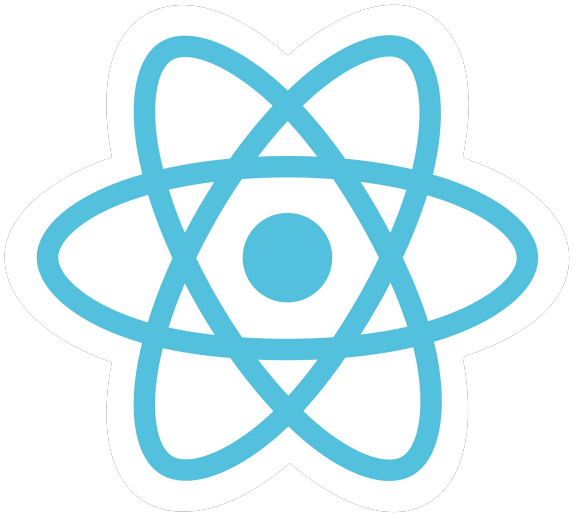
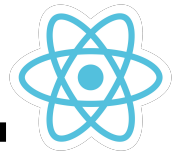


REACT

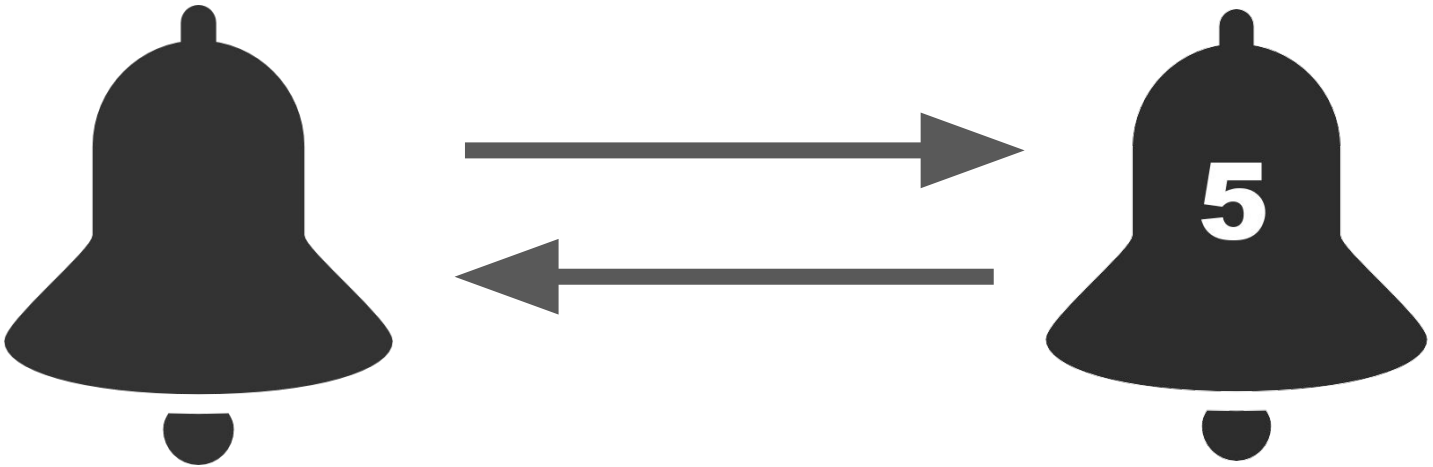



NETFLIX


airbnb


YAHOO!


