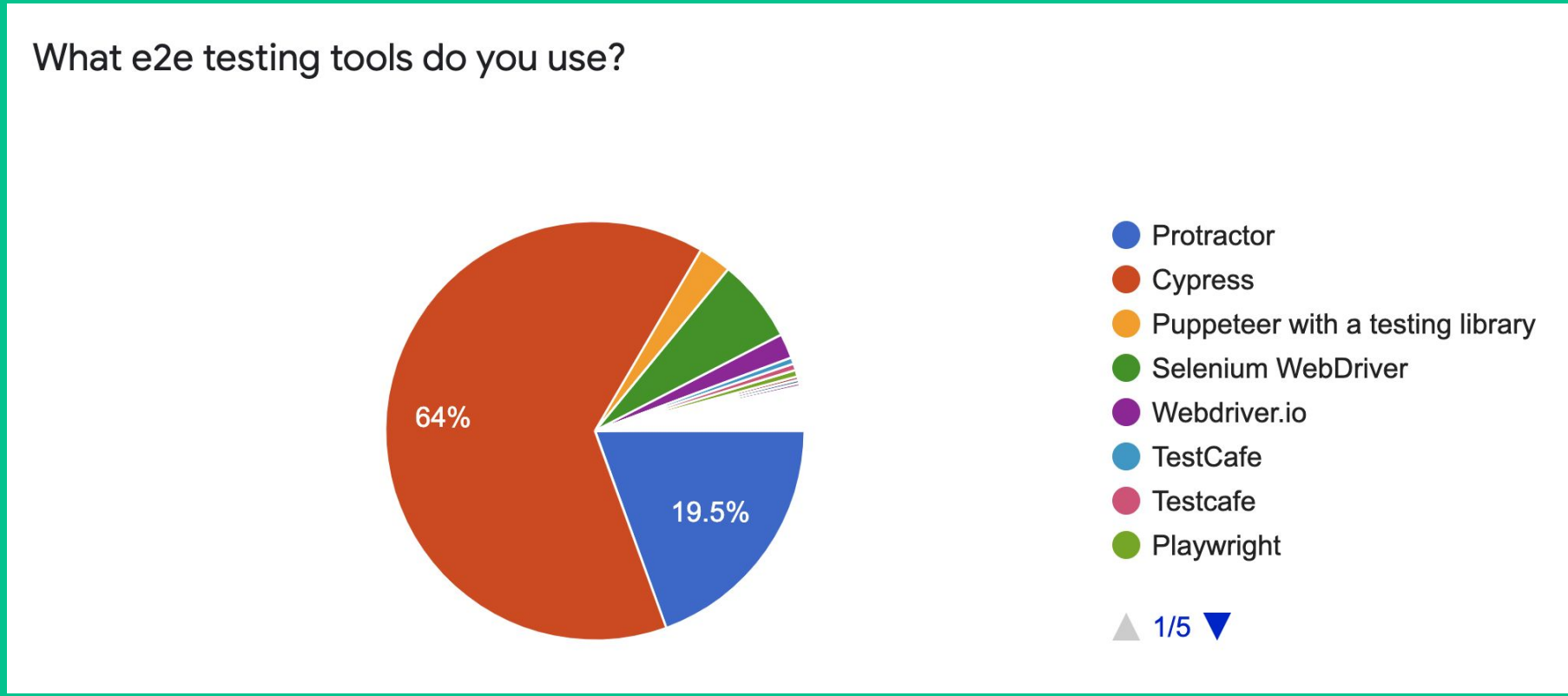
“Fast, easy and reliable testing for anything that runs in a browser.”

