



cypress.io

versus

Yann Helleboid

Orange Innovation 2023



orange™



Playwright

Cypress, Playwright, what ?

Test automation tools

RPA capabilities

Cypress, Playwright, why ?

Open source / Free

Easy / Powerful

Cypress, Playwright, what for ?

HTTP oriented functional tests

GUI and API



Chrome



Firefox



Edge



Electron



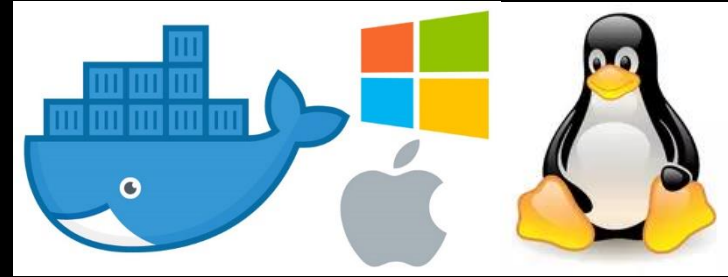
Brave



Webkit / Safari

Cypress, Playwright, where ?

Most platforms



Your machine or on a VM or on a cloud

Cypress, Playwright, how ?

Hands-on training

see by yourselves



means it's your turn to type

Let's go !

Cypress, Playwright, what is different ?



friendly ui
better integrated js/browser
standard libs included



step-by-step debug possible
js + python + others
browser tab management

exercise 1



exercise 1

goal : install the software

install the environment



install a proper editor or IDE

notepad++, eclipse, vscode, vim ...

install node.js

Go to <https://nodejs.org/en>

Follow the install guide for your OS

configure for Orange proxies



eburo => dev access VPN

yourdev => set proxy.rd.francetelecom.fr

launch : 2 modes

open in a GUI => design, debug and analyse
use headed browser

run as a CLI => run tests
use headless browser (or headed)

Cypress, Playwright and browsers



Cypress is shipped with a
electron browser

Cypress can use installed
standard browsers

Cypress can control headed or
headless browsers

Playwright is shipped with out
any browser

You must install specific
browsers for Playwright

Playwright can control headed or
headless browsers

install



```
λ mkdir cypress_training
```

```
λ cd cypress_training
```

```
λ npm init -y
```

```
λ npm install cypress
```



exercise 1

```
λ mkdir playwright_training
```

```
λ cd playwright_training
```

```
λ npm init playwright
```

Getting started with writing end-to-end tests with Playwright:
Initializing project in '.'

- ✓ Do you want to use TypeScript or JavaScript? · JavaScript
- ✓ Where to put your end-to-end tests? · tests
- ✓ Add a GitHub Actions workflow? (y/N) · false
- ✓ Install Playwright browsers (can be done manually via 'npx playwright install')? (Y/n) · true

install



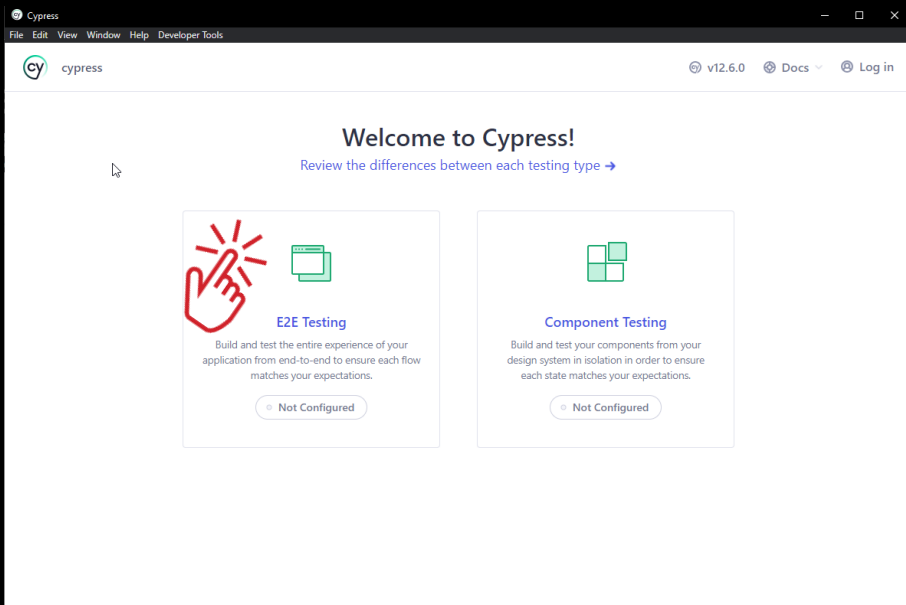
```
λ npx cypress open
```



```
λ npx playwright test --ui
```

```
λ npx playwright test --debug
```

install



exercise 1

install



exercise 1

Cypress


File Edit View Window Help Developer Tools

cy cypress > E2E Testing v12.6.0 Docs Log in

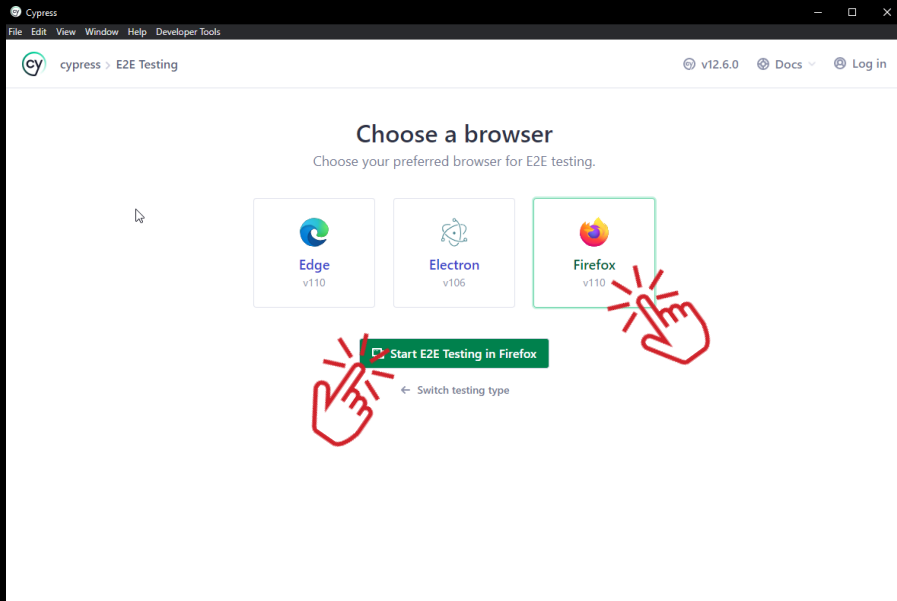
Configuration files

We added the following files to your project:

