REACTJS*, JAVASCRIPT*

Краткое руководство по React JS

follower 26 января 2015 в 12:04 366k

Статья даст вам краткий обзор того, как строятся интерфейсы с помощью React JS.
Вы можете параллельно писать код с помощью starter kit, или просто продолжать читать.

Концепции

React имеет небольшой API, он прост в изучении использовании. Однако, сперва давайте взглянем на несколько концепций.

Элементы — это объекты JavaScript, которые представляют HTML-элементы. Их не существуют в браузере. они описывают DOM-элементы, такие как h1, div, или section.

Компоненты — это элементы React, написанные разработчиком. Обычно это части пользовательского интерфейса, которые содержат свою структуру и функциональность. Например, такие как NavBar, LikeButton, или ImageUploader.

JSX — это техника создания элементов и компонентов React. Например, это React-элемент, написанный на JSX:

```
<h1>Hello</h1>
```

Такой же элемент может быть написан на JavaScript:

```
React.DOM.h1(null, 'Hello');
```

С JSX требуется гораздо меньше усилий, он трансформируется в JavaScript перед запуском в браузере.

Virtual DOM — это дерево React элементов на JavaScript. React отрисовывает Virtual DOM в браузере, чтоб сделать интерфейс видимым. React следит за изменениями в Virtual DOM и автоматически изменяет DOM в браузере так, чтоб он соответствовал виртуальному.

С пониманием этих концепций мы можем продолжать использовать React. Мы напишем ряд пользовательских интерфейсов, каждый из которых будет добавлять слой функциональности к предыдущему. В качестве примера, мы напишем фото-ленту на подобии Instagram.

Рендеринг

Первым делом происходит рендер виртуального элемента (элемента, или компонента React). Помните, пока виртуальный элемент содержится только в памяти JavaScript. мы должны явно сообщить React отрисовать его в браузере.

```
React.render(<img src='http:tinyurl.comlkevsb9' />, document.body);
```

Функция *render* принимает два аргумента: виртуальный элемент и реальный узел DOM. React берёт виртуальный элемент и добавляет его в указанный узел. Теперь изображение можно увидеть в браузере.

Компоненты

Компоненты — это душа и сердце React. Это пользовательские элементы. Обычно, они имеют свою уникальную структуру и функциональность.

```
var Photo = React.createClass({

  render: function() {
    return <img src='http:tinyurl.comlkevsb9' />
  }
});

React.render(<Photo />, document.body);
```

Функция *createClass* принимает объект, который реализует функцию *render*.
Компонент **Photo** создан и отрисован в теле документа.

Этот компонент делает не больше, чем строчка в предыдущем примере, но теперь его можно дополнять функциональностью.

Свойства

Под свойствами можно понимать опции компонента. Они предоставляются в качестве аргументов компонента и выглядят так же, как атрибуты HTML.

```
var Photo = React.createClass({

  render: function() {
    return (
      <div className='photo' />
        <img src={this.props.src}>
        <span>{this.props.caption}</span>
      </div>
    )
  }
});

React.render(<Photo imageURL='http:tinyurl.comlkevsb9' caption='New
York!' />, document.body);
```

В функции *render* компоненту **Photo** переданы 2 свойства: *imageURL* и *caption*.

Внутри пользовательской функции *render*, свойство *imageURL* используется в качестве *src* для изображения. Свойство *caption* используется как текст элемента span.

Стоит отметить, что свойства компонента неизменяемы. Если у компонента есть изменяемые свойства, используйте состояние.

Состояние

Состояние — это специальный объект внутри компонента. Он

хранит данные, которые могут изменятся с течением времени

```
var Photo = React.createClass({

  toggleLiked: function() {
    this.setState({
      liked: !this.state.liked
    });
  },

  getInitialState: function() {
    return {
      liked: false
    }
  },

  render: function() {
    var buttonClass = this.state.liked ? 'active' : '';

    return (
      <div className='photo'>
        <img src={this.props.src} />

        <div className='bar'>
          <button onClick={this.toggleLiked} className={buttonClass}
/>

          </button>
          <span>{this.props.caption}</span>
        </div>
      </div>
    )
  }
});

React.render(<Photo src='http:tinyurl.comlkevsb9' caption='New
York!' />, document.body);
```

Наличия состояния у объекта вносит небольшую сложность. У компонента появляется новая фунцкия *getInitialState*. React вызывает её после инициализации компонента. В ней

устанавливается начальное состояние (что подразумевает название функции).

Также у компонента появляется функция *toggleLiked*. Она с помощью функции *setState*, переключает состояние *liked*.

Внутри функции *render*, переменной *buttonClass* присваивается значение «active», либо пустая строка, в зависимости от состояния *liked*.

buttonClass используется, как класс кнопки элемента. У кнопки также имеется обработчик события onClick, который вызывает функцию *toggleLiked*.