*“Fast, easy and reliable testing for **anything that runs in a browser**.”*

*“Fast, easy and reliable testing for anything that runs in a **browser**.”*

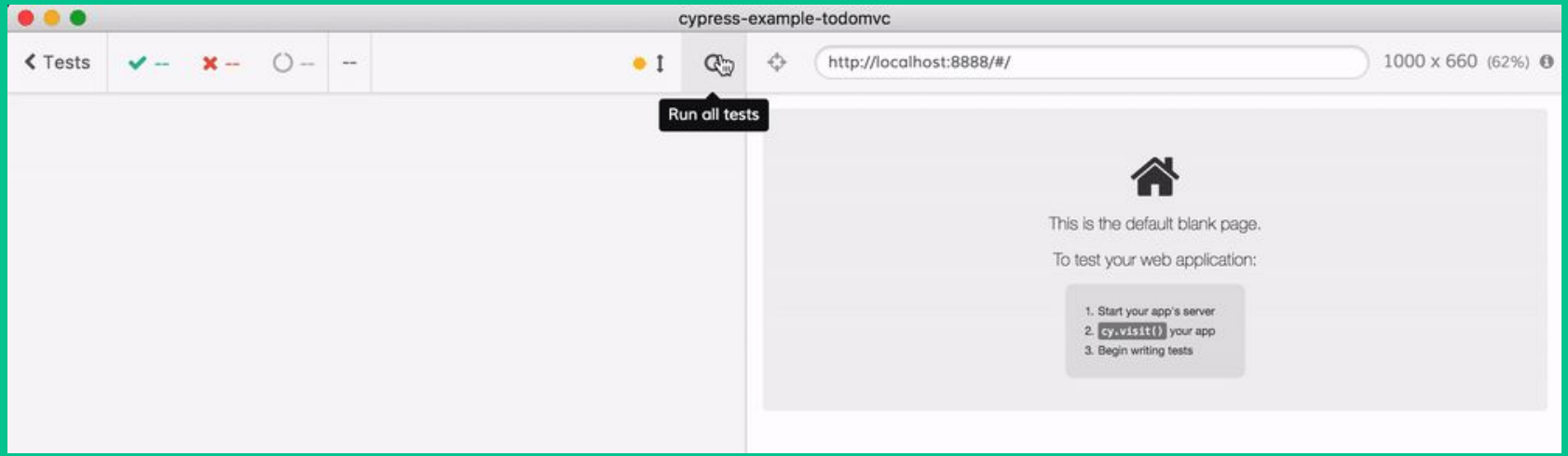
https://github.com/angular/protractor/issues/5502

survey zu e2e (survey von protractor -> angular community)

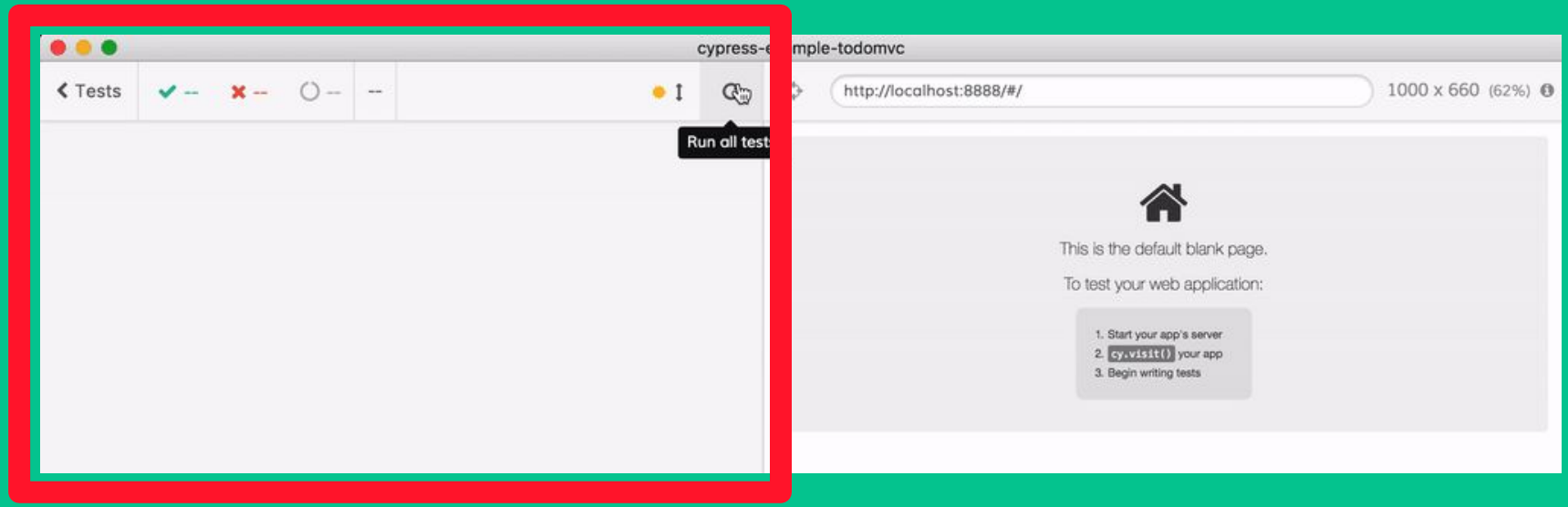


Was genau macht Cypress ?

