

Jest

- Alejandro León Pereira
- Andrés Fernández González
- María González Gancedo
- Iñaki Salgado Uralde
- Pablo González Argüelles


SemEs1-05

¿Qué es Jest?

- ▶ Framework de JavaScript que permiten realizar pruebas automatizadas
- ▶ Enfoque simple
- ▶ Creado por los ingenieros de Facebook
- ▶ Viene integrado en React
- ▶ Funciona en proyectos con:

TypeScript

 React

 Vue.js

 BABEL

 node JS

 ANGULAR

Factores importantes:

- ▶ Sin configuración
- ▶ Gran API : Bien documentado, bien mantenido.
- ▶ Aislamiento: pruebas ejecutadas en sus propios procesos
- ▶ Instantáneas / Snapshots: rastrea objetos grandes con facilidad

Características:



Rápido y seguro



Cobertura de código



Mock Functions



Grandes excepciones



Charlas y documentos



Open Collective

¿Quién usa Jest?



Ejemplo

1

```
# yarn add --dev jest  
# npm install --save-dev jest
```

3

sum.test.js

```
const sum = require('./sum');  
  
test('adds 1 + 2 to equal 3', () => {  
  expect(sum(1, 2)).toBe(3);  
});
```

5

```
# yarn test  
# npm run test
```

2

sum.js

```
function sum(a, b) {  
  return a + b;  
}  
module.exports = sum;
```

4

package.json

```
{  
  "scripts": {  
    "test": "jest"  
  }  
}
```

6

```
PASS ./sum.test.js  
✓ adds 1 + 2 to equal 3 (5ms)
```

► Ingenieros de Facebook

► Desarrolladores

Stakeholders



► El equipo Fundación
JavaScript de Facebook

► Clientes

Principales atributos de calidad y restricciones

Atributos de Calidad

- ▶ Comportamiento temporal

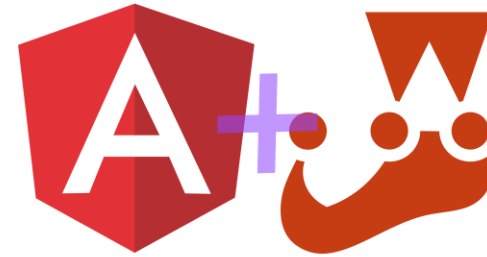
Velocidad 6x

- ▶ Capacidad para ser instalado

Simple dependencia

- ▶ Adaptabilidad

Cualquier framework o biblioteca JS



Restricciones

- ▶ Poca flexibilidad en proyectos muy grandes
- ▶ Difícil personalización
- ▶ No está optimizado para pruebas end2end
- ▶ Falta de madurez

Aspectos de desarrollo

- ▶ ¿Qué quiere decir esto?
- ▶ Módulos y arquitectura
- ▶ Issues

Aspectos de desarrollo

- ▶ ¿Cómo ayudar con el desarrollo?
- ▶ Código de conducta para desarrollar
- ▶ Good first issues

Aspectos de desarrollo

[Code](#) [Issues 829](#) [Pull requests 113](#) [Actions](#) [Projects 0](#) [Wiki](#) [Security](#) [Insights](#)

Want to contribute to facebook/jest? [Dismiss](#)

If you have a bug or an idea, read the [contributing guidelines](#) before opening an issue.
If you're ready to tackle some open issues, [we've collected some good first issues for you](#).

[Filters](#)

is:open label:"good first issue"

[Labels 39](#)

[Milestones 1](#)

New issue

[X](#) Clear current search query, filters, and sorts

🔔 27 Open ✓ 181 Closed

Author ▾ Label ▾ Projects ▾ Milestones ▾ Assignee ▾ Sort ▾

🔔

Include ts-jest mock util functions in jest core

Feature Request

Help Wanted

TypeScript

good first issue

4

#9534 opened on 7 Feb by ahnpnl

🔔

Async testing fails with async/await

Feature Request

Help Wanted

good first issue

7

#9240 opened on 28 Nov 2019 by yanickrochon

🔔

Create a Dockerfile for contributing

Documentation

Help Wanted

good first issue

1

2

#9105 opened on 26 Oct 2019 by Alex-Cannon

🔔

Running CLI command with "-U" results in a message containing `Did you mean "\$0`

good first issue

8

#9007 opened on 4 Oct 2019 by Tokimon

Módulos y Componentes

- ▶ Módulos aislados
- ▶ Mocking
- ▶ Objeto Jest

Módulos aislados

- ▶ Jest tiene la capacidad de simular los módulos que desarrollamos en nuestra aplicación
- ▶ Esta técnica se denomina **Mocking**
- ▶ El objeto Jest está al alcance de todos los archivos de prueba

