



Лекция 1

Разработка с CRA.

Тестирование 1

Мартин Комитски



ЦРИТО
Центр Развития
ИТ-Образования



План на сегодня

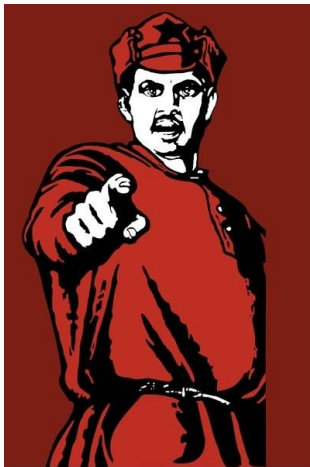


- Разработка с CRA
 - Все фичи
 - Стили, SCSS
 - UI Kits
 - Proxy
 - ...
- Storybook
- Тестирование
 - Что это и зачем?
 - Виды
 - TDD
 - Jest
 - Puppeteer
- TDD/BDD

Минутка бюрократии



- Внимание
- Отметки о посещении занятий
- Обратная связь о лекциях





Все фичи CRA



Зависимости

CRA. Упрощенный список зависимостей



```
1.  "dependencies": {
2.    "@babel/core": "7.2.2", // babel
3.    "eslint": "5.12.0", // lint
4.    "jest": "23.6.0", // tests
5.    "react": "^16.8.6", // react
6.    "react-dom": "^16.8.6", // react-dom
7.    "react-app-polyfill": "^0.2.2", // Polyfills
8.    "react-dev-utils": "^8.0.0", // utils https://www.npmjs.com/package/react-dev-utils
9.    "html-webpack-plugin": "4.0.0-alpha.2", // work with html
10.   "mini-css-extract-plugin": "0.5.0", // optimize styles
11.   "optimize-css-assets-webpack-plugin": "5.0.1", // optimize styles
12.   "postcss-loader": "3.0.0", // postcss preset-env { autoprefixer }
13.   "sass-loader": "7.1.0", // load scss styles
14.   "style-loader": "0.23.1", // load styles
15.   "css-loader": "1.0.0", // load styles
16.   "@svgr/webpack": "4.1.0", // load files
17.   "file-loader": "2.0.0", // load files
18.   "url-loader": "1.1.2", // load files
19.   "webpack": "4.28.3", // webpack
20.   "webpack-dev-server": "3.1.14", // dev server
21.   "webpack-manifest-plugin": "2.0.4", // pwa manifest
22.   "workbox-webpack-plugin": "3.6.3" // service workers
23. },
24.
```

CRA. Babel



BABEL

DocsSetupTry it outVideosBlogSearchDonateTeam

SETTINGS

☐ Evaluate

☒ Line Wrap

☐ Minify

☐ Prettify

☐ File Size

☐ Time Travel

Source Type

Module

PRESETS

☒ es2015

```
1 const something = [1, 2 ** 3];
2 const anotherOne = [...something];
3
4 class Human {
5   constructor(props) {
6     this.name = props.name;
7     this.age = props.age;
8   }
9 }
10
11 const user = new Human({ name: 'Martin', age: '55'
12 });
13 const { name, age } = user;
```

```
1 "use strict";
2
3 var something = [1, Math.pow(2, 3)];
4 var anotherOne = [].concat(something);
5
6 var Human = function Human(props) {
7   this.name = props.name;
8   this.age = props.age;
9 };
10
11 var user = new Human({
12   name: 'Martin',
13   age: '55'
14 });
15 var name = user.name,
16   age = user.age;
```

CRA. ESLint



```
-OSX:memphisw-eslint gspake$ ./node_modules/.bin/eslint .

/Users/gspake/code/memphisw-eslint/src/App.js
  5:1  error  Component should be written as a pure function  react/prefer-stateless-function
  8:7  error  JSX not allowed in files with extension '.js'   react/jsx-filename-extension

/Users/gspake/code/memphisw-eslint/src/App.test.js
  5:1  error  'it' is not defined          no-undef
  6:15 error  'document' is not defined     no-undef
  7:19 error  JSX not allowed in files with extension '.js'   react/jsx-filename-extension

/Users/gspake/code/memphisw-eslint/src/index.js
  7:17 error  JSX not allowed in files with extension '.js'   react/jsx-filename-extension
  7:26 error  'document' is not defined     no-undef

/Users/gspake/code/memphisw-eslint/src/registerServiceWorker.js
  12:3  error  'window' is not defined          no-undef
  14:5  error  'window' is not defined          no-undef
  16:5  error  'window' is not defined          no-undef
  17:63 error  Missing trailing comma           comma-dangle
  18:6  error  Missing trailing comma           comma-dangle
  22:67 error  'navigator' is not defined       no-undef
  24:27 error  'URL' is not defined             no-undef
  24:55 error  'window' is not defined          no-undef
  25:30 error  'window' is not defined          no-undef
  32:5  error  'window' is not defined          no-undef
  37:9  error  'registerValidSW' was used before it was defined  no-use-before-define
  40:9  error  'checkValidServiceWorker' was used before it was defined  no-use-before-define
  47:3  error  'navigator' is not defined       no-undef
  49:11 error  Expected parentheses around arrow function argument having a body with curly braces  arrow-parens
  50:7  error  Assignment to property of function parameter 'registration'  no-param-reassign
  54:17 error  'navigator' is not defined       no-undef
  59:15 warning Unexpected console statement  no-console
  64:15 warning Unexpected console statement  no-console
  70:12 error  Expected parentheses around arrow function argument having a body with curly braces  arrow-parens
  71:7  warning Unexpected console statement  no-console
  77:3  error  'fetch' is not defined           no-undef
  78:11 error  Expected parentheses around arrow function argument having a body with curly braces  arrow-parens
  85:9  error  'navigator' is not defined       no-undef
  85:44 error  Expected parentheses around arrow function argument having a body with curly braces  arrow-parens
  87:13 error  'window' is not defined          no-undef
  96:7  warning Unexpected console statement  no-console
  97:72 error  Missing trailing comma           comma-dangle
  103:26 error  'navigator' is not defined       no-undef
  104:5  error  'navigator' is not defined       no-undef
  104:40 error  Expected parentheses around arrow function argument having a body with curly braces  arrow-parens

* 37 problems (33 errors, 4 warnings)
  8 errors, 0 warnings potentially fixable with the '--fix' option.
```




FAIL src/components/App.test.js

Examining the syntax of Jest tests

✗ sums numbers (182ms)

- Examining the syntax of Jest tests > sums numbers

expect(received).toEqual(expected)

Expected value to equal:

5

Received:

4

```
3 |         it('sums numbers', () => {
4 |             expect(1 + 2).toEqual(3);
> 5 |             expect(2 + 2).toEqual(5);
    |                               ^
6 |         });
7 |     });
```

at Object.toEqual (src/components/App.test.js:5:18)

Test Suites: 1 failed, 1 total

Tests: 1 failed, 1 total

Snapshots: 0 total

Time: 0.965s, estimated 1s

Ran all test suites related to changed files.

