Введение в React

№ урока:	1	**Курс:**	Введение в React

Средства обучения: Текстовый редактор или IDE, браузер

Обзор, цель и назначение урока

В данном уроке, мы рассмотрим историю возникновения React, его преимущества, разберемся с тем, как работает виртуальный DOM, напишем свой первый React компонент, научимся обрабатывать события и напишем простое приложение на React.

Изучив материал данного занятия, учащийся сможет:

- Строить простые компоненты на React
- Компоновать компоненты в приложении
- Использовать JSX для рендеринга компонентов

Содержание урока

1. Немного о курсе
2. Что такое React?
3. Когда и почему стоит использовать React
4. Что такое виртуальный DOM и как он работает
5. Рендеринг данных в React - JSX!
6. Первый компонент - Hello world.
7. Props у компонентов
8. State компонентов, методы getInitialState и setState
9. Пример. Отображение списка контактов
10. Обработка событий в React, синтетические события
11. Пример. Динамический поиск по списку контактов

Резюме

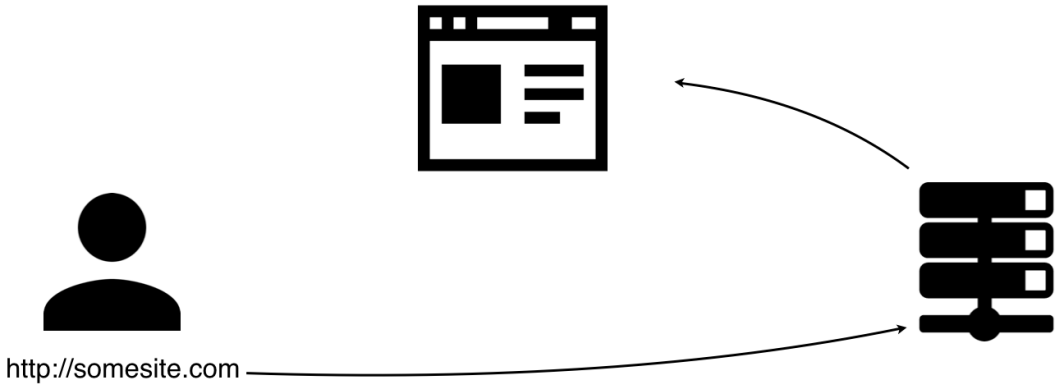
Что такое React?

ReactJS – это JavaScript фреймворк для построения пользовательских интерфейсов. Это не MVC фреймворк. К нему можно применить только V из этой аббревиатуры. Такая узкая сфера применения дает свободу использования React в различных системах в комбинации с другими библиотеками.

React был представлен Facebook в 2013 году, и очень быстро обрел популярность. Сегодня его используют многие известные компании включая Instagram, Airbnb, Ebay, Netflix, Yahoo и другие.

Основным отличием React от других JavaScript фреймворков является то, как он управляет состоянием приложения. Если вспомнить, как пользователи взаимодействовали с веб-страницами еще 10-15 лет назад, то увидим такую картину:

CyberBionic systematics
CyberBionic Systematics ® 2016
19 Mariny Raskovoy Str., 5 floor
Kyiv, Ukraine
t. +380 (44) 361-8473
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com



Сервер всегда возвращал статическую страницу, и реакцией на действия пользователя была полная перезагрузка страницы. Преимуществами такого подхода, была простота в реализации и понимании, недостатками – скорость работы, отзывчивость, UX и потеря состояния при каждой перезагрузке.

Все очень изменилось с появлением AJAX, это подход к построению интерактивных веб-приложений, заключающийся в «фоновом» обмене данными браузера с веб-сервером. То есть, в фоновом режиме отправляются запросы на сервер, приходят с него ответы, изменяется состояние приложения и, соответственно, внешний вид. Именно такой подход породил понятие Single Page Application.