- [cypress.config.js](#)
The Cypress config file for E2E testing.
- [cypress/support/e2e.js](#)
The support file that is bundled and loaded before each E2E spec.
- [cypress/support/commands.js](#)
A support file that is useful for creating custom Cypress commands and overwriting existing ones.
- [cypress/fixtures/example.json](#)
Added an example fixtures file/folder

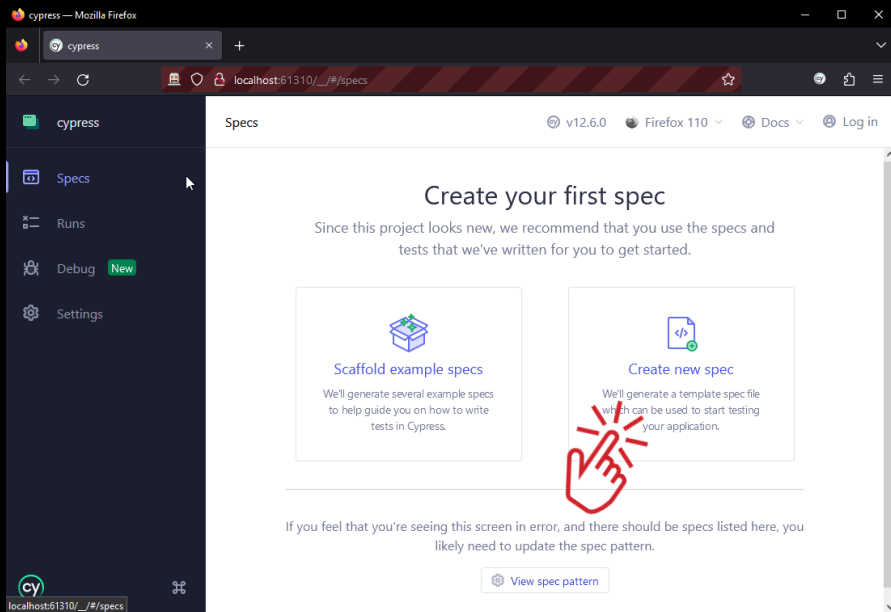
 [Continue](#)

