ITECHART GROUP

Тестирование React Native приложений

Подготовила Чобот Наталья, 2019

Тестирование — процесс исследования, испытания программного продукта, имеющий своей целью проверку соответствия между реальным поведением программы и её ожидаемым поведением на конечном наборе тестов, выбранных определенным образом.

Manual testing

It is the process of manually **testing** software for defects. It requires a tester to play the role of an end user whereby they use most of the application's features to ensure correct behavior.

Unit testing

Unit testing helps to check that individual unit of code (mostly functions) work as expected.

Integration Tests

Integration tests are tests where individual units/
features of the app are combined
and tested as a group.

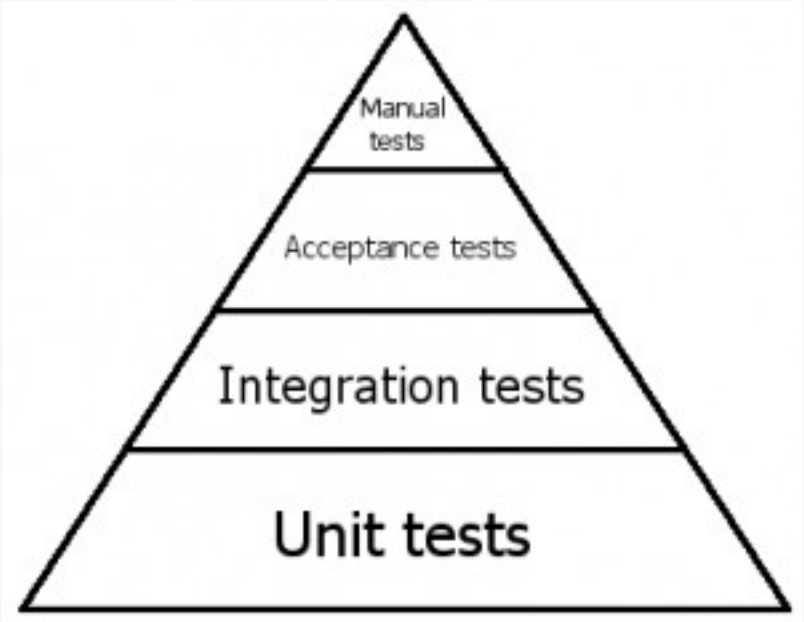
End-to-End Tests

This test helps to confirm that entire features work from the user's perspective when using the actual application.

