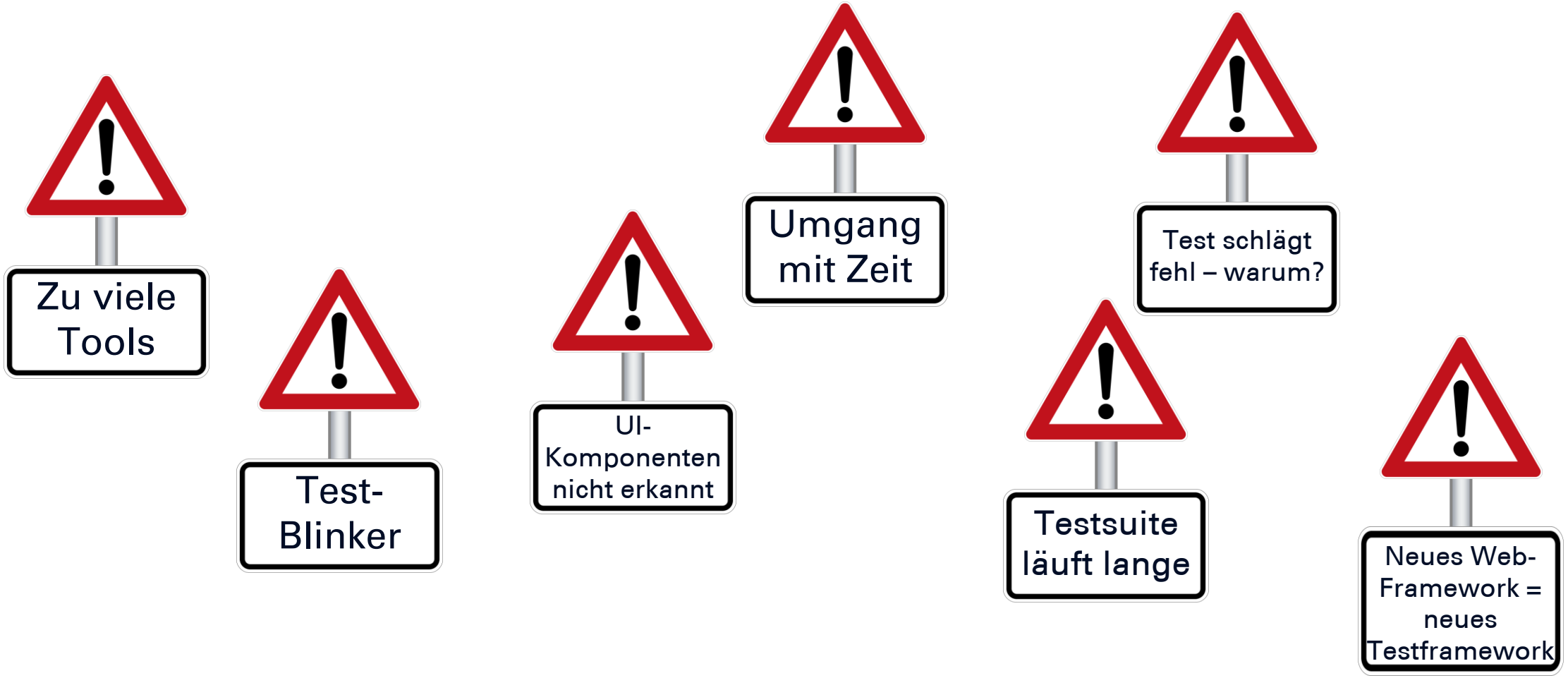
Cypress überall

Ein einziges Automatisierungswerkzeug für alle Teststufen?!

Dehla Sokenou

48. Treffen der GI-Fachgruppe TAV

Kennt ihr das auch?



Fragt man einen Backend-Entwickler…

Unit-Tests?

xUnit!

Integrationstest?

xUnit!

E2E-Tests?

xUnit!?

Fragt man einen Frontend-Entwickler…

Unit-Tests?

Jasmine/Karma?

Enzyme?

Jest?

Vitest?

Integrationstest?

Jasmine/Karma?

Enzyme?

Jest?

Vitest?

E2E-Tests?

Cypress?

Selenium?

Playwright?

Geht das nicht besser?

Vielleicht so…?

Unit-Tests?

Cypress!

