



Attractor
Software

Оптимизация React приложений



<https://github.com/teimurjan/react-optimization-presentation>

- Размер **bundle**
- Вызовы **render**

ExtractTextPlugin

```
const ExtractTextPlugin = require('extract-text-webpack-plugin');

module.exports = {
  module: {
    rules: [
      {
        oneOf: [{
          test: /\.css$/,
          loader: ExtractTextPlugin.extract({/* your loader here */)
        }]
      }
    ]
  }
}
```

Минимизация CSS

```
loader: ExtractTextPlugin.extract({  
  use: [{  
    loader: require.resolve('css-loader'),  
    options: {  
      minimize: true  
    }  
  }]  
})
```

*.css -= 45 KB (~25%)

185 KB



140 KB



NODE_ENV = production

```
const webpack = require('webpack');

module.exports = {
  plugins: [
    new webpack.DefinePlugin({
      'process.env': {
        NODE_ENV: JSON.stringify('production')
      }
    })
  ]
}
```

*.js -= 630 KB (~69%)

920 KB



290 KB



UglifyJsPlugin

```
const webpack = require('webpack');

module.exports = {
  plugins: [
    new webpack.optimize.UglifyJsPlugin({
      /* settings */
    })
  ]
}
```

*.js -= 136 KB (~47%)

290 KB



154 KB



Другие методы

- Убрать prop types из **bundle** -
babel-plugin-transform-react-remove-prop-types
- Установить ограничения по поддержке браузеров для
babel

Анализ bundle с помощью

- source-map-explorer

```
npm install -g source-map-explorer  
source-map-explorer bundle.js bundle.js.map
```


А также с помощью

- webpack-bundle-analyzer

```
const BundleAnalyzerPlugin = require('webpack-bundle-analyzer').BundleAnalyzerPlugin;

module.exports = {
  plugins: [
    new BundleAnalyzerPlugin()
  ]
}
```

Treemap sizes:

Stat **Parsed** Gzipped

Show chunks:

- ☒ All (151.02 KB)
- ☒ main.1449552c.js (149.61 KB)
- ☒ manifest.7f23a882.js (1.41 KB)

main.1449552c.js

node_modules

react-dom

cjs

react-dom.production.min.js





Делим bundle на vendor и client

```
const webpack = require('webpack');

module.exports = {
  entry: {
    client: './src/index.js',
    vendor: ['react', 'react-dom', 'react-router', 'react-router-dom']
  },
  plugins: [
    new webpack
      .optimize
      .CommonsChunkPlugin({name: 'vendor', filename: 'vendor.[chunkhash].js'})
  ]
};
```

Делим `client` на чанки

```
const webpack = require('webpack');

module.exports = {
  output: {
    path: buildPath,
    filename: '[name].[chunkhash:8].js',
    chunkFilename: '[name].[chunkhash:8].chunk.js'
  }
};
```

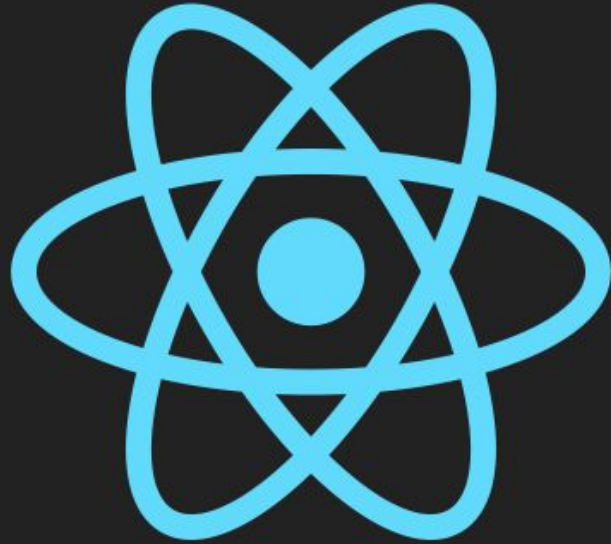
```
// Home.js
import React from 'react';

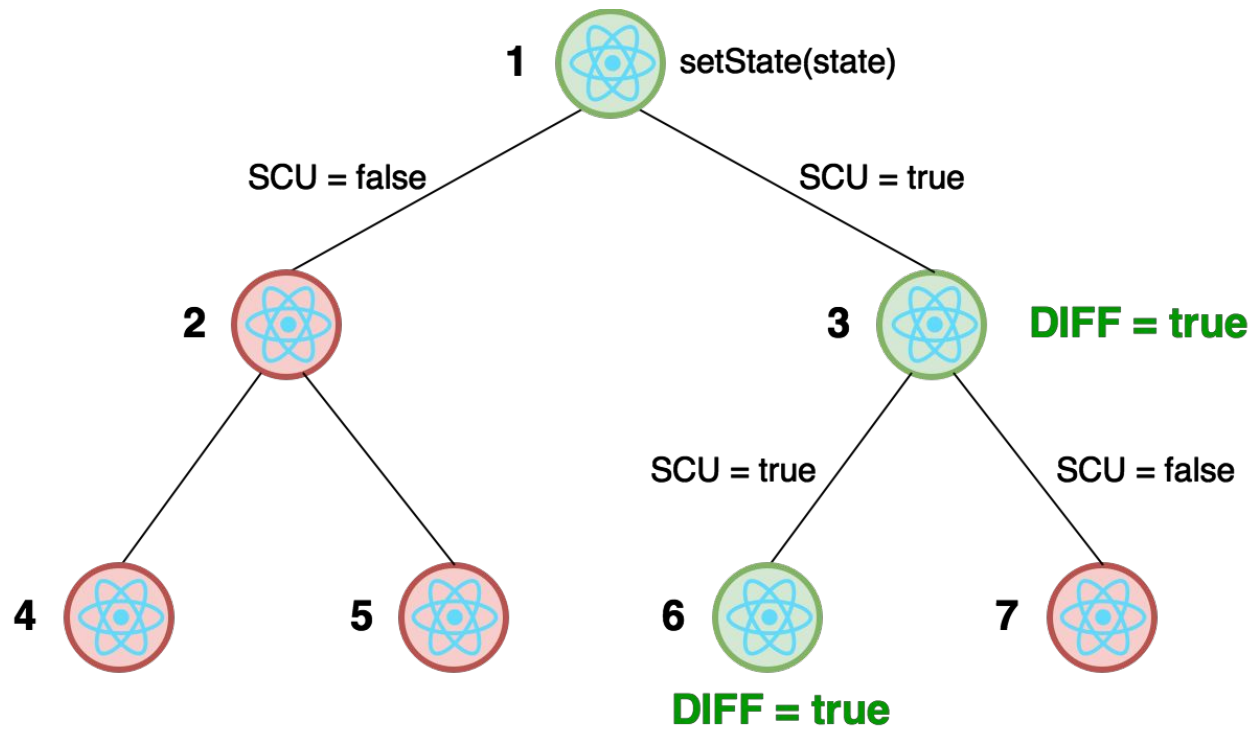
export default () => (
  <div className="wrapper">
    <h2>Home</h2>
  </div>
);
```

```
// HomeContainer.js
import Loadable from 'react-loadable';
import Loading from '../Common/Loading';

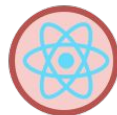
export default Loadable({
  loader: () => import ('../Home' /* webpackChunkName: 'home' */),
  loading: Loading
})
```

