Федеральное государственное образовательное бюджетное

учреждение высшего образования

**«Финансовый университет при Правительстве Российской Федерации»**

**(Финансовый университет)**

Колледж информатики и программирования

**ОТЧЕТ О ВЫПОЛНЕНИИ ЗАДАНИЯ № 8**

**Тема: Доклад**

Студент: Романов.Н.М

Группа: 4ИСИП-619

Преподаватель: Сибирев.И.В

Дата: 28.02.23

**Доклад Javascript библиотека React**

React.js — это библиотека для языка программирования javascript с открытым исходным кодом для разработки пользовательских интерфейсов. Она помогает быстро и легко реализовать реактивность — явление, когда в ответ на изменение одного элемента меняется все остальное.

У React открытый исходный код и мощное сообщество. Это одна из самых популярных библиотек для веб-разработки.

**Кто пользуется React**

React применяют фронтенд-разработчики. В модели MVC, название которой расшифровывается как Model-View-Controller, интерфейс — это View, представление, внешнее отображение, с которым взаимодействует пользователь, та часть сайта или приложения, которая видна человеку. Специалисты, которые с ней работают, в частности, используют React.js. Также с React могут работать верстальщики, тестировщики и другие специалисты, задействованные в создании веб-интерфейсов.

React используют для создания одностраничных и многостраничных приложений, разработки крупных сайтов. Например, с его помощью написан стриминговый сервис Netflix и реализованы новостные ленты крупнейших социальных сетей. Библиотека предназначена:

* для создания функциональных интерактивных веб-интерфейсов, работая с которыми, не нужно постоянно обновлять страницу;
* быстрой и удобной реализации отдельных компонентов и страниц целиком — элементы в React легко использовать повторно;
* легкой разработки сложных программных структур — их просто описывать, если использовать реализованный в React подход;
* доработки новой функциональности с любым изначальным стеком технологий: библиотека не зависит от остального инструментария и будет хорошо работать, на чем бы ни был написан код;
* разработки одностраничных и многостраничных приложений (SPA и PWA). Это приложения, которые функционируют как программы и веб-сервисы и имеют соответствующий интерфейс;
* работы с серверной частью сайта или разработки мобильных приложений. В таких случаях React используют совместно с инструментами, адаптирующими веб-технологии под другие цели.

## **Особенности библиотеки React**

React имеет ряд особенностей, которые делают его гибким и мощным инструментом. Мы разберем некоторые из них.

### **Декларативность**

Декларативный стиль означает, что достаточно один раз описать, как будут выглядеть результаты работы кода — элементы в разных состояниях. Ему не нужно фокусироваться на способах достижения результатов: большую часть задач выполнит библиотека. React.js будет автоматически обновлять элементы в зависимости от условий, главная задача — грамотно описать их. Удобный и понятный подход облегчает написание и отладку кода.

### **Виртуальное DOM-дерево**

Любой веб-интерфейс основан на HTML-документе и CSS-стилях, к которым подключен код на JavaScript. Структура HTML-документа, точнее его модель, называется **DOM-деревом** (DOM расшифровывается как Document Object Mode, объектная модель документа). Это древовидная модель, в которой в иерархическом виде собраны все используемые на странице элементы.

Особенность React в том, что он создает и хранит в кэше виртуальное DOM-дерево — копию DOM, которая изменяется быстрее, чем реальная структура. Это нужно для того, чтобы быстро обновлять страницы. Если пользователь выполнит действие или наступит какое-либо событие, DOM должна измениться, так как изменятся объекты на странице. Но реальная объектная модель может быть огромной, ее обновление — медленный процесс. Поэтому React работает не с ней, а с виртуальной копией в кэше, которая весит меньше.

Когда происходит событие, из-за которого код должен обновить объект, изменение быстро отображается в виртуальном DOM. После этого обновляется реальная объектная модель. Для пользователя это значит, что изменения на странице отобразятся мгновенно, а не после долгой загрузки.

### **Обновление DOM по частям**

Чтобы улучшить быстродействие, React обновляет DOM не полностью. Он хранит в памяти две облегченных копии: актуальную и предыдущую. Когда что-то обновляется, библиотека сравнивает версии между собой и изменяет только ту часть дерева, которая действительно поменялась. Это нужно, чтобы не перезагружать DOM целиком и не замедлять страницу. Подход кажется сложным, но он важен для оптимизации загрузки.

### **Возможность повторно использовать компоненты**

React основан на компонентах — отдельных элементах веб-интерфейса. Компоненты инкапсулированы, то есть самостоятельны: в каждом из них размещены все необходимые методы и данные. Подробнее про инкапсуляцию можно почитать в нашей статье про ООП: это один из ключевых критериев объектно-ориентированного подхода. В случае с компонентами React инкапсуляция также означает, что состояние элемента хранится в нем самом, а не в отдельном объекте.

Инкапсулированные самостоятельные компоненты можно использовать повторно, размещать в другом месте кода, в ином разделе или на другой странице. Данные можно переносить по всему приложению, использовать вне DOM конкретной страницы. Это ускоряет разработку и сокращает количество действий для создания функционирующего интерфейса. Благодаря отсутствию сложных зависимостей инкапсуляция также облегчает отладку.

### **Нисходящий поток данных**

Компоненты могут передавать свойства и данные друг другу, но только в одном направлении — от «родительских» к дочерним. Это помогает реализовать четкую иерархию, облегчает отладку. Однонаправленный поток данных означает, что программист всегда может понять, откуда именно к элементу поступили данные.

