Understanding Jest Mocks

Daniel Stern
CODE WHISPERER

@danieljackstern

“We ape, we mimic, we mock.”

– Laurence Olivier

Why Mocking?

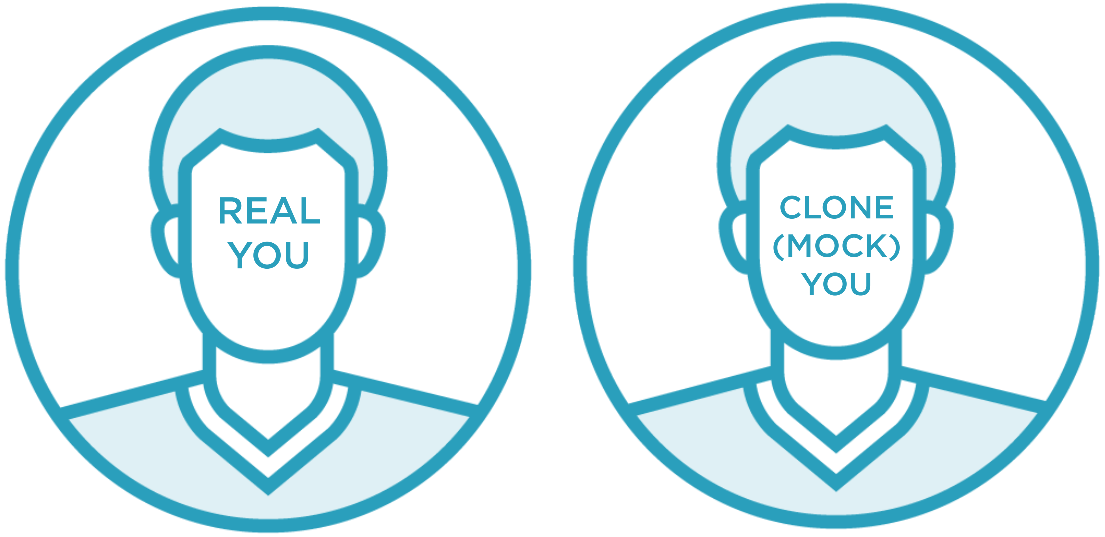
Reduce dependencies required by tests (faster execution)

Prevent side-effects during testing

Build custom mocks to facilitate desired testing procedures

A Mocking Example

A Scientist



Real You	Mock You
Looks like you	Also looks like you
Real feelings	Cold, heartless
Life has purpose	Only purpose is to convince others that he is you

What Is a Mock?

A convincing duplicate of an object with no internal workings

Can be automatically or manually created

Has same API as original, but no side-effects

Spies and other mock features simplify testing

The Mocking Process

Scan the original object for methods, give the new object spy methods with the same names

Ensure that any methods which returned a promise still return a promise in the mock

Create mocks for any complex values that are returned from methods which are required for tests

Mock Functions

Also known as “spies”

No side-effects

Counts function calls

Records arguments passed when called

Can be “loaded” with return values

Return value must approximate original

Creating Mock Files

Appropriately named NPM mocks are loaded automatically

Mocks must reside in a __*mocks*__ folder next to mocked module

NPM modules and local modules can both be mocked

Demo

Create tests for question fetching saga

Create a mock implementation of the *isomorphic-fetch* NPM package

Add custom functionality which allows us to preload chosen values

Note that mock is automatically substituted in for any tests

We will mock locally defined components in a later module

Automatic and Manual Mocking

Automatic / Manual Mocking

In some setups, any require statements will have mocks generated automatically

If a manual mock file exists, it will be used as the mock instead of the automatic version

Most apps require some combination of manual and automatic mocking

Manual Mocks

Exists as a separate file alongside the file being mocked

Manual mocks will be used automatically for NPM modules

Manual mocks are more work than automatic mocks

Needs to be updated when mocked file changes

Automatic Mocking

Most modules can be automatically replaced with mocks

Mocks are usually generated correctly, but sometimes not

Greatly reduced likelihood of side-effects during tests

Developer must use discretion

Automatic Mocking Challenges

Methods returning a specific and complex value often can't be mocked automatically

Methods that are not part of your module at compile-time won't be mocked

Modules that you did not expect to be mocked may be mocked

Summary

A mock module or function is a convincing duplicate of the original with no inner functionality

Simple modules can be automatically mocked, advanced ones may need manual mocking

Manual mocks are created by placing a correctly named file in the *__mocks__* directory

Coming up in the Next Module…

Say Cheese – Snapshot Testing Explained

Advantages and Disadvantages of Snapshot Testing, The Big Picture

Updating Snapshots