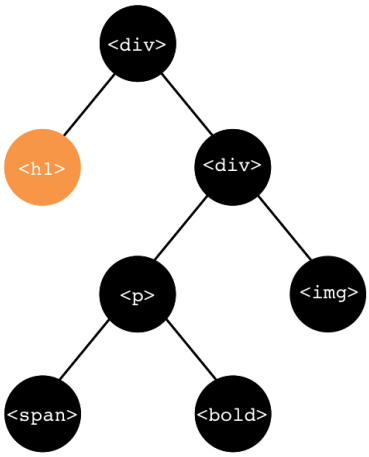
Но каждое визуальное изменение на странице соответствует изменению ее DOM дерева. Не секрет, что все манипуляции с DOM деревом являются очень ресурсоемкими операциями, т.к. изначально DOM дерево было статическим и никакой динамики не предусматривало.

CyberBionic systematics

CyberBionic Systematics ® 2016
19 Mariny Raskovoy Str., 5 floor
Kyiv, Ukraine

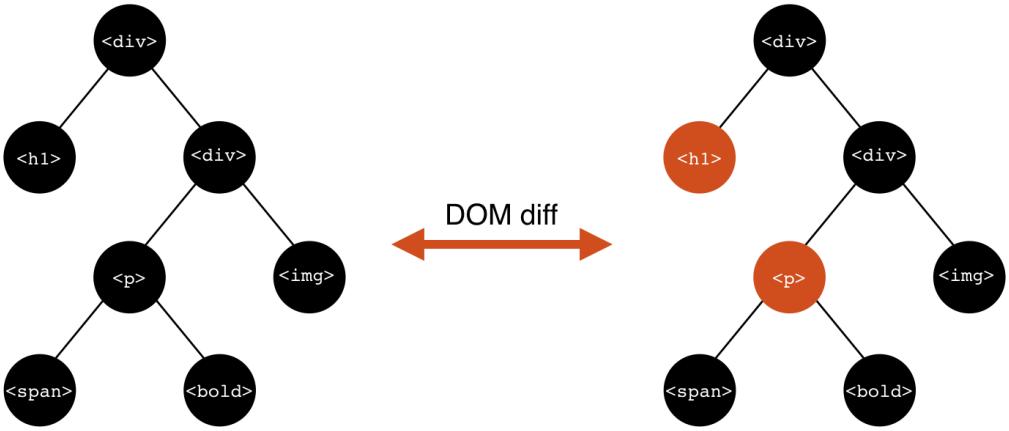
t. +380 (44) 361-8473
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

Страница



DOM дерево

Именно поэтому в React используется виртуальный DOM. Это такая легковесная копия реального DOM дерева на Javascript. Таким образом, React манипулирует не с реальным