5


5

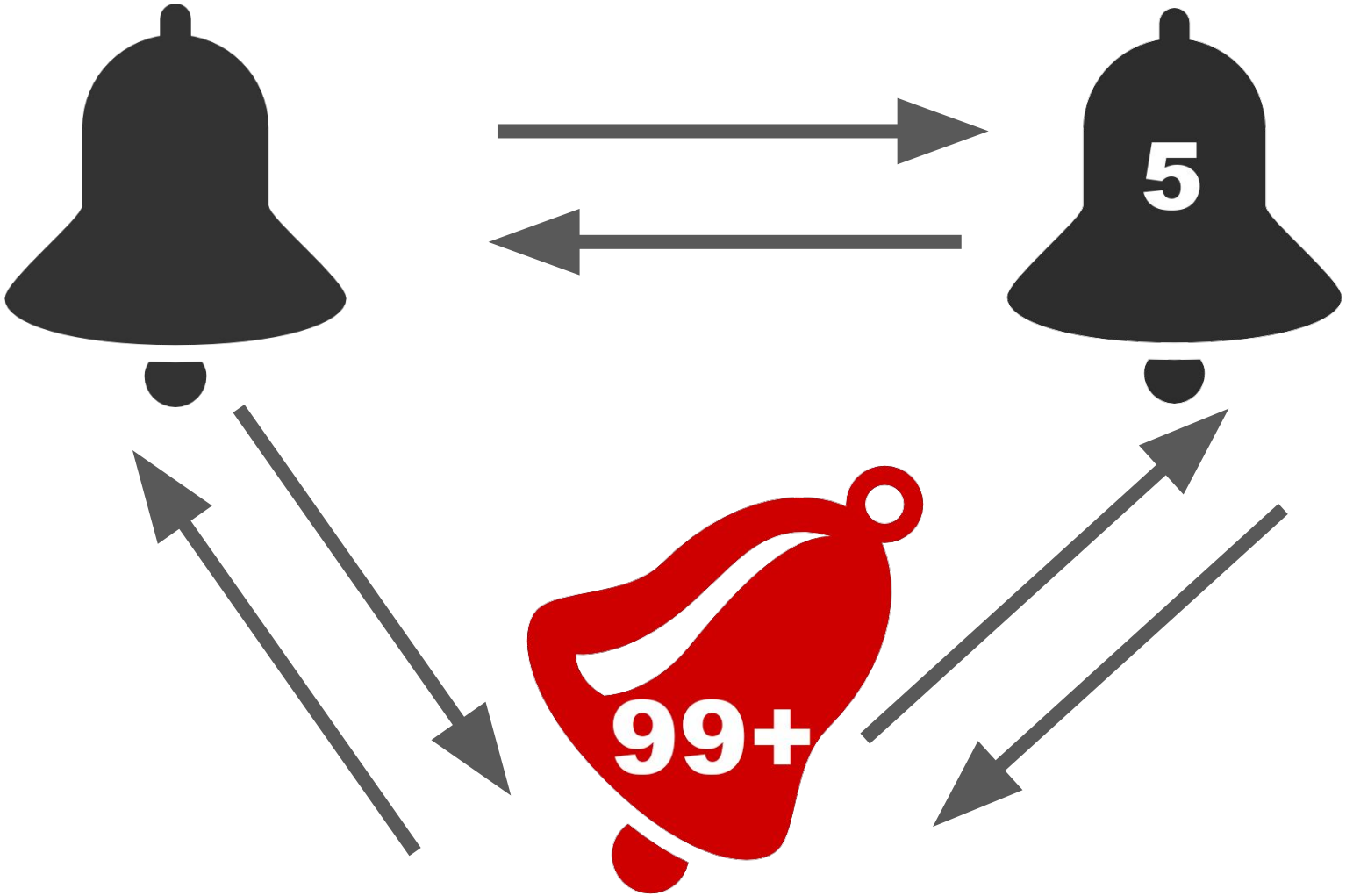

99+


5

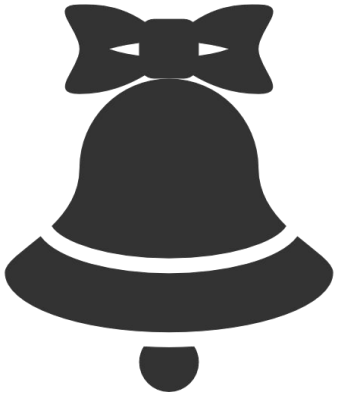

99+

Сколько здесь состояний?

