



Cypress or the end of laborious end-to-end testing



@AlainChautard - angulartraining.com

ANGULAR
TRAINING



About me - Alain Chautard (or just AI)

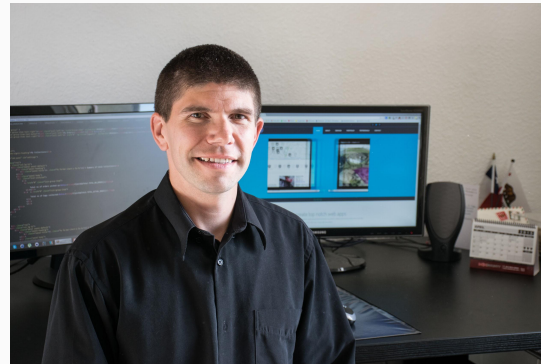
Google Developer Expert in Web technologies / Angular / Google Maps

Java developer since 2006

Angular JS addict since 2011

Web consultant / trainer @ angulartraining.com

Organizer of GDG Sacramento, California



Useful links

- Set-up instructions:

<http://bit.ly/at-cypress-setup>

- Link to my slides:

<http://bit.ly/ac-cypress>

- Repository with code examples:

<https://github.com/alcfeoh/ng-store-cypress>

What is Cypress?

What is Cypress?

- A complete end-to-end testing experience for anything that runs in a browser
- Tests are written with Cypress
- Tests are run with Cypress
- Tests can be debugged with Cypress
- Tests failures are recorded by Cypress


We are going to test a web app: <http://store.angulartraining.com>

[License Plate Store](#) [Home](#) [My cart](#)

Welcome to our store

Browse our collection of License Plates below


2008 Georgia license plate



Sunt occaecat ex nisi reprehenderit dolore esse. Excepteur laborum fugiat sint tempor et in magna labore quis exercitation consequat nulla tempor occaecat. Sit cillum deserunt eiusmod proident labore mollit. Cupidatat do ullamco ipsum id nisi mollit pariatur nulla dolor sunt et nostrud qui.

\$8


2015 New Jersey license plate



A beautiful license plate from the Garden State. Year is 2015.

\$11

2013 California My Tahoe license plate



Sunt irure nisi excepteur in ea consequat ad aliqua. Lorem duis in duis nisi sit. Cillum eiusmod ipsum mollit veniam consectetur ex eiusmod nisi laborum amet anim mollit non nulla. Lorem ea est exercitation nostrud consectetur officia laborum fugiat sint. Nostrud consequat magna officia minim et aute nostrud.

\$9

Let's write a basic test with Cypress

- Cypress is based on **Mocha** and **Chai**, two popular JS libraries for testing

```
// A describe function is a suite of tests
```

```
describe('License plate store home page', () => {
```

```
// A it function is an individual test
```

```
it('displays the right main title', () => {
```

```
// Let's visit the app - actual test goes here
```

```
cy.visit('http://store.angulartraining.com');
```

```
});
```

```
});
```

DEMO #1: First test run

```
npx cypress open
```


Cypress: contains

- **Contains** selects an element by its text contents
- Cypress will wait to find that element till a timeout is reached (4 secs by default)
- The test fails if such element can't be found after that timeout

```
it('displays the right main title', () => {  
  cy.visit('http://store.angulartraining.com');  
  cy.contains('Welcome to our store');  
});
```

Cypress: Assertions

- We can define assertions (and chain those assertions) on every single selected element using **should**
- <https://docs.cypress.io/guides/references/assertions.html#Common-Assertion>
s

```
it('displays the right main title', () => {  
  cy.visit('http://store.angulartraining.com');  
  cy.contains('Welcome to our store')  
    .should('be.visible')  
    .should('have.css', 'font-weight', '300');  
});
```

Lab #1: Testing the title of our app



- Set-up instructions: <http://bit.ly/at-cypress-setup>
- Open a terminal in your project and run **npx cypress open**
- When the Cypress window opens, double click on the file **cypress/labs/1-home-page.spec.js**
- Then complete the test in that file so it checks that:
 - The text “Welcome to our store” is visible
 - Its **font weight** is **300** and **font size** is **72px**
 - A tough one: Its **font family** should match **Segoe UI**

Cypress: get

- **Get** selects an element using a CSS selector
- Cypress provides a selector tool to help find a proper CSS selector
- Just like contains, **get** is going to be retried when needed

```
it('tests something', () => {  
  cy.get('h1');  
});
```