CRA. React/ReactDOM



```
1 import React from 'react';
2 import ReactDOM from 'react-dom';
3 import App from './App';
4 import registerServiceWorker from './registerServiceWorker';
5 import './index.css';
```

```
6
7 react
```

```
8 React [x] React import React
```

```
9 regi [x] ReactDOM
```

```
10
    RequestCache
    • RegExpMatchArray
    • URLSearchParams
    RequestCache
    • RTCIceGatherCandidate
    • RTCRtcpFeedback
    • FrameRequestCallback
    • RTCIceCandidateComplete
    • RTCMediaStreamTrackStats
    • RTCIceCandidatePairChangedEvent
    • RTCIceTransportStateChangeEvent
```

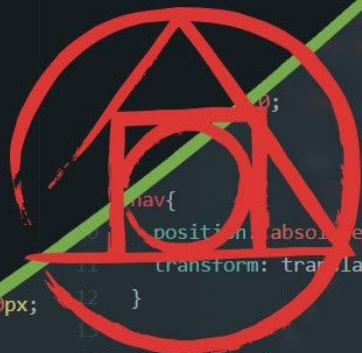
CRA. PostCSS {Autoprefixer}



```
gulpfile.js  X  bundle.css  app.css  index.html

1  var gulp = require('gulp');
2  var postcss = require('gulp-postcss');
3  var rename = require('gulp-rename');
4
5  // PostCSS Plugins
6  var processors = [
7    require('precss'),
8    require('rucksack-css'),
9    require('css-mqpacker'),
10   require('lost'),
11   require('autoprefixer')({ browsers: ['last 5 version']}),
12   require('postcss-sprites')({
13     stylesheetPath: './',
14     spritePath: './img/sprite.png',
15     retina: true,
16     outputDimensions: true,
17     filterBy: function(image) {
18       return /sprite/gi.test(image.url);
19     }
20   })
21 ]
22
23 .main{
24   lost-column: 1/3;
25   background-color: #ff0000;
26 }
27
28 .responsive{
29   font-size: responsive 14px 60px;
30 }
31
32 .container{
33   lost-center: 980px;
34   background-color: #ff0000;
35 }
```

PostCSS



CRA. SVGR



SVGR: The SVG to JSX transformer

Secure <https://svgr.now.sh>

Made with ❤ by Smooth Code

Global

- ☒ SVGO
- ☒ Prettier
- ☒ Expand props
- ☐ Icon
- ☐ React Native
- ☐ Ref

REPLACE ATTRIBUTE VALUE

#063855=currentColor

SVGO

- ☒ Title
- ☒ ViewBox
- ☐ Ids

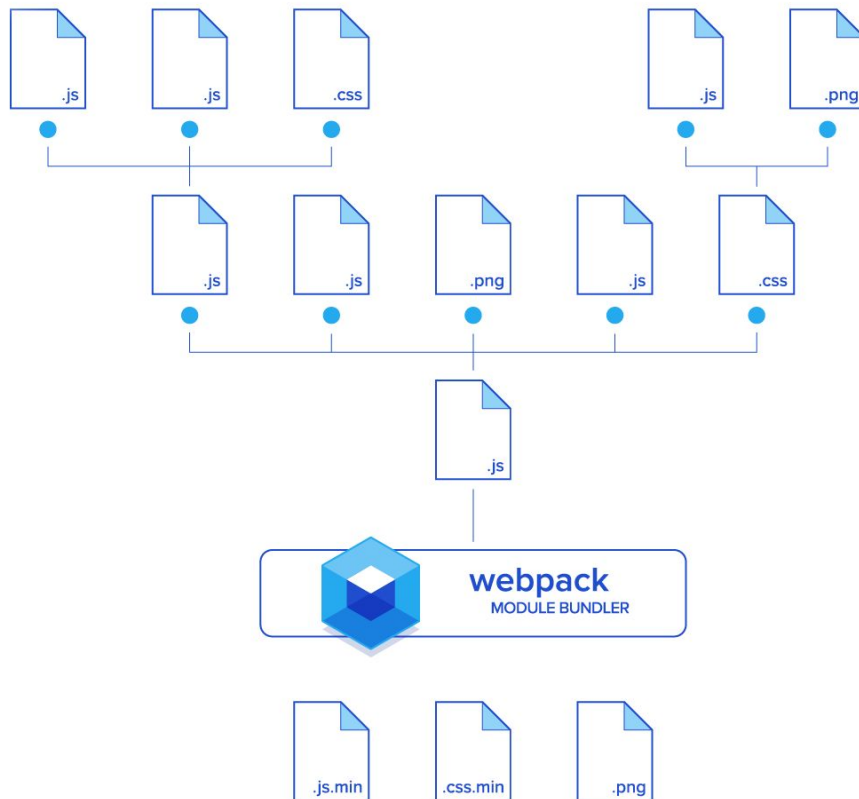
PRECISION

3

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <svg width="48px" height="1px" viewBox="0 0 48 1" version="1.1" xmlns="http://www.w3.org/2000/svg">
3   <!-- Generator: Sketch 46.2 (44496) - http://www.bohemiancoding.com/sketch -->
4   <title>Rectangle 5</title>
5   <desc>Created with Sketch.</desc>
6   <defs></defs>
7   <g id="Page-1" stroke="none" stroke-width="1" fill="none" fill-rule="evenodd">
8     <g id="19-Separator" transform="translate(-129.000000, -156.000000)">
9       <g id="Controls/Settings" transform="translate(80.000000, 0)">
10        <g id="Content" transform="translate(0.000000, 64.000000)">
11          <g id="Group" transform="translate(24.000000, 56.000000)">
12            <g id="Group-2">
13              <rect id="Rectangle-5" x="25" y="36" width="1" height="1"></rect>
14            </g>
15          </g>
16        </g>
17      </g>
18    </g>
19  </g>
20 </svg>
```