Objeto Jest

► disableAutomock()

```
// __tests__/disableAutomocking.js
import utils from '../utils';

jest.disableAutomock();

test('original implementation', () => {
  // now we have the original implementation,
  // even if we set the automocking in a jest configuration
  expect(utils.authorize()).toBe('token');
});
```

► enableAutomock()

```
// __tests__/disableAutomocking.js
jest.enableAutomock();

import utils from '../utils';

test('original implementation', () => {
  // now we have the mocked implementation,
  expect(utils.authorize._isMockFunction).toBeTruthy();
  expect(utils.isAuthorized._isMockFunction).toBeTruthy();
});
```


Objeto Jest

► genMockFromModule(modulename)

```
// __tests__/genMockFromModule.test.js
const utils = jest.genMockFromModule('../utils').default;
utils.isAuthorized = jest.fn(secret => secret === 'not wizard');

test('implementation created by jest.genMockFromModule', () => {
  expect(utils.authorize.mock).toBeTruthy();
  expect(utils.isAuthorized('not wizard')).toEqual(true);
});
```

► mock(moduleName,factory,options)

```
jest.mock('../moduleName', () => {
  return jest.fn(() => 42);
});

// This runs the function specified as second argument
const moduleName = require('../moduleName');
moduleName(); // Will return '42';
```

Objeto Jest

- ▶ `unmock(modulename)`
- ▶ `doMock(moduleName,factory,options)`
- ▶ `dontMock(modulename)`

```
test('moduleName 1', () => {  
  jest.doMock('../moduleName', () => {  
    return {  
      __esModule: true,  
      default: 'default1',  
      foo: 'foo1',  
    };  
  });  
  return import('../moduleName').then(moduleName => {  
    expect(moduleName.default).toEqual('default1');  
    expect(moduleName.foo).toEqual('foo1');  
  });  
});
```

Objeto Jest

► setMock(modulename,moduleExports)

► requireActual(moduleName)

► requireMock(modulename)

```
jest.mock('../myModule', () => {  
  // Require the original module to not be mocked...  
  const originalModule = jest.requireActual(moduleName);  
  
  return {  
    __esModule: true, // Use it when dealing with esModules  
    ...originalModule,  
    getRandom: jest.fn().mockReturnValue(10),  
  };  
});
```

Objeto Jest

- ▶ resetModules()

```
const sum1 = require('../sum');  
jest.resetModules();  
const sum2 = require('../sum');  
sum1 === sum2;
```

- ▶ isolateModules(fn)

```
let myModule;  
jest.isolateModules(() => {  
  myModule = require('myModule');  
});  
  
const otherCopyOfMyModule = require('myModule');
```

Arquitectura que hay detrás de Jest

https://jestjs.io/docs/en/architecture#__docusaurus

En la página oficial de Jest se puede encontrar un vídeo detallando su arquitectura, cómo funciona Jest por debajo.

The screenshot shows the Jest 25.1 documentation page. The browser address bar displays `https://jestjs.io/docs/en/architecture#__docusaurus`. The page header includes the Jest logo, version 25.1, and navigation links for Docs, API, Help, Blog, and English. A search bar and a GitHub link are also present. The left sidebar contains a table of contents with sections like Introduction, Guides, and More Resources. The main content area is titled "Architecture" and includes a video player. The video is titled "Jest Architecture" and shows a man writing on a whiteboard. The video player has a progress bar at 15:34 / 51:00 and a subtitle that reads "builds up today's context and then if you've stopped information". Below the video player are two buttons: "TROUBLESHOOTING" and "TESTING REACT APPS".