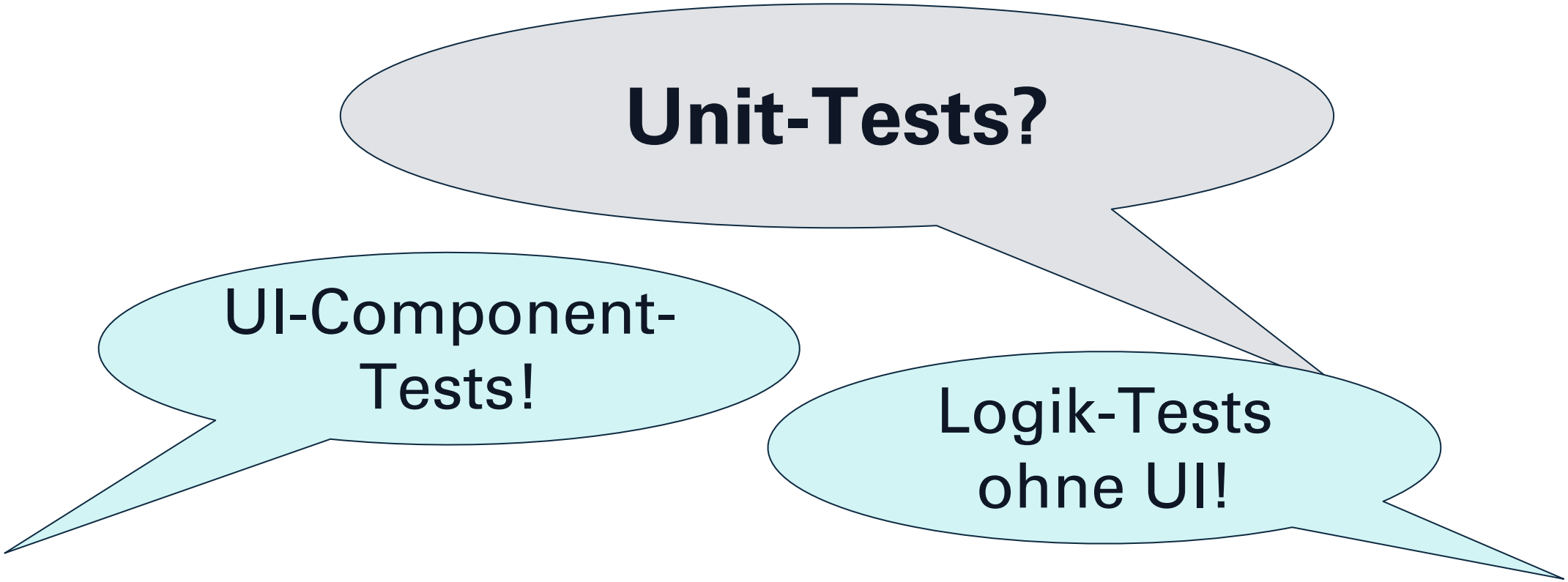
Integrationstest?

Cypress!

E2E-Tests?

Cypress!

Unit-Tests im Frontend…?

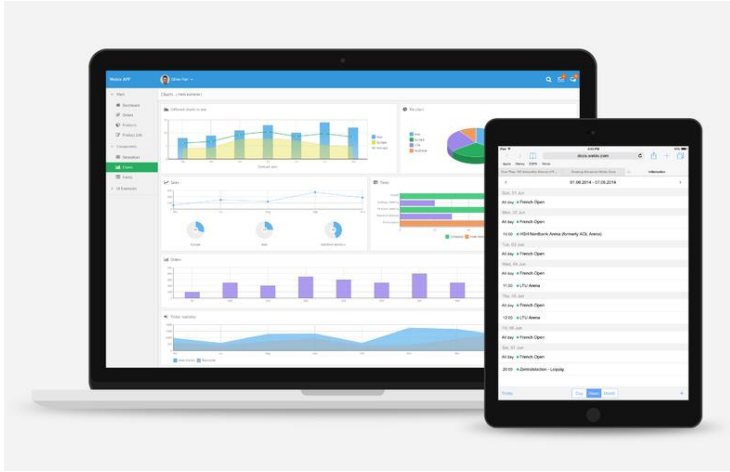


Cypress – ein kleiner historischer Abriss

- 2014/15 gestartet als Gegenpol zu Selenium und anderen Capture & Replay Werkzeugen, Open Source
- 2017 Version 1.0 → E2E Testing
- 2021 Version 7.0 → Component Testing (alpha)
- 2022 Version 10.0 → Component Testing (beta), sukzessiver Ausbau der Framework-Unterstützung (React, Vue, Angular, …)

Zielplattformen

- Cypress E2E ist für jede Art der Webanwendung geeignet
  - Unterstützt insbesondere auch moderne Single-Page-Apps, bei denen eine Reihe anderer Werkzeuge schwächelt
- Cypress Component Testing muss spezifisch für das Webframework bereitgestellt werden
  - Dann aber: einheitliche Technik, unabhängig davon, ob man ein Angular, Vue, React, … Projekt hat
  - Jest unterstützt auch verschiedene Frameworks, aber nur Unit-, Integration, Component Tests, kein E2E



Wie funktioniert Cypress?

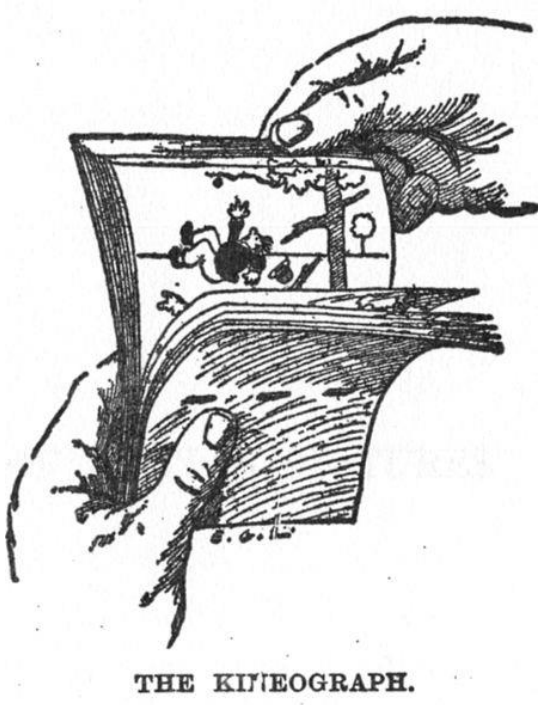
- Tests werden implementiert → auf den ersten Blick ein Nachteil, z.B. im Vergleich zu Capture & Replay, BDD oder Keyword-Driven Testwerkzeugen
  - Aber: wer nutzt das Testwerkzeug tatsächlich?
- Dieselbe Technik kann auf allen Testebenen (zumindest im Frontend) wiederverwendet werden
- Dieselbe Technik kann für alle gängigen Web-Frameworks (zumindest im Frontend) wiederverwendet werden

Und (wie) löst das jetzt unsere Probleme?