```
1 import React from "react";
2
3 const SvgComponent = props => (
4   <svg width={48} height={1} viewBox="0 0 48 1" {...props}>
5     <title>Rectangle 5</title>
6     <path d="M0 0h48v1H0z" fill="#063855" fillRule="evenodd" />
7   </svg>
8 );
9
10 export default SvgComponent;
11
```

CRA. Webpack



CRA. Webpack Dev Server



```
webpack: bundle is now VALID.  
webpack: bundle is now INVALID.  
Hash: 7dfc6925eca25650b928  
Version: webpack 1.12.13  
Time: 59ms
```

Asset	Size	Chunks	Chunk Names
bundle.js	496 kB	0 [emitted]	main
chunk {0} bundle.js (main)	470 kB	[rendered]	
[74] ./src/index.js	283 bytes	{0} [built]	
+ 75 hidden modules			

```
webpack: bundle is now VALID.
```

webpack

```
index.js buffers  
1 import $ from 'jquery';  
2  
3 let message = 'webpack';  
4 $('body').html(`<h1>${message}</h1>`);  
~  
~  
~  
~
```

CRA. WorkBox



localhost:8080

← → ↻ localhost:8080

Hello PWA with Next.js

Check the console for the Service Worker registration status.

Application

- Manifest
- Service Workers**
- Clear storage

Storage

- Local Storage
- Session Storage
- IndexedDB
- Web SQL
- Cookies

Service Workers

☐ Offline ☐ Update on reload ☐ Bypass for network ☐ Show all

localhost [Update](#) [Unregister](#)

Source [sw.js](#)

Received 2/17/2018, 9:38:04 PM

Status ● #1774 activated and is running [stop](#)

Console

top Filter Default levels

- workbox** Welcome to Workbox! [workbox-core.dev.js:423](#)
- service worker registration successful [index.js:1](#)
- workbox** Are your precached assets revisioned? [workbox-core.dev.js:423](#)
- workbox** Precaching 0 files. 3 files are already cached. [workbox-core.dev.js:423](#)
- workbox** View URLs that were already precached. [workbox-core.dev.js:423](#)
- workbox** /_next/5cad3299-fe3c-42b0-ada0-0a45d42ad9df/page/_error.js [workbox-core.dev.js:423](#)
- workbox** /_next/5cad3299-fe3c-42b0-ada0-0a45d42ad9df/page/index.js [workbox-core.dev.js:423](#)
- workbox** /_next/62361c45736061b7ed5258b537109ac9/app.js [workbox-core.dev.js:423](#)



Разработка

CRA. Разработка



```
1. npx create-react-app my-app
2.
3. # доступные скрипты
4. npm start
5. npm test
6. npm run build
7. npm run eject
8.
9. // package.json
10. + "analyze": "source-map-explorer 'build/static/js/*.js'"
11. + "start": "HTTPS=true react-scripts start"
```

Продвинутая конфигурация.

Поддержка всех современных браузеров, есть возможность поддержки IE9+ при помощи [react-app-polyfill](#).

Синтаксис:

- Exponentiation Operator (ES2016).
- Async/await (ES2017).
- Object Rest/Spread Properties (ES2018).
- Dynamic import() (stage 4 proposal)
- Class Fields and Static Properties (part of stage 3 proposal).
- JSX, Flow and TypeScript.

```
my-app/
  README.md
  node_modules/
  package.json
  public/
    index.html
    favicon.ico
  src/
    App.css
    App.js
    App.test.js
    index.css
    index.js
    logo.svg
```

CRA. Разработка. browserslist



```
1.  /* помогает указывать целевые браузеры, под которые будет компилироваться код */
2.  "browserslist": {
3.    "production": [
4.      ">0.2%",
5.      "not dead",
6.      "not op_mini all"
7.    ],
8.    "development": [
9.      "last 1 chrome version",
10.     "last 1 firefox version",
11.     "last 1 safari version"
12.   ]
13. }
14.
15. https://browserl.ist/
16.
```

CRA. Разработка. Обновление



Create React App разделен на 2 пакета:

- *create-react-app* - это глобальная cli утилита для создания проектов
- *react-scripts* - это набор скриптов, которые позволяют разрабатывать, собирать и тестировать проект

```
npx create-react-app my-app === npm i create-react-app -g && create-react-app
```

Для обновления CRA в существующих проектах в большинстве случаев можно просто обновить версию пакета *react-scripts*.

В редких случаях потребуется что-то делать руками. Информация всегда описана в [changelog](#).



Button.css

```
.Button {  
  padding: 20px;  
}
```

Button.js

```
import React, { Component } from 'react';  
import './Button.css'; // Tell Webpack that Button.js uses these styles  
  
class Button extends Component {  
  render() {  
    // You can use them as regular CSS styles  
    return <div className="Button" />;  
  }  
}
```

CRA. Стили. CSS Modules



Button.module.css

```
.error {  
  background-color: red;  
}
```

another-stylesheet.css

```
.error {  
  color: red;  
}
```

Button.js

```
import React, { Component } from 'react';  
import styles from './Button.module.css'; // Import css modules stylesheet as styles  
import './another-stylesheet.css'; // Import regular stylesheet  
  
class Button extends Component {  
  render() {  
    // reference as a js object  
    return <button className={styles.error}>Error Button</button>;  
  }  
}
```

CRA. Стили. SCSS



```
$ npm install node-sass --save
```

```
src/App.css -> src/App.scss
```

```
1. // src/App.scss
2. @import 'styles/_colors.scss'; // Импортирование общих стилей
3. @import '~nprogress/nprogress'; // Импортирование стилей из внешних пакетов
4.
5. // Можно и комбинировать с CSS Modules!
6.
```

CRA. Стили. Post CSS. Autoprefixer



```
.App {  
  display: flex;  
  flex-direction: row;  
  align-items: center;  
}
```

```
.App {  
  display: -webkit-box;  
  display: -ms-flexbox;  
  display: flex;  
  -webkit-box-orient: horizontal;  
  -webkit-box-direction: normal;  
  -ms-flex-direction: row;  
  flex-direction: row;  
  -webkit-box-align: center;  
  -ms-flex-align: center;  
  align-items: center;  
}
```

CRA. Стили. Внешние UI kits



```
$ npm install --save bootstrap
```

```
1.  // App.js
2.  import 'bootstrap/dist/css/bootstrap.css';
3.
4.  // style.scss
5.  // Переопределение переменных из внешних UI kits
6.  $body-bg: #000;
7.
8.  // Импорт стилей в самом scss
9.  @import '~bootstrap/scss/bootstrap.scss';
10.
```

<https://material-ui.com/getting-started/installation/>

<https://hackernoon.com/23-best-react-ui-component-libraries-and-frameworks-250a81b2ac42>

CRA. Разработка. Дополнительные параметры CRA



