**Rendering Elements**

|  |  |
| --- | --- |
| Наименьший строительный блок |  |
| Описывать |  |
| На экране |  |
| В отличие |  |
| Простые объекты |  |
| Соответствовать React элементам |  |
| Перепутать с |  |
| Более известная концепция |  |
| Сделан из |  |
| Где-то в вашем файле |  |
| Корневой узел DOM |  |
| Управлять |  |
| Приложения, написанные с использованием только React |  |
| Корневой узел |  |
| Встраивать в (интегрировать в) |  |
| Существующее приложение |  |
| Изолированный, отдельный, независимый |  |
| Для отрисовки React элемента |  |
| Передать что-то в функцию ReactDOM.render | pass |
| Создать элемент |  |
| Изменить потомков или атрибуты |  |
| В определенный момент времени | certain |
| Обновить пользовательский интерфейс |  |
| Сравнивать с |  |
| Применять обновления |  |
| Привести к желаемому состоянию |  |

Elements are the smallest building blocks of React apps.

An element describes what you want to see on the screen:

const element = <h1>Hello, world</h1>;

Unlike browser DOM elements, React elements are plain objects, and are cheap to create. React DOM takes care of updating the DOM to match the React elements.

**Note:**

One might confuse elements with a more widely known concept of “components”. Elements are what components are “made of”.

**Rendering an Element into the DOM**

Let’s say there is a <div> somewhere in your HTML file:

<div id="root"></div>

We call this a “root” DOM node because everything inside it will be managed by React DOM.

Applications built with just React usually have a single root DOM node. If you are integrating React into an existing app, you may have as many isolated root DOM nodes as you like.

To render a React element into a root DOM node, pass both to ReactDOM.render():

const element = <h1>Hello, world</h1>;

ReactDOM.render(element, document.getElementById('root'));

**Updating the Rendered Element**

React elements are [immutable](https://en.wikipedia.org/wiki/Immutable_object). Once you create an element, you can’t change its children or attributes. An element is like a single frame in a movie: it represents the UI at a certain point in time.

With our knowledge so far, the only way to update the UI is to create a new element, and pass it to **ReactDOM.render().**

**React Only Updates What’s Necessary**

React DOM compares the element and its children to the previous one, and only applies the DOM updates necessary to bring the DOM to the desired state.

**Connect the word and its description:**

|  |  |
| --- | --- |
| 1. The state of knowing about or being familiar with something | 1. Application |
| 1. The way that something is performed, written, drawn, etc. | 1. Knowledge |
| 1. Different from | 1. Isolated |
| 1. Costing little money or less than is usual or expected | 1. Immutable |
| 1. Happening or existing before something or someone else | 1. Previous |
| 1. A computer program that is designed for a particular purpose | 1. Cheap |
| 1. Happening alone, separately, or only once | 1. Rendering |
| 1. Not changing, or unable to be changed | 1. Unlike |

**Translate into Englis**h:

1. Для отрисовки React-элемента в корневой узел DOM, вызовите ReactDOM.render()
2. После создания элемента React, нельзя менять его потомков или атрибуты.
3. Элементы React неизменяемы.
4. React-элемент отражает состояние интерфейса в конкретный момент времени.
5. Элементы – это мельчайшие блоки React-приложений.
6. Элемент описывает то, что вы хотите увидеть на экране.