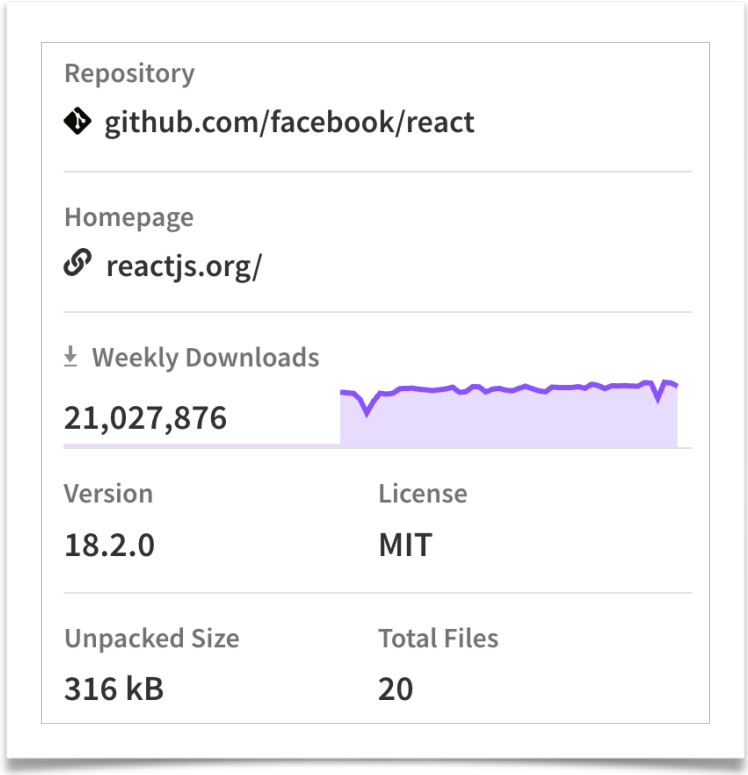
Веб-программирование

React

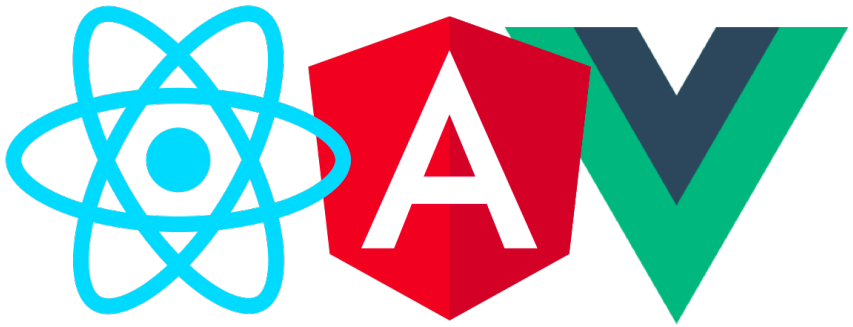
Давайте знакомиться

React – JavaScript **библиотека** для разработки пользовательских интерфейсов.

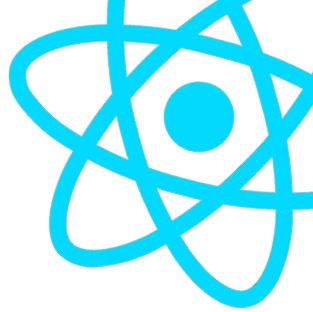


Преимущества

- **Низкий порог входа**
- …
- …
- …
- …

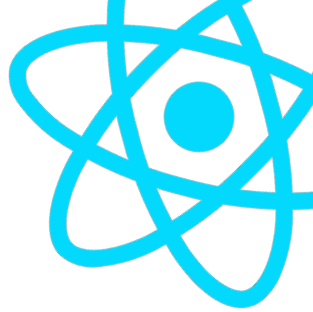


Преимущества



- Низкий порог входа
- **Легко добавить в уже готовый проект**
- …
- …
- …

Преимущества



- Низкий порог входа
- Легко добавить в уже готовый проект
- **Декларативный подход**
- …
- …

Императивный / Декларативный

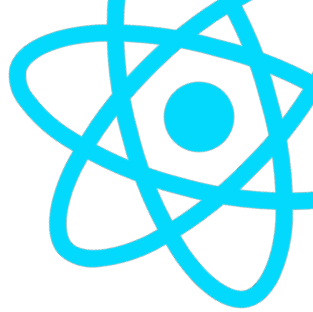
Стили программирования (подходы):

- Императивный — "Как делать?"
- Декларативный — "Что делать?"

React использует **декларативный подход**

```
// Как?
const sumImperative = (numbers) => {
    let result = 0;

    for (let i = 0; i < numbers.length; i++) {
        result += numbers[i];
    }

    return result;
};

// Что?
const sumDeclarative = (numbers) => {
    return numbers.reduce((acc, number) => acc + number, 0);
};
```