Test Runner

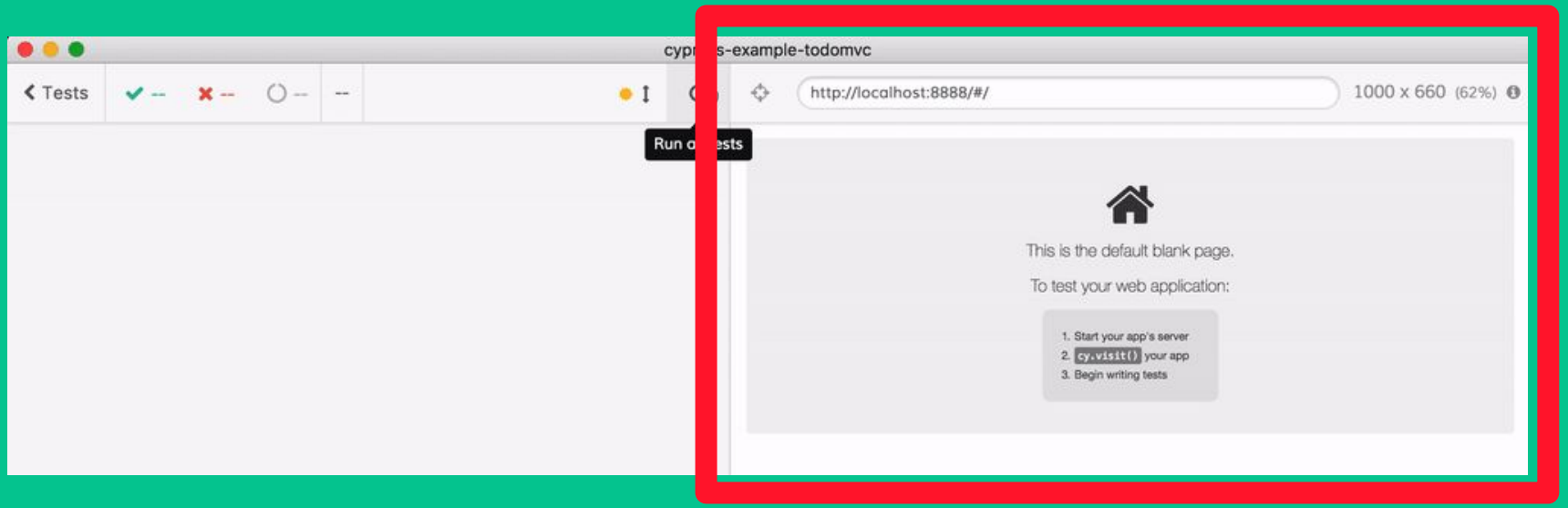

cypress-example-todomvc
Tests
Run all tests
http://localhost:8888/#/
1000 x 660 (62%)
This is the default blank page.
To test your web application:
1. Start your app's server
2. cy.visit() your app
3. Begin writing tests

Test Runner



durchlaufende Tests

Test Runner



Webapplikation


Email eingeben
Passwort eingeben
Login Button drücken
...


Email eingeben
Passwort eingeben
Login Button drücken
...

Email eingeben → Passwort eingeben → Login Button drücken → ...

\+  (State)

\+  (State)

\+  (State)

\+  (State)

Email eingeben → Passwort eingeben → Login Button drücken → ...

\+ 📸 (State) + 📸 (State) + 📸 (State) + 📸 (State)

Wie ist ein Test aufgebaut ?

```
describe('Example Application', () => {

    beforeEach(() => {
        cy.visit('https://localhost:3000');
    })

    it('allows user to sign in', () => {
        cy.get('Sign in').click();
        cy.get('h1').should('contain.text', 'Sign in');
        ...
    })

    it('does something else', () => {
        ...
    })

})
```

```
describe('Example Application', () => {

    beforeEach(() => {
        cy.visit('https://localhost:3000');
    })

    it('allows user to sign in', () => {
        cy.get('Sign in').click();
        cy.get('h1').should('contain.text', 'Sign in');
        ...
    })

    it('does something else', () => {
        ...
    })

})
```

Tests organisieren

```
describe('Example Application', () => {

    beforeEach(() => {
        cy.visit('https://localhost:3000');
    })

    it('allows user to sign in', () => {
        cy.get('Sign in').click();
        cy.get('h1').should('contain.text', 'Sign in');
        ...
    })

    it('does something else', () => {
        ...
    })

})
```