Вот, что происходит после отрисовки компонента:

— После нажатия кнопки вызывается *toggleLiked*
— Изменяется состояние *liked*
— React перерисовывает компонент в Virtual DOM
— Новый Virtual DOM сравнивается с предыдущим
— React изолирует изменения и обновляет их в DOM браузера

В данном случае, React изменит имя класса кнопки.

Композиция

Композиция означает комбинирование меньших компонентов при формировании большего. Например, компонент **Photo** может быть использован внутри компонента **PhotoGallery**. Примерно таким

образом:

```
var Photo = React.createClass({

  toggleLiked: function() {
    this.setState({
      liked: !this.state.liked
    });
  },

  getInitialState: function() {
    return {
      liked: false
    }
  },

  render: function() {
    var buttonClass = this.state.liked ? 'active' : '';

    return (
      <div className='photo'>
        <img src={this.props.src} />

        <div className='bar'>
          <button onClick={this.toggleLiked} className=
{buttonClass}>

          </button>
          <span>{this.props.caption}</span>
        </div>
      </div>
    )
  }
});

var PhotoGallery = React.createClass({

  getDataFromServer: function() {
    return [{
      url: 'http:tinyurl.comlkevsb9',
      caption: 'New York!'
    },
    {
      url: 'http:tinyurl.commxkwh56',
      caption: 'Cows'
    },
```

```
    {
      url: 'http:tinyurl.comnc7jv28',
      caption: 'Scooters'
    }];
  },

  render: function() {
    var data = this.getDataFromServer();

    var photos = data.map(function(photo) {
      return <Photo src={photo.url} caption={photo.caption} />
    });

    return (
      <div className='photo-gallery'>
        {photos}
      </div>
    )
  }
});

React.render(<PhotoGallery />, document.body);
```

Компонент **Photo** остался точно таким же, как и был.

Появился новый компонент **PhotoGallery**, который генерирует компоненты **Photo**. В примере используются поддельные данные сервера, которые возвращают массив из 3 объектов с url и заголовком.

В цикле создаются 3 компонента Photo, которые затем подставляются в возвращаемое значение функции *render*.

Заключение

Этого вполне достаточно для того, чтоб начать строить

пользовательские интерфейсы с использованием React. Более подробное описание есть в документации и она крайне рекомендуется к прочтению.
В руководстве также не было деталей по настройке локального окружения. Вы можете узнать это из документации, либо использовать мой boilerplate.

Оригинал статьи: The React Quick Start Guide

Похожие публикации

Честный realtime на React и Redux, как основа автоаукциона

30

jumanji2277 • 15 августа 2017 в 16:49

Практическое руководство по использованию CSS Modules в React приложениях

16

ИЗ ПЕСОЧНИЦЫ

Alexey_Solomatin • 9 августа 2017 в 13:15

Работа с периферией из JavaScript: от теории к практике

39

EFS_programm • 15 июня 2017 в 17:14

Популярное за сутки

Яндекс открывает Алису для всех разработчиков. Платформа Яндекс.Диалоги (бета)

69

BarakAdama • вчера в 10:52

Почему следует игнорировать истории основателей успешных стартапов

20

ПЕРЕВОД

m1rko • вчера в 10:44

Как получить телефон (почти) любой красотки в Москве, или интересная особенность MT_FREE

24

ИЗ ПЕСОЧНИЦЫ

cab404 • вчера в 20:27

Java и Project Reactor

10

zealot_and_frenzy • вчера в 10:56

Пользовательские агрегатные и оконные функции в PostgreSQL и Oracle

6

erogov • вчера в 12:46

Лучшее на Geektimes

Как фермеры Дикого Запада организовали телефонную сеть на колючей проволоке

31

NAGru • вчера в 10:10

Энтузиаст сделал новую материнскую плату для ThinkPad X200s

49

alizar • вчера в 15:32

Кто-то посылает секс-игрушки с Amazon незнакомцам. Amazon не знает, как их остановить

85

Pochtoycom • вчера в 13:06

Илон Маск продолжает убеждать в необходимости создания колонии людей на Марсе

139

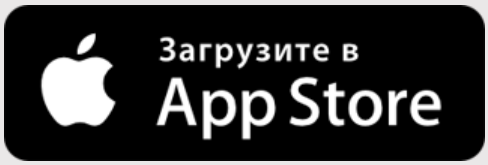
marks • вчера в 14:19

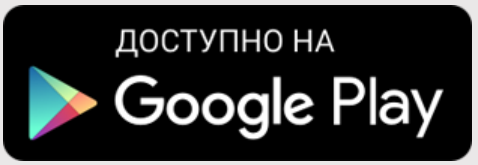
Дела шпионские (часть 1)

16

TashaFridrih • вчера в 13:16

Мобильное приложение





Полная версия