Кто занимается тестированием

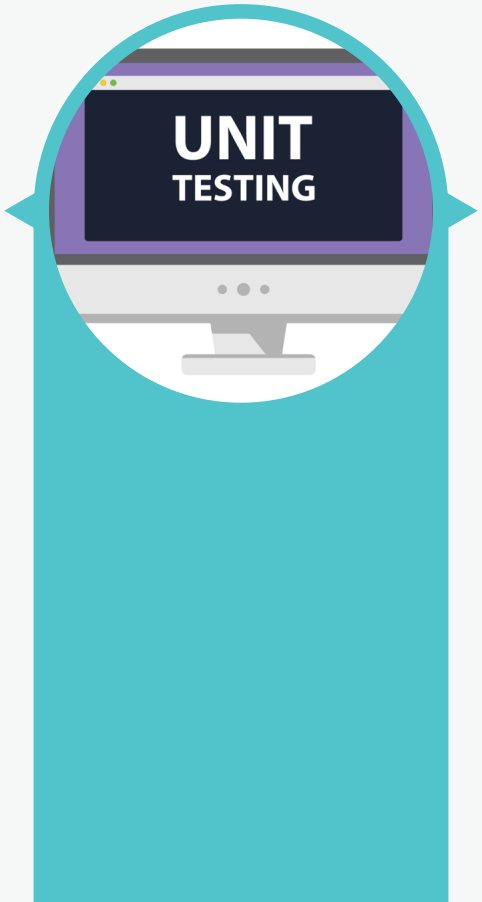
Зачем проекту
и тесты, и тестировщики

Про какие тесты мы будем говорить


Manual tests
Acceptance tests
Integration tests
Unit tests

С чего начать

1



Выберите тестовый фреймворк, который подходит вам


MOCHA


chai


ENZYME

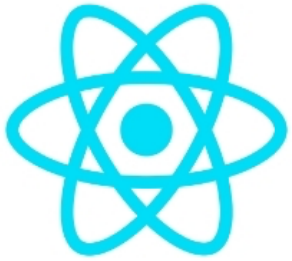
Jest + enzyme

https://jestjs.io/docs/en/tutorial-react-native

https://airbnb.io/enzyme/docs/guides/react-native.html

Enzyme

- JavaScript Testing utility for React
- Open sourced by Airbnb (2016)
- Makes testing UI easier



npm i —save-dev
enzyme enzyme-adapter-react-16

npm i —save-dev react-dom

1

npm i —save-dev enzyme-to-json

2

npm install --save-dev jest

Create setupTests.js
(new Adapter)

3

4

Create jest.config.js


Структура
1
Именование файлов и самих тестов
2
Что тестировать?
3
Как тестировать?
4
Везде ли использовать снепшоты?
5
Что такое TDD?
6