Besondere Eigenschaften von Cypress – im Allgemeinen

Einige Eigenschaften von Cypress, die es besonders macht

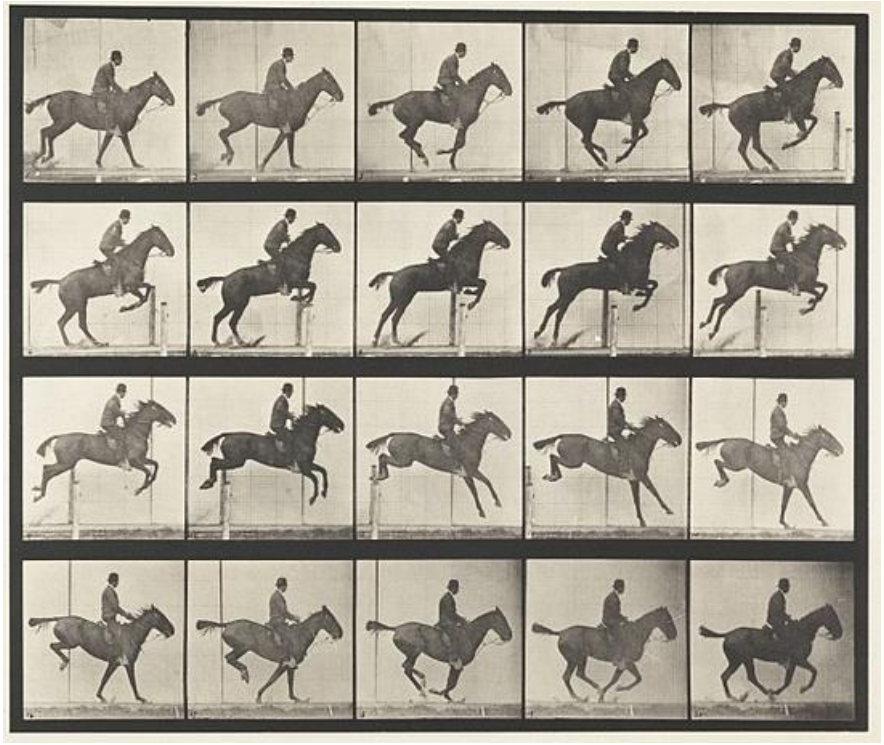
- Daumenkino („Time Travel“)
- Automatische Screenshots / DOMs im Fehlerfall
  → bessere Fehleranalyse
- Clock
- Mocks (Spies, Stubs)
- Automatisches Warten
- Tippen in Benutzergeschwindigkeit



Besondere Eigenschaften von Cypress – E2E-Tests

Einige Eigenschaften von Cypress, die es besonders macht

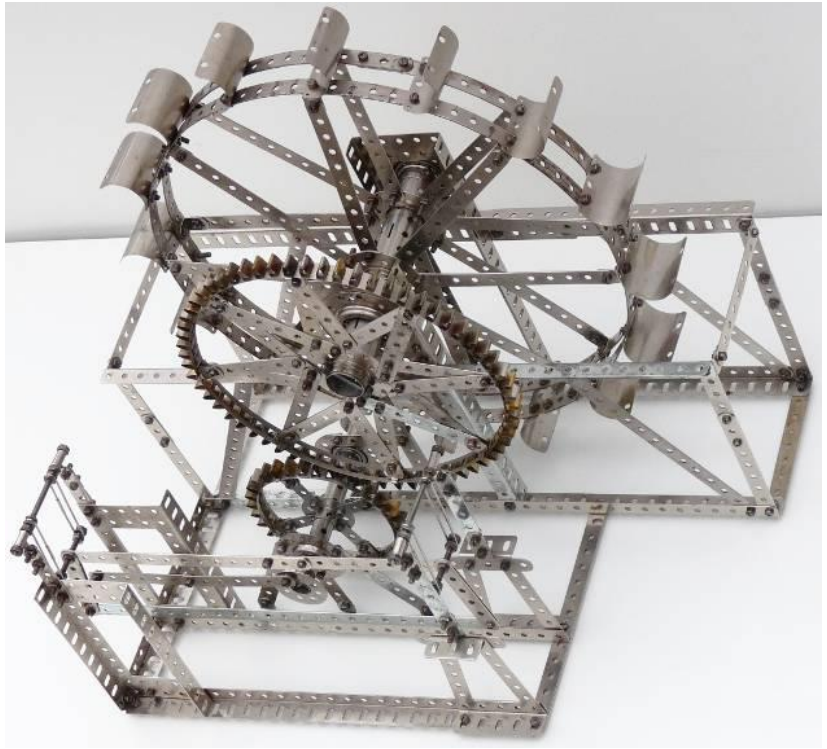
- Sehr stabil, nur wenige Test-Blinker
- Sehr schnell
- Cross-Browser-Tests (im echten Browser)
- Läuft mit der Anwendung in einer Umgebung und erlaubt Kontrolle von innen
- Mocking von Netzwerkverkehr auch im E2E-Bereich