До

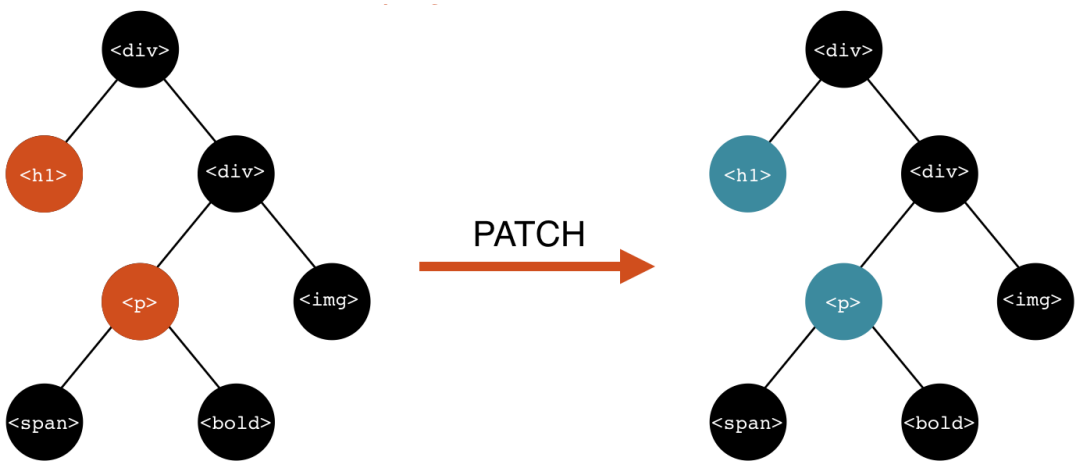
После

(синоним – медленным) DOM деревом, а с виртуальным.
Он сравнивает предыдущее состояние виртуального DOM дерева с его следующим состоянием и находит минимальное количество манипуляций, которые можно произвести уже с реальным DOM, чтобы обновить вид приложения согласно его новому состоянию.

CyberBionic systematics

CyberBionic Systematics ® 2016
19 Mariny Raskovoy Str., 5 floor
Kyiv, Ukraine

t. +380 (44) 361-8473
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com



Виртуальный DOM

Реальный DOM

И это действительно быстро работает. А все что вам нужно делать - это просто менять состояние вашего приложения, а все остальное React сделает уже за вас!

JSX

Для рендеринга данных в React используется JSX. JSX нужен для JavaScript XML — разметки в стиле XML внутри компонентов React. React работает и без JSX, но именно JSX поможет сделать ваши компоненты более читаемыми, поэтому рекомендуется использовать его.

```
// с JSX
var app = <Nav color="blue">
    <Profile>click</Profile>
</Nav>;

// без JSX
var app = React.createElement(
    Nav,
    {color:"blue"},
    React.createElement(
        Profile,
        null,
        "click"
    )
);
```

JSX позволяет вам описывать структуру компонентов с помощью понятного синтаксиса, а затем все написанное вами преобразуется в цепочку javascript функций.

В JSX можно использовать переменные, условные конструкции и вызывать функции. Для этого нужно использовать фигурные скобки.

```
// переменная
var myName = 'Katya';
var app = <h1> My name is {myName}! </h1>;

// результат вызова функции
function getMyName() {
    return 'Katya';
}
```


CyberBionic systematics

CyberBionic Systematics ® 2016
19 Mariny Raskovoy Str., 5 floor
Kyiv, Ukraine

t. +380 (44) 361-8473
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

```
var app = <h1>
    My name is {getMyName()}!
</h1>;

// условная конструкция – тернарный оператор
var age = 20;

var app = <h1>
    Hi! { age > 18 ? 'Your age is more than 18!' : 'Your age is less than
18!' }
</h1>;
```

Компоненты

React использует компонентную модель, то есть все большое приложение делится на небольшие независимые компоненты, которыми гораздо легче управлять.

```
// простой компонент
var HelloWorld = React.createClass({
    render: function() {
        return (
            <h1> Hello world! </h1>
        );
    }
});
```

У каждого компонента есть один обязательный метод - render, который возвращает JSX разметку, соответствующую виду компонента.

Параметры aka props

Каждый компонент может принимать параметры. Они передаются из выше стоящих компонентов. К параметрам компонента можно обращаться используя this.props.propName.

```
var Heading = React.createClass({
    render: function() {
        return <h1> My name is {this.props.name}!</h1>;
    }
});

var Hello = React.createClass({
    render: function() {
        return <Heading name="Katya" />;
    }
});
```

Запомните: Модифицировать *this.props* крайне нежелательно!

Состояние aka state

Также, каждый компонент может хранить свое состояние. К нему можно получить доступ, обратившись к this.state. В состоянии компонента стоит хранить данные, от которых компонент напрямую зависит внешний вид компонента, и при изменении которых его внешний вид тоже должен меняться (будет вызываться метод render). Состояние компонента доступно только внутри самого компонента. Для объявления начального состояния компонента, нужно использовать метод getInitialState. Этот метод вызывается до того, как компонент отобразиться в доме и определяет первоначальное значение состояния компонента. Для того, чтобы модифицировать состояние компонента нужно вызвать метод this.setState({ /* новое состояние */ }), тогда состояние компонента измениться и вызовется метод render.