Test Runner beauftragen

```
describe('Example Application', () => {

    beforeEach(() => {
        cy.visit('https://localhost:3000');
    })

    it('allows user to sign in', () => {
        cy.get('Sign in').click();
        cy.get('h1').should('contain.text', 'Sign in');
        ...
    })

    it('does something else', () => {
        ...
    })

})
```

Ergebnisse checken

```
describe('Example Application', () => {

    beforeEach(() => {
        cy.visit('https://localhost:3000');
    })

    it('allows user to sign in', () => {
        cy.get('Sign in').click();
        cy.get('h1').should('contain.text', 'Sign in');
        ...
    })

    it('does something else', () => {
        ...
    })

})
```

```
describe('Example Application', () => {

    beforeEach(() => {
        cy.visit('https://localhost:3000');
    })

    it('allows user to sign in', () => {
        cy.get('Sign in').click();
        cy.get('h1').should('contain.text', 'Sign in');
        ...
    })

    it('does something else', () => {
        ...
    })

})
```

Setup

Interagieren

Evaluieren

Wie macht cypress  ?

Test Runner

Test Runner

Cypress Dashboard

- Manager für Test Runs
- Nutzer & Projekte managen
- Tests parallel laufen lassen
- Load Balancing
- Kostenlos testbar (3 Nutzer - bis 500 Testergebnisse (500x *‘it’*) / Monat)

Tradeoffs

- Browser Support ist limitiert
- kein generelles Automatisierungstool
- nur Javascript (für immer)
- nicht mehrere Tabs gleichzeitig
- nicht mehrere Browser gleichzeitig