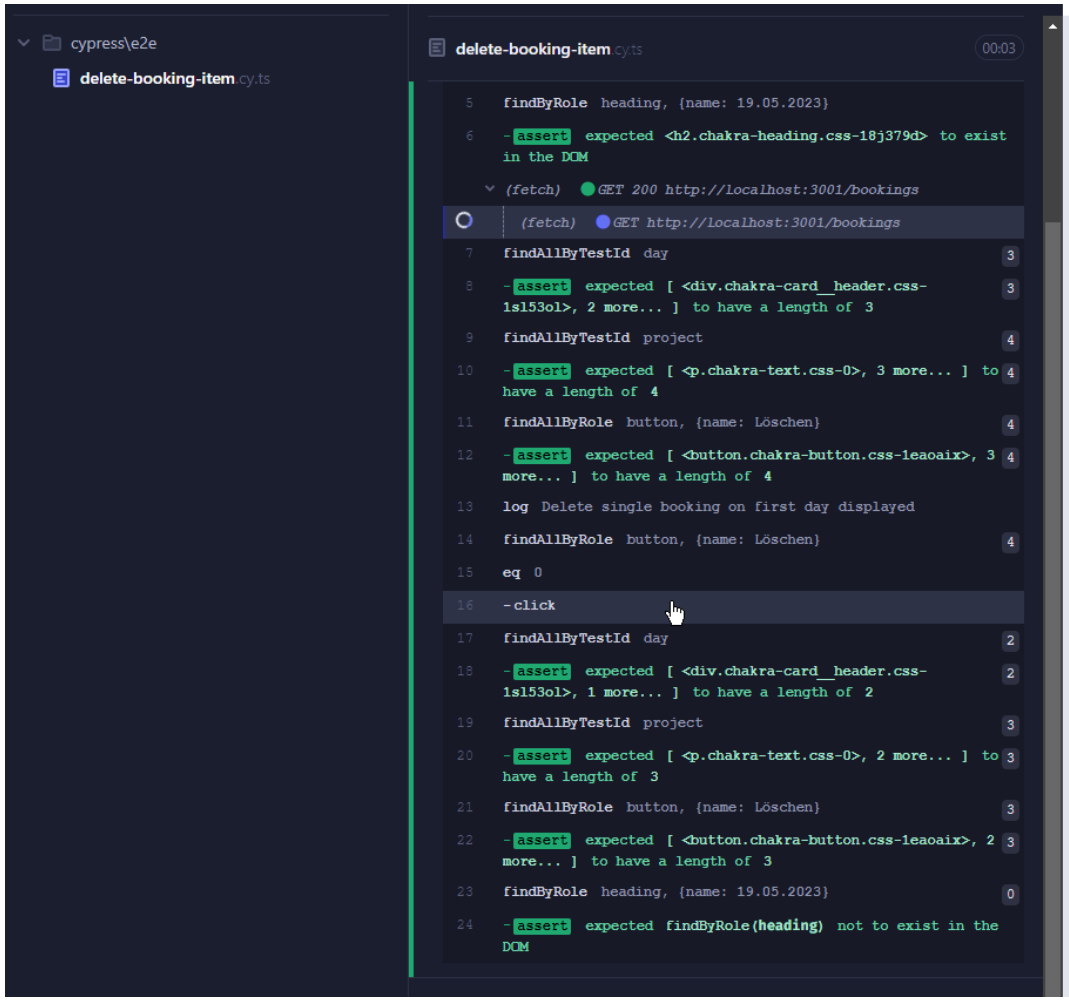
Besondere Eigenschaften von Cypress – Component- und Integration-Tests

