УДК 004.4

СРАВНЕНИЕ КЛАССИЧЕСКОГО ПРОЦЕССА РЕАЛИЗАЦИИ ВЕБ-ПРИЛОЖЕНИЙ И ПОДХОДА С ИСПОЛЬЗОВАНИЕМ БИБЛИОТЕКИ REACT

А. А. Горбачев , Е. С. Горбачева
Донской государственный технический университет (г. Ростов-на-Дону, Российская Федерация)

Рассмотрены два способа реализации веб-приложений: классический и с использованием библиотеки React. Проведено их сравнение по трем критериям: создание страницы с минимальным содержимым, реализация и поддержка больших веб-приложений, скорость отрисовки страниц. Даны характеристики обоих подходов, названы их преимущества и недостатки.
Ключевые слова: библиотека React, веб-технологии, веб-разработка, JSX, JavaScript.

UDC 004.4

COMPARISON OF THE CLASSICAL WEB APPLICATION IMPLEMENTATION PROCESS AND THE REACT LIBRARY APPROACH

A. A. Gorbachev, E. S. Gorbacheva
Don State Technical University (Rostov-on-Don, Russian Federation)

The article describes two ways to implement web applications: classic and using the React library. They were compared according to three criteria: the creation of a page with minimal content, the implementation and support of large web applications, the speed of page rendering. The characteristics of both approaches, their advantages and disadvantages are given.
Keywords: React library, web technologies, web development, JSX, JavaScript.

Введение. В настоящее время веб-технологии находятся на пике своего развития. Веб-приложения уже начинают полностью поглощать десктопные программы. Последние необходимо устанавливать на компьютер пользователя, где они запускаются локально и выполняют свой код. Веб-приложения же, как правило, не требуют инсталляции дополнительного программного обеспечения на рабочую станцию.

Существует два основных способа реализации веб-приложений: классический и с использованием различных инструментов, таких как Angular, React, Vue и другие. Цель данной работы — рассмотреть подробнее классический подход и подход с использованием библиотеки React, провести их сравнение, чтобы выделить преимущества и недостатки.

Классический способ написание веб-приложений появился еще во времена создания Интернета. Под ним понимают написание css и js файлов с их последующим объединением в html документе. Несмотря на то, что этот подход достаточно старый, его все еще используют из-за низкого порога вхождения. С помощью данного способа разработки трудно реализовывать интерактивные веб-приложения, он больше подходит для написания статических страниц [1].

React — библиотека JavaScript с открытым исходным кодом для создания пользовательских интерфейсов. Разрабатывается и поддерживается Facebook. React делает создание интерактивных интерфейсов проще. Данная технология эффективно обновляет и отображает только необходимые компоненты при изменении данных. На сегодняшний день использование библиотеки React и связанных с ней пакетов является одним из самых популярных подходов в построении веб-приложений [2].

Проведено сравнение данных способов разработки веб-приложений.

Создание страницы с минимальным содержимым. Для написание базовой страницы с помощью классического подхода достаточно создать файл с расширением html и написать в нем структуру документа. Пример кода показан на рис. 1. После этих действий страницу можно открыть в браузере.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
</head>
<body>
    hello world
</body>
</html>
```

Рис. 1. Структура html документа

Запуск базовый страницы с использованием библиотеки React имеет значительно больше шагов. Для подготовки каркаса приложения потребуется:

— установить программную платформу Node.js версии 6 и выше;

— через интерпретатор командной строки (cmd) воспользоваться пакетным менеджером (npm), установленным вместе с Node.js. Выполнить команду *npx create-react-app my-app*, которая автоматически генерирует структуру React приложения, где *my-app* — имя проекта;

— перейти в созданный проект с помощью команды *cd my*-app;

— запустить веб-страницу в браузере через команду *npm start* [3].

Реализация и поддержка больших веб-приложений. Используя классический подход написания веб-приложений, необходимо искусственно разделять технологии, помещая разметку и логику в разные файлы. Это усложняет процесс написания динамических страниц, так как связь между логикой и разметкой слабая. Также разделение технологий влечет за собой увеличение объема веб-приложений, следовательно, сложность дальнейших процессов разработки и поддержки экспоненциально растет.

При написании React-приложений рекомендуется использовать надстройку языка JavaScript – JSX, её синтаксис показан на рис. 2. Она позволяет легко разобраться в коде и представить DOM-модель. JSX максимально понятно описывает UI интерфейс и включает в себя весь потенциал языка JavaScript [4].