Reconciliation





- RERENDERED



- NOT RERENDERED

SCU - shouldComponentUpdate

React.PureComponent

```
import React from 'react';

export default class extends React.PureComponent {
  shouldComponentUpdate(nextProps, nextState) {
    return !(
      shallowEqual(nextProps, this.props) &&
      shallowEqual(nextState, this.state)
    );
  }
}
```

Проблемы `React.PureComponent`

- `Arrow`-функции и `.bind(this)`
- Создание и модификация массивов и объектов


```
class TodoList extends React.Component {  
  render() {  
    return (  
      <ul>  
        {this.props.todos.map(todo => (  
          <Todo  
            todo={todo}  
            onClick={() => this.props.onTodoClick(todo.id)}  
          />  
        ))}  
      </ul>  
    );  
  }  
}
```

```
class TodoList extends React.Component {  
  render() {  
    return (  
      <ul>  
        {this.props.todos.map(todo => (  
          <Todo  
            todo={todo}  
            onClick={this.props.onTodoClick}  
          />  
        ))}  
      </ul>  
    );  
  }  
}
```

```
class Todo extends React.PureComponent {  
  handleClick = e => {  
    this.props.onClick(this.props.todo.id);  
  }  
  
  render() {  
    return (  
      <li className="todo" onClick={this.handleClick}>  
        {this.props.todo}  
      </li>  
    )  
  }  
}
```

```
render() {  
  return (  
    <ul>  
      {this.props.todos.map(todo => (  
        <Todo  
          todo={todo}  
          onClick={this.props.handleClick.bind(this)}  
        >  
      ))}  
    </ul>  
  );  
}
```

```
constructor(props) {  
  super(props);  
  this.handleClick = this.handleClick.bind(this);  
}  
  
render() {  
  return (  
    <ul>  
      {this.props.todos.map(todo => (  
        <Todo  
          todo={todo}  
          onClick={this.props.handleClick}  
        />  
      ))}  
    </ul>  
  );  
}
```

```
handleClick = (todoId) => {  
  this.setState({  
    [todoId]: {clicked: true}  
  });  
}
```

```
const arr1 = [1, 2, 3];
```

```
const arr2 = [1, 2, 3];
```

```
arr1 === arr2 // false
```

```
const obj1 = {foo: 'bar'};
```

```
const obj2 = {foo: 'bar'};
```

```
obj1 === obj2 // false
```

```
render() {  
  return (  
    <TodoList  
      options={{  
        wrap: false,  
        maximizeOnFocus: true  
      }}  
    />  
  );  
}
```



```
const TODO_LIST_OPTIONS = {
  wrap: false,
  maximizeOnFocus: true
};

export default class extends React.Component {
  render() {
    return (
      <TodoList
        options={TODO_LIST_OPTIONS}
      />
    );
  }
}
```

Использование индекса элемента как **key**

```
const elements = [  
  {type: 'div', key: 0, textContent: 'Container #0'},  
  {type: 'div', key: 1, textContent: 'Container #1'},  
  {type: 'div', key: 2, textContent: 'Container #2'},  
  {type: 'div', key: 3, textContent: 'Container #3'},  
];
```

// Удалим элемент под индексом 1

```
const elements = [  
  {type: 'div', key: 0, textContent: 'Container #0'},  
  // Render будет бесполезно вызван для всех элементов ниже  
  {type: 'div', key: 1, textContent: 'Container #2'},  
  {type: 'div', key: 2, textContent: 'Container #3'},  
];
```

Решение проблемы

- Использование существующих стабильных ключей
- Генерация стабильных ключей во время `mount` (Или в каком случае во время `render`!)

Пасхалочка

```
import React from 'react';

const TodoFactory = ({ todo, onClick }) => (
  <li className="todo" onClick={onClick}>
    {todo.title}
  </li>
);

export default ({ todos }) => (
  <ul>
    {todos.map(todo => TodoFactory({ todo, onClick: console.log })))}
  </ul>
)
```