install



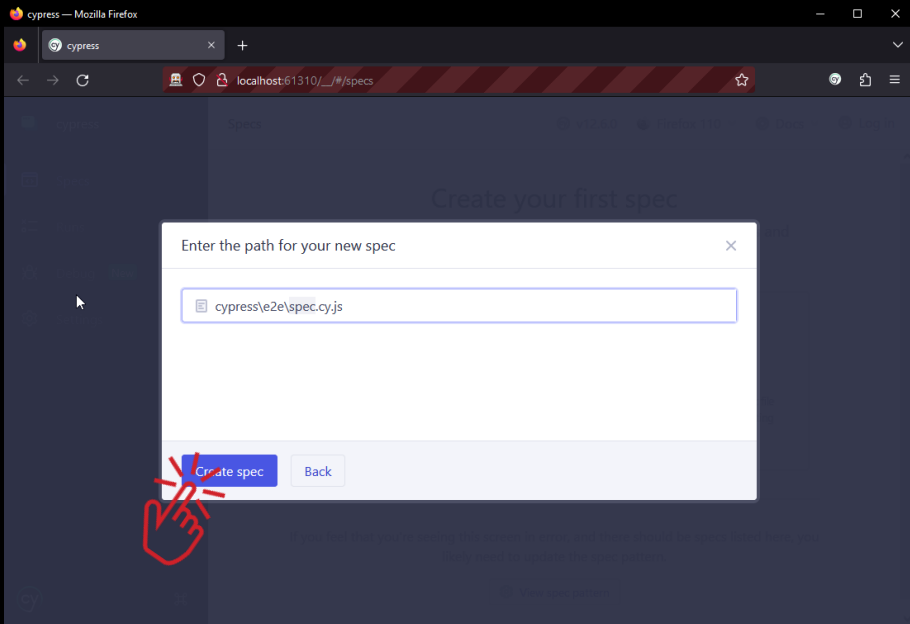
exercise 1

install



exercise 1

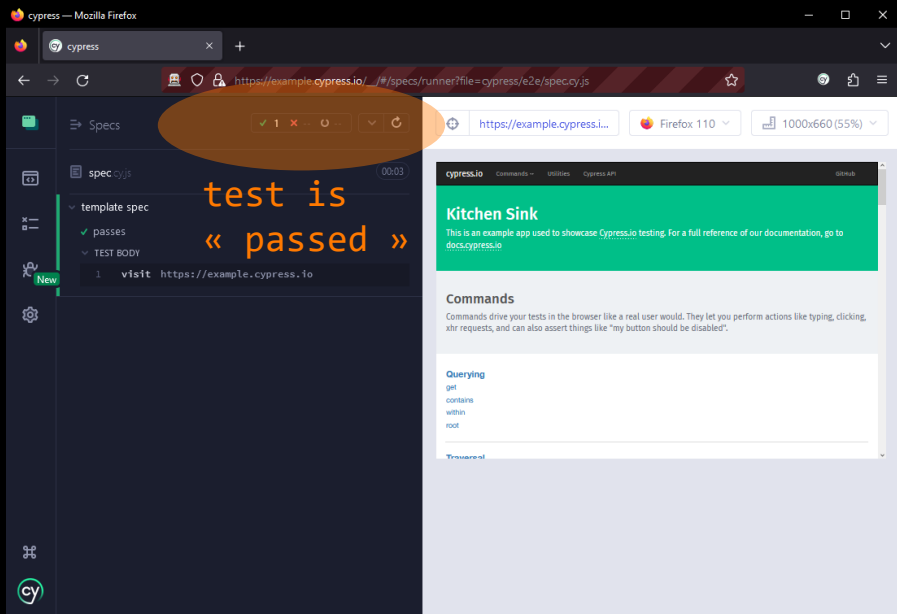
install



exercise 1



install

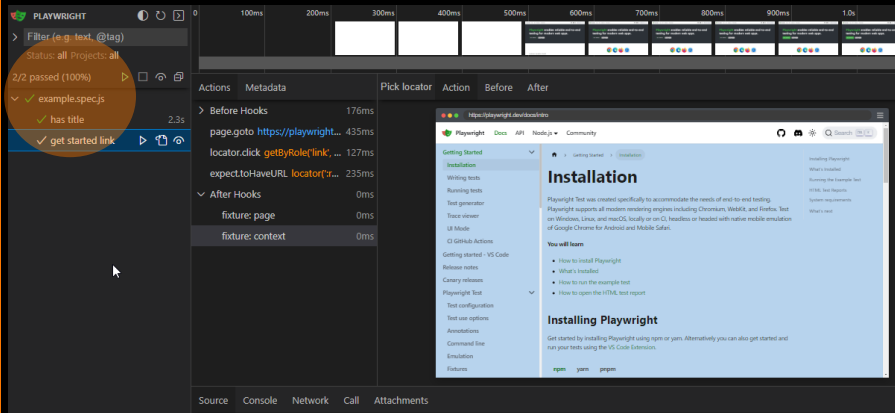


20

Orange Restricted



exercise 1



run headless

headless run will be used in a ci pipeline

exercise 2



exercise 2

goal : headless run

run headless



```
λ npx cypress run --e2e  
--browser firefox
```



exercise 2

```
λ npx playwright test  
--project=firefox
```

run headless



```
D:\dev\cypress
λ npx cypress run --e2e --browser firefox
```

(Run Starting)

```
Cypress:      12.6.0
Browser:      Firefox 110 (headless)
Node Version: v19.7.0 (C:\Program Files\nodejs\node.exe)
Specs:        1 found (spec.cy.js)
Searched:     cypress/e2e/**/*.cy.{js,jsx,ts,tsx}
```

Running: spec.cy.js (1 of 1)

template spec
✓ passes (1789ms)

1 passing (4s)

(Results)

```
Tests:        1
Passing:      1
Failing:      0
Pending:      0
Skipped:      0
Screenshots:  0
Video:        false
Duration:     3 seconds
Spec Ran:     spec.cy.js
```

(Run Finished)

| Spec | Duration | Tests | Passing | Failing | Pending | Skipped |
|---------------------|----------|-------|---------|---------|---------|---------|
| ✓ spec.cy.js | 00:03 | 1 | 1 | - | - | - |
| ✓ All specs passed! | 00:03 | 1 | 1 | - | - | - |



exercise 2

```
D:\dev\playwright (playwright@1.0.0)
λ npx playwright test
```

Running 3 tests using 3 workers
3 passed (9.1s)

To open last HTML report run:

```
npx playwright show-report
```



configuration



Cypress is shipped with a default config :

`cypress.config.js`

Playwright is shipped with a default config :

`playwright.config.js`

exercise 3



exercise 3

goal : configure the tools

configuration



exercise 3

```
λ cat cypress.config.js
const { defineConfig } = require("cypress");
module.exports = defineConfig(
{
  video: false,
  e2e:
  {
    baseUrl: "https://ozh.github.io/cookieclicker",
    setupNodeEvents(on, config) {}
  }
});
```

control video creation
(run mode only)

default url

```
λ cat playwright.config.js
const { defineConfig, devices } =
require('@playwright/test');
module.exports = defineConfig({
  /* Run tests in files in parallel */
  fullyParallel: false,
  use: {
    actionTimeout: 0,
    video: 'off',
    /* Base URL to use in actions like
       `await page.goto('/')`. */
    baseURL: 'https://ozh.github.io',
  },
  /* Configure projects for major browsers */
  projects: [
    { name: 'chromium',
      use: { ...devices['Desktop Chrome'] } },
    { name: 'firefox',
      use: { ...devices['Desktop Firefox'] } },
    { name: 'webkit',
      use: { ...devices['Desktop Safari'] } }
  ]
});
```

run test in //
CAUTION !!!



environment variables



Cypress can use
environment variables from

- `.env` file (using `dotenv`)
- `cypress.config.js`
- `cypress.env.json`
- `CYPRESS_*` system variables
- `cypress cli`



Playwright can use
environment variables from

- `.env` file (using `dotenv`)
- system variables

exercise 4



exercise 4

goal : use environment variables

environment variables



exercise 4

```
λ cat cypress.env.json
{
  "url": "https://ozh.github.io/cookieclicker"
}
```

create variable «url»

```
λ cat cypress/e2e/spec.cy.js
describe('test suite', () =>
{
  it('test', () =>
  {
    cy.visit(Cypress.env('url'));
  })
})
```

use variable «url»

```
λ npm install dotenv
λ cat playwright.config.js
require('dotenv').config();
λ cat .env
url="https://ozh.github.io/cookieclicker"
```

```
λ cat tests/example.spec.js
const {test, expect} = require('@playwright/test');
test.describe('test suite', () =>
{
  test('test', async ({ page }) =>
  {
    await page.goto(process.env.url);
  });
});
```



log



log in Cypress

```
cy.log('my trace');
```

log in the browser console

```
console.log('my trace');
```



log on stdout

```
console.log('my trace');
```

log in the browser console

```
await page.evaluate(() =>  
  console.log('my trace'));
```

exercise 5



exercise 5

goal : log debug traces

log



exercise 5

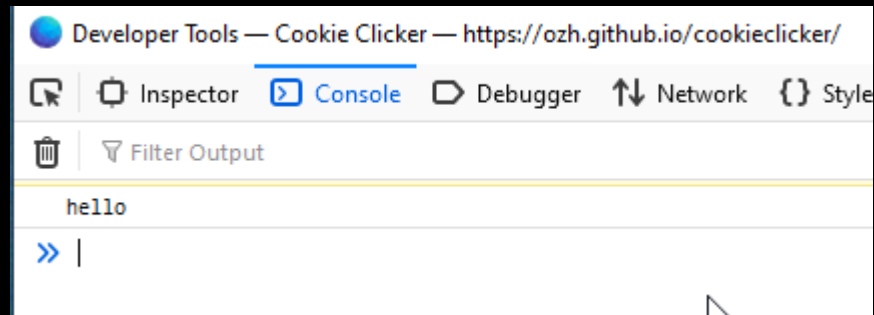
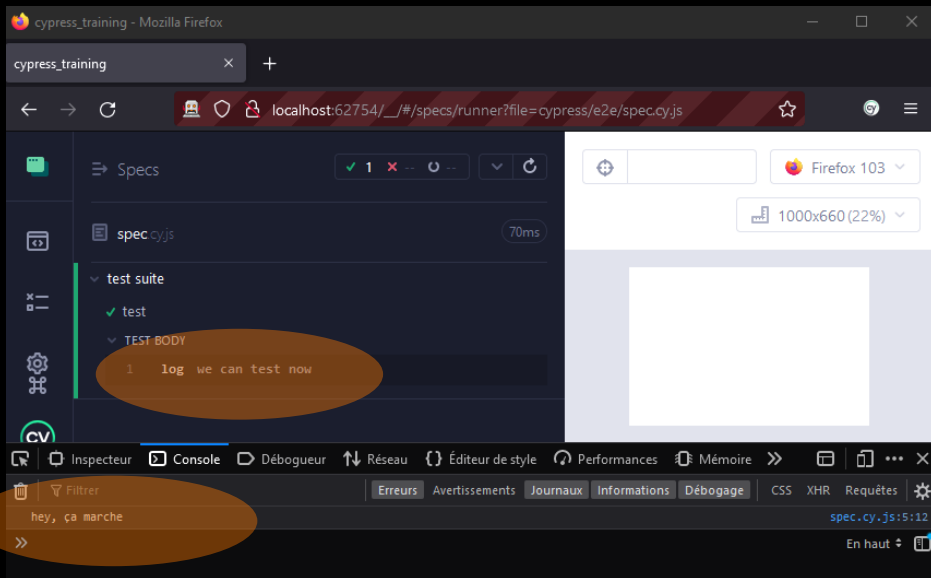
```
λ cat cypress/e2e/spec.cy.js
describe('test suite', () =>
{
  it('test', () =>
  {
    console.log('hey, its working');
    cy.log('we can test now');
  });
});
```

```
λ cat tests/example.spec.js
const {test, expect} = require('@playwright/test');
test.describe('test suite', () =>
{
  test('test', async ({ page }) =>
  {
    await page.evaluate(()=> console.log('hello'));
    await page.waitForTimeout(20000);
  });
});
```