JEST 25.1

Docs API Help Blog English Search GitHub

Architecture

If you are interested in learning more about how Jest works, what the architecture behind the framework is, and how Jest is split up into individual reusable packages, check out this video:

Jest Architecture

Ver más tarde Compartir

MÁS VÍDEOS builds up today's context and then if you've stopped information

15:34 / 51:00 YouTube

← TROUBLESHOOTING TESTING REACT APPS →

jest

my-test.js

config + argv

normalize

jest-config

Global Config
Project Config
1...N

jest-cli
jest

2.

3.

jest-haste-map

HasteContext
HasteFS

watchman

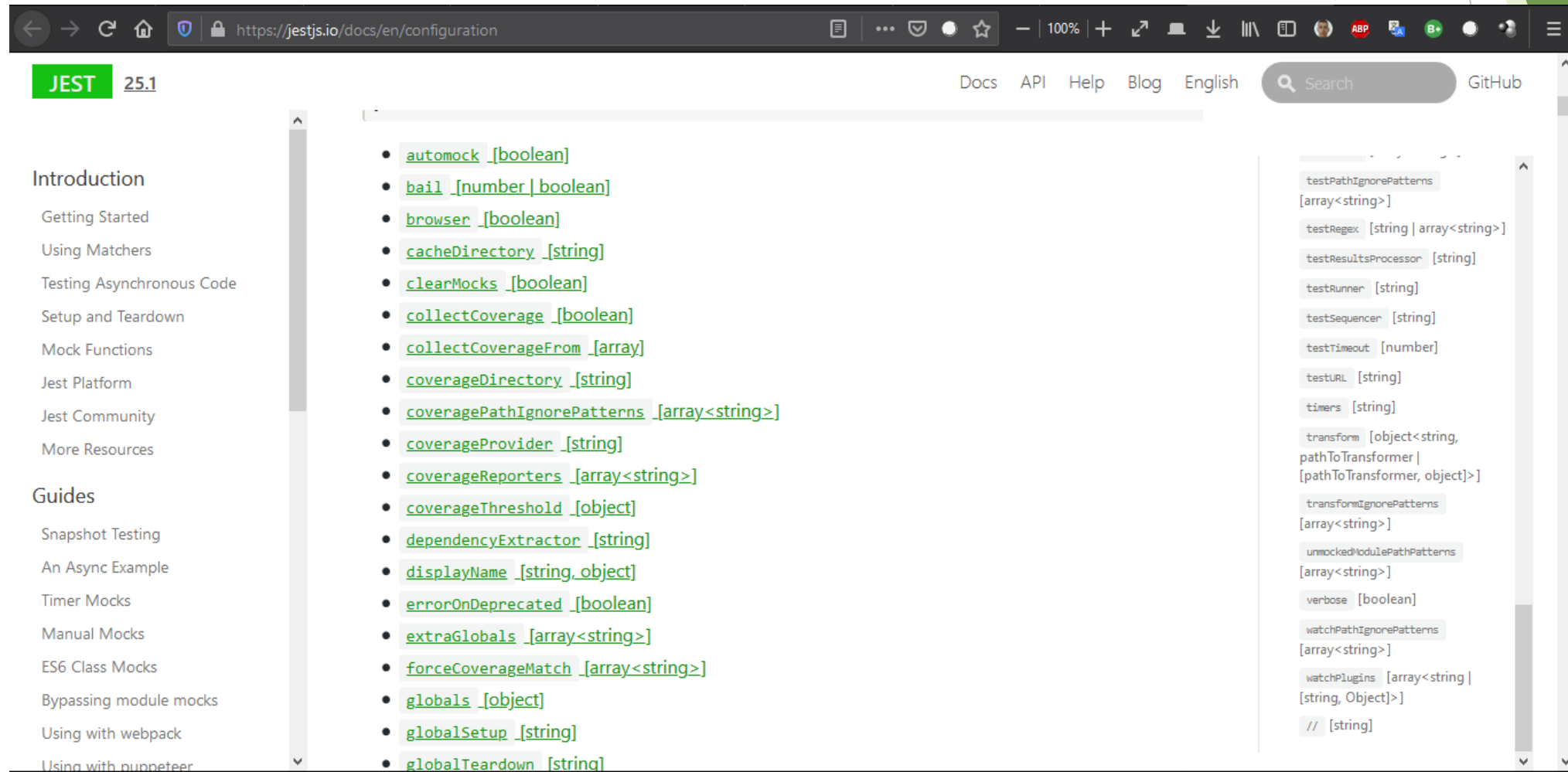
node.js

jest-worker

4.