TDD

Пишем тест на несуществующее поведение

Запускаем – он **падает**

Пишем нужное поведение

Запускаем тест – **зеленый**

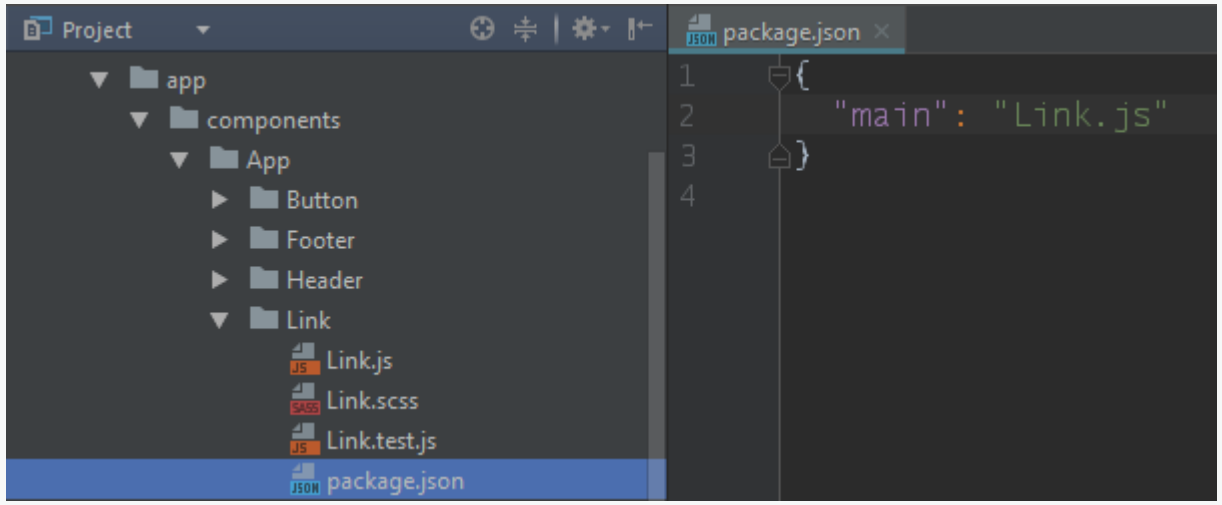
Рефакторим поведение, очищая код

Запускаем тест – зеленый

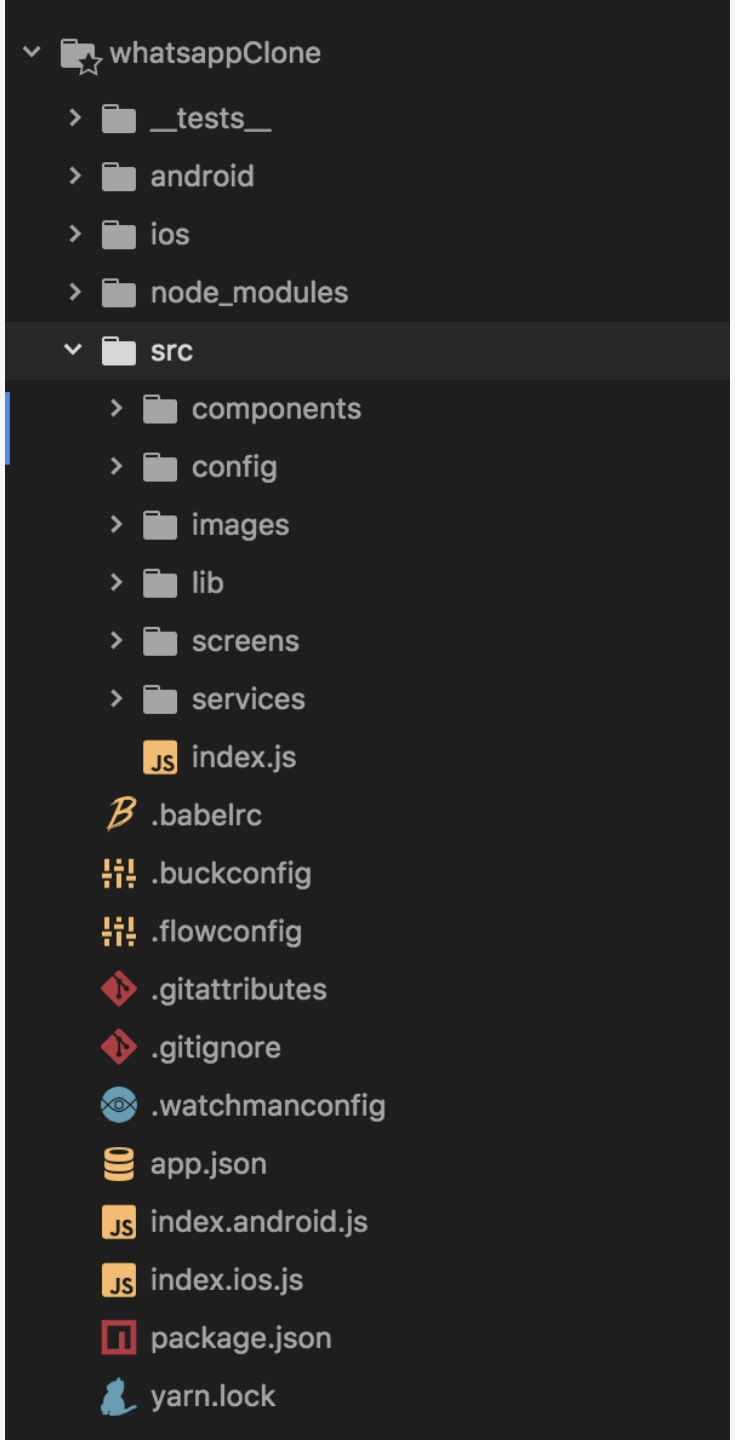
Комитим наши изменения