log



exercise 5



only seen in Playwright
debug mode



load an url

load a url (web site or api route) with the browser



```
cy.visit('https://site.com');  
  or  
cy.visit('/');
```



```
await page.goto('https://site.com');  
  or  
await page.goto('/');
```



use default «baseUrl» defined in config

exercise 6



exercise 6

goal : load an url

load an url



```
λ cat cypress/e2e/spec.cy.js
describe('test suite', () =>
{
  it('test', () =>
  {
    cy.visit('/');
  });
});
```



exercise 6

```
λ cat tests/example.spec.js
const { test, expect } =
require('@playwright/test');
test.describe('test suite', () =>
{
  test('test', async ({page}) =>
  {
    await page.goto('/cookieclicker');
  });
});
```

load an url



exercise 6

cypress_training - Mozilla Firefox

Specs

spec.cy.js

test suite

test

TEST BODY

```
1 visit /  
(xhr) GET 200  
https://pagead2.googlesyndication.com/getconfig  
/sodar?sv=200&tid=gda&tv=r20220804&st=env  
(fetch) GET 200  
https://pagead2.googlesyndication.com  
/pcs/activeview?xai=AKA0jsti8JDaVzNd_ufr4m3m6vU  
aZohq4U2HMcBtA8yn8FEES7vseN6R04Ns15xAMNUNUIByW  
RqwcBTmanEn4u6r0QT5fCwYn6H5jXD5OGN7gdzAcxFLnMhu  
VjrxjyWSpG2BoZg&sa=AMFL-YTMs cmz9hvh-sB2bIxf-  
oE6NZvXdlzidbzcyUIqn7wA3VrPB-  
2mairbLH4LAVefb9XNay5cRBeQa3AleaSd6-  
oCBhyg8IPG2xu1Nyu1cQbRPL_sp5Fjmw0&  
sig=Cg0ArKJ5zHFYq1NV-M_oEAE8  
cid=CAAS3ORornyFLX14yFLS9rYnPoTX9HK8xmQPXCTEV5j  
d1EL-LOGfvA&id=Lidar2&mcvt=1004&p=0,0,250,300&  
mtos=1004.1004.1004.1004.1004&nc=1004.0.0.0.0
```

PLAYWRIGHT

Filter (e.g. text, @tag)

Status: all Projects: firefox

1/1 passed (100%)

example.spec.js

test

Before Hooks

page.goto / cookieclicker

After Hooks

fixture: page

fixture: context

Loading...

This is taking longer than expected.
Slow connections! If not, please make sure your playwright is enabled, then refresh.
If problems persist, this might be on our side - wait a few minutes, then hit ctrl-f5!

note the difference
Cypress ends the test when
everything is loaded
Playwright ends the test when
the « loaded » event is fired



get, click and auto retry

get a DOM element using a selector
retrying (no wait needed)



```
cy.get(selector);
```

```
await page.locator(selector);
```



click on an actionable element
retrying (no wait needed)



```
cy.get(selector).click();
```

```
await page.locator(selector).click(
```



exercise 7



goal : select a DOM element and click on it

get, click and auto retry



exercise 7

```
λ cat cypress.env.json
{
  "lang": "#langSelect-EN",
  "care": ".cc_btn_accept_all"
}
```

use specific test ids whenever it is possible
and encourage developers to add them or add them yourself

```
λ cat cypress/e2e/spec.cy.js
describe('test suite', () =>
{
  ...
  cy.get(Cypress.env('lang')).click();
  cy.get(Cypress.env('care')).click();
  ...
});
```

```
λ cat .env
lang2="#langSelect-EN"
care=".cc_btn_accept_all"
```

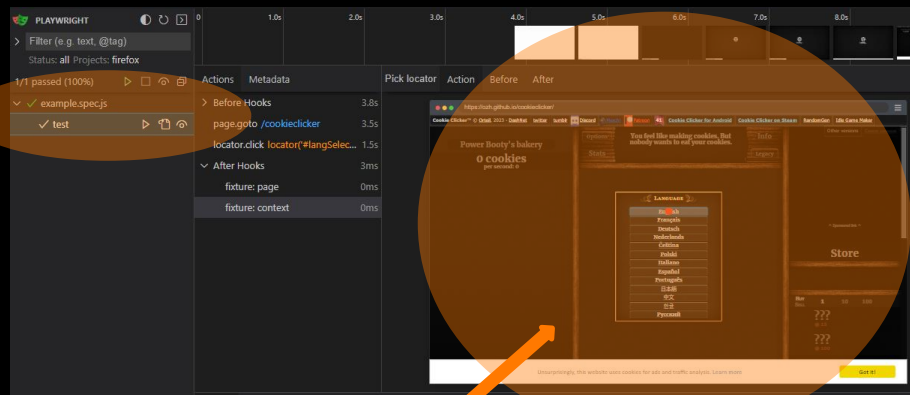
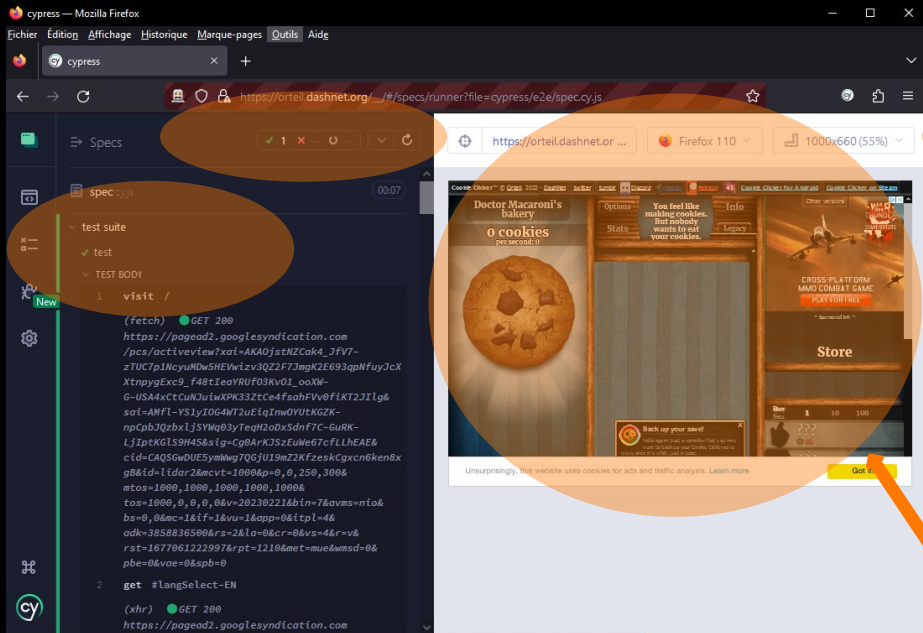
CAUTION : dotenv does not override
existing varenv lang is existing
so use another name

```
λ cat tests/example.spec.js
test.describe('test suite', () =>
{
  ...
  await page.locator(process.env.lang2).click();
  await page.locator(process.env.care).click();
  ...
});
```

