
Globant
we are ready

UI Engineering Studio. Day 15

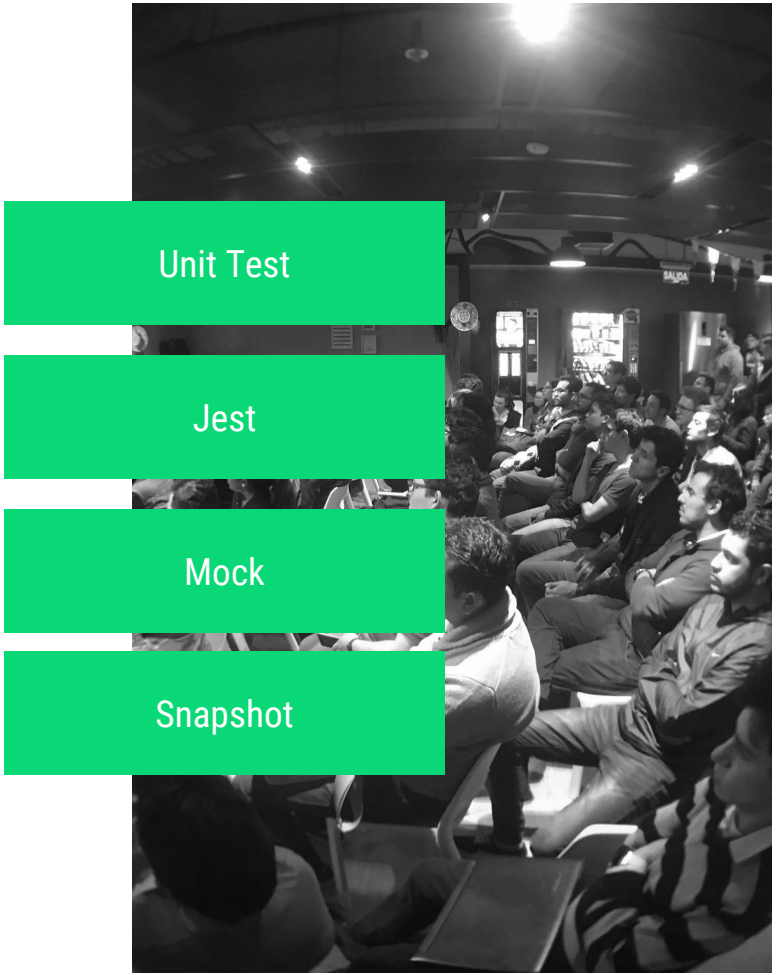
Bootcamp: Jest

Globant

What is a Unit Test?

Unit testing is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. **A unit is the smallest testable part of any software.**

source



UI Boot Camp: Jest

What is Jest?

Jest is a tool created and maintained by the open source community with the support of Facebook. Jest emerged with the aim of adding unit tests to projects with react but it has been so important that it is increasingly used in more projects with other frameworks.

source

UI Boot Camp: Jest

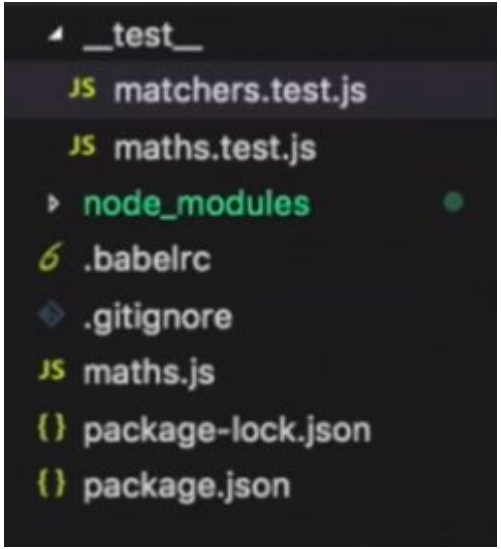
Advantages of using Jest

- Developer ready
- Instant feedback
- Snapshot testing
- Zero configuration testing platform
- Powerful mocking library

UI Boot Camp: Jest

Example

By default, Jest searches for the tests within the "***__test__***" folder, but this can be configured in the package where to look for them.



UI Boot Camp: Jest

Example

describe breaks your test suite into components. Depending on your test strategy, you might have a describe for each function in your class, each module of your plugin, or each user-facing piece of functionality.

Test or it is where you perform individual tests. You should be able to describe each test like a little sentence, such as "it calculates the area when the radius is set". You shouldn't be able to subdivide tests further-- if you feel like you need to, use **describe** instead.

```
describe('Comparadores comunes', () => {
    const user = {
        name: "oscar",
        lastname: "barajas"
    }
    const user2 = {
        name: "oscar2",
        lastname: "barajas2"
    }

    test('igualdad de elementos', () => {
        expect(user).toEqual(user2);
    });

    test('No son exactamente iguales', () => {
        expect(user).not.toEqual(user2);
    })

});
```

UI Boot Camp: Jest

Example

Output of information in console of the previous test. You can visualize correct tests and which failed.

Read more about different expect in jest

UI Boot Camp: Jest

What is a mock?

Mock is a method/object that simulates the behavior of a real method/object in controlled ways. Mock objects are used in unit testing. Often a method under a test calls other external services or methods within it. These are called dependencies. Once mocked, the dependencies behave the way we defined them.

source

With spyOn()

```
const mathjs = require('mathjs')

test(`The mathjs log function`, () => {
  const spy = jest.spyOn(mathjs, 'log')
  const result = mathjs.log(10000, 10)

  expect(mathjs.log).toHaveBeenCalled()
  expect(mathjs.log).toHaveBeenCalledWith(10000, 10)
})
```

With fn()

```
const mathjs = require('mathjs')

mathjs.log = jest.fn(() => 'test')
test(`The mathjs log function`, () => {
  const result = mathjs.log(10000, 10)
  expect(result).toBe('test')
  expect(mathjs.log).toHaveBeenCalled()
  expect(mathjs.log).toHaveBeenCalledWith(10000, 10)
})
```

UI Boot Camp: Jest

Snapshot

Snapshot tests are a very useful tool whenever you want to make sure your UI does not change unexpectedly.

We can update the snapshot: *jest --updateSnapshot*

source

```
import React from 'react';
import Link from '../Link.react';
import renderer from 'react-test-renderer';

it('renders correctly', () => {
  const tree = renderer
    .create(<Link page="http://www.facebook.com">Facebook</Link>)
    .toJSON();
  expect(tree).toMatchSnapshot();
});
```

UI Boot Camp: Jest

Homework!

We're going to continue with the TODO list.

- Let's implement Jest in ALL your components :D


Globant
we are ready