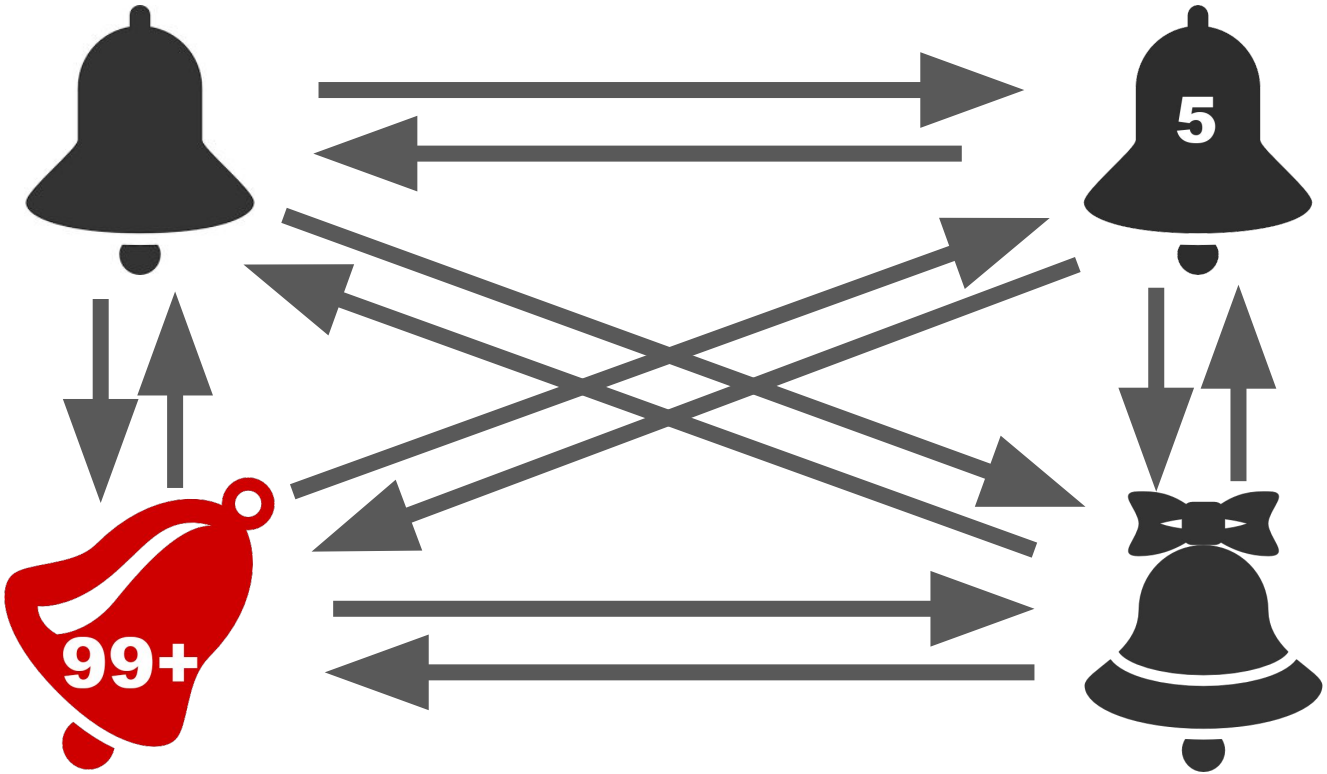
А сколько здесь возможно переходов состояний?


5
99+

Как насчет еще одного состояния?



А сколько переходов состояния теперь?



N - состояние

O - переход

$$O = N^2 - N$$

```
var notifier = {
      state: EMPTY
};

...
notifier.state = NORMAL;
...
notifier.state = ALERT;
...
notifier.state = NEW_ONE;
```

Мутирование данных

(здесь должен был быть слайд про черепашек-ниндзя, но...)

Мутирование данных

- ❏ Долго
- ❏ Дорого
- ❏ Сложно

Как обычно происходит?

Изменили состояние ===> Следует изменить UI

т.е. при малейшем изменении данных следует пересчитать и перестроить DOM заново.

Как происходит в React?

Как происходит в React?

Изменили состояние ===> Следует изменить UI

т.е. при малейшем изменении данных следует пересчитать и перестроить DOM заново.

Какая самая дорогая операция на клиенте?

Какая самая дорогая операция на клиенте?