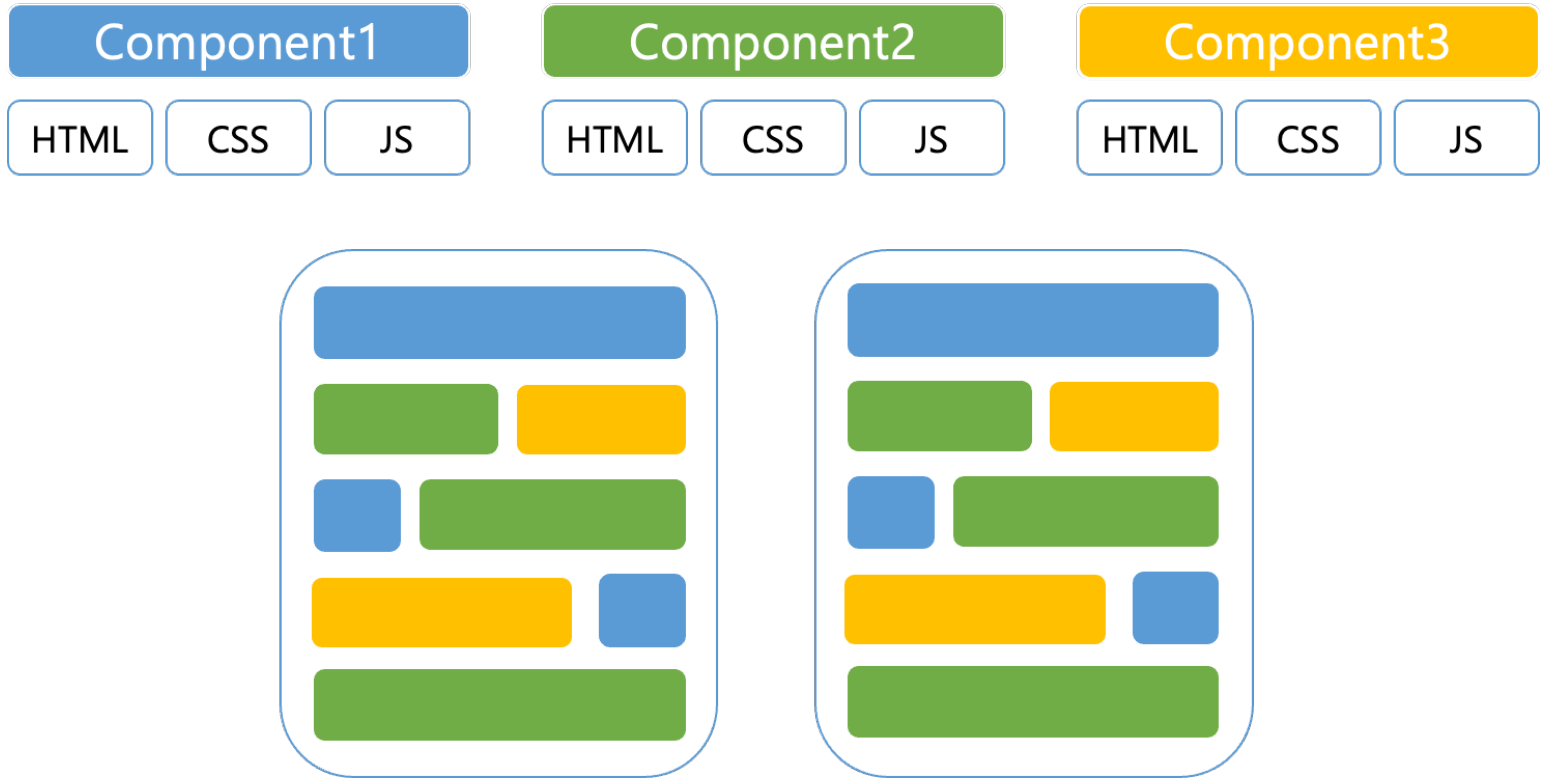
Преимущества



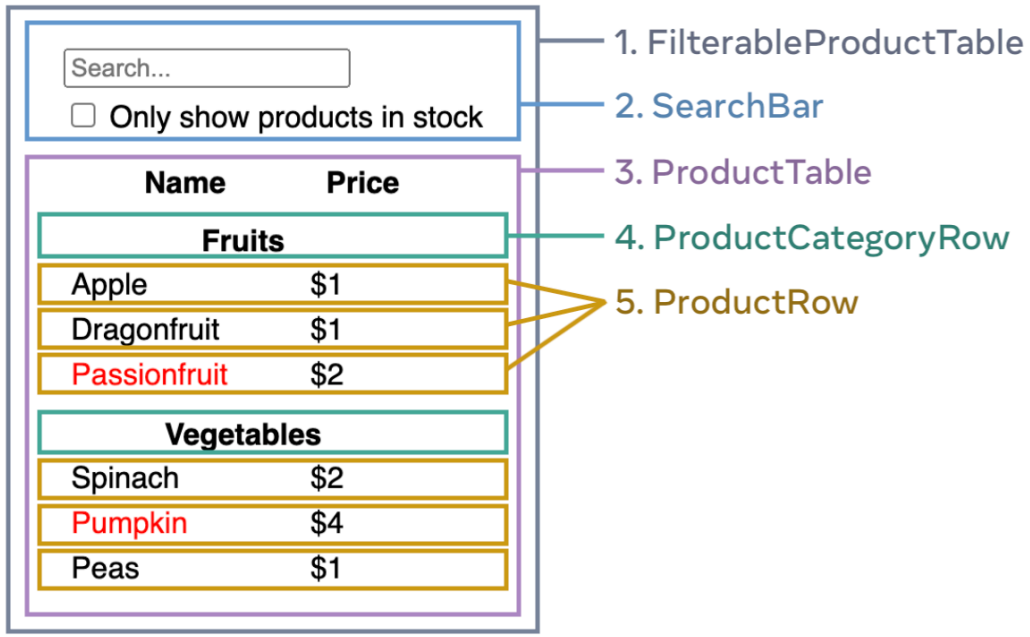
- Низкий порог входа
- Легко добавить в уже готовый проект
- Декларативный подход
- **Компонентный подход**
- …