```
// CRA + flow
$ npm install --save flow-bin

// CRA + TS
$ npx create-react-app my-app --typescript

// CRA + Relay
$ npm install --save babel-plugin-relay@dev
```

CRA. Разработка. Прoxy в бекенд



```
1. // package.json
2. ...
3. "proxy": "http://localhost:4000",
4. ...
```

`fetch('/api/todos') -> http://localhost:4000/api/todos`

CRA. Разработка в изоляции. Storybook



Обычно, в приложении имеется большое количество ui компонентов, каждый из них имеет множество состояний. Например, простой компонент кнопки может иметь следующие свойства:

- Обычное состояние, с текстом
- Выключенное состояние
- Состояние загрузки (preloader)

Чаще всего бывает сложно увидеть эти состояния без запуска самого приложения или тестовых примеров на его основе.

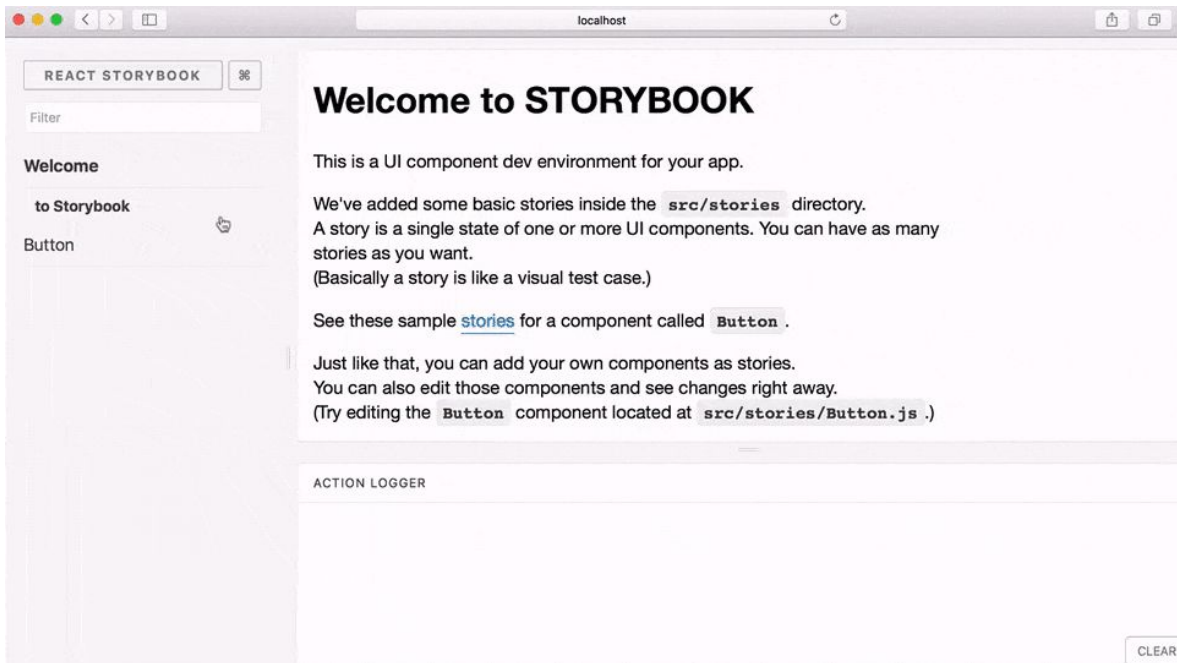
<https://learnstorybook.com/>

<https://storybook.js.org/basics/introduction/>

CRA. Разработка в изоляции. Storybook