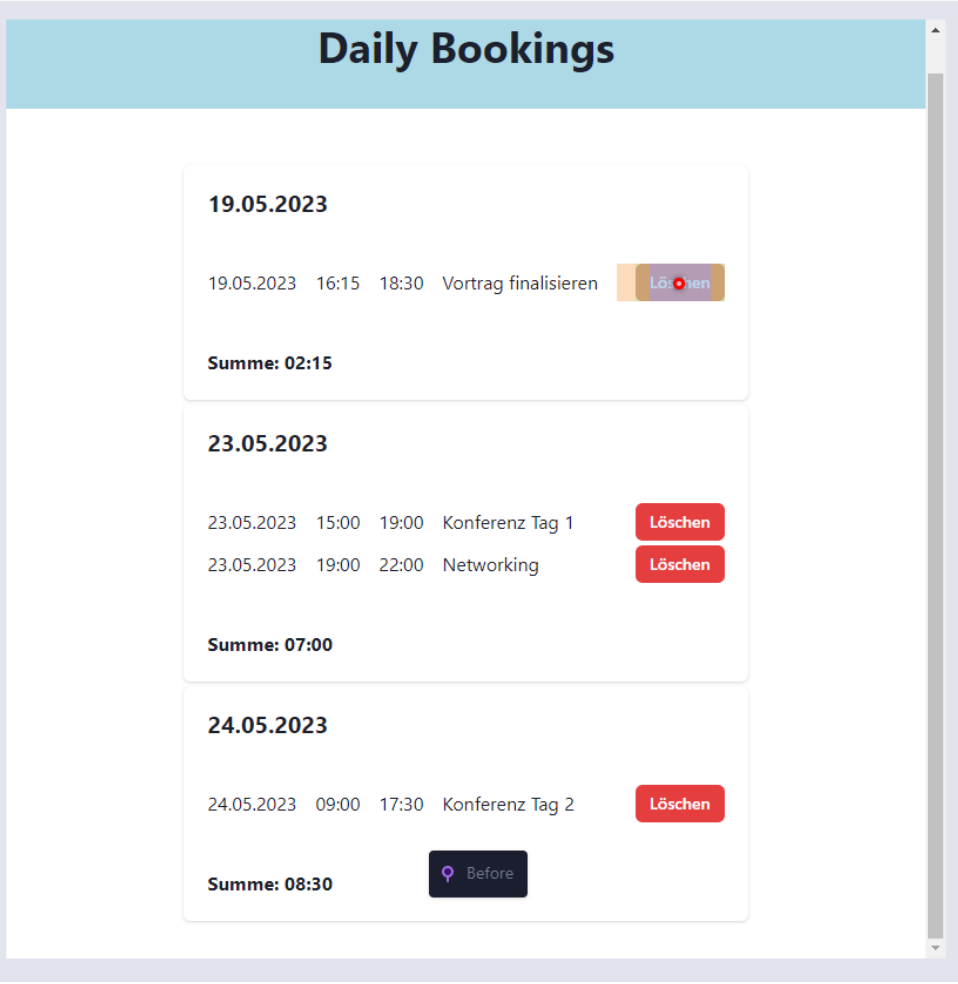
Einige Eigenschaften von Cypress, die es besonders macht

- Sehr stabil, nur wenige Test-Blinker, inbesondere bei viel Nutzerinteraktion
- Man sieht, was getestet wird (wie bei **Karma**, im Gegensatz bspw. zu **Jest**)
- Es ist trotzdem schnell (wie bei **Jest**, im Gegensatz bspw. zu **Karma**)
- Breite Framework-Unterstützung (wie bei **Jest**)



Einblicke in Code


cypress\e2e
delete-booking-item.cy.ts
delete-booking-item.cy.ts
00:03
findByRole heading, {name: 19.05.2023}
-assert expected <h2.chakra-heading.css-18j379d> to exist in the DOM
(fetch) GET 200 http://localhost:3001/bookings
(fetch) GET http://localhost:3001/bookings
findAllByTestId day
-assert expected [<div.chakra-card__header.css-1sl53ol>, 2 more...] to have a length of 3
findAllByTestId project
-assert expected [<p.chakra-text.css-0>, 3 more...] to have a length of 4
findAllByRole button, {name: Löschen}
-assert expected [<button.chakra-button.css-1eaoaix>, 3 more...] to have a length of 4
log Delete single booking on first day displayed
findAllByRole button, {name: Löschen}
eq 0
-click
findAllByTestId day
-assert expected [<div.chakra-card__header.css-1sl53ol>, 1 more...] to have a length of 2
findAllByTestId project
-assert expected [<p.chakra-text.css-0>, 2 more...] to have a length of 3
findAllByRole button, {name: Löschen}
-assert expected [<button.chakra-button.css-1eaoaix>, 2 more...] to have a length of 3
findByRole heading, {name: 19.05.2023}
-assert expected findByRole(heading) not to exist in the DOM


Daily Bookings
19.05.2023
19.05.2023 16:15 18:30 Vortrag finalisieren Löschen
Summe: 02:15
23.05.2023
23.05.2023 15:00 19:00 Konferenz Tag 1 Löschen
23.05.2023 19:00 22:00 Networking Löschen
Summe: 07:00
24.05.2023
24.05.2023 09:00 17:30 Konferenz Tag 2 Löschen
Summe: 08:30
Before

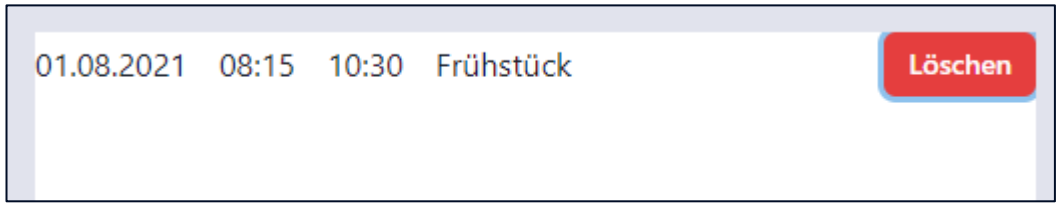
Demo

Cypress überall…

```
describe('Delete Booking Items', () => {
 it('should delete single booking on a day and remove card', () => {
   cy.viewport(800, 800);
   cy.visit('/');

   cy.findByText('Daily Bookings').should('exist‘);
   cy.findByRole('heading', { name: '19.05.2023' }).should('exist');
   cy.findAllByTestId('day').should('have.length', 3);
   cy.findAllByTestId('project').should('have.length', 4);

   cy.log('Delete single booking on first day displayed');
   cy.findAllByRole('button', { name: 'Löschen' }).eq(0).click();

   cy.findAllByTestId('day').should('have.length', 2);
   cy.findAllByTestId('project').should('have.length', 3);
   cy.findByRole('heading', { name: '19.05.2023' }).should('not.exist');
 }); });
```