get, click and auto retry



exercise 7



again note the difference
Cypress ends the test when
everything is loaded
Playwright ends the test
immediately



asserting

asserting is a major feature of an automated testing tool



Cypress is providing the major standard assertion libs out of the box :

- chai (BDD and TDD)

- chai-jquery

- sinon-chai

- common assertions (should)



Playwright is providing only 1 build-in assertion system. Any standard other lib must be downloaded and specifically imported.

exercice 8

asserting



exercise 8

```
λ cat cypress.env.json
{
  ...
  "nbcookies": "#cookies"
}

λ cat cypress/e2e/spec.cy.js
describe('test suite', () =>
{
  ...
  cy.get(Cypress.env('nbcookies'))
    .contains('0 cookie');
  ...
});
```

```
λ cat .env
...
nbcookies="#cookies"

λ cat tests/example.spec.js
test.describe('test suite', () =>
{
  ...
  await
  expect(page.locator(process.env.nbcookies))
    .toContainText('0 cookie');
  ...
});
```

Re

asserting



Cypress — Mozilla Firefox

https://ortel.dashnet.org/_/#/specs/runner?file=cypress/e2e/spec.cy.js

Specs

spec.cy.js

```
(xhr) GET 200 /patreon  
/grab.php?nocache=1677063780525  
get #cookies  
-contains 0 cookies  
(fetch) GET 200  
https://pagead2.googlesyndication.com  
/pagead/js/pagead_async.js  
YTLR0m0u7a1L0u0p0B0L0c3W1unN0G1z0cFKAx-  
bnpY0uhez_0P0Kd0kV19e2zXj040cV50Bz4N_2D179v0C  
0pW7x8BkQ0LcJ223W0cJF0U1ChjFq7RvSXXR-8  
s1g=Cg8ArK3S2b13Mj=60JFACAE&  
cid=CAQSPADUESym8Zchh9270v40owlfAv0LGLsEajqLL  
nq8SpowfiCHn70T0UakjK1f-  
gYokh9YtqAtiq3Kyxq8&id=ldar2mcvt=1007&  
p=0,0,250,300&mtos=1007,1007,1007,1007,1007&  
tos=1007,0,0,0,0&v=20230221&bin=7&avms=nto&  
bs=0,0&mc=1&f=1&vu=1&app=0&itpl=22&  
adk=3858836500&rs=2&lo=0&cr=0&vs=4&r=v&  
rst=1677063778215&rpt=2141&mt=mue&msd=0&  
pbe=0&vae=0&spb=0
```

Bearded Parrot's bakery

0 cookies

You feel like making cookies, but nobody wants to eat your cookies.

Store

Back up your name!

46

Orange Restricted



exercise 8

Cypress — Mozilla Firefox

https://ortel.dashnet.org/_/#/specs/runner?file=cypress/e2e/spec.cy.js

Specs

spec.cy.js

```
(xhr) GET 200 /patreon  
/grab.php?nocache=1677063780525  
get #cookies  
-contains 0 cookies  
(fetch) GET 200  
https://pagead2.googlesyndication.com  
/pagead/js/pagead_async.js  
YTLR0m0u7a1L0u0p0B0L0c3W1unN0G1z0cFKAx-  
bnpY0uhez_0P0Kd0kV19e2zXj040cV50Bz4N_2D179v0C  
0pW7x8BkQ0LcJ223W0cJF0U1ChjFq7RvSXXR-8  
s1g=Cg8ArK3S2b13Mj=60JFACAE&  
cid=CAQSPADUESym8Zchh9270v40owlfAv0LGLsEajqLL  
nq8SpowfiCHn70T0UakjK1f-  
gYokh9YtqAtiq3Kyxq8&id=ldar2mcvt=1007&  
p=0,0,250,300&mtos=1007,1007,1007,1007,1007&  
tos=1007,0,0,0,0&v=20230221&bin=7&avms=nto&  
bs=0,0&mc=1&f=1&vu=1&app=0&itpl=22&  
adk=3858836500&rs=2&lo=0&cr=0&vs=4&r=v&  
rst=1677063778215&rpt=2141&mt=mue&msd=0&  
pbe=0&vae=0&spb=0
```

Delicious Ninja's bakery

0 cookies

You feel like making cookies, but nobody wants to eat your cookies.

Store

Back up your name!



test organization and context management

Cypress and Playwright are clearing the context by default

- the DOM

- the cookies

- the local storage

between each test !

test organization and context management



Cypress proposes 2 mechanisms for context management session

tests are isolate by default, context is saved using a `cy.session` call and restored using another `cy.session` call
restore is taking a bit of time and the entire test must be in an `it` scope

test isolation



preferred
solution

the user controls if Cypress clears the context or not for every test suite aka `describe` scope
test content can be splitted over several `it` calls
it runs faster

test organization and context management



Playwright proposes to use the `beforeAll` and `afterAll` functions to open and close the page.