DEMO #2: Test #2

Store should display 8 license plates

beforeEach

- Instead of repeating setup steps in each test, a **beforeEach** function can be used. It runs before every single **it** function

```
describe('License plate store home page', () => {  
  // beforeEach runs before every single test  
  beforeEach(() => {  
    cy.visit('http://store.angulartraining.com');  
  });  
  
  it('displays the right main title', () => {  
    ...  
  });  
});
```

cypress.json

- Cypress can be configured through **cypress.json**
- <https://docs.cypress.io/guides/references/configuration.html#Options>
- Example of **cypress.json** config:

```
{  
  "baseUrl": "http://store.angulartraining.com",  
  "defaultCommandTimeout": 10000  
}
```

DEMO #3: beforeEach

Let's refactor the set-up code

Lab #2: Testing the products displayed in the app



- Open a terminal in your project and run **`npx cypress open`**
- When the Cypress window opens, double click on the file **`cypress/labs/2-check-license-plates.spec.js`**
- Then complete the second test in that file so it checks that:
 - The 8 license plates are displayed in the right order
 - Both **title** and **price** show up with right values for those 8 license plates

BONUS

Cypress commands

- We can interact with a web page through Cypress commands:

```
cy.get('.login').type(email);  
cy.get('.password').type(password);  
cy.get('button[type="submit"]').click();
```

- It's also possible to simulate keys being pressed

(<https://docs.cypress.io/api/commands/type.html#Arguments>) :

```
cy.get("form").type("{enter}");
```

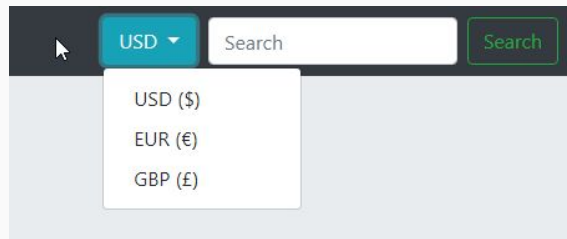
DEMO #4: Clicks

Let's check that we can
navigate to other screens

Lab #3: Testing the currency switcher



- Using the Cypress window, double click on the file **`cypress/labs/3-check-currency-switcher.spec.js`**
- Then write a new test that checks that:
 - The currency can be switched to EUR or GBP by using the dropdown in the nav bar
 - Changing the currency updates all prices and currency symbols



Custom commands

- Instead of repeating common assertions/selectors, it is possible to define custom commands defined in **cypress/support/commands.js**

```
Cypress.Commands.add("login", (email, password) => {  
  cy.visit('/login');  
  cy.get('.login').type(email);  
  cy.get('.password').type(password);  
  cy.get('button[type="submit"]').click();  
});
```

- This command would be used like this:

```
cy.login("test@email.com", "pa$$w0rd");
```

DEMO #5: Custom command

Let's simplify our code with a
custom command

Lab #4: Testing the cart features



- Using the Cypress window, double click on the file **`cypress/labs/5-cart-features.spec.js`**
- All instructions for the tests to write are added as TODOs in that file
- Implement all tests described in that file, and even more if you feel like it

Mocking server responses

- Very often we don't have multiple back-ends or databases to test different scenarios. Sometimes we don't even have one at all!
- Cypress allows us to fix that by mocking server responses:

```
cy.server();  
cy.route('/demos/angular/rates.php', {EUR: 1.5, GBP: 2})
```



URL to mock

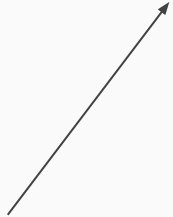


Mocked response

Mocking server responses using fixture files

- Our mock data does not have to be hard-coded in Javascript though
- Cypress allows us to return the contents of a file too:

```
cy.server();  
cy.route('/demos/angular/plates', 'fixture:plates.json');
```



Name of any file located in
cypress/fixtures

Cypress aliases

- Cypress allows us to define aliases on any selected element:


```
beforeEach(function () {  
  cy.get('button').as('myButton')  
});
```

Declaration of the alias



```
it('has access to text', function () {  
  this.myButton // is now available  
});
```

Note that the following syntax only works with regular functions, not arrow functions



Cypress aliases and wait

- Aliases can also be used as a means to wait for a request to complete:

```
beforeEach(() => {  
  cy.server();  
  cy.route('/demos/angular/rates',  
           {EUR: 1.5, GBP: 2}).as('rates');  
  cy.visit('/');  
  cy.wait('@rates');  
});
```



Wait for that data request to complete

DEMO #6: Mocking data

Let's use mocked data so that exchange rates are always the same in our tests

Thanks for your attention

Link to these slides:

<http://bit.ly/ac-cypress>

More tutorials and Angular
content:

<https://blog.angulartraining.com>