Cypress überall…

```
describe('BookingItemViewer', () => {

 it('should be possible to delete an item', () => {
  // arrange
  const bookingItem: BookingItem = {
   id: 1,
   start: 1627798500000,
   end: 1627806600000,
   project: 'Frühstück',
  };
  const handleDelete = cy.stub().as('onDelete');
  cy.mount(
    <BookingItemViewer bookingItem={bookingItem} onDelete={handleDelete} />
  );

  // act
  cy.findByRole('button', { name: 'Löschen' }).click();

  // assert
  cy.get('@onDelete').should('be.calledOnceWith', 1);
 });
});
```

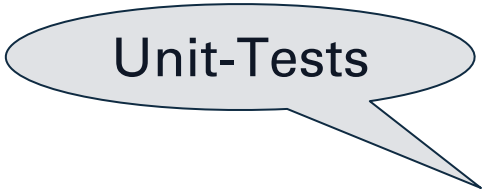


Component-Test

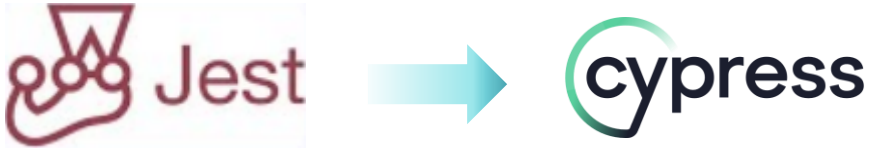
Integrationstest

Cypress überall…

```
describe('Date Util', () => {
  const emptyBookings: BookingItem[] = [];
  const singleBookings: BookingItem[] = […];
  const multipleBookings: BookingItem[] = […];

  [
    { bookingItems: emptyBookings, result: '00:00' },
    { bookingItems: singleBookings, result: '01:00' },
    { bookingItems: multipleBookings, result: '03:30' },
  ].forEach(({ bookingItems, result }) => {
    it('should calculate correct daily sum for bookings', () => {
      // act
      const formattedSum = formatSum(bookingItems);

      // assert
      expect(formattedSum).to.eq(result);
    });
  });
});
```

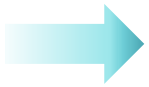


Migration bestehender Testsuiten



- Für welche Tests lohnt es sich?
  - Wir haben vorrangig die Wackelkandidaten / Langläufer migriert
  - Alle weiteren Tests direkt in Cypress
- Wie viel Aufwand ist das?
  - Gleich: ein Beispiel (Teaser: 90% reine Übersetzungsarbeit, 10% Gehirnschmalz)
  - Visueller Input, deshalb schneller bei neuen Tests
- Wie geht man am besten vor?
  - Durch Benennung die Tests trennen (was ist schon konvertiert, was muss noch?)
- Wie ist der aktuelle Stand von Cypress (z.B. im Hinblick auf Stabilität der API, False-Negatives)?

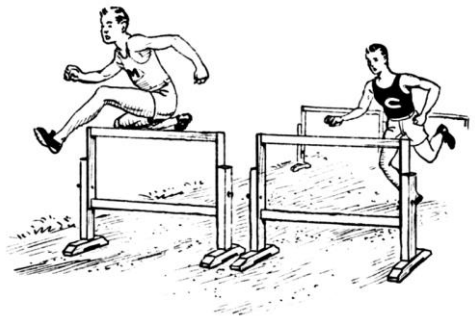
Code-Beispiel Migration



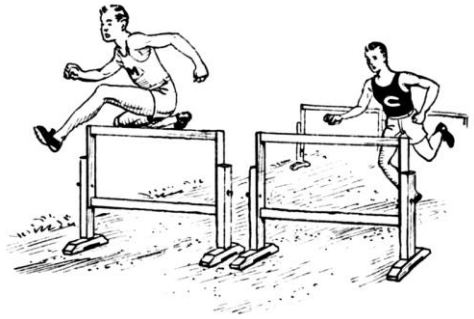
```
it('should be possible to delete an item', async () => {

 fetch.mockResponse(initialBookings);

 const onDelete = jest.fn();

 render(
   <MyComponent onDelete={onDelete} />);

 expect(screen.getByTestId('date')).toHaveTextContent('01.08.2021‘);

 const sums = screen.getAllByTestId('sum‘);
 expect(sums[0]).toHaveTextContent('01:00');

 fireEvent.click(await screen.findByRole('button', { name: 'Löschen' }));

 expect(onDelete).toHaveBeenCalledWith(1);

 expect(formattedSum).to.eq(result);
});
```

```
it('should be possible to delete an item', () => {

 cy.intercept(url, initialBookings);

 const onDelete = cy.stub().as('onDelete')

 cy.mount(
   <MyComponent onDelete={onDelete} />);

 cy.findByTestId('date').should('have.text', '01.08.2021‘);

 cy.findAllByTestId('sum').as('sum‘);
 cy.get('@sum').eq(0).should('contain.text', '01:00');

 cy.findByRole('button', { name: 'Löschen' }).click();

 cy.get('@onDelete').should('be.calledWith', 'id');

 expect(formattedDate).toEqual(result);
});
```

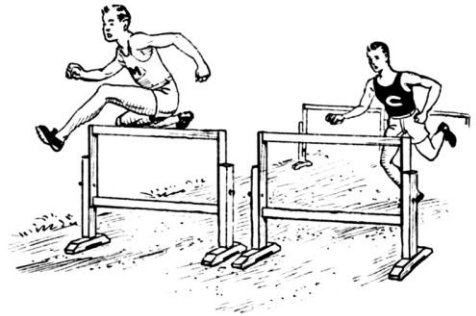
Migration bestehender Testsuiten: Hindernisse?