До компонентного подхода

Компонентный подход

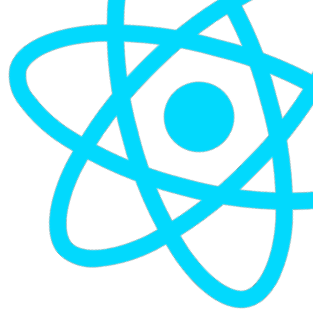


Компонентный подход



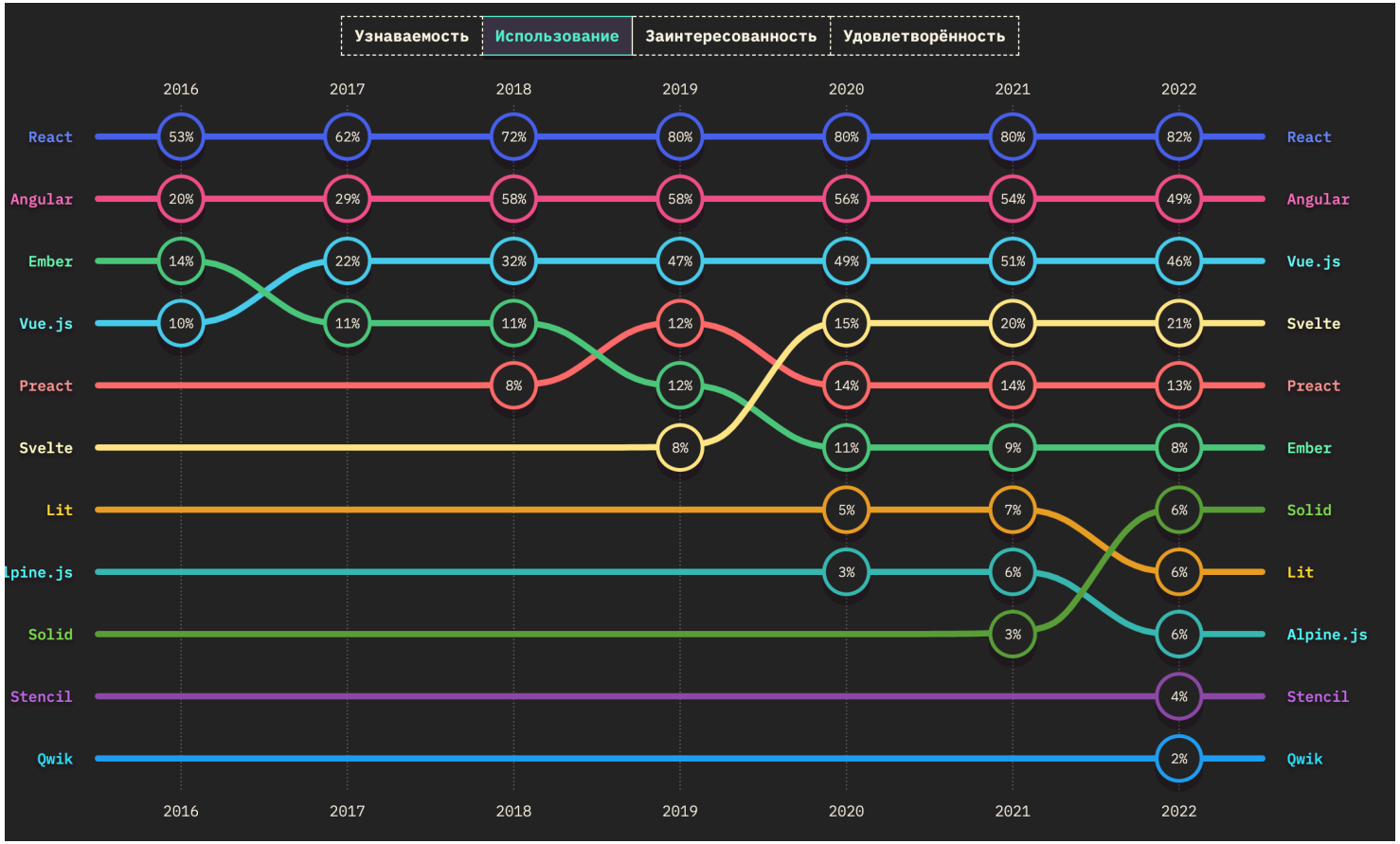
1. **FilterableProductTable** – контейнер, содержащий пример целиком
2. **SearchBar** – поле пользовательского ввода
3. **ProductTable** – отображает и фильтрует список данных на основе пользовательского ввода
4. **ProductCategoryRow** – наименование категорий
5. **ProductRow** – отдельно взятый товар

Преимущества



- Низкий порог входа
- Легко добавить в уже готовый проект
- Декларативный подход
- Компонентный подход
- **Лидер по использованию**

Преимущества



https://2022.stateofjs.com/ru-RU/libraries/front-end-frameworks/

Как начать использовать?

Для уже имеющегося проекта:

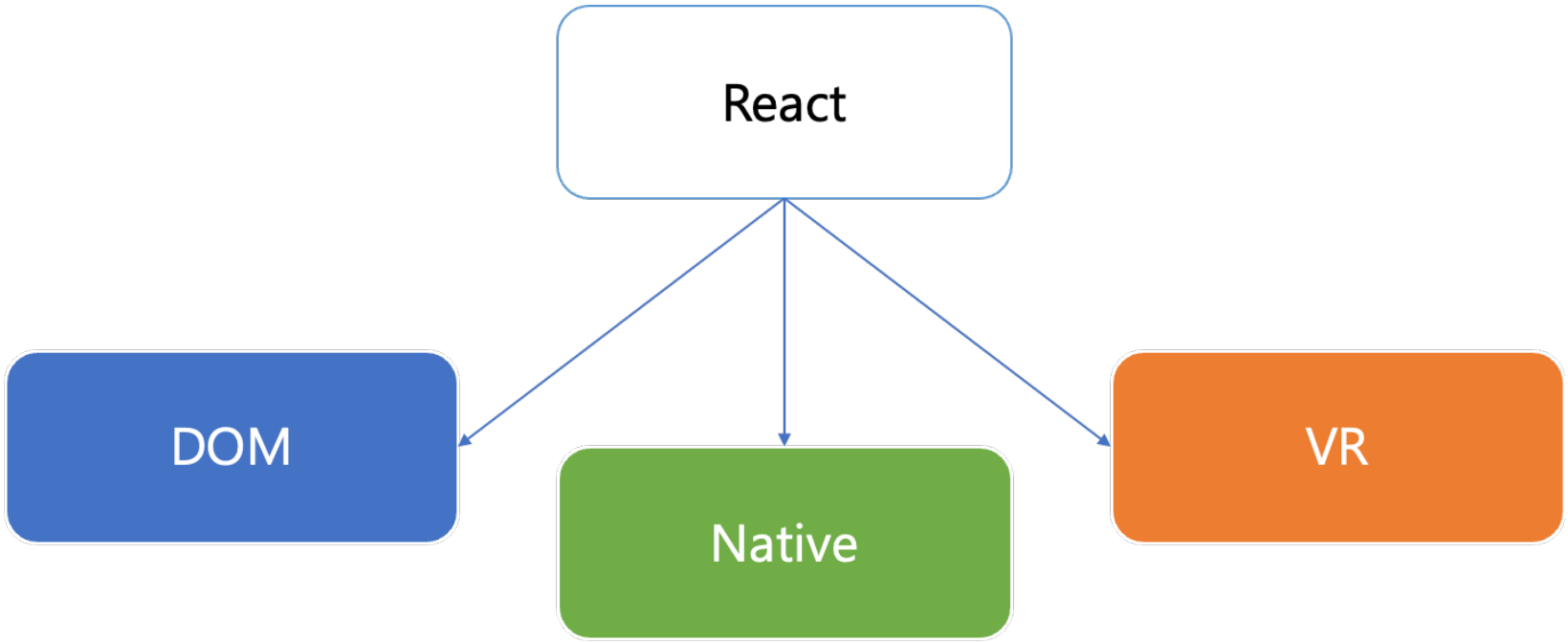
```
<script crossorigin src="https://unpkg.com/react@18/umd/react.development.js"></script>
<script crossorigin src="https://unpkg.com/react-dom@18/umd/react-dom.development.js"></script>
```

С чистого листа (один из множества вариантов):

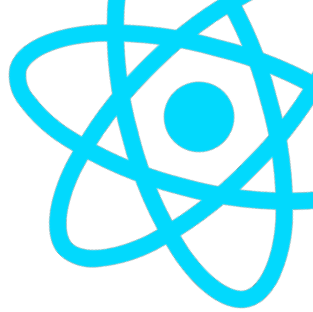
```bash
$ npm create vite@latest
```

https://react.dev/learn/start-a-new-react-project

Почему библиотеки две?