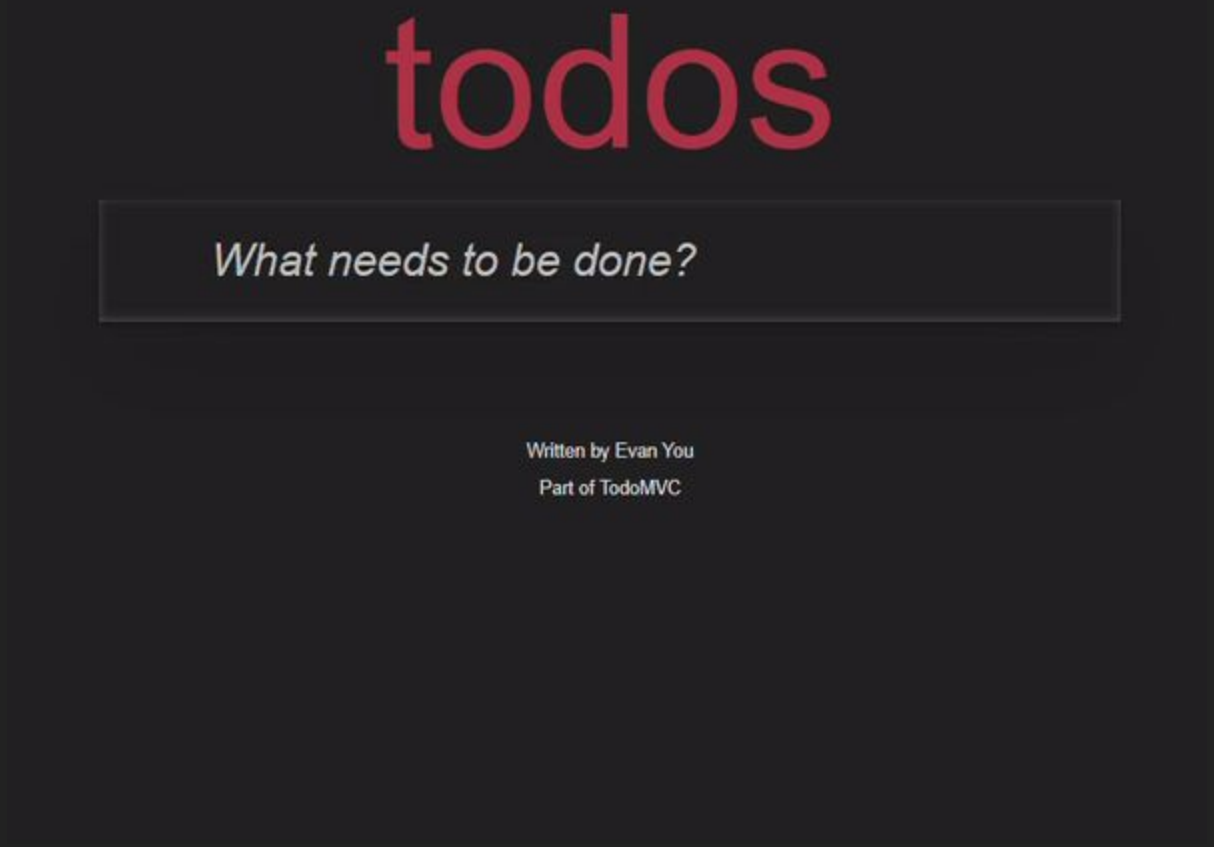
Nice to Know

- Videos von Tests exportieren
- Access auf Network Layer
- Komponenten Tests (E2E-Unit-Tests) möglich
- Headless Mode
- Dokumentation ist SEHR gut
- funktioniert auf jeder beliebigen Website
- sonstiges → gleich

Fragen ?

Wie läuft der Workshop ab ?

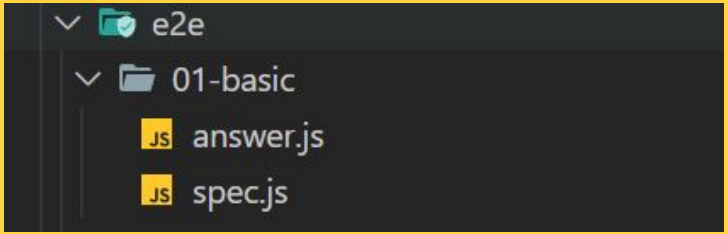
Ping Pong Prinzip


todos
What needs to be done?
Written by Evan You
Part of TodoMVC

Repository & Aufgaben

Installationsanleitung in README.md

Aufgaben in */cypress/e2e*



TodoMVC unter */todomvc*

Lass mich mal machen !

Basics

Cypress Selectors

- CSS Selector
- Sehr ähnlich zu jQuery $(...)
- Überprüfbar in DevTools

```
cy.get('div.className')
cy.get('.className')
cy.get('a#id')
cy.get('[data-cy=custom-id]’)
cy.contains('a#id',’Text’)
```

Basics

- cy.visit(URL)
- cy.get(selector)
- cy.contains(selector?, content)

Hands On!
01-basic

Adding items

Cypress Assertions

Chai Assertion Library

```
cy.contains('a#id',’link’).should('be.hidden’)
cy.get('a#id').should('be.visible)
cy.get('a#id').should('have.class', 'class')
cy.get('a#id').should('not.have.value','val’)
```

Cypress Chains

- cy.[command] startet eine Promise Chain
- Chain ist asynchron
- Timeouts

```
cy.get('.slow-selector',{ timeout: 10000 })
cy.get('button').click()
cy.contains('a#id',link’).should('be.hidden’)
```

Adding items

- cy.type(content)
- cy.check()
- cy.click()
- cy.should(chainers, value)
- cy.find(selector)

Hands On!
02-adding-items

Reset State

Reset State

cy.writeFile(filePath, contents, encoding)

```
cy.writeFile('path/to/message.txt', 'Hello World')
```

cy.task(name)

cy.fixture(path, encoding, options)

Cypress Plugins

- werden in Node Environment ausgeführt
- Möglichkeit eigenen Code in Cypress Lifecycle einzubinden mit cy.task()

```
cy.task('hello', { greeting: 'Hello', name: 'World' })
```

```
module.exports = (on, config) => {
  on('task', {
    // deconstruct the individual properties
    hello({ greeting, name }) {
      console.log('%s, %s', greeting, name)
        return null
    },
  })
}
```

Cypress Fixtures

Laden von Daten aus einer Datei

Hilfreich in Automatisierung

Asynchron

```
cy.fixture('users').as('usersJson')
cy.fixture('logo.png').then((logo) => {
  // load data from logo.png })
```

