Секция «Информационные технологии»

Хук состояния useState, React

Гишлакаев Сайфулла Умарович
Студент (бакалавр)
Чеченский государственный университет, Факультет информационных технологий,
Грозный, Россия
E-mail: sgishlakaev@mail.ru

Разберем на примере простого счетчика как работает хук состояния useState.

Для начала чтобы хук работал нам необходимо импортировать библиотеку классов React.

```
import React, { useState } from 'react'; // импорт библиотекb классов React.

function App() {
  // Объявляем хук с двумя переменными
  const [count, setCount] = useState(0);

  return (
    <div>
      <p>На счетчик было нажато{count} раз(а)</p>
       // тут делаем кнопку и передаем ей функцию которую мы создали выше,
       // и setCount функция в свою очередь меняет count, при вызове
      <button onClick={() =>
setCount(count + 1)
}>
        Кликни на меня
      </button>
    </div>
  );
}
```

Это был функциональный подход, давайте разберём пример на классах, при работе обычно классы не используются, но для понимания общей работы в начале все же рекомендуется познакомился и с классами. Классовый подход при обучении позволяет лучше понять в общем всю картину.

```
      class App extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      count: 0
    };
  }

  render() {
    return (
      <div>
        <p>Вы кликнули {this.state.count} раз(а)</p>
```

```
        <button onClick={() => this.setState({ count: this.state.count + 1 })}>
          Нажми на меня
        </button>
      </div>
    );
  }
}
```

При первом рендеринге приложения count будет равно 0, но при каждом каждом клике ао кнопке <button>, будет срабатывать метод из HTML onClick, внутри которого мы и передаём this.setState, с указанием логики. this.setState({ count: this.state.count + 1 }) Логика тут простая, мы просто увеличиваем count, на единицу, каждый раз при клике по <button>.

Так что же хук конкретно? Хук - это такая функция, с помощью которой мы можем зацепиться либо хукнуться за функциональные возможности React. В переводе с английского Hook - это крюк, зацепка. Хук который мы рассматриваем позволяет на зацепиться за функционал useState. Эта хук состояния с помочью него можно предоставлять доступ компонентам к состоянию.

Давайте разберем в каких ситуациях и где лучше всего использовать useState хук состояния. Обычно, когда программисты разрабатывали функциональный компонент и у них возникала необходимость прицепить к компоненту состояние, то им необходимо было переписывать функциональный компонент под класс. С последним обновлением React вы можете использовать хуки состояния прямо в функциональном компоненте. И через них навешивать состояния на что угодно.

Благодаря useState можно не напрягаясь перерендарить необходимые участки приложения, которые подверглись изменению, эта особенность в значительной степени увеличивает производительность и скорость работы приложения. Нам не нужно обновлять всё приложение, а только необходимые куски приложения.

Допустим вы хотите создать переменную состояния, допустим это будет count значение которого 0. Если мы это делаем в классе, то нужно внутри конструктора присвоить state обьект под названием которое вы хотите, для нас это будет { count: 0 }

Например

```
class App extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      count1: 0
// тут count1 превращается в переменную состояния
    };
  }
```

Если бы мы работали в функции то в ней нам не был бы доступно ключевое слово this, поэтому для функционального подхода данный метод инициализации не подойдет. Мы не можем в функциональном компоненте считать состояние таким образом this.state. В функионаьном компоненте мы просто вызываем сразу хук useState, прямо без каких либо this.

Например:

```
import React, { useState } from 'react';
```

```
function App() {
  // Декомпозиция useState и объявление переменной состояния
  const [count, setCount] = useState(0);
```

Разберем же конкретно делает useState. Когда мы его вызываем мы прописываем переменную и функцию для декомпозиции. Переменная инициализируется, например, count, либо подругому можно назвать эту переменную, но разберем наш пример выше. И есть функция setCount которая при вызове в дальнейшем в коде, будет менять значение count, и перерендерит элемент внутри которого функция указана. Проще говоря useState - это обновленный способ использования классового подхода this.state. В программировании распространено что при вызове функции переменные исчезают которые были внутри него, но с переменная состояния работает иным образом, она сохраняется внутри React.

Давайте разберем какие аргументы мы передаем конкретно в саму функцию useState(). По сути мы можем передать в useState(0) число, массив useState([]), объект useState({}). Проще говоря тот тип данных которые мы укажем в useState() по сути и определяет тип данных для count. В классовом подходе с использованием this.state, мы всегда обязаны передавать объект.

Источники и литература

1) Николас З. ECMAScript 6 для разработчиков. 2017 г.
2) Прасти Н. Введение в ECMAScript 6. 2016г.
3) Марейн Х. Выразительный JavaScript. Современное веб-программирование. 3-е издание. 2020 г.
4) Рыбаков Е. JavaScript и Node.js для Web-разработчиков. 2022 г.
5) Черный Б. Профессиональный TypeScript. Разработка масштабируемых JavaScript-приложений. 2021 г.