[Lesson 9]

[DAN.IT EDUCATION]

[Что мы рассмотрим:]

- Тестирование React компонентов при помощи Jest и Enzym

[DAN.IT] EDUCATION

[Тестирование]

Чаще всего тестирование в React проводят при помощи **Jest** и **Enzym**, но это не значит, что вы не можете использовать какие-либо другие утилиты для тестирования.

Jest - Test Runner
https://jestjs.io/

Enzyme - тестовая утилита для React
https://airbnb.io/enzyme/

[Shallow rendering]

Shallow rendering (*неглубокая отрисовка*) - отрисовывает только сам компонент, без дочерних компонентов. Поэтому, если что-то измените в дочернем компоненте, то это не изменит неглубокий вывод вашего компонента. И даже баг, находящийся в вашем дочернем компоненте, не изменит результат тестирования вашего компонента родителя.

[Shallow rendering]

Компонент

```
const ButtonWithIcon = ({icon, children}) => (
    <button><Icon icon={icon} />{children}</button>
);}
```

React отрисует его так:

```
<button>
    <i class="icon icon_coffee"></i>
    Hello Jest!
</button>
```

С shallow rendering вот так:

```
<button>
    <Icon icon="coffee" />
    Hello Jest!
</button>
```

[DAN.IT] EDUCATION

Тестирование снимками

Снимки Jest - это отрисованный вывод вашего компонента, хранящийся в виде текстового файла.

Вы говорите Jest, что вы хотите быть уверены в том, что вывод компонента не изменится случайно, и Jest сохраняет его в файл. После чего, при каждой сборке компонента, полученный результат будет сверяться со снимком и если что-либо изменится, вы будете проинформированны.

[Тестирование снимками]

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

./src/setupTests.js

```
// Сделаем функции Enzyme доступными во всех файлах тестов без необходимости импорта importing
import { configure, shallow, render, mount } from 'enzyme';
import Adapter from 'enzyme-adapter-react-16';
configure({ adapter: new Adapter() });
global.shallow = shallow;
global.render = render;
global.mount = mount;
// Обрушим тест при любой ошибке
console.error = message => {
   throw new Error(message);
};
```

[DAN.IT] EDUCATION

[Тестирование снимками]

./src/components/Feedback/index.spec.js

```
import React from 'react'
import Feedback from './index'

describe('Feedback form container', () => {
    it('Feedback form render', () => {
        const wrapper  = shallow(<Feedback />)
        expect(wrapper).toMatchSnapshot()
    })
})
```

[Тестирование снимками]

./src/components/Feedback/__snapshots__/index.spec.js.snap

```
ShallowWrapper {
  Symbol(enzyme.__root__): [Circular],
  Symbol(enzyme.__unrendered__): <Feedback />,
  Symbol(enzyme.__renderer__): Object {
    "batchedUpdates": [Function],
    "checkPropTypes": [Function],
    "getNode": [Function],
    "render": [Function],
    "simulateError": [Function],
    "simulateEvent": [Function],
    "unmount": [Function],
  },
  Symbol(enzyme.__node__): Object {
```

[DAN.IT] EDUCATION

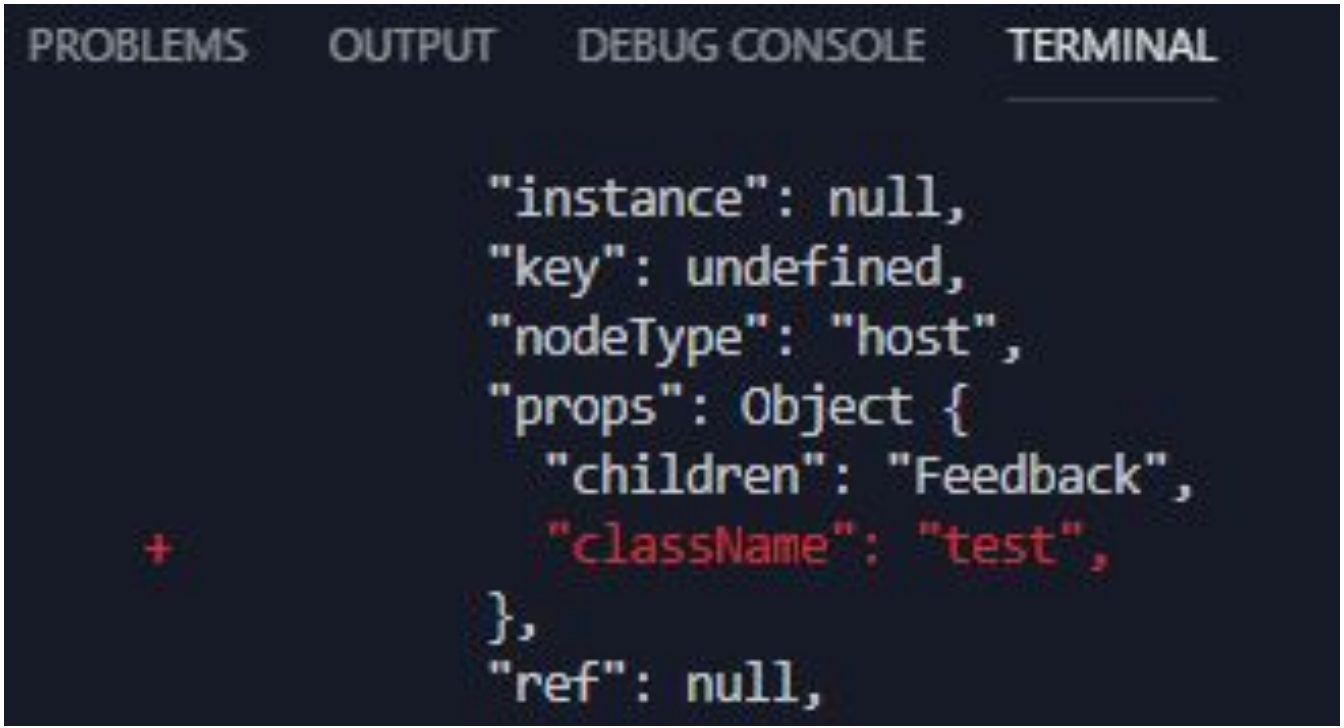
[Тестирование снимками]

./src/components/Feedback/index.js

```
render() {
    return (
      <div>
        <h1 className="test">Feedback</h1>
        <FeedbackForm onSubmit={this.submit} initialValues={this.getInitialValues()} />
      </div>
    )
  }
```

[DAN.IT] EDUCATION

[Тестирование снимками]



Тестирование снимками

./src/components/Mail/index.spec.js

```
import React from 'react'
import Mail from './index'

describe('Mail page render', () => {
    const props = {
        mails: {
            sent: [{ id: 123, subject: 'Test', text: 'Test text' }]
        },
        match: { params: { id: 123, folder: 'sent' } },
        chgMailStatus: () => {}
    }
    it('Mail page with dummy data', () => {
        const wrapper  = shallow(<Mail {...props} />)
        expect(wrapper).toMatchSnapshot()
    })
})
```