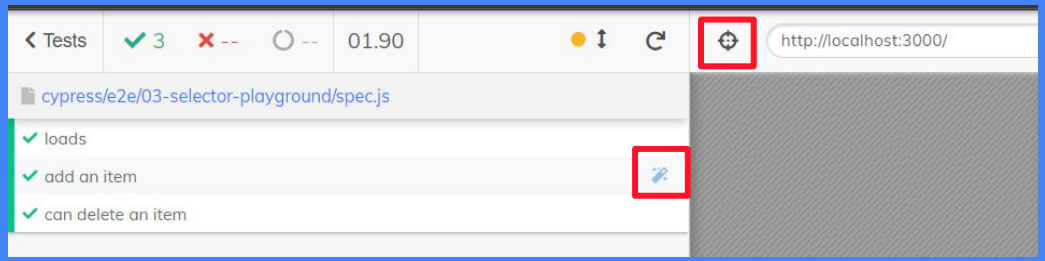
Hands On!
04-reset-state

Selector Playground

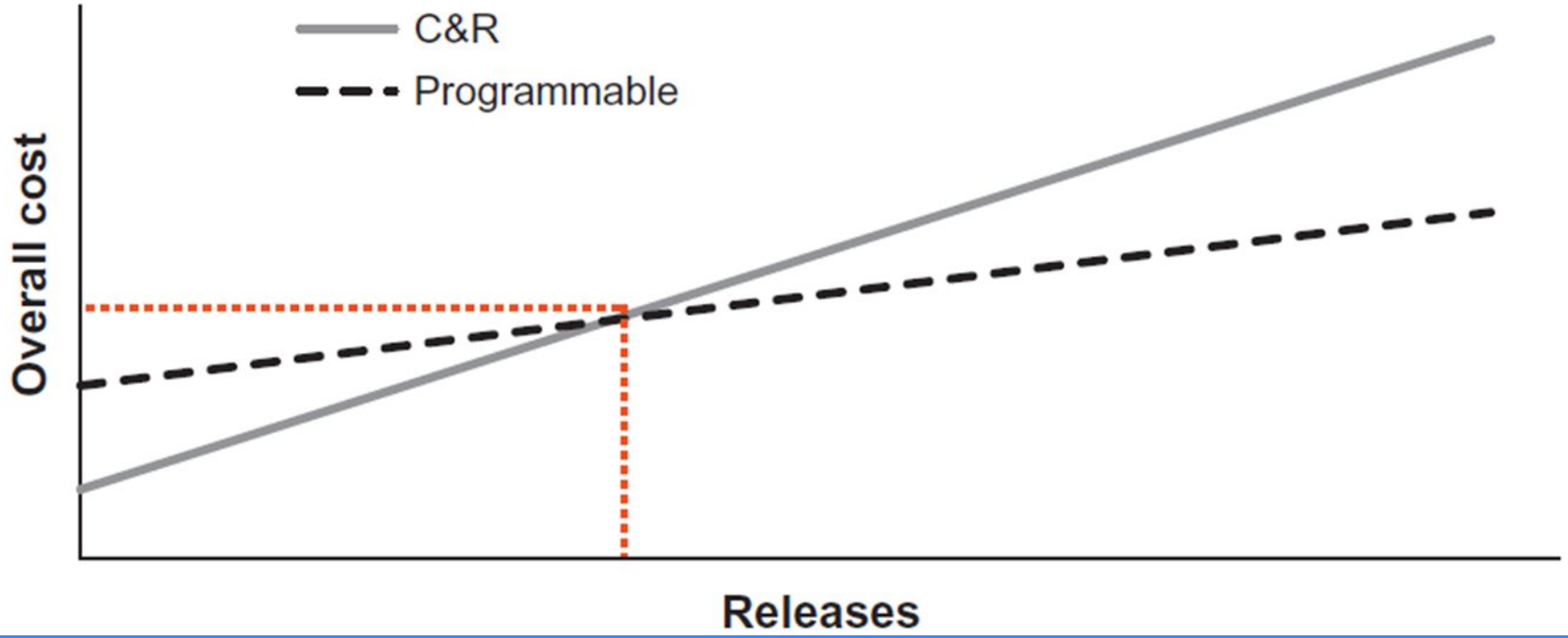
Cypress Studio

Vorschau von möglichen CSS Selektoren

Capture replay



C&R vs. Programmable



Wrap-Up

Takeaways

- E2E Tests simulieren User Experience
- Grundlagen für das Verständnis von Cypress gelegt
- Cypress bietet noch viele Möglichkeiten

Cypress.io Workshop

20 Lessons

Aufteilug in Easy, Intermediate & Hard

https://github.com/cypress-io/testing-workshop-cypress

Danke für eure Teilnahme !

Fragen ?

Referenzen

https://www.cypress.io/

https://itnext.io/front-end-testing-strategy-5fddfd463feb

https://www.scnsoft.de/blog/vorgehensmodelle-der-softwareentwicklung

https://mochajs.org/

https://www.chaijs.com/

https://todomvc.com/

https://opensource.org/licenses/MIT