Don't forgot to run the test suite in "serial mode" to avoid Playwright running the tests in parallel.

test organization and context management



```
describe('clear env', { testIsolation:true }, () =>
{
  it('clear', () => { cy.log('context cleared'); });
});
```

← clears context

```
describe('testing Cookie Clicker', { testIsolation:false }, () =>
{
  it('test1', () => { // do something });
  it('test2', () => { // do something else in the same context });
});
```

← keeps context

test organization and context management



```
test.describe('test suite', () =>
{
  test.describe.configure({ mode: 'serial' });
  let page;
  test.beforeAll(async ({ browser }) => { page = await browser.newPage(); });
  test.afterAll(async () => { await page.close(); });

  test('test1', async () => { // do something with page });
  test('test2', async () => { // do something else in the same context });

  test('debug only', async () => { await page.waitForTimeout(5000); });
});
```

use only for debug to avoid Playwright ending the test immediately and thus preventing to see the proper screenshot at the end

exercise 9



exercise 9

goal : organize tests and control context

test organization and context management



exercise 9

```
λ cat cypress.env.json
{
  ...
  "bigcookie": "#bigCookie"
}
```

```
λ cat .env
{
  ...
  bigcookie="#bigCookie"
}
```

test organization and context management



exercise 9

```
describe('clear env', { testIsolation:true }, () => {
  it('clear', () => { cy.log('env cleared'); }); });

describe('test suite', { testIsolation:false }, () => {
  it('init', () => {
    cy.visit('/');
    cy.get(Cypress.env('lang')).click();
    cy.get(Cypress.env('care')).click();
    cy.get(Cypress.env('nbcookies')).contains('0 cookie'); });
  it('bake', { scrollBehavior: false }, () => {
    cy.get(Cypress.env('bigcookie')).click();
    cy.get(Cypress.env('nbcookies')).contains('1 cookie'); });
});
```

test organization and context management



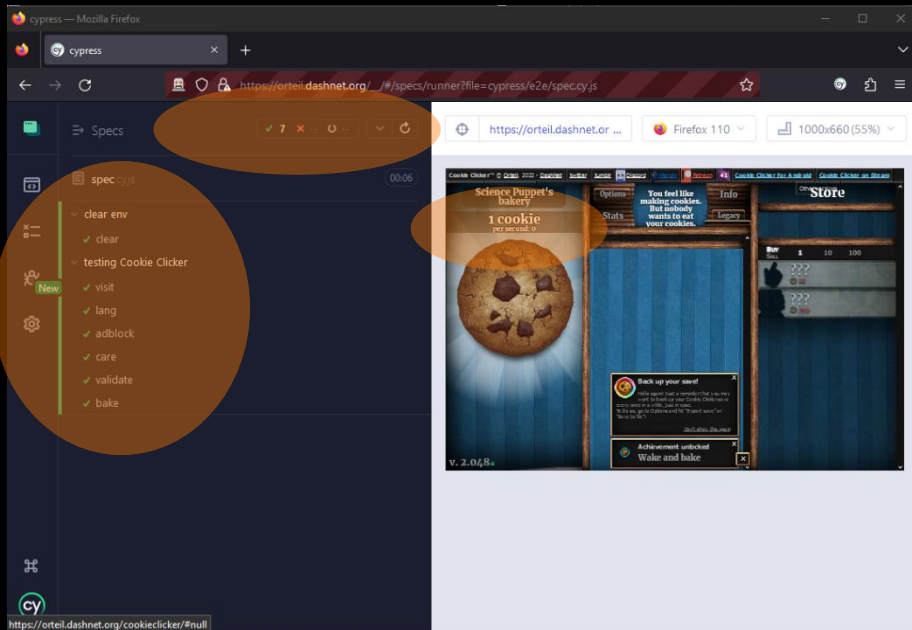
exercise 9

```
test.describe('test suite', () => {
  test.describe.configure({ mode: 'serial' });
  let page;
  test.beforeAll(async ({ browser }) => { page = await browser.newPage(); });
  test.afterAll(async () => { await page.close(); });
  test('init', async () => {
    await page.goto('/cookieclicker');
    await page.locator(process.env.lang2).click();
    await page.locator(process.env.care).click();
    await expect(page.locator(process.env.nbcookies)).toContainText('0 cookie'); });
  test('bake', async () => {
    await page.locator(process.env.bigcookie).click();
    await expect(page.locator(process.env.nbcookies)).toContainText('1 cookie'); });
});
```

test organization and context management



exercise 9



keyboard interaction

whether it is for filling a form or any other keyboard usage



```
cy.get(selector).type('{ctrl}s');
```



```
await  
page.keyboard.press('Control+s');
```

press and type
are different in
Playwright !

exercise 10



goal : simulate keyboard action

keyboard interaction

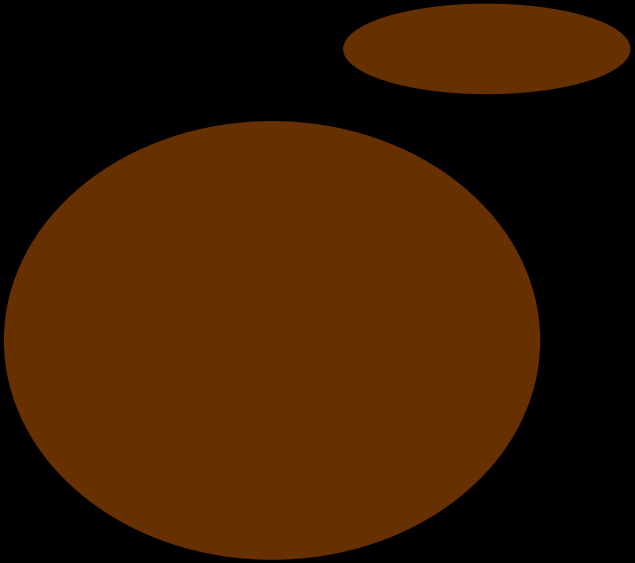


exercise 10

```
λ cat cypress/e2e/spec.cy.js
describe('test suite', () =>
{
  ...
  it('save', () =>
  {
    cy.get('body').type('{ctrl}s');
  });
});
```

```
λ cat tests/example.spec.js
test.describe('test suite', () =>
{
  ...
  test('save', async () =>
  {
    await
page.keyboard.press('Control+s');
  });
});
```

Cypress : keyboard interaction



Cypress : reload

reload the current page



```
cy.reload();
```

```
cy.reload(true);  
// clears the cache
```



```
await page.reload();
```

exercise 11



exercise 11

goal : reload the current page

reload



```
λ cat cypress/e2e/spec.cy.js
describe('test suite', () =>
{
  ...
  it('reload', () =>
  {
    cy.reload();
  });
  ...
});
```