Las diferentes opciones de configuración de Jest (Jest CLI options) pueden encontrarse en:

► <https://jestjs.io/docs/en/configuration>



The screenshot shows the Jest documentation website at <https://jestjs.io/docs/en/configuration>. The page title is "JEST 25.1". The navigation bar includes links for Docs, API, Help, Blog, English, a search bar, and a GitHub link. The left sidebar contains a table of contents with sections like Introduction, Guides, and various Jest features. The main content area lists 25 configuration options, each with its name and type in brackets. The right sidebar shows a partial view of the same list.

JEST 25.1

Docs API Help Blog English Search GitHub

Introduction

- Getting Started
- Using Matchers
- Testing Asynchronous Code
- Setup and Teardown
- Mock Functions
- Jest Platform
- Jest Community
- More Resources

Guides

- Snapshot Testing
- An Async Example
- Timer Mocks
- Manual Mocks
- ES6 Class Mocks
- Bypassing module mocks
- Using with webpack
- Using with puppeteer

Configuration Options:

- `automock` [boolean]
- `bail` [number | boolean]
- `browser` [boolean]
- `cacheDirectory` [string]
- `clearMocks` [boolean]
- `collectCoverage` [boolean]
- `collectCoverageFrom` [array]
- `coverageDirectory` [string]
- `coveragePathIgnorePatterns` [array<string>]
- `coverageProvider` [string]
- `coverageReporters` [array<string>]
- `coverageThreshold` [object]
- `dependencyExtractor` [string]
- `displayName` [string, object]
- `errorOnDeprecated` [boolean]
- `extraGlobals` [array<string>]
- `forceCoverageMatch` [array<string>]
- `globals` [object]
- `globalSetup` [string]
- `globalTeardown` [string]
- `testPathIgnorePatterns` [array<string>]
- `testRegex` [string | array<string>]
- `testResultsProcessor` [string]
- `testRunner` [string]
- `testSequencer` [string]
- `testTimeout` [number]
- `testURL` [string]
- `timers` [string]
- `transform` [object<string, pathToTransformer | [pathToTransformer, object]>]
- `transformIgnorePatterns` [array<string>]
- `unmockedModulePathPatterns` [array<string>]
- `verbose` [boolean]
- `watchPathIgnorePatterns` [array<string>]
- `watchPlugins` [array<string | [string, Object]>]
- `//` [string]

Y pueden utilizarse añadiéndose directamente al package.json o a un archivo jest.config

