React в контексте JavaScript

Полезные ссылки

https://learn.javascript.ru — бесплатный учебник по JavaScript

По теме React и JavaScript:

- React Docs: Conditional Rendering
- React Docs: Rendering Lists

По теме Prettier:

- Официальный сайт Prettier — документация и параметры настройки.
- Расширение Prettier для VSCode (установите обязательно!).

React — это "Just JavaScript"

- Важное понимание: **React не изобретает новый язык программирования.**
- Всё, что вы пишете в React-компонентах (кроме JSX-разметки) — это чистый JavaScript.
- Чем лучше вы знаете JS (массивы, объекты, функции, методы map, filter), тем мощнее и элегантнее будет ваш код на React.

"React — это просто JavaScript с библиотекой для рендеринга."

JSX — синтаксический сахар

- **JSX (JavaScript XML)** — это расширение синтаксиса JavaScript, которое позволяет писать HTML-подобную разметку прямо в JS-файлах.
- Браузер не понимает JSX. Инструменты сборки (Vite, Babel) превращают его в обычные вызовы `React.createElement()`.
- *Важно:* Внутри фигурных скобок { } в JSX можно писать любые JavaScript-выражения.

```jsx
const name = "студент";
const element = <h1>Привет, {name}!</h1>; // name - это JS
```

Как JavaScript работает внутри React-компонента

- Компонент — это функция, которая возвращает JSX.
- Внутри функции можно использовать весь арсенал JS:

```jsx
function WelcomeMessage() {
  // 1. Объявляем переменные
  const isLoggedIn = true;
  const userName = "Студент";
  // 2. Используем условную логику
  if (!isLoggedIn) {
    return <p>Пожалуйста, войдите</p>;
  }
  // 3. Возвращаем JSX с вставками JS
  return <h2>Добро пожаловать, {userName}!</h2>;
}
```

Практикум (JavaScript → React)

Лабораторная работа 2_4
«Создание массива идей»

Шаг 1: Создаем массив идей в App.tsx

- Создаем массив объектов с данными об идеях
- Каждый объект содержит свойства:
  - никнейм автора,
  - название и
  - описание

```
// Создаем массив идей
const ideas = [
  {
    nick: 'user1',
    name: 'Идея 1',
    description: 'Описание первой идеи'
  },
  // добавляем остальные объекты...
];
```

- Ниже используем *map()* для динамического создания элементов
- Важно указать уникальный *key* для каждого элемента

```
// Используем map для создания элементов
{ideas.map((idea, index) => (
  <div key={index} className="idea-card">
    <h2>{idea.name}</h2>
    <p>{idea.description}</p>
    <p><strong>Автор:</strong> {idea.nick}</p>
  </div>
))}
```

Шаг 2: Стилизация компонентов

- Открываем файл стилей с расширением *.module.css*
- Используем локальные классы, которые автоматически получают уникальные имена
- Подключаем стили в компоненте через импорт (если не подключен)
- Применяем стили к элементам через пропс *className*

```css
/* Стили для карточки идеи */
.idea-card {
  background-color: #f9f9f9;
  padding: 15px;
  border-radius: 8px;
  box-shadow: 0 2px 4px rgba(0,0,0,0.1);
}

/* Стили для заголовков и текста */
.idea-card h2 {
  margin: 0 0 10px;
  color: #333;
}

.idea-card p {
  margin: 5px 0;
  color: #666;
}
```

```
import styles from './App.module.css’;

…

<div key={index} className={styles.ideaCard}>
```

Лабораторная работа 2_5

«Объявление и вывод массива»

Лабораторная работа: Объявление и вывод массива

Цель: Научиться встраивать JavaScript-структуры данных в React-компоненты и выводить их на экран.

Задача:

Объявить массив данных с помощью JavaScript.

Встроить его в React-компонент.

Вывести элементы массива в виде списка на страницу.

Пошаговый алгоритм выполнения (Часть 1)

1. Создайте проект vite + React + JS (через `npm create vite@latest`), в название проекта добавьте свою фамилию.
2. Очистите `App.jsx` от лишнего. Оставьте базовую структуру.
3. **Объявите массив данных** внутри компонента `App` (до `return`):

```jsx
function App() {
  // Массив с данными (чистый JavaScript)
  const students = [
    { id: 1, name: 'Иван', group: 'ФИиТ-11' },
    { id: 2, name: 'Мария', group: 'ФИиТ-12' },
    { id: 3, name: 'Алексей', group: 'ФИиТ-11' }
  ];

  return ( ... )
}
```

Пошаговый алгоритм выполнения (Часть 2: Вывод в JSX)

4. **Выведите массив** в JSX, используя метод `map()`.

* Помните: в JSX можно вставлять JS выражения внутри `{ }`.
* Для списков React требует уникальный атрибут `key`.

```
  return (
    <div>
      <h1>Список студентов ФИиТ</h1>
      <ul>
        {students.map((student) => (
          <li key={student.id}>
            {student.name} — группа {student.group}
          </li>
        ))}
      </ul>
    </div>
  );
}
```

Ожидаемый результат

- После сохранения файла (браузер с Vite обновится автоматически), вы должны увидеть:
  - Заголовок "Список студентов ФИиТ"
  - Маркированный список:
    - Иван — группа ФИиТ-11
    - Мария — группа ФИиТ-12
    - Алексей — группа ФИиТ-11
- **Важно:** Если вы измените массив (добавите/удалите студента), интерфейс обновится автоматически. Это и есть реактивность React.