Создали проект. Что дальше?

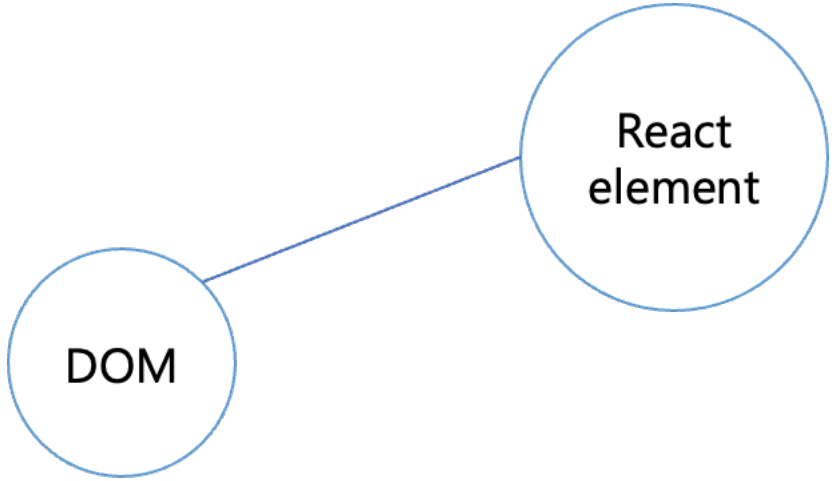


Точка входа в приложение:

```jsx
import React from 'react'
import ReactDOM from 'react-dom/client'
import App from './App.jsx'
import './index.css'

ReactDOM.createRoot(document.getElementById('root')).render(/* ... */);
```

```html
<body>
  <div id="root"></div>
  <script type="module" src="/src/main.jsx"></script>
</body>
```

Строительные блоки



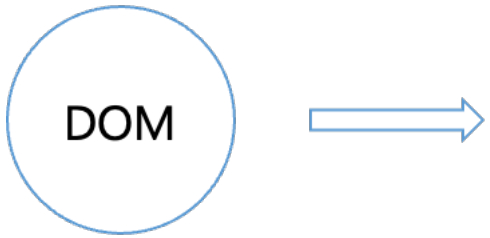
<input/>
<button>

…

Не являются DOM-элементами!

(но отвечают за них)

React Element DOM




```
ReactDOM.createRoot(document.getElementById('root')).render(
    React.createElement(
        'span',
        {
            children: 'Hello, React!'
        }
    )
);
```

React Element DOM

```
ReactDOM.createRoot(document.getElementById('root')).render(
    React.createElement(
        'div',
        {
            children: [
                React.createElement(
                    'span',
                    { children: 'title', style: { color: 'white' } }
                ),
                React.createElement(
                    'a',
                    {
                        href: '/',
                        children: 'link',
                        style: { textAlign: 'center' }
                    }
                )
            ],
            style: { display: 'flex' }
        }
    )
);
```

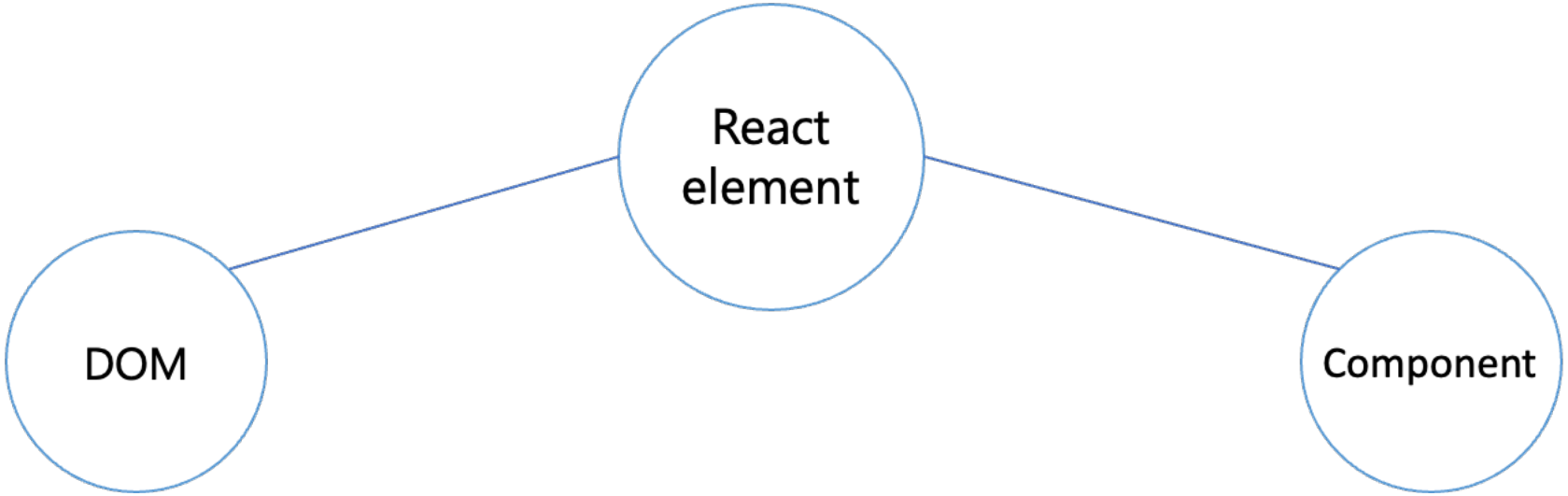
JSX

JSX — расширение синтаксиса React (синтаксический сахар)

```
ReactDOM.createRoot(document.getElementById('root')).render(
    React.createElement(
        'div',
        {
            children: [
                React.createElement(
                    'span',
                    { children: 'title', style: { color: 'white' } }
                ),
                React.createElement(
                    'a',
                    {
                        href: '/',
                        children: 'link',
                        style: { textAlign: 'center' }
                    }
                )
            ],
            style: { display: 'flex' }
        }
    )
);
```

```
ReactDOM.createRoot(document.getElementById('root')).render(
    <div style={{ display: 'flex' }}>
        <span style={{ color: 'white' }}>
            title
        </span>
        <a href='/' style={{ textAlign: 'center' }}>
            link
        </a>
    </div>
);
```