```
$ npx -p @storybook/cli sb init
```



CRA?



Вопросы?





“

Перерыв! (10 минут)

*Препо*д (с)



Тестирование

Тестирование. Обеспечение качества (Quality Assurance)



QA - совокупность мероприятий, охватывающих все этапы разработки, выпуска и эксплуатации и предпринимаемых на разных стадиях жизненного цикла ПО для обеспечения качества выпускаемого продукта.

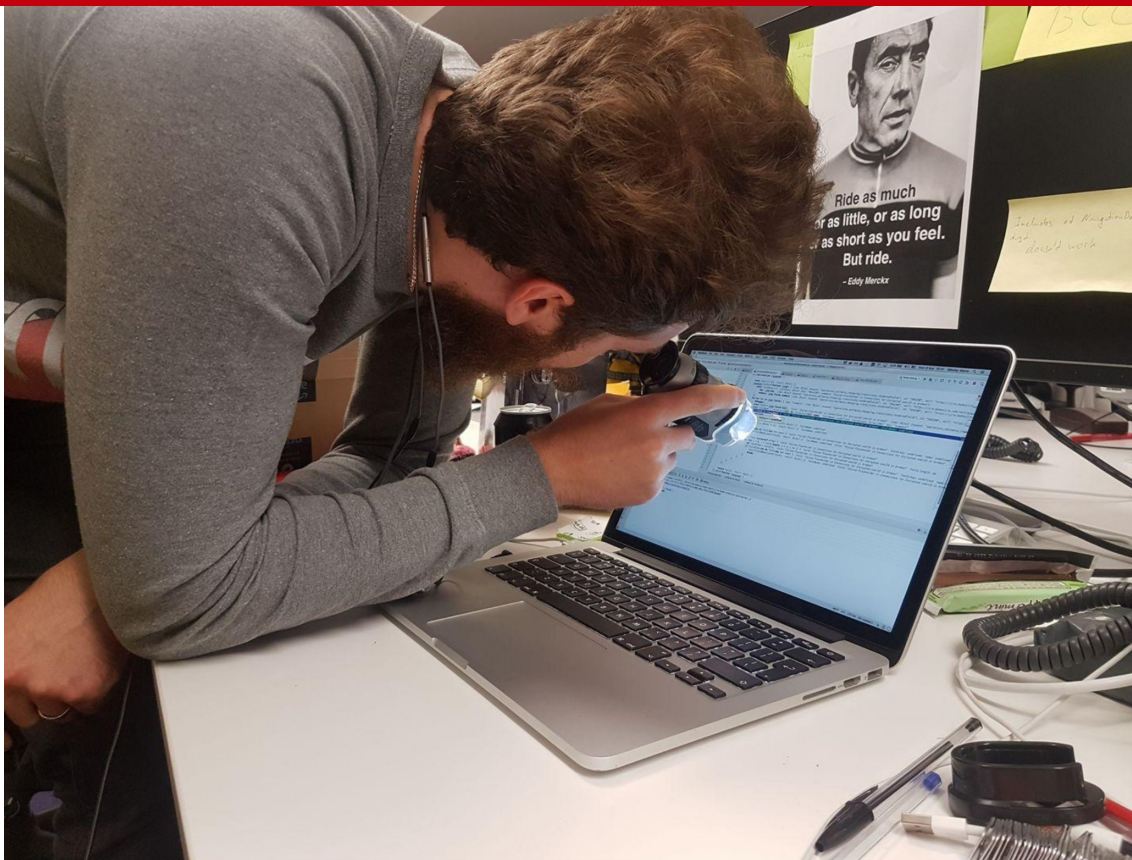
Тестирование. Обеспечение качества (Quality Assurance)



Этапы обеспечения качества ПО. **Внутренний:**

- Тестирование документации
- Unit-тесты
- Code-review

Тестирование. Code-review



Тестирование. Обеспечение качества (Quality Assurance)



Этапы обеспечения качества ПО. **Внешний:**

- Обработка обратной связи
- Тестирование
- Менеджерская приемка

Тестирование. QA. Обработка обратной связи



- Взаимодействие в соц.сетях/магазинах и пр.
- Обработка заявок в саппорт
- Фокус тестирование
- α/β тестирование
- A/B тестирование

Тестирование. Контроль качества (Quality Control)



Quality Control - совокупность мероприятий, проводимых над объектом в процессе разработки для постоянного получения информации об актуальном состоянии объекта и его соответствии требованиям и заявленному уровню качества.

Тестирование - процесс исследования ПО с целью получения актуальной информации о качестве продукта.

Цели:

- Проверка на соответствие требованиям
- Повышение качества ПО
- Предотвращение дефектов
- Выявление дефектов