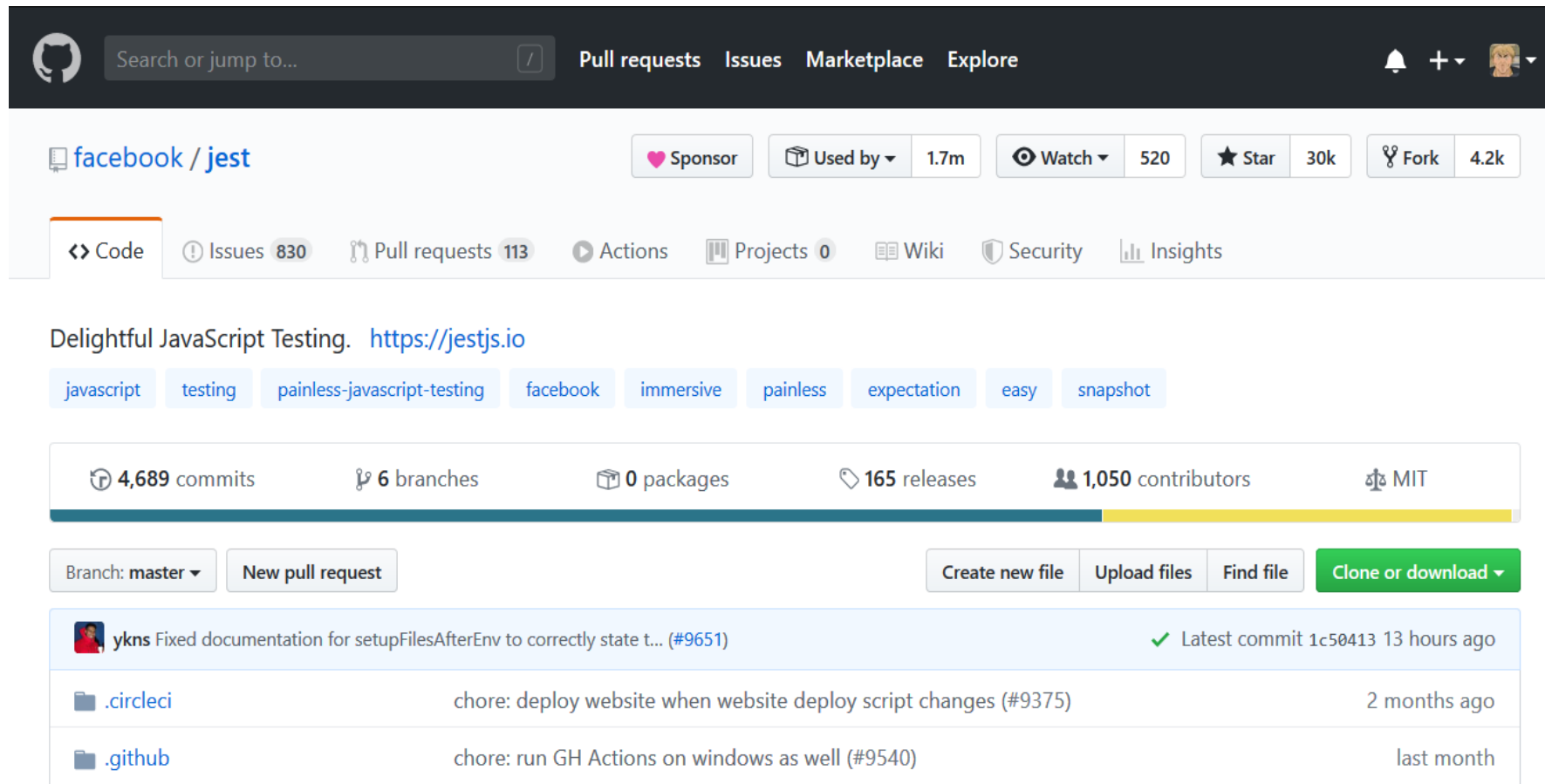
```
{  
  "name": "my-project",  
  "jest": {  
    "verbose": true  
  }  
}
```

Por ejemplo, verbose se utiliza para mostrar información por consola del estado actual de los test

Detrás de Jest, está Facebook

<https://opensource.facebook.com/>

<https://github.com/facebook/jest>



The screenshot shows the GitHub repository page for `facebook/jest`. The repository is described as "Delightful JavaScript Testing" with a link to <https://jestjs.io>. It features various tags such as `javascript`, `testing`, `painless-javascript-testing`, `facebook`, `immersive`, `painless`, `expectation`, `easy`, and `snapshot`. The repository statistics include 4,689 commits, 6 branches, 0 packages, 165 releases, 1,050 contributors, and the MIT license. The page also shows a list of recent commits, including one by `ykns` titled "Fixed documentation for setupFilesAfterEnv to correctly state t..." (#9651) and two others by `.circleci` and `.github` related to website deployment and GH Actions.