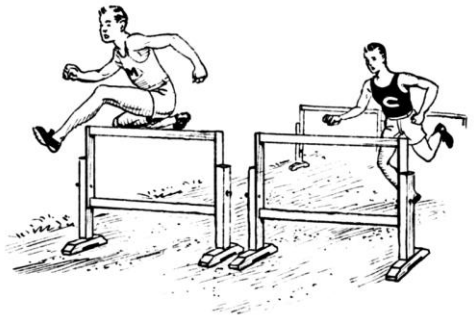
- Hochfahren Cypress-UI und Tests langsam



- Failen von Tests in Cypress-UI



- Unterstützung in IDEs z.T. nicht gegeben (inkl. Debugging)



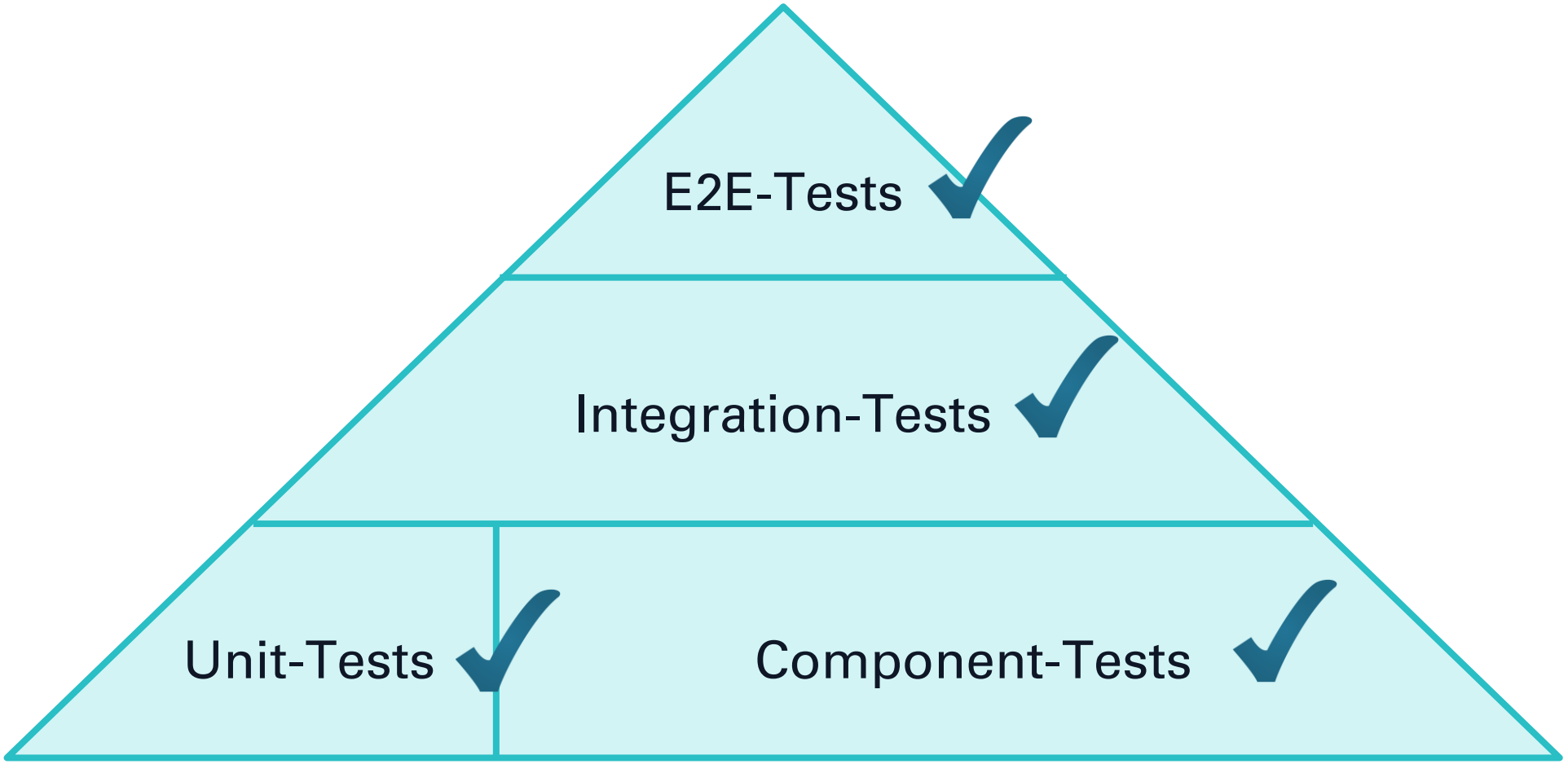
- Manche E2E-Test-Szenarien nicht unterstützt

Aber:

- Andere Arbeitsweise
  - Cypress-UI in Watch-Mode
  - Auto-Restart der Tests
  - Play-Button in IDE nur selten vermisst
- Beschleunigung z.B. durch Vite beschleunigt auch Teststart

Bewertung

Ist es nun ein Werkzeug für alle Teststufen (im Frontend)?