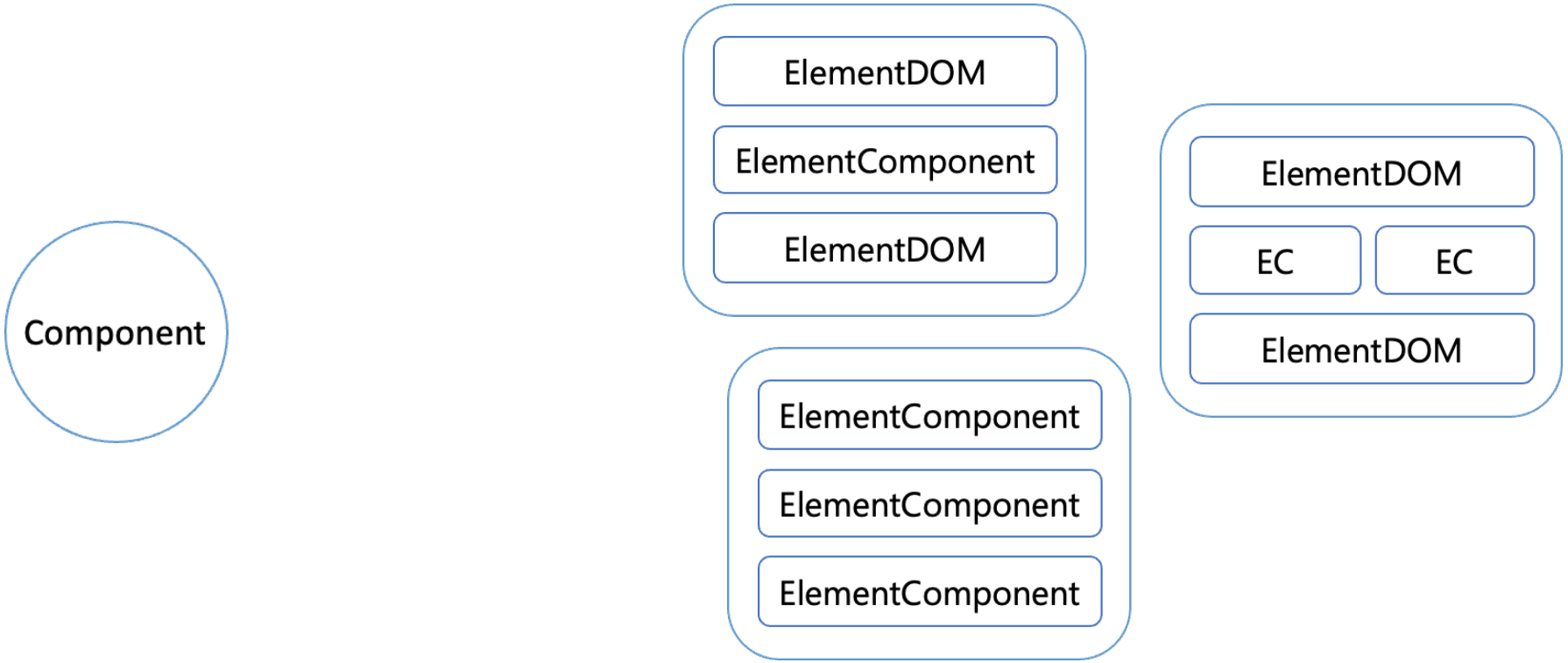
Строительные блоки



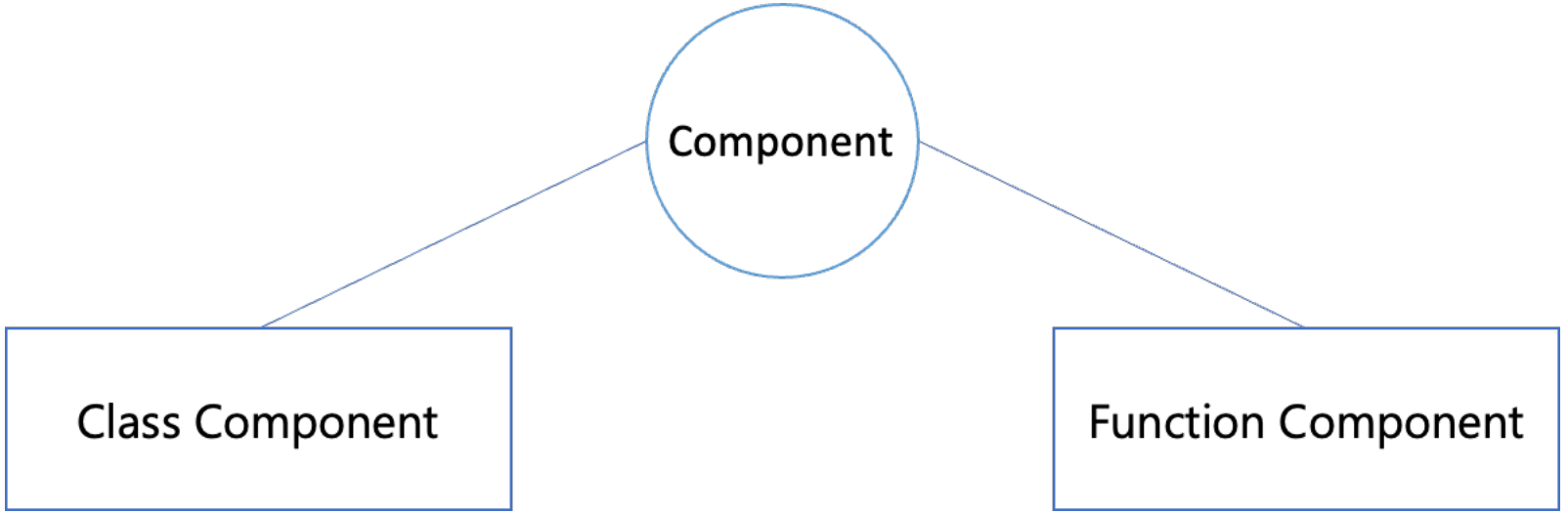
"Коробочка" с React Element DOM

React Element Component

Переиспользуемость, DRY, etc.