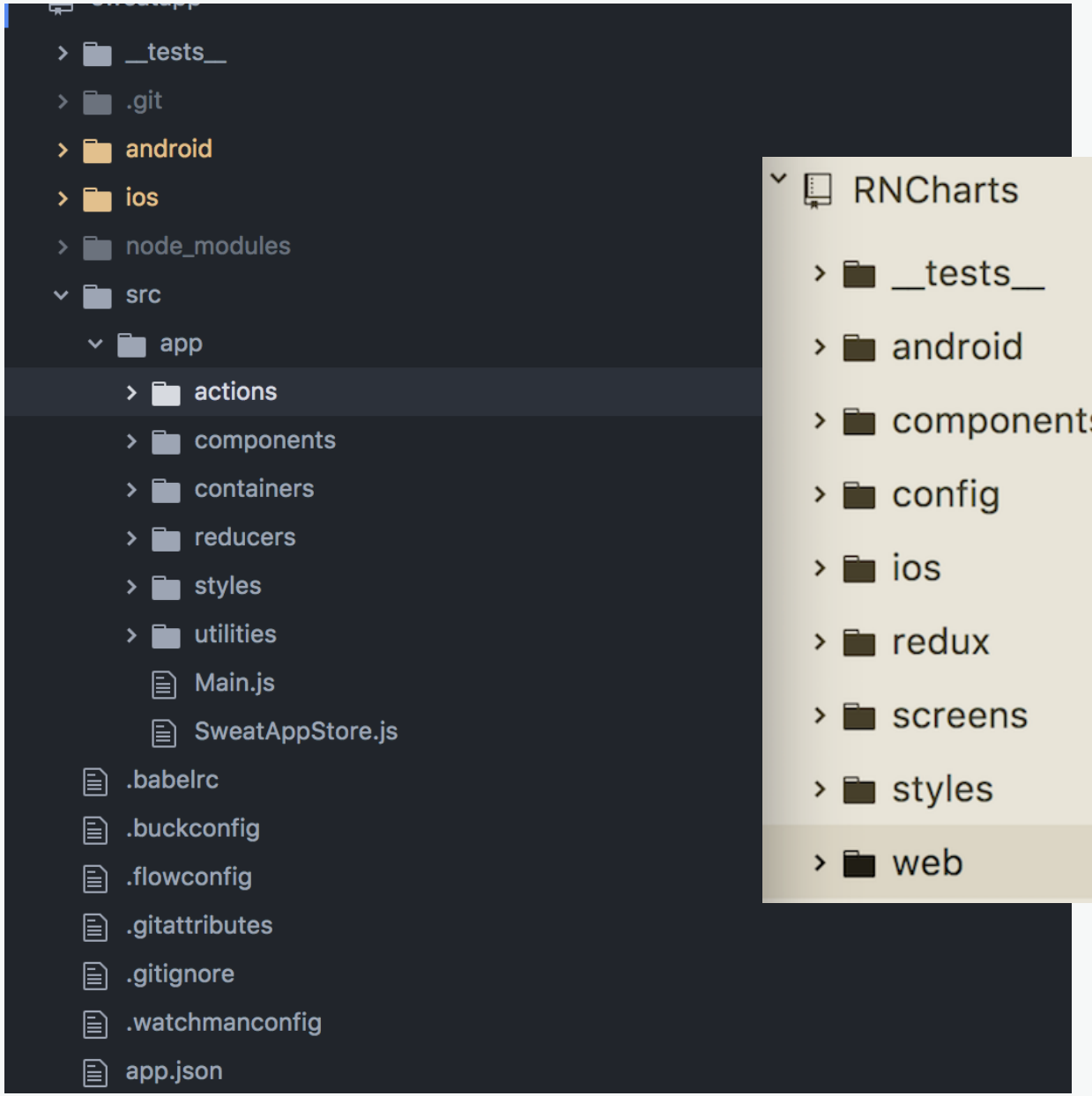
Структура тестов в проекте

- src
  - Components
    - Board
      - board.js
      - board.test.js
    - Game
      - game.js
      - game.test.js
    - Square
      - square.js
      - square.test.js
  - index.css
  - index.js