Тестирование. Виды тестирования



Тестирование. Виды тестирования по целям



Функциональные:

- Функциональное/End-to-end тестирование
- Тестирование безопасности
- Интеграционное тестирование
- Unit-тесты
- Smoke-тесты
- Тестирование установки

Тестирование. Виды тестирования по целям



Нефункциональные:

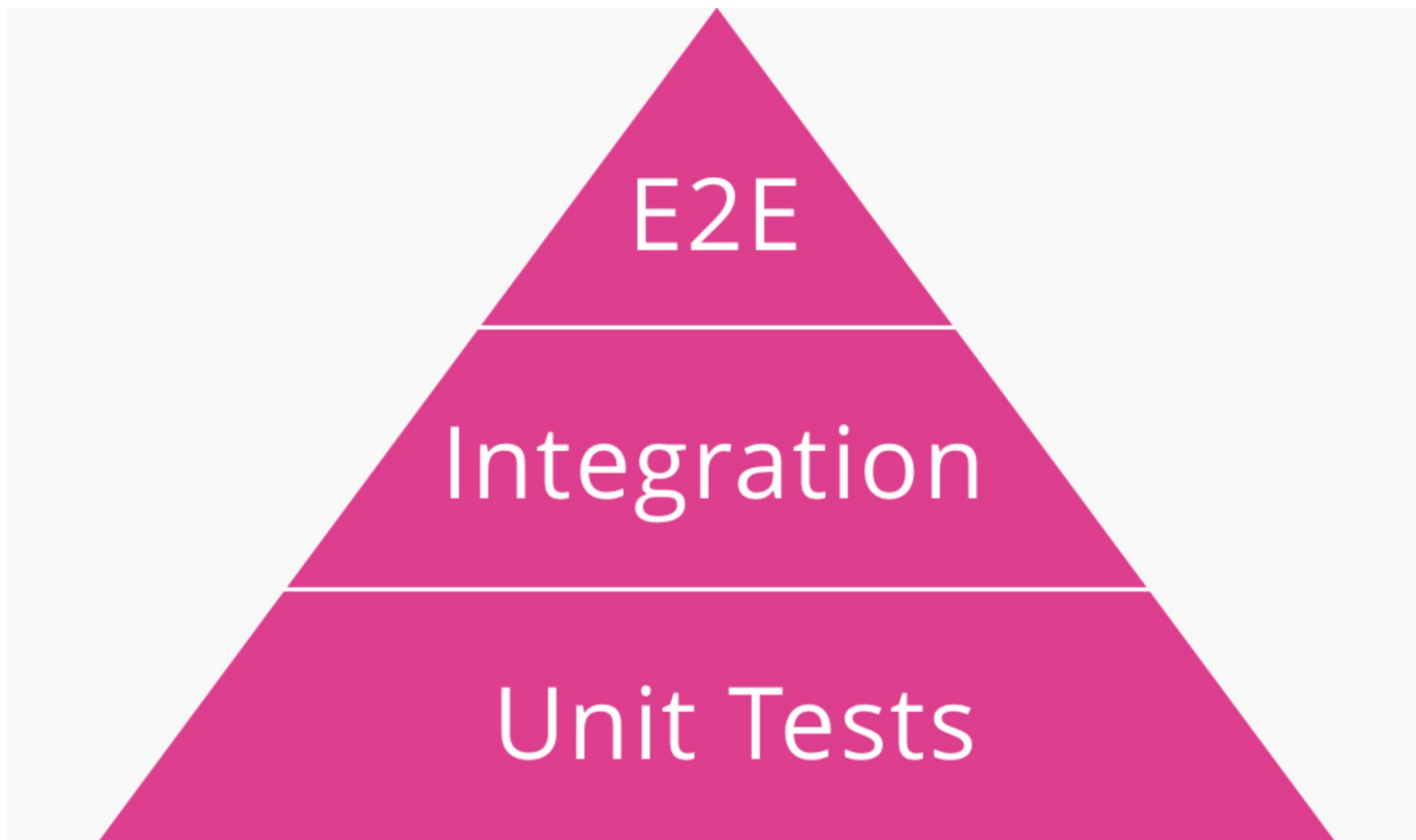
- Тестирование отказа и восстановления после сбоев
- Нагрузочное тестирование
- Тестирование удобства использования ([UX](#))
- Тестирование локализации

Тестирование. Виды тестирования по целям



По хронологии:

- Дымовое
- Тестирование сборки
- Регрессионное
- Приемочное тестирование

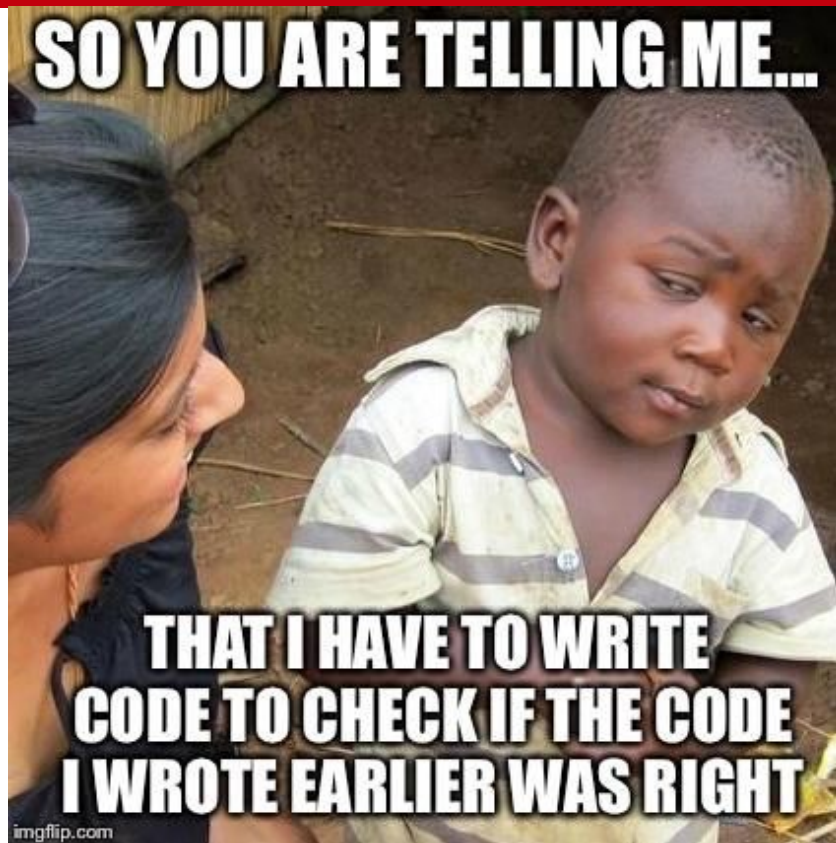


Тестирование. Unit



Unit-тесты - тесты, позволяющие проверить на корректность отдельные модули исходного кода программы, наборы из одного или более программных модулей вместе с соответствующими управляющими данными, процедурами использования и обработки.

Тестирование. Unit. Зачем?



Тестирование. Unit. Зачем?



- Качество кода, быстрый фидбек
- Если есть тесты, то вносить изменения легче => уверенный разработчик
- Повышение скорости разработки (производительность)
- Минимизация человеческого фактора
- Документация
- Возможность автоматизации (CI/CI)

Тестирование. Unit



Unit-тесты не проверяют:

- БД
- Другие либы и пакеты

Почему их надо писать?

- Проверяют логику и всё, где есть условия
- Быстро и легко написать
- Быстро проходят

Тестирование. Пример Unit-теста