React Element Component



- Появились раньше
- На классах
- Имеет методы жизненного цикла
- По большей части, в legacy

- На функциях
- Популярнее
- Проще (для чтения, разработки)
- Не просаживает performance
- Имеет хуки (как альтернативу методам жизненного цикла)

React Element Component

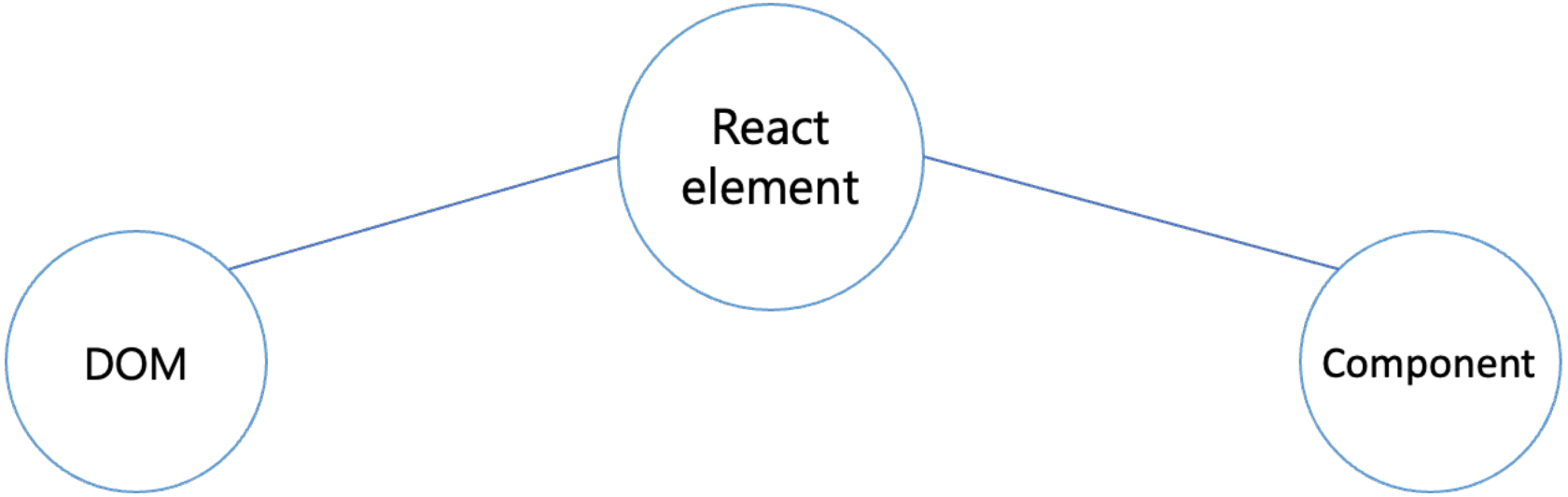
```jsx
class Example extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      count: 0
    };
  }

  componentDidMount() {
    document.title = `You clicked ${this.state.count} times`;
  }
  componentDidUpdate() {
    document.title = `You clicked ${this.state.count} times`;
  }

  render() {
    return (
      <div>
        <p>You clicked {this.state.count} times</p>
        <button onClick={() => this.setState({ count: this.state.count + 1 })}>
          Click me
        </button>
      </div>
    );
  }
}
```

Классовый компонент

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

function Example() {
  const [count, setCount] = useState(0);

  // Similar to componentDidMount and componentDidUpdate:
  useEffect(() => {
    // Update the document title using the browser API
    document.title = `You clicked ${count} times`;
  });

  return (
    <div>
      <p>You clicked {count} times</p>
      <button onClick={() => setCount(count + 1)}>
        Click me
      </button>
    </div>
  );
}
```

Функциональный компонент

Строительные блоки



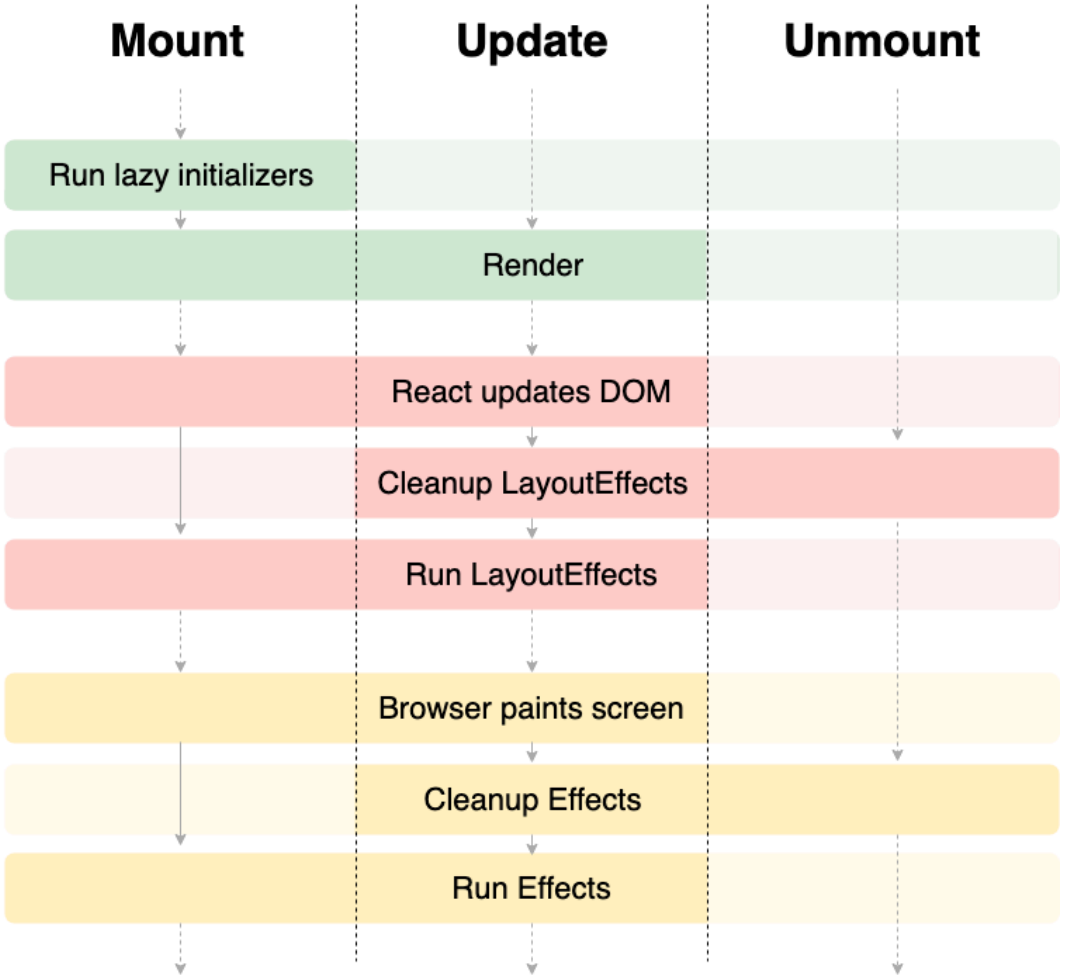
(с маленькой буквы)