```
const element = <div>Привет, React!</div>;
```

Рис. 2. Синтаксис JSX

React автоматически учитывает важные механизмы UI-интерфейса:

— обработка пользовательских событий;

— динамическое изменение состояний;

— подготовка данных к последующей отрисовке на экране.

Для осуществления данных пунктов библиотека React использует компоненты, которые совмещают в себе разметку и логику.

Чтобы встроить JavaScript код в JSX, его достаточно заключить в фигурные скобки. На рис. 3 показана циклическая подстановка элемента массива в тег div.

```
render() {
    const elements = ['React', 'Читатель', 'Друг'];
    return elements.map( callbackfn: (el,i) => <div key={i}>Привет {elements}!</div>)
}
```

Рис. 3. Внедрение JS кода в JSX

Скорость отрисовки страниц. Классический способ реализации веб-приложений при различных манипуляциях осуществляет полную перезагрузку страницы, так как, когда меняется контент или происходит переход на другую страницу сайта, с сервера приходят новые данные. Это связано с тем, что возникает необходимость полностью перерисовать структуру страницы (DOM-дерево). Данный процесс занимает достаточно много времени, особенно если страница содержит много объектов.

Библиотека React дает возможность создавать интерактивные веб-приложения, которые выполняют большое количество действий и максимально быстро дают ответ на них. При этом все данные обновляются в режиме реального времени, и страница не требует полной перезагрузки.

Быстрота работы React-приложения обеспечивается за счет механизма, который отслеживает измененные элементы и обновляет только их, а не всю страницу. Также на скорость влияет виртуальное DOM-дерево, которое является легковесной копией настоящего DOM-дерева. Вместо того, чтобы работать с DOM напрямую, изменения вносятся в копию, а после этого применяются к реальному DOM. При этом происходит сравнение DOM-дерева с его виртуальной копией. Определяются различия деревьев, и перерисовывается только то, что было изменено. Такой подход работы с DOM-деревом значительно быстрее, так как исключает тяжелые части настоящего DOM [1].

Заключение. Были рассмотрены два способа реализации веб-приложений: классический и с использованием библиотеки React. Проведено их сравнение по выбранным критериям.

Создать страницу с минимальным содержимым проще и быстрее с помощью классического подхода, так как не требуется ничего устанавливать. Это может быть удобно для создания небольших страниц. Однако данная процедура совершается единожды за весь проект, поэтому такое преимущество незначительно при разработке крупных приложений.

Разработка и поддержка приложений с помощью React происходит значительно легче, чем при классическом подходе. Это осуществляется за счет встроенных в React механизмов.

По скорости отрисовки страниц также выигрывает способ разработки веб-приложений с использованием React. Загрузка такой страницы в среднем происходит в три раза быстрее, чем аналогичной, разработанной с помощью классического подхода.

Таким образом, так как веб-технологии стремительно развиваются, не стоит отказываться от новых библиотек и фреймворков. Они делают процесс разработки веб-приложений значительно проще и качественнее.

Библиографический список

1. Резинг, Д. JavaScript для профессионалов, 2-е издание / Д. Резинг, Р. Фергюсон, Д. Пакстон. — Москва : ООО «И. Д. Вильямс», 2016. — 240 с.

2. Бэнкс, А. React и Redux: функциональная веб-разработка / А. Бэнкс, Е. Порселло. — Санкт-Петербург : Питер, 2018. — 336 с.

3. Стефанов, С. React.js. Быстрый старт / С. Стефанов. — Санкт-Петербург: Питер, 2017. — 304 с.

4. Знакомство с JSX [Электронный ресурс] / Lean-React JS. — Режим доступа: https://learn-reactjs.ru/basics/introduction-to-jsx/ (дата обращения: 20.11.2019).

Об авторах:

Горбачев Андрей Александрович, студент Донского государственного технического университета (344000, РФ, г. Ростов-на-Дону, пл. Гагарина, 1), andrew.gorba4ev2018@yandex.ru

Горбачева Елена Сергеевна, магистрант Донского государственного технического университета (344000, РФ, г. Ростов-на-Дону, пл. Гагарина, 1), lukashewa.alena@yandex.ru