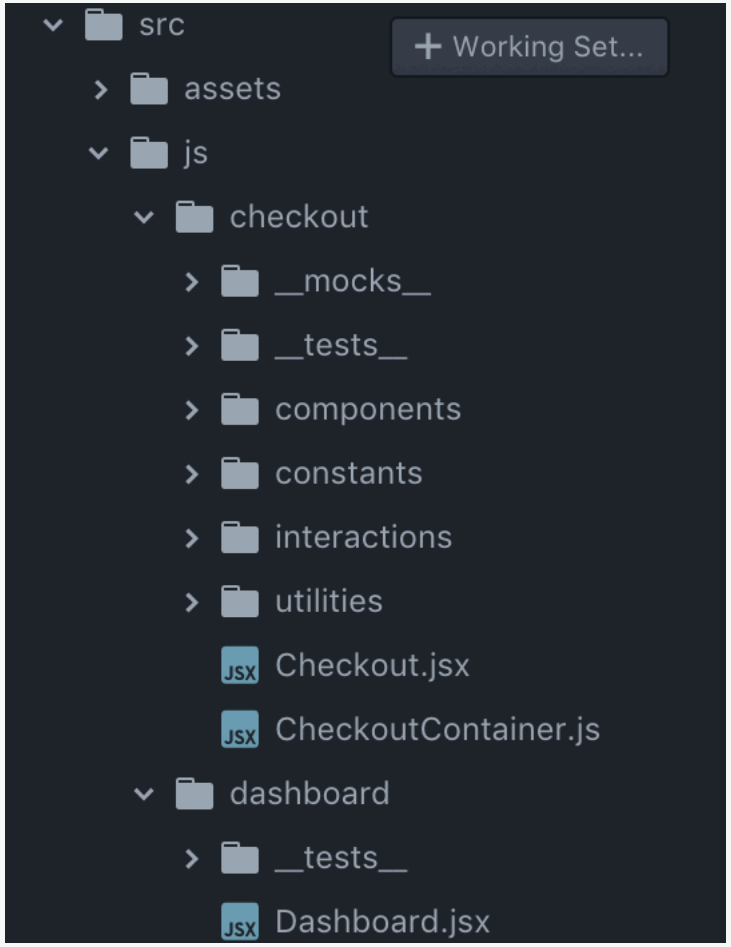


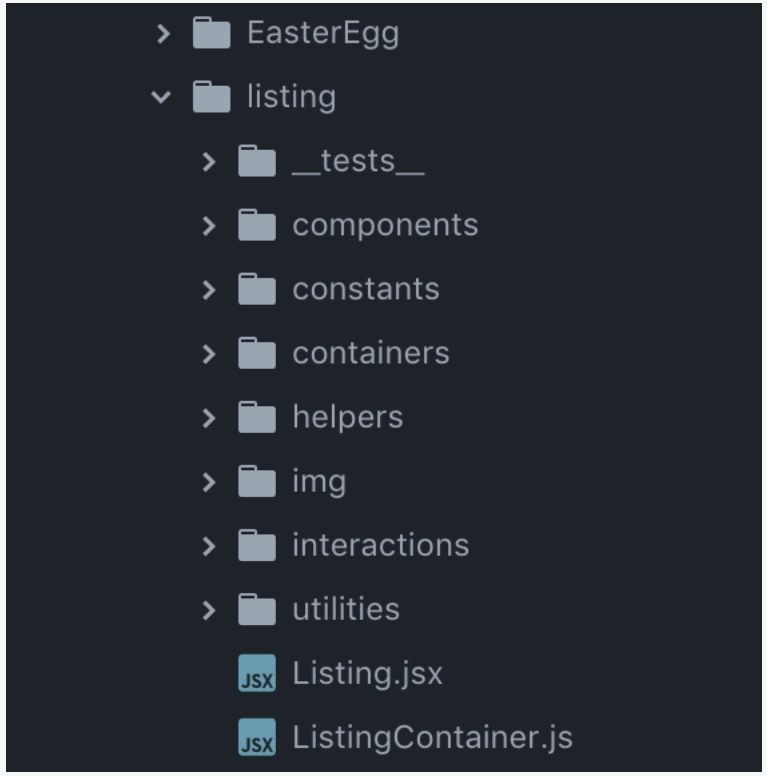
Структура тестов в проекте

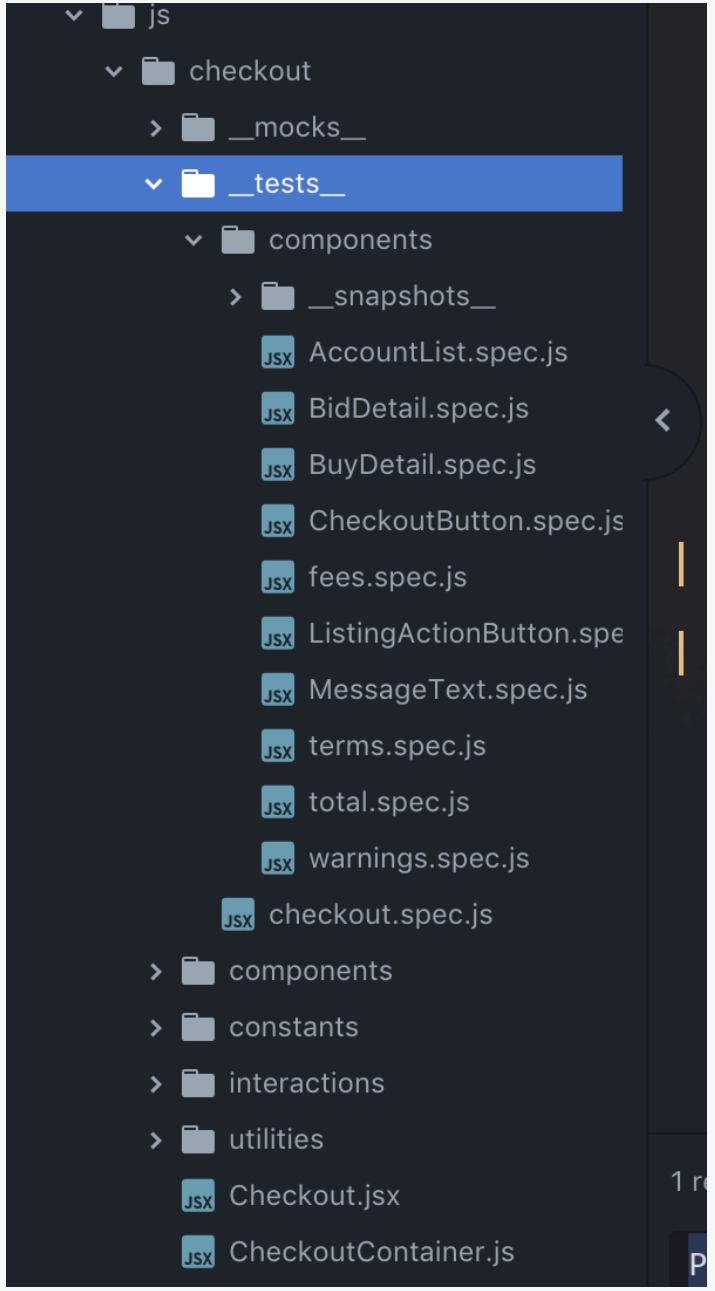




Структура тестов в проекте







Jest matchers

.toBe()	compares numbers, strings - primitives
.toEqual()	compares objects deeply
.toBeDefined()	checks for the existence of a variable or a function
.toBeClicked()	checks for button: if it was clicked or not
.toHaveLength()	like .toBe() but for checking length of arrays (more preferable)
.toHaveBeenCalled()	if method has been called
.not.toHaveBeenCalled()	if method hasn't been called
.toHaveBeenCalledTimes(number)	compare to how many times method has been called
.toContain	

Нужен ли тесты?

Больше уверенности в коде

Можно смелее рефакторить

Спасибо за внимание!