```
ReactDOM
    .createRoot(document.getElementById('root'))
    .render(<input/>)
```

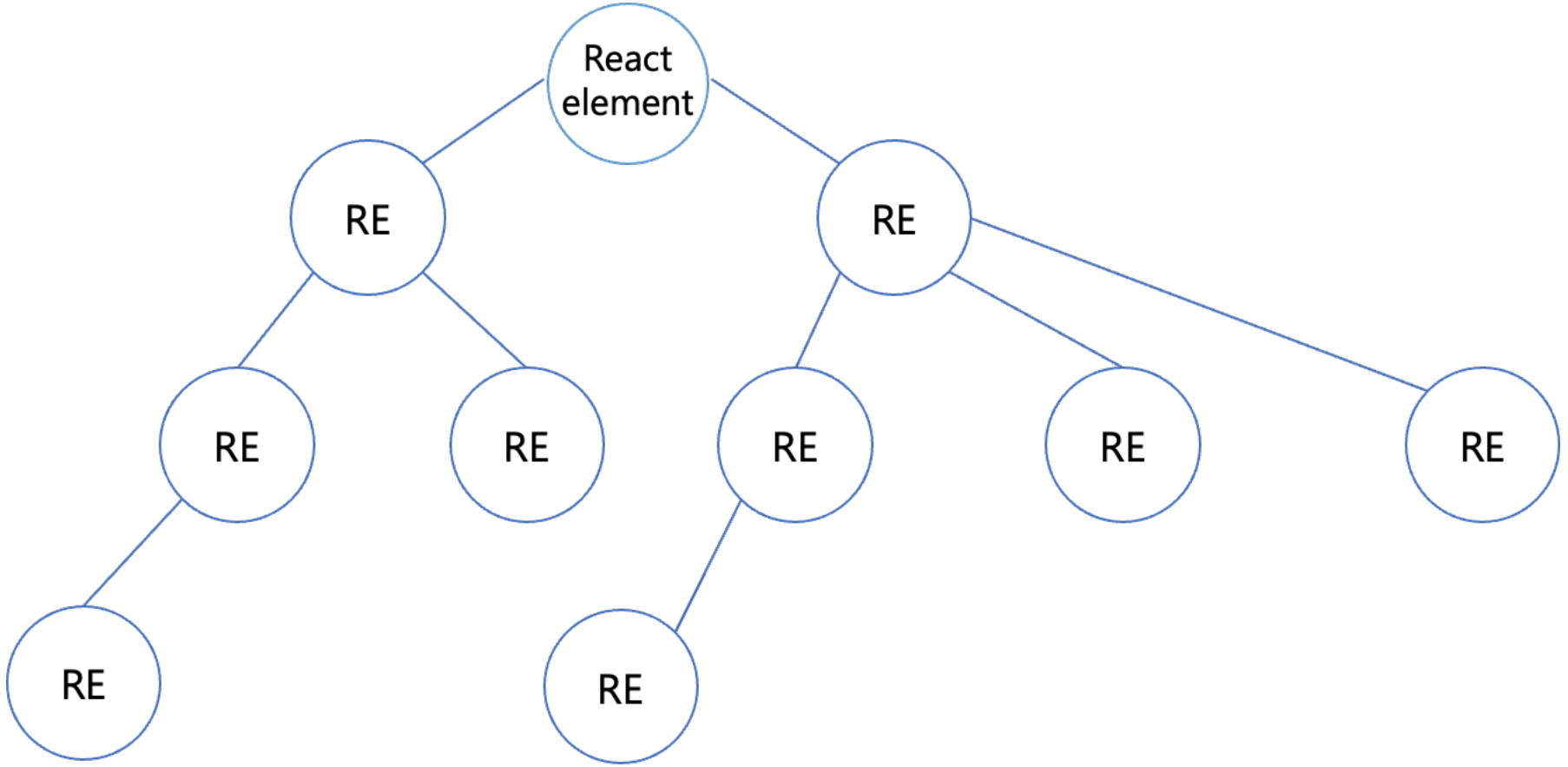
(с большой буквы)