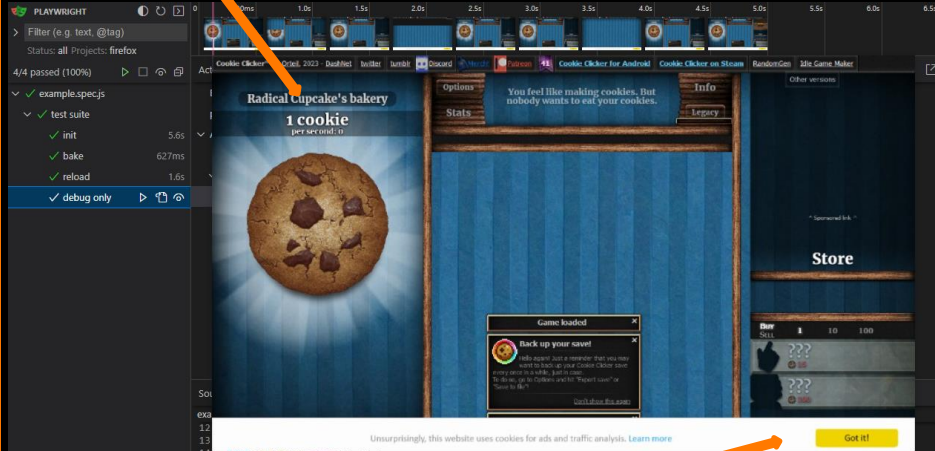
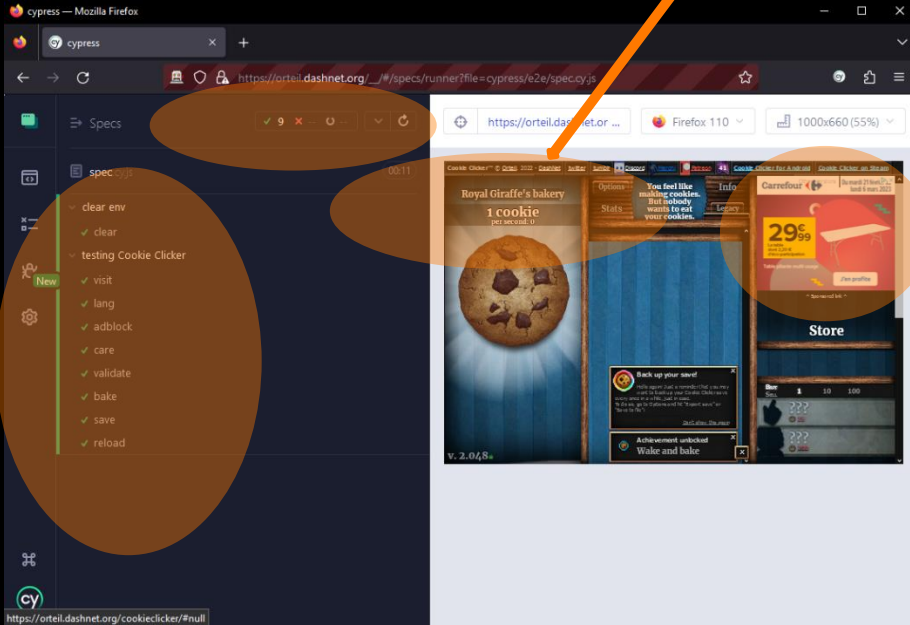
exercise 11

```
λ cat tests/example.spec.js
test.describe('test suite', () =>
{
  ...
  test('reload', async () =>
  {
    await page.reload();
  });
  ...
});
```

reload



note that the éléments linked to the context (cookie, local storage) are recovered



problem in Playwright, cookies are not restored ?



scrolling

to simulate scrolling :



```
cy.scrollTo(position|x,y);
```

```
cy.scrollToView();
```



```
await page.evaluate(() =>  
  document  
    .querySelector(locator)  
    .scrollBy(0, 1000)  
);
```

nothing specific in Playwright,
fallback on standard js, a bit awkward ...

exercise 12



exercise 12

goal : scroll the page

scrolling



exercise 12

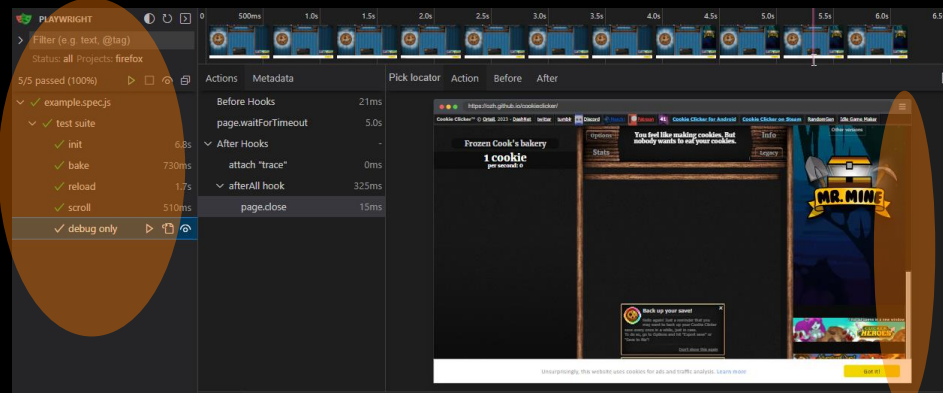
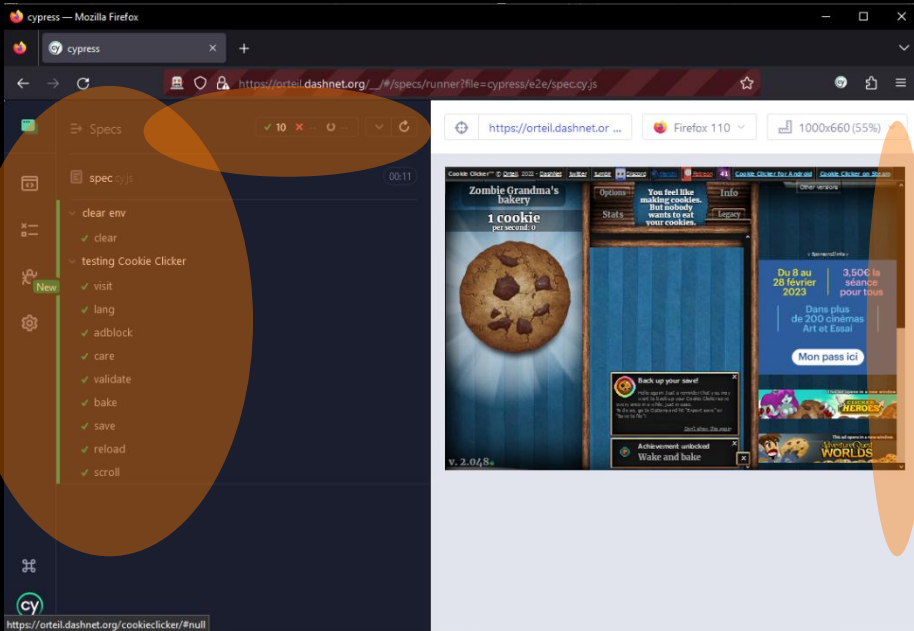
```
λ cat cypress.env.json
{
  ...
  "sectionright": "#sectionRight"
}
λ cat cypress/e2e/spec.cy.js
describe('test suite', () => {
  ...
  it('scroll', () => {
    cy.get(Cypress.env('sectionright'))
      .scrollTo('bottom');
  });
});
```

```
λ cat .env
...
  sectionright="#sectionRight"
λ cat tests/example.spec.js
describe('test suite', () => {
  ...
  await page.evaluate((selector) =>
    document.querySelector(selector).scrollBy(0,
    1000), process.env.sectionright)
  });
});
```

