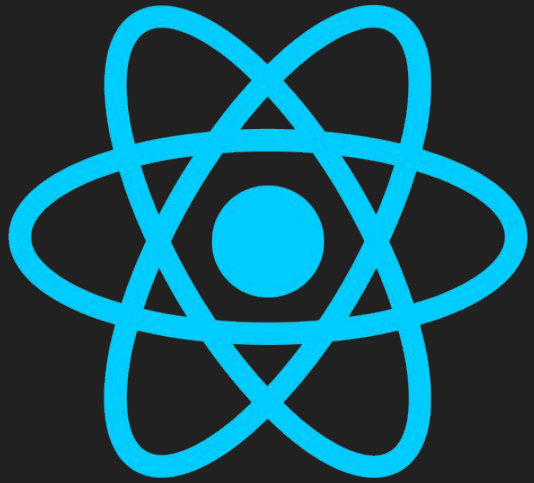


React

JavaScript-библиотека для создания пользовательских интерфейсов

Что такое React

React - это JavaScript библиотека, разработанная Facebook*, которая позволяет создавать пользовательские интерфейсы веб-приложений. Она позволяет вам собирать сложный UI из маленьких изолированных кусочков кода, называемых «компонентами».

```
class ShoppingList extends React.Component {
  render() {
    return (
      <div className="shopping-list">
        <h1>Список покупок для {this.props.name}</h1>
        <ul>
          <li>Instagram</li>
          <li>WhatsApp</li>
          <li>Oculus</li>
        </ul>
      </div>
    );
  }
}
```

“Shopping list” - Пример классового компонента

*-запрещен в РФ; принадлежат корпорации Meta, которая признана в РФ экстремистской

В чем плюсы использования

1. React использует виртуальную DOM-модель для обновления только измененных элементов на странице, вместо обновления всего DOM-дерева. Это позволяет улучшить быстродействие веб-приложений.

2. React использует компоненты для организации кода. Вместо того чтобы писать один огромный скрипт, вы создаете множество небольших, многоразовых компонентов. Это делает код чище, более организованным и легким для сопровождения.

3. React применяет односторонний поток данных, что означает, что данные (state) движутся только в одном направлении, от родительского компонента к дочерним. Это упрощает отслеживание и управление состоянием приложения, что облегчает отладку и предотвращает неоднозначности данных.

4. React имеет огромное сообщество разработчиков по всему миру, что означает, что вы получаете поддержку, документацию, учебные ресурсы и ряд сторонних библиотек, инструментов и плагинов.

5. React может использоваться для разработки не только веб-приложений, но и мобильных приложений с использованием React Native.

Какие компоненты есть в React

1. Простыми словами, функциональные компоненты - это функции JavaScript. Написав функцию JavaScript, мы можем создать функциональный компонент в React. Чтобы сделать приложение React эффективным, мы используем функциональный компонент только тогда, когда уверены, что нашему компоненту не нужно взаимодействовать с другими компонентами. Функциональные компоненты не требуют данных от других компонентов. Ниже показан пример функционального компонента:

```
function Welcome(props) {
  return <h1>Привет, {props.name}</h1>;
}
```

2. Компоненты класса похожи на функциональные компоненты, но имеют некоторые дополнительные возможности, которые делают компоненты класса немного сложнее функциональных компонентов. Функциональные компоненты не заботятся о других компонентах вашего приложения, в то время как компоненты класса могут работать друг с другом. Мы можем передавать данные от одного компонента класса к другому компоненту класса. Ниже показан пример компонента класса в React:

```
class Welcome extends React.Component {
  render() {
    return <h1>Привет, {this.props.name}</h1>;
  }
}
```

Какие компоненты есть в React

3. Компонент высшего порядка (Higher-Order Component, HOC) — это один из продвинутых способов для повторного использования логики. HOC не являются частью API React, но часто применяются из-за композиционной природы компонентов. Говоря просто, компонент высшего порядка — это функция, которая принимает компонент и возвращает новый компонент.

```
const hoc = (MyComponent) => (props) => {
  return (
    <div>
      <MyComponent {...props}>
        {props.children.toUpperCase()}
      </MyComponent>
    </div>
  )
}
```

4. Презентационные компоненты часто называют функциональным компонентом без состояния, который принимает параметры и отображает пользовательский интерфейс. Функциональные компоненты без состояния - это обычные функции JavaScript, которые не имеют состояния. Компоненты, которые получают состояние от компонента более высокого порядка, будут работать как презентационные компоненты. Состояние передается им, и они условно отображают пользовательский интерфейс на его основе. Ниже показан пример презентационного компонента в React:

```
const List = props => (<ul>
  {props.list.map(user => (<li>{items}</li>))}
</ul>)
```

Какие компоненты есть в React

5. Компонент без состояния гораздо эффективнее, чем компонент с состоянием, поскольку для его рендеринга не требуется так много компьютерных ресурсов (память, CPU и GPU в приложениях с интенсивной графикой). Ниже показан пример такого компонента в React:

```
export default Title => () => (
  <h1>I am Title</h1>
);
```