```
ReactDOM
    .createRoot(document.getElementById('root'))
    .render(<App/>)
```

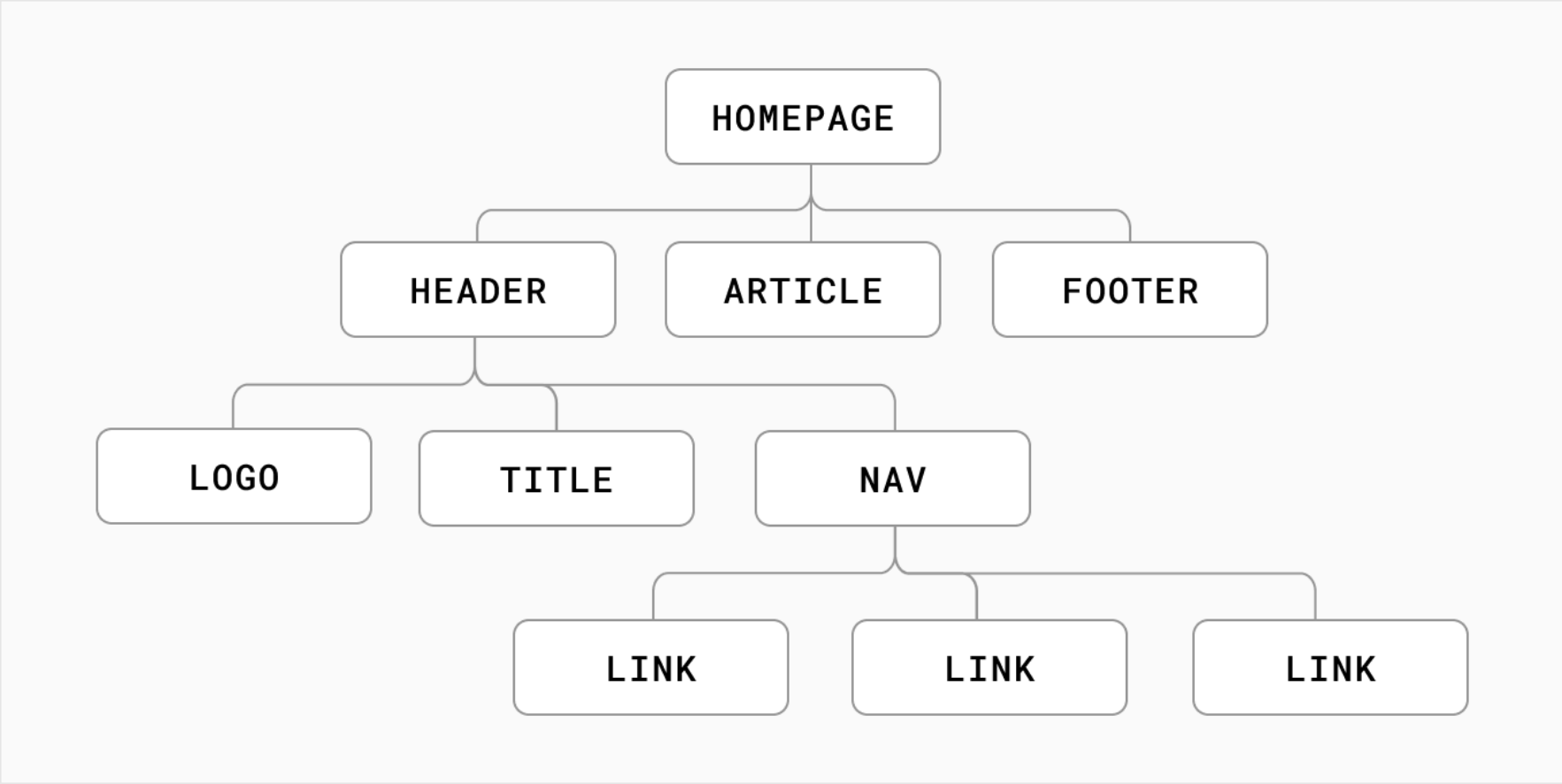
Жизненный цикл компонента



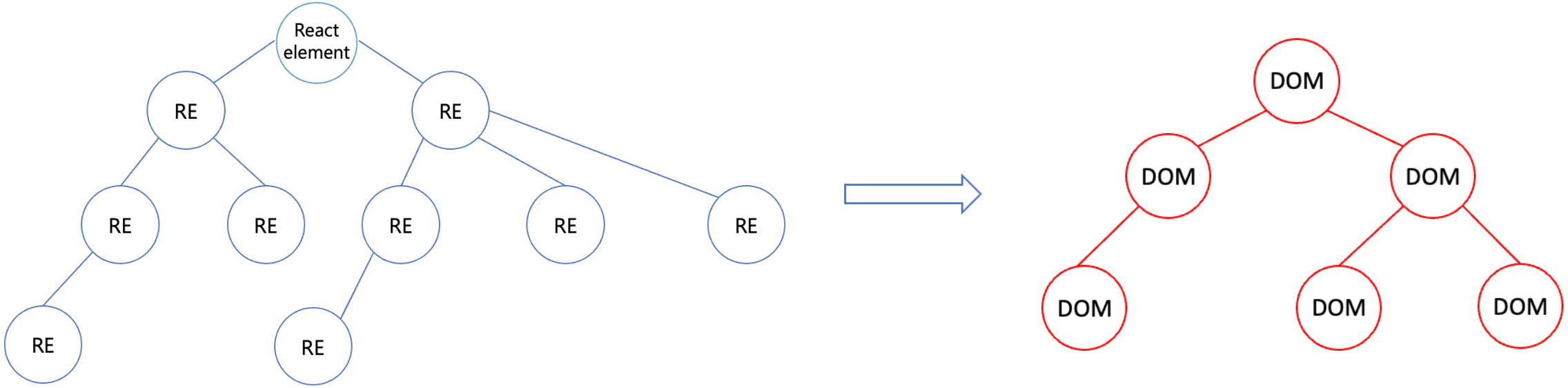
React Element Tree

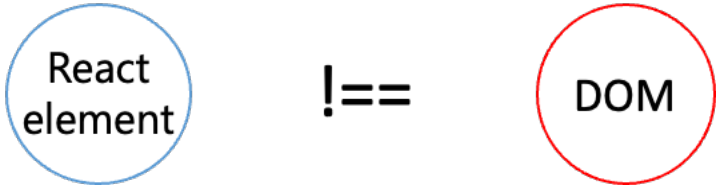


React Element Tree

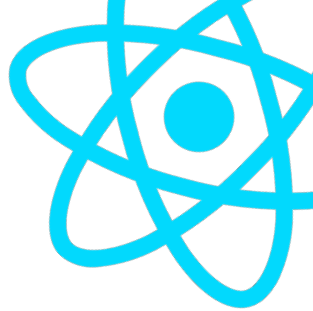


React Element Tree !== DOM Tree



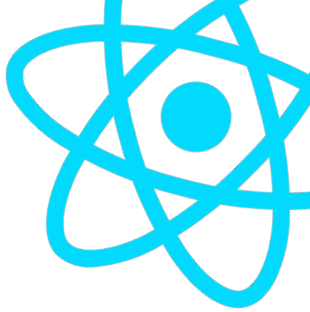


Лайвкодинг



https://github.com/lyaplyap/yatodo/tree/main/react

Полезные материалы



https://react.dev/ — официальная документация React

Базовый React (ШРИ 2023) — лекция школы разработки интерфейсов Академии Яндекса

https://vitejs.dev/ — документация Vite