CyberBionic Systematics ® 2016
19 Mariny Raskovoy Str., 5 floor
Kyiv, Ukraine
t. +380 (44) 361-8473
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

```
var Component = React.createClass({
  getInitialState : function() {
    return {
      name : "Katya"
    };
  },

  handleClick : function() {
    this.setState({
      name : "Vasya"
    });
  },

  render : function() {
    return <div onClick={this.handleClick}>
      Hello, {this.state.name}
    </div>;
  }
});
```

Состояние нужно использовать только там, где это действительно необходимо!

НИКОГДА не нужно модифицировать *this.state* напрямую!

Обработка событий

Если вы захотите сделать свои компоненты динамическими, то вам не обойтись без использования событий. Обычно, для каждого события описывается обработчик, в котором вы можете произвести какие-то действия.

```
var HelloComponent = React.createClass({
  handleClick : function() {
    alert('Hello stranger!');
  },

  render : function() {
    return <button onClick={this.handleClick}> Say hello </button>;
  }
});
```

Во все такие обработчики событий в качестве аргумента приходит объект *SyntheticEvent*. Это объект, который является кроссбраузерной оберткой над стандартными событиями.

```
var EventComponent = React.createClass({
  handleClick : function(event) {
    alert('Event handled - ' + event.type); // Event handled - click
  },

  render : function() {
    return <button onClick={this.handleClick}> Click me </button>;
  }
});
```

О том, какие события поддерживаются можно прочитать в официальной документации.

CyberBionic systematics

CyberBionic Systematics ® 2016
19 Mariny Raskovoy Str., 5 floor
Kyiv, Ukraine

t. +380 (44) 361-8473
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

Закрепление материала

1. Как работает виртуальный DOM?
2. Как можно создать компонент? Какая функция используется для этого?
3. Какой метод является обязательным для каждого компонента?
4. Как использовать JavaScript конструкции внутри JSX?
5. Зачем нужны параметры у компонентов?
6. Как можно обратиться к параметрам компонента?
7. Как изменить состояние компонента?
8. Как определить первоначальное состояние компонента?
9. Зачем нужен key в React?

Самостоятельная деятельность учащегося

Задание 1: Сделать компонент для отображения статьи

Уровень сложности: низкий

В компонент передается заголовок, имя автора и текст, а он их отображает, как статью. Внешний вид на ваше усмотрение.

Задание 2: Написать динамический Hello World.

Уровень сложности: низкий

В поле ввода вводится текст. Снизу надпись, которая отображает “Hello, <введенный текст>!”. Текст должен меняться по мере ввода. Если в поле ничего не введено, то должно быть выведено “Hello, stranger!”.

Задание 3: Дополнительная информация о контактах

Уровень сложности: средний

Усовершенствовать список контактов, написанный на уроке. При клике на контакт, он должен разворачиваться (увеличиваться в высоте) и должна отображаться какая-то дополнительная информация о контакте (например, адрес и email). При повторном клике, информация должна сворачиваться.

Задание 4: Простой калькулятор

Уровень сложности: выше среднего

В два поля вводится числа a и b. Нужно написать простой калькулятор, который умеет их складывать, вычитать и умножать. Отображение на ваше усмотрение.

Рекомендуемые ресурсы

Все примеры из урока и пояснения к домашним заданиям можно найти тут:
https://github.com/krambertech/react-essential-course

Официальная документация React - http://facebook.github.io/react/docs

Советую ознакомиться с JavaScript Style Guide - https://github.com/airbnb/javascript

CyberBionic systematics

CyberBionic Systematics ® 2016
19 Mariny Raskovoy Str., 5 floor
Kyiv, Ukraine

t. +380 (44) 361-8473
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com