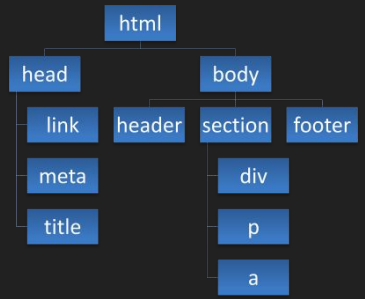
6. Умные компоненты - это компоненты с состоянием и работают аналогично компонентам класса. При работе с Babel или React в стиле ES6 мы привыкли понимать под этим любой объект класса, который расширяет Component, также компонент может быть записан и в функциональном стиле. Ниже показан пример умного компонента в React, где `Other.Component` заботится о состоянии компонента:

```
export default title => class MyComponent extends Other.Component {
  render() {
    return (
      <div>
        <h1>Your Cart</h1>
        <ShoppingCart cart={ this.state.cart } onItemRemove={ this.removeCartItem } onItemChangeQty={ this.changeCartItemQty } />
        <ShippingInfo cart={ this.state.cart } address={ this.state.userShippingAddress }/>
        <BuyNowButton cart={ this.state.cart } onBuyClick={ this.cartPurchased } />
      </div>
    );
  }
}
```

Что такое DOM

DOM — это объектная модель документа, которую браузер создает в памяти компьютера на основании HTML-кода, полученного им от сервера. Иными словами, это представление HTML-документа в виде дерева тегов.

Виртуальная DOM (Virtual DOM) - это концепция, используемая React, которая позволяет эффективно обновлять и отображать пользовательский интерфейс веб-приложений. Вместо непосредственного изменения реального DOM (Document Object Model) браузера, React создает виртуальное представление DOM в памяти, которое называется виртуальной DOM.



Пример DOM-дерева

Как происходит обновление данных на сервере в React

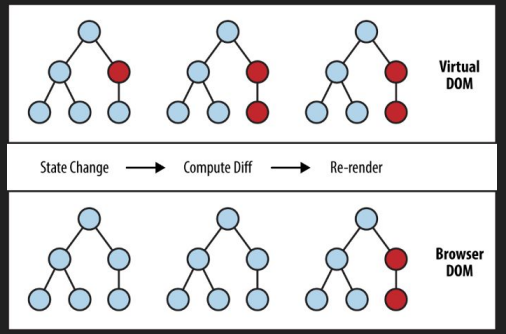
1. Изначально React создает виртуальное представление всего пользовательского интерфейса в виде виртуальной DOM, используя компоненты и их состояния.

2. Когда происходят изменения данных (например, при обновлении состояния компонента), React сравнивает новую виртуальную DOM с предыдущей.

3. React вычисляет разницу между новым и предыдущим состоянием виртуальной DOM, называемую изменениями (diffing).

4. Затем React обновляет только те части реального DOM, которые действительно изменились, основываясь на вычисленных изменениях виртуальной DOM. Это делается с использованием минимального количества операций обновления DOM, что приводит к оптимизации производительности веб-приложения.

5. Наконец, React обновляет виртуальную DOM с показанным состоянием, готовым для следующих изменений.



наглядный пример

Какие плюсы по сравнению с аналогами (Vue.js)

1. Синтаксис: React и Vue.js имеют различный синтаксис. React использует JSX (JavaScript XML), предоставляя возможность писать компоненты, объединяя JavaScript и HTML-подобный синтаксис. В Vue.js используется шаблонный синтаксис, который представляет собой комбинацию HTML и директив Vue.js.

2. Размер и вес библиотеки: React включает только необходимый минимум, чтобы создавать компоненты пользовательского интерфейса, в то время как Vue.js включает больше функциональности по умолчанию, таких как система маршрутизации и управление состоянием. Это делает Vue.js независимым и легковесным по сравнению с React.

3. Компонентная модель: Обе библиотеки предлагают компонентную модель разработки, но с некоторыми отличиями. В React компоненты имеют более явную и гибкую структуру, что может быть полезно для больших проектов. В Vue.js компоненты более масштабируемы и создаются с использованием единого объекта-конфигурации.

4. Реактивность данных: Vue.js имеет встроенную реактивность, что означает, что изменение данных автоматически обновляет связанные с ними элементы интерфейса. React требует явного определения обновлений и перерисовки компонентов при изменении состояния.

5. Экосистема и сообщество: Оба фреймворка имеют развитые экосистемы, но React, разработанный Facebook*, на сегодняшний день имеет более широкое сообщество и больше сторонних библиотек и инструментов.

*-запрещен в РФ; принадлежат корпорации Meta, которая признана в РФ экстремистской

Какие плюсы по сравнению с аналогами (AngularJS)

1. Рендеринг: В React используется виртуальная DOM, которая обновляет только измененные элементы интерфейса. В Angular используется реальный DOM, и обновления происходят после каждого изменения данных. React, таким образом, обеспечивает более эффективное обновление и лучшую производительность в случае частых изменений данных.

2. Язык: React разрабатывается на JavaScript, тогда как Angular использует TypeScript, расширение JavaScript с добавлением статической типизации. Использование TypeScript в Angular может способствовать более строгой и надежной разработке.

3. Архитектура и компоненты: React предлагает более гибкую компонентную модель с использованием функциональных и классовых компонентов. Angular предлагает строгую и полноценную модель компонентов с использованием классов.

4. Синтаксис и разработка: React использует JSX - комбинацию JavaScript и XML-подобного синтаксиса, которая может быть непривычной для разработчиков, не знакомых с ней. Angular использует шаблонный синтаксис, более близкий к обычному HTML-коду.

5. Размер и вес: React считается более легковесным и имеет меньший размер по сравнению с Angular, что позволяет легче встраивать React в существующие проекты или использовать его с другими библиотеками.

Спасибо за внимание