Search or jump to... Pull requests Issues Marketplace Explore

facebook / jest Sponsor Used by 1.7m Watch 520 Star 30k Fork 4.2k

Code Issues 830 Pull requests 113 Actions Projects 0 Wiki Security Insights

Delightful JavaScript Testing. <https://jestjs.io>

javascript testing painless-javascript-testing facebook immersive painless expectation easy snapshot

4,689 commits 6 branches 0 packages 165 releases 1,050 contributors MIT

Branch: master New pull request Create new file Upload files Find file Clone or download

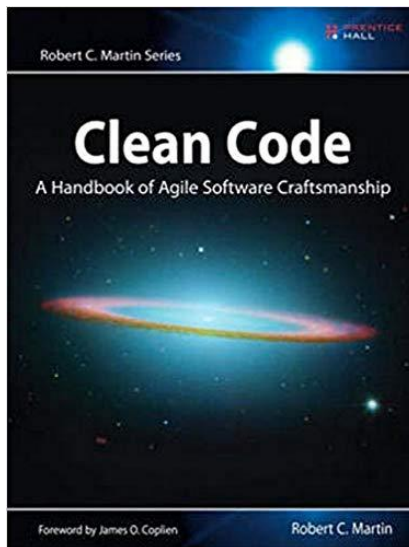
ykns Fixed documentation for setupFilesAfterEnv to correctly state t... (#9651) Latest commit 1c50413 13 hours ago

.circleci chore: deploy website when website deploy script changes (#9375) 2 months ago

.github chore: run GH Actions on windows as well (#9540) last month

Arquitectura limpia

- ▶ Robert C. Martin
 - ▶ Creador de los principios del software limpio
 - ▶ Obras destacadas:
 - ▶ Clean code: A Handbook of Agile Software Craftsmanship



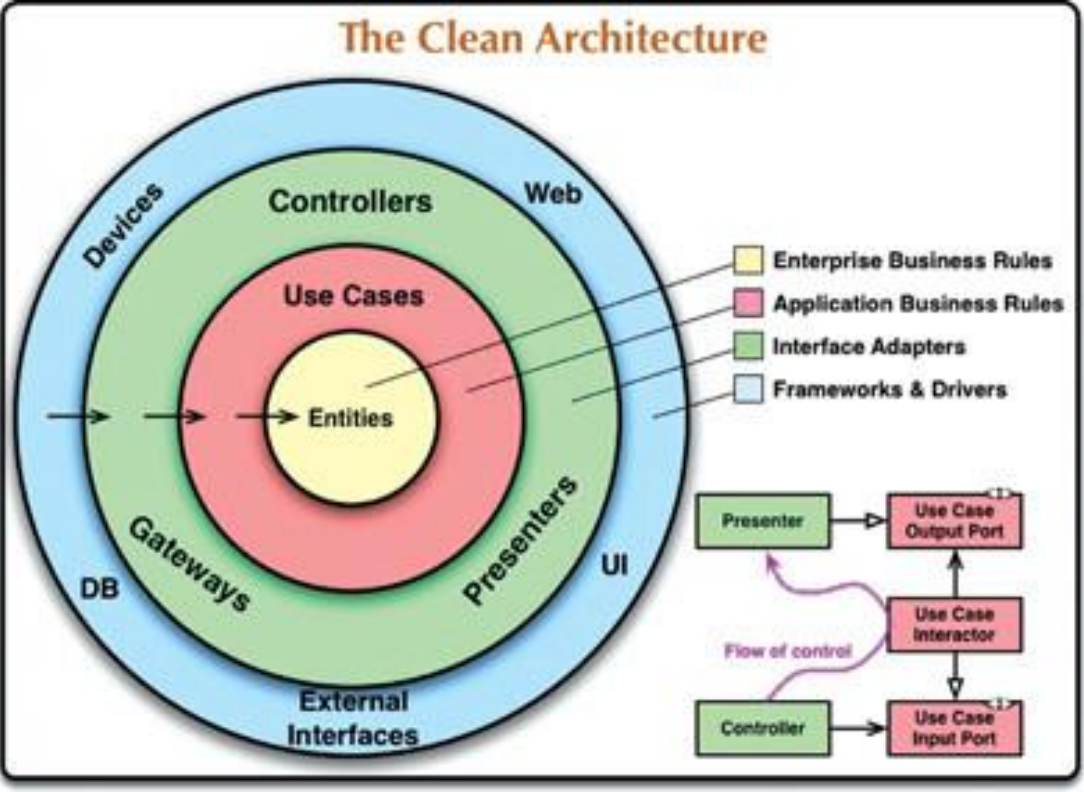
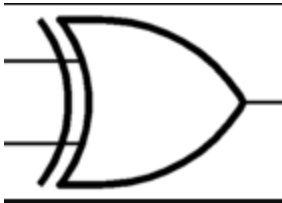
Características de una arquitectura limpia