Cypress : scrolling



exercise 12



mobile web

to simulate a mobile web device :



set the viewport size

```
cy.viewport(screen);
```

set the user agent

```
cy.intercept('/', (req) =>  
  req.headers['user-agent'] = useragent);
```



set the viewport size

```
test.use({ viewport:  
  { width: 123, height: 456 } });
```

set the user agent

```
test.use({ userAgent: useragent });
```

exercise 13



exercise 13

goal : simulate a mobile browser

mobile web



exercise 13

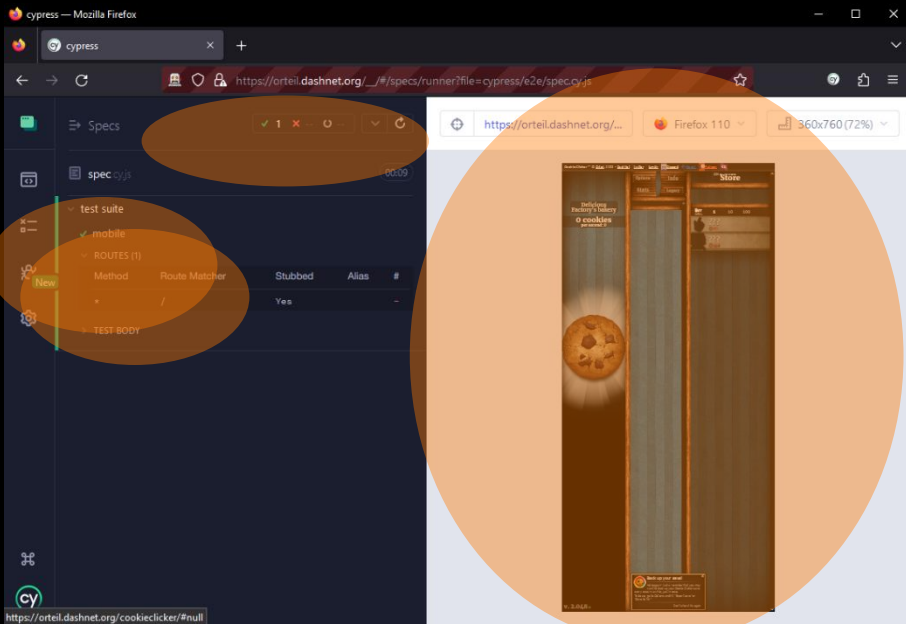
```
λ cat cypress/e2e/spec.cy.js
const config = {screen:'samsung-s10',
                 useragent:"Mozilla/5.0 (Android 9;
Mobile; rv:97.0) Gecko/20100101 Firefox/97.0"};

describe('test suite', () => {
  it('mobile', () => {
    cy.viewport(config.screen);
    cy.intercept('/', (req) => req.headers['user-agent'] = config.useragent);
    cy.visit('/');
    cy.get(Cypress.env('lang')).click();
    cy.get(Cypress.env('care')).click();
    cy.get(Cypress.env('nbcookies')).contains('0 cookie');
  }); });
```

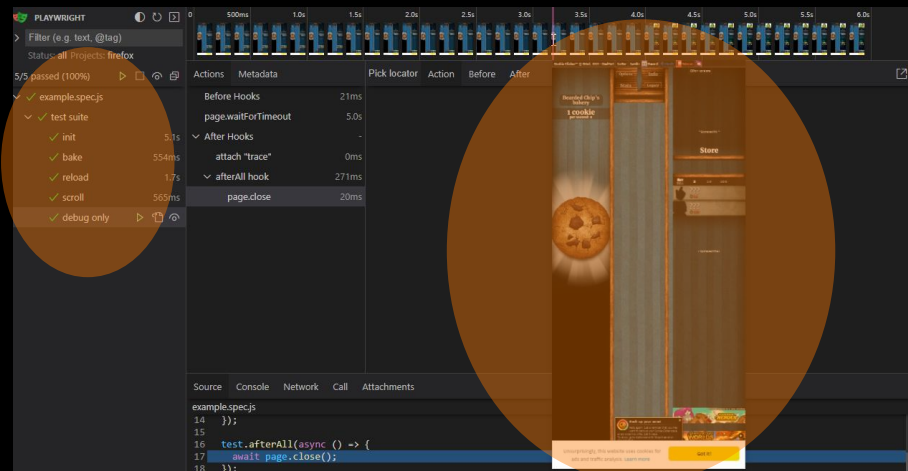
```
λ cat tests/example.spec.js
const config = {screen:{width:360, height:760},
                 useragent:"Mozilla/5.0 (Android 9;
Mobile; rv:97.0) Gecko/20100101 Firefox/97.0"};

test.describe('test suite', () => {
  test.use({ userAgent: config.useragent });
  test.use({ viewport: config.screen });
  test('test', async ({page}) => {
    await page.goto('/cookieclicker');
    await page.locator(process.env.lang2).click();
    await page.locator(process.env.care).click();
    await expect(page.locator(process.env.nbcookies))
      .toContainText('0 cookie');
    await page.waitForTimeout(20000);
  }); });
```

mobile web



exercise 13



wrap-up

to wrap-up, I propose a exercise to
simulate really playing Cookie Clicker 😊

exercise 14



exercise 14

goal : play Cookie Clicker

play Cookie Clicker



- 1- start with code for the exercise 9
- 2- click on the big cookie until you have enough cookies to buy a cursor, then buy the cursor
- 3- click on the big cookie until you have enough cookies to buy a grandma or an upgrade, then buy the latest
- 4- carry on until you have plenty of cookies 😊

Thanks