```
describe('Basic Mocha String Test', function () {  
  it('should return number of charachters in a string', function () {  
    assert.equal("Hello".length, 4);  
  });  
  
  it('should return first charachter of the string', function () {  
    assert.equal("Hello".charAt(0), 'H');  
  });  
});
```

Тестирование. Принципы написания автотестов



- Атомарность
- Независимость
- Изолированность / герметичность

Тестирование. Принципы написания автотестов. Атомарность



```
1 describe('isUndefined function', ()=> {
2     it('should return true or false when object is undefined', () => {
3         expect(isUndefined(undefined)).toEqual(true);
4         expect(isUndefined(true)).toEqual(false);
5     });
6 });
```



```
1 describe('isUndefined function', ()=> {
2     it('should return true when object is undefined', () => {
3         expect(isUndefined(undefined)).toEqual(true);
4     });
5
6     it('should return false when object has a boolean value', () => {
7         expect(isUndefined(true)).toEqual(false);
8     });
9 });
```



Тестирование. Принципы написания автотестов. Независимость



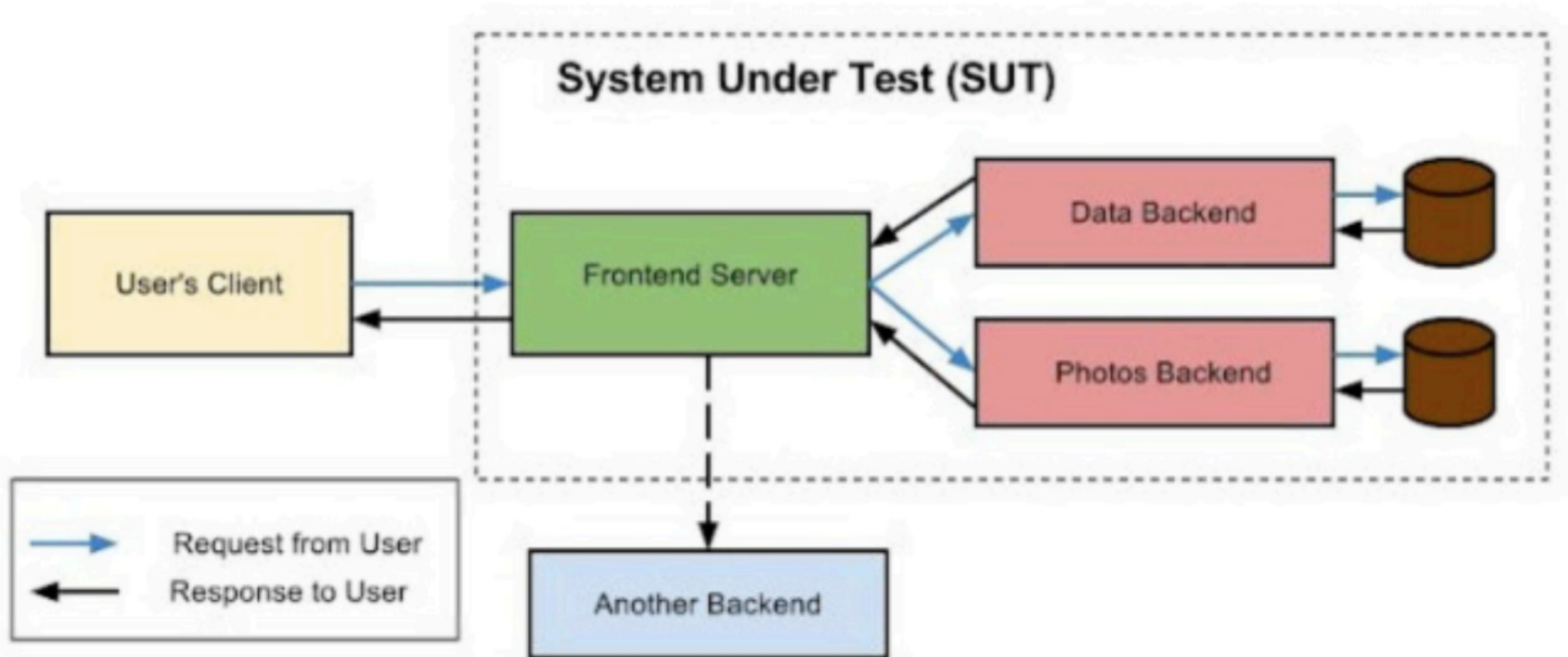
- Test 1
- Test 2
- Test 3

- Test 3
- Test 2
- Test 1

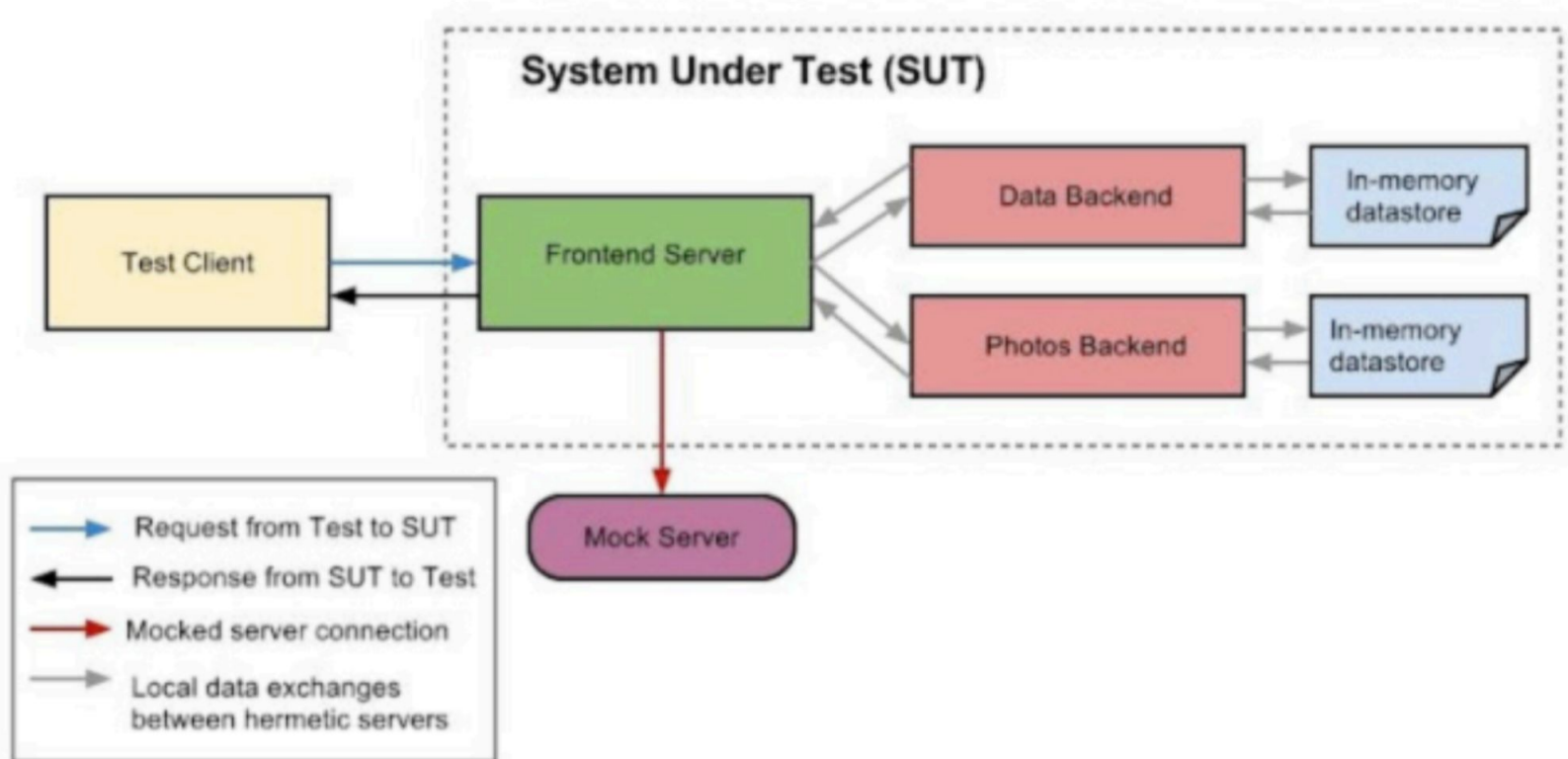
- Test 2
- Test 3
- Test 1

...

Тестирование. Принципы написания автотестов. Изолированность



Тестирование. Принципы написания автотестов. Изолированность



Тестирование. Что еще за Mock server? Заглушки



- Fake – имитация, “упрощенная” рабочая реализация
- Mock – эмуляция объекта, не рабочая реализация
- Stub – заглушка для объекта
- Spy – это Stub с записью данных или вызовов, поступающих из тестируемого объекта
- Dummy - пустой объект

Тестирование. Фреймворки для Unit



Тестирование. Jest



- Скорость
- Все уже есть (assertions, mocks, spies)
- Можно использовать другие либы
- Snapshot testing
- Code coverage
- Интерфейс понятен и удобен

Тестирование. Jest. Пример Unit-теста



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

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

<https://jestjs.io/docs/ru/api>

Тестирование. Jest. Пример snapshot-теста



```
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();
});