Además de cumplir los principios anteriormente descritos, una arquitectura limpia se caracteriza por:

- **Independiente de los frameworks.** Los frameworks deberían ser herramientas, y no obligarnos a actuar de una determinada manera debido a sus restricciones.
- **Testable.** Debemos poder probar nuestras reglas de negocio sin pensar en base de datos, interface gráfica u otros componentes no esenciales de nuestro sistema.
- **Independiente de la UI.** Si la UI cambia a menudo esto no puede afectar al resto de nuestro sistema, que tiene que ser independiente.
- **Independiente de la base de datos.** Deberíamos poder cambiar de Oracle, a SQL Server, a MongoDB, a Casandra o a cualquier otra base de datos sin que afectara demasiado a nuestro sistema.
- **Independiente de cualquier entidad externa.** No deberíamos saber nada de entidades externas, por lo que no deberemos depender de ellas.



EXOR



Código limpio VS Jest

Código limpio

- ▶ Mejor mantenimiento de proyectos a largo plazo
- ▶ Reducción del coste de mantenimiento del proyecto, debido a que es más accesible, ya que no se necesita gente experta en ningún framework determinado.

Jest

- ▶ Los tests corren en paralelo
- ▶ La API está documentada
- ▶ Fácil configuración
- ▶ Control de la cobertura de código

Aún así, Jest tiene sus ventajas...

Jest VS no-framework testing

Jest (cobertura de código)

```
const sum = require('./sum');

test('adds 1 + 2 to equal 3', () => {
  expect(sum(1, 2)).toBe(3);
});
```

```
~/d/p/j/j/e/timer $ yarn jest --coverage
yarn run v1.12.3
$ /Users/ortatherox/dev/projects/jest/jest/examples/timer/node_modules/.bin/jest --coverage
PASS  __tests__/timer_game.test.js
PASS  __tests__/infinite_timer_game.test.js
```

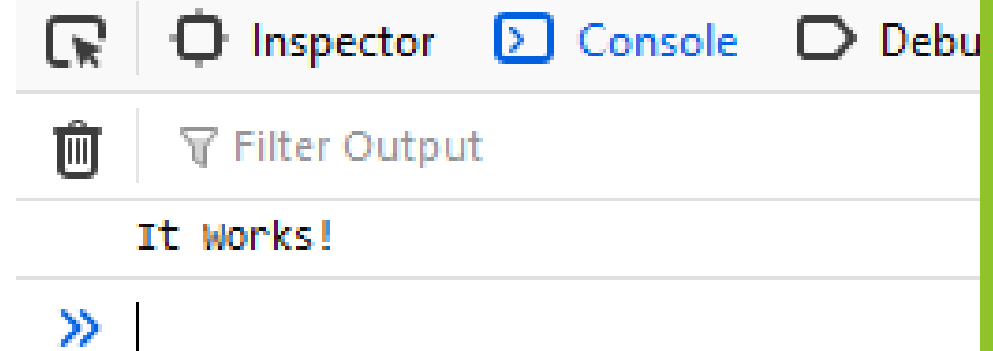
File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Line #s
All files	87.5	100	80	87.5	
infiniteTimerGame.js	80	100	66.67	80	12
timerGame.js	100	100	100	100	

```
Test Suites: 2 passed, 2 total
Tests:       4 passed, 4 total
Snapshots:   0 total
Time:        0.878s, estimated 1s
Ran all test suites.
```

Clean code testing (sencillez)

```
function testCalculate(){
  if(calculate(1,1)==2)
    console.log('It Works!');
  else
    console.log('Test failed');
}
```

```
testCalculate();
```



Jest makes testing delightful.

- Jest Core Team