Was man berücksichtigen sollte…

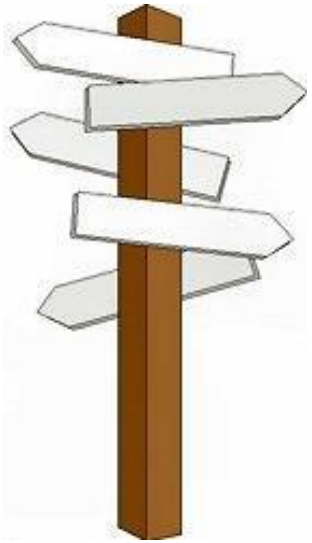
- Wenn man sich so auf ein Werkzeuge festlegt, sind auch folgende Fragestellungen wichtig:
  - Zukunftsfähigkeit (lebt das Projekt?)
  - Harte Brüche zwischen den Versionen mit hohem Anpassungsbedarf?
  - Einschränkungen der Open-Source-Version im Vergleich zur kommerziellen Version?
    - Nicht Open-Source: Cypress Cloud mit Flaky-Test-Management, Parallelisierung und Auto-Rerun / Auto-Cancel
      (aber es gibt Workarounds)

Unser Fazit

- Für wen eignet sich Cypress Component Tests und für wen eher nicht?
  - Ja, wenn Projekte mit unterschiedlichen Frontend-Technologien eingesetzt werden
  - Ja, wenn die Tests sehr flaky sind
  - Vielleicht, wenn man Cypress bereits als E2E-Werkzeug einsetzt
  - Vielleicht, wenn Tests ohne visuelles Feedback nur schwer zu schreiben sind
  - Vielleicht nicht, wenn es neben den Web- auch andere UIs gibt
  - Nein, wenn die automatisierten Tests aktuell problemlos laufen

Unser Fazit

- Was zeigt Cypress, was ein Testwerkzeug können muss?
  → Hilfreich für Bewertungen auch anderer Testwerkzeuge
  - Visuelles Feedback, Daumenkino und Screenshots sind hilfreich für die Fehleranalyse
  - Tests sollten im echten Browser laufen
    - Sonst findet man browserabhängige Fehler nicht
    - Sonst sind einige Tests wegen der Emulation langsam oder flaky
  - Robustheit der Tests ist essentiell
  - Unterstützung durch verbreitete Third-Party-Libraries sinnvoll und erleichtert Portierung von Tests
    - Bsp. Testing-Library



Anhang

Links

Test-Lösungen für die Web-Frontendentwicklung

- Cypress: https://www.cypress.io/
- Cypress Dashboard (Open Source-Variante): https://sorry-cypress.dev/
- Selenium: https://www.selenium.dev/
- Playwright: https://playwright.dev/
- Jest: https://jestjs.io/
- Vitest: https://vitest.dev
- Karma / Jasmine: https://angular.io/guide/testing
- Testing Library: https://testing-library.com/

Bildnachweise (in the order of appearance)

Bilder:

- https://www.cypress.io/
- https://commons.wikimedia.org/wiki/File:Strider_Linkage_Robot_Climbing.gif (CC BY-SA 4.0)
- https://commons.wikimedia.org/wiki/File:Zeichen_101_-_Gefahrstelle,_StVO_1970.svg (Public Domain)
- https://commons.wikimedia.org/wiki/File:Software_Developer_at_work_03.jpg (CC BY-SA 4.0)
- https://commons.wikimedia.org/wiki/File:Software_developer_at_work_02.jpg (CC BY-SA 4.0)
- https://commons.wikimedia.org/wiki/File:Zeichen_206_-_Halt!_Vorfahrt_gew%C3%A4hren!_StVO_1970.svg (Public Domain)
- https://commons.wikimedia.org/wiki/File:Pyramides_et_le_Sphinx_MET_DP113869.jpg (CC0 1.0)
- https://commons.wikimedia.org/wiki/File:JavaScript_UI_widgets_library_for_building_desktop_and_mobile_web_apps.png (CC BY-SA 4.0)
- https://commons.wikimedia.org/wiki/File:Victorinox_climber.jpg (CC BY-SA 4.0)
- https://commons.wikimedia.org/wiki/File:Linnet_kineograph_1886.jpg (Public Domain)
- https://commons.wikimedia.org/wiki/File:Horse_Daisy_jumping_a_hurdle,_saddled_with_a_rider_(rbm-QP301M8-1887-639).jpg (Public Domain)
- https://commons.wikimedia.org/wiki/File:Stabil_Modell_751_klein.jpg (CC BY-SA 4.0)
- https://jestjs.io/
- https://commons.wikimedia.org/wiki/File:Hurdle_(PSF).png (Public Domain)
- https://commons.wikimedia.org/wiki/File:Jack-in-the-box.jpg (Public Domain)
- https://commons.wikimedia.org/wiki/File:Confused_man.jpg (CC BY-SA 2.5)
- https://commons.wikimedia.org/wiki/File:BlackMarble_2016_rotating_globe_at_night_transparent.gif (Public Domain)

Lizenzen:

- CC0 1,0: https://creativecommons.org/publicdomain/zero/1.0/deed.en
- CC BY-SA 2.5: https://creativecommons.org/licenses/by-sa/2.5/deed.en
- CC BY-SA 4.0: https://creativecommons.org/licenses/by-sa/4.0/deed.en