Rendering

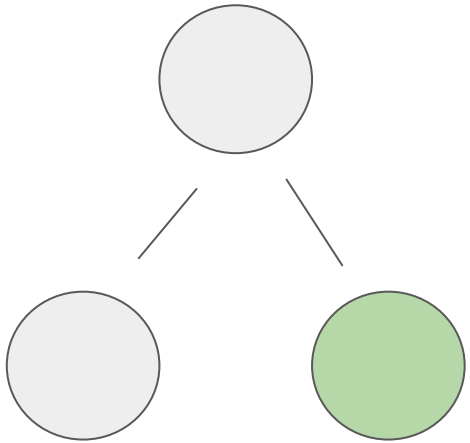
Есть одно отличие...

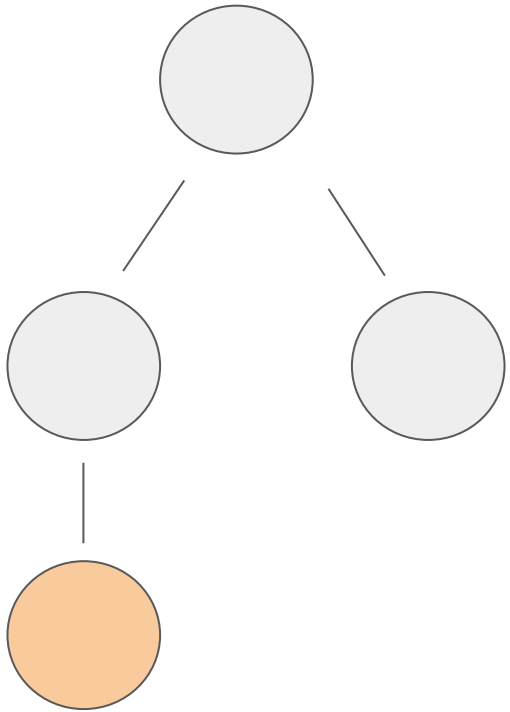
Virtual DOM

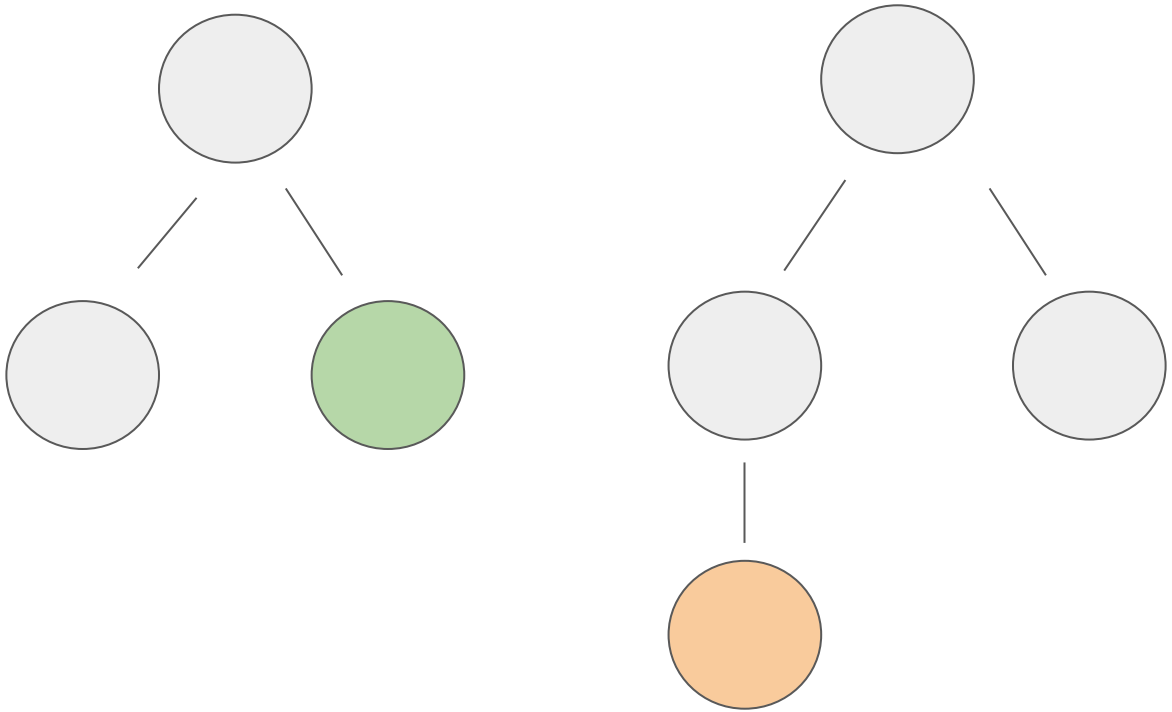
Virtual DOM

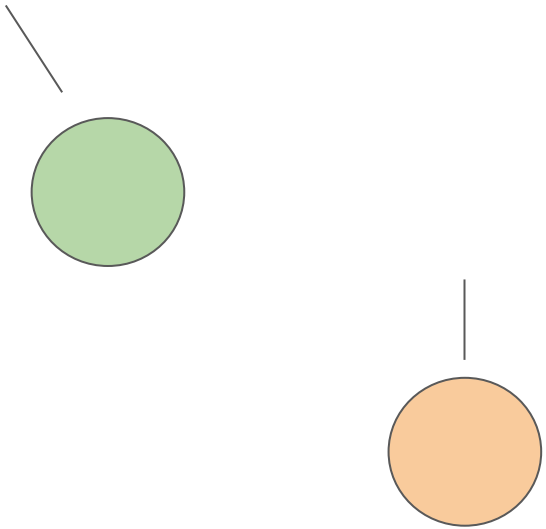
Виртуалный DOM - легковесная версия DOM,
которая хранится в памяти.
По сути, это обычный JavaScript-объект.

Изменения в Virtual DOM !== Изменения в DOM









React-приложение

- ❏ Простое
- ❏ Производительное
- ❏ Легко масштабируемое

Hello World

Hello World

npm install --save react react-dom

Hello World

```
import React from 'react';

import ReactDOM from 'react-dom';
```

Hello World

```
<script src="https://unpkg.com/react@15.5.4/dist/react.js"></script>

<script src="https://unpkg.com/react-dom@15.5.4/dist/react-dom.js" ...
```

```
var element = React.createElement(
    'h1',
    {
        id: 'my-awesome-react-element',
        className: 'my-awesome-react-element-class'
    },
    'Hello World'
);
```

```
var element = React.createElement(...);

ReactDOM.render(
    element,
    document.querySelector('#react-root')
);
```

```
var ReactElement = function(...) {
    // немного магии и...
    var element = {
        type, props, key, ref
    };
     ...



    return element;
};
```

```
var ReactElement = function(...) {
    // немного магии и...
    var element = {
        type, props, key, ref