// snapshot
exports[`renders correctly 1`] = `
<a
  className="normal"
  href="http://www.facebook.com"
  onMouseEnter={[[Function]]}
  onMouseLeave={[[Function]]}
>
  Facebook
</a>
`;
```

Тестирование. Integration



Integration-тесты - вид тестирования, при котором на соответствие требований проверяется интеграция модулей, их взаимодействие между собой, а также интеграция подсистем в одну общую систему.

Для интеграционного тестирования используются компоненты, уже проверенные с помощью модульного тестирования, которые группируются в множества. Данные множества проверяются в соответствии с планом тестирования, составленным для них, а объединяются они через свои интерфейсы.

[Пример](#)



Тестирование. E2E



E2E-тесты - вид тестирования, где проверяется работоспособность не отдельных юнитов, а всей системы сразу. Это гарантирует точное взаимодействие приложения и обеспечивает правильное функционирование на разных платформах и средах.

Тестирование. E2E. Пример Puppeteer-теста

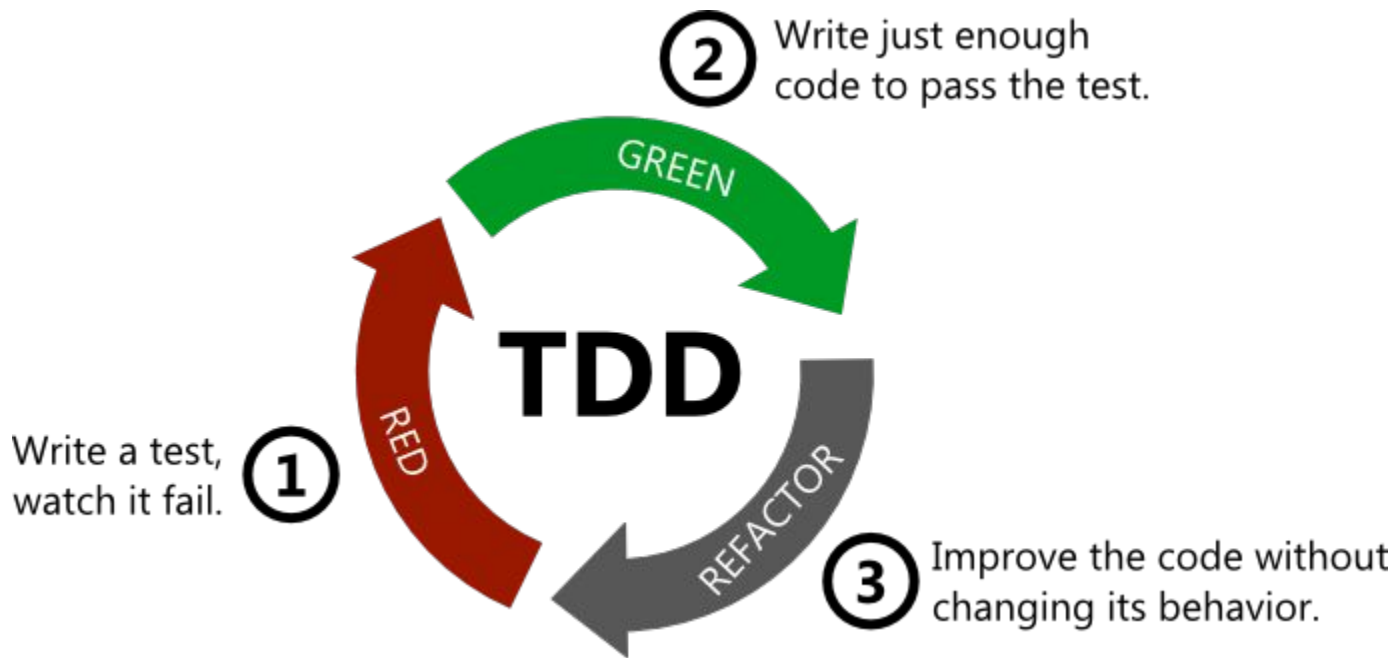


```
$ npm i jest-puppeteer --save

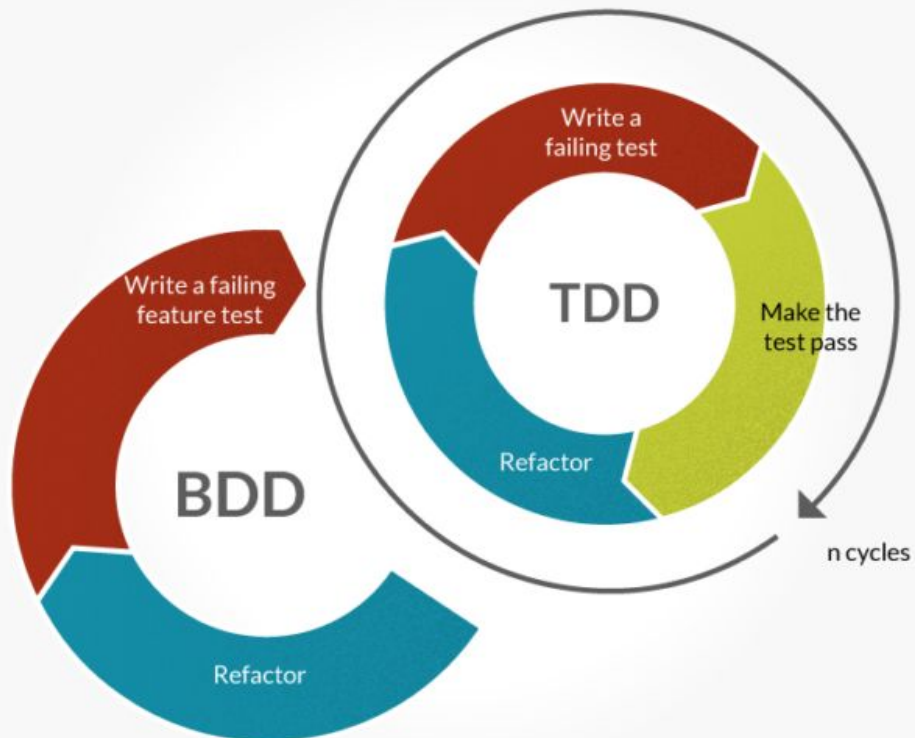
describe('Google', () => {
  beforeAll(async () => {
    await page.goto('https://google.com');
  });

  it('should be titled "Google"', async () => {
    await expect(page.title()).resolves.toMatch('Google');
  });
});
```

Тестирование. TDD



Тестирование. BDD



Тестирование. Заключение



- Автотесты пишутся людьми и не лишены ошибок
- Автотесты чувствительны к среде исполнения
- Тестирование нестабильного приложения вызывает частые падения
- Разбор логов может потребовать больше времени, чем ручной прогон тестов



Практика

Практика



- Разработать тест на один из компонентов
- Сделать мок для запроса за сообщениями
- Storybook для атомарного компонента

Домашнее задание № 1



1. Обновить CRA
2. Разработать тесты для остальных атомарных компонентов
3. Разработать Storybook stories для атомарных компонентов

Срок сдачи

27 февраля



Спасибо за внимание!