    };
     ...
    Object.freeze(element.props);
    Object.freeze(element);

    return element;
};
```

```
var element = React.createElement(
    'h1',
    {
        id: 'my-awesome-react-element',
        className: 'my-awesome-class'
    },
    'Hello World'
);
```

```
var element2 = React.createElement(
    'h2'
    {
        id: 'my-another-element',
        className: 'my-another-class'
    },
    'Very important component'
);
```

```
var App = React.createElement(
    'div',
     null,
     element,
     element2
);

ReactDOM.render(
    App,
    document.querySelector('#react-root')
);
```

```
var App = {
    type, key, ref,
    props: {
        children:[
            element = {...},
            element2 = {
                type, key, ref,
                props:{
                    children:[...]
                }
            }
        ]
    }
};
```

JSX

JSX

JSX - синтаксический сахар для метода React.createElement.

Это **НЕ** очередной шаблонизатор.

JSX

```
<div> Hello world! </div>
```

JS

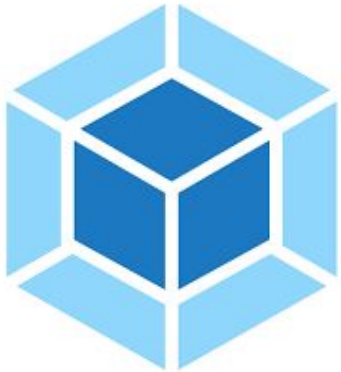
```
'use strict'

React.createElement(
        "div",
        null,
         "Hello world");
```

Наследование?

Композиция.


ES
6


BABEL



```
npm install -g create-react-app

create-react-app my-app
cd my-app/
npm start
```

(не реклама)

strizhak.sergey@gmail.com

sergey.strizhak