Еще одна особенность потока данных — отсутствие возможности изменять свойства напрямую. Для этого существуют специальные callback-функции. Так называются части исполняемого кода, которые передаются от одного компонента другому в виде параметра функции. Это позволяет поддерживать стабильность: свойства оказываются неизменяемыми после создания, а callback-функция как бы пересоздает их. В результате использования коллбэков поток данных выглядит так: свойства и информация передаются сверху вниз, а события происходят снизу вверх.

### **Синтаксис JSX**

JSX расшифровывается как JavaScript XML. Это расширение языка JavaScript, которое помогает описывать HTML-подобные элементы с помощью кода на React. С помощью синтаксиса на React создают компоненты страницы и гибко управляют ими.

Несмотря на то что элементы похожи на HTML, это по-прежнему язык JavaScript с возможностью быстро и легко изменять DOM с помощью кода. И все же JSX воспроизводится как HTML: по сути разработчик описывает нужный компонент на языке разметки, а тот остается JavaScript-объектом с широкой функциональностью. Это удобно, упрощает программирование, но может запутать начинающих.

### **React Hooks**

В старых версиях управлять состояниями можно было с помощью классов — специальных конструкций, о которых можно подробнее прочесть в статье про ООП. Сейчас в React.js есть поддержка хуков — так называются специальные функции-«крючки», которые «цепляются» за состояние элемента или за метод. Изменение состояния или вызов метода «тащит» за собой эти функции, и они автоматически выполняются — это помогает избежать использования классов, облегчает и упрощает написание кода.

В React есть встроенные хуки, но разработчик может создавать и собственные — это помогает повторно использовать код и уменьшить его количество.

### **React Developer Tools**

Так как React.js — очень популярная технология, его создатели разработали бесплатные расширения для браузера с инструментами для проверки и отладки. Фронтендеры часто пользуются консолью и панелью разработчика в браузере, чтобы проверить, как работает их код. React Developer Tools облегчают задачу и расширяют возможности. Например, можно просматривать прямо в браузере компоненты с большим уровнем вложенности и не искать их в коде долгое время. React Developer Tools существуют для новых версий популярных браузеров Firefox и Google Chrome.

## **Преимущества React**

### **Популярность**

Это один из трех самых распространенных инструментов для фронтенд-разработки. Кроме него, популярны Vue.js и Angular, но первый пока не так распространен, а второй намного сложнее в изучении. Еще одна популярная технология — jQuery, но она постепенно уходит в прошлое. На React написано множество сайтов и приложений. Ее используют такие крупные компании, как Яндекс, Uber, Сбербанк, Авито, BBC, Airbnb, Netflix и другие. Вакансий много, и они высоко оплачиваются даже на начальных уровнях.

### **Огромное сообщество**

Это еще одно следствие популярности: React-разработчиков много, в том числе опытных, многие из них пишут туториалы и статьи, помогают новичкам влиться в процесс разработки. В ходе обучения начинающий всегда может обратиться на тематические ресурсы, где ему постараются помочь. Людей в сообществе хватает, можно получить ответы на большинство вопросов. Документация поддерживается и обновляется, существует в том числе на русском языке.

### **Развитая экосистема**

Создатели React активно побуждали пользователей вкладываться в развитие проекта, и энтузиасты писали свои технологии для совместного использования с библиотекой. В результате сейчас сформировалась целая экосистема: десятки библиотек и других инструментов, которые можно использовать в разработке. Это делает процесс проще и легче, позволяет решать огромное количество задач.

### **Простота создания**

С помощью компонентов проекта можно быстро и легко собрать интерактивный, чутко реагирующий на любые изменения интерфейс сайта или приложения любой сложности. Это намного проще, чем прописывать реакции на всевозможные события вручную, и снижает количество ошибок. Так как компоненты можно использовать повторно в любом месте кода, задача становится еще легче. Например, для создания сложного многостраничного сайта или приложения не нужно многократно писать один и тот же компонент.

### **Реактивность**

Это одно из ключевых преимуществ проекта, вынесенное в название. Библиотека реагирует на обновление компонента и автоматически отображает его изменения в дереве документа. Изменения могут происходить в ответ на действия пользователя, какие-то внешние перемены или другие события. В результате сайты и приложения становятся более привлекательными для пользователя.

### **Эффективность**

Благодаря виртуальному DOM библиотека экономит ресурсы. Чтобы изменить состояние элементов, не нужно полностью перезагружать все DOM-дерево, лишний раз тратить трафик пользователя и нагружать браузер. Изменяются только конкретные элементы, это происходит через виртуальное DOM-дерево — сайты и приложения становятся «легче» и удобнее.

### **Высокая скорость**

Сайты и приложения, написанные на React, быстрые и отзывчивые благодаря виртуальному DOM. Дерево занимает меньше места, быстрее обновляется.

Также ускоряется работа программистов. Компоненты легко создавать и переиспользовать, между ними уже продумана логика сообщения. Главная задача — грамотно описать состояния. Не нужно прописывать всю логику и принципы с нуля, это за разработчика уже сделали создатели React.

### **Удобная отладка**

Ошибок с React бывает меньше, чем при написании кода на «чистых» языках без библиотек. Это происходит, потому что многое реализуется автоматически, соответственно, уменьшается влияние «человеческого фактора». Но даже если при разработке программист допустил ошибку, ее довольно легко отследить и исправить благодаря четкой логике потока данных, дополнительным инструментам, понятному синтаксису. Код на React легко читать, понимать и отлаживать.

## **Недостатки React**

### **Запутанность синтаксиса**

Технология JSX удобная и широко используется, но сначала она может вызвать сложности в изучении. Людям непривычно «смешивать» друг с другом HTML и JavaScript, поэтому в синтаксисе легко запутаться и ошибиться. Со временем путаница и ошибки проходят — к этой особенности нужно просто привыкнуть.

### **Неполное соответствие MVC**

React отвечает только за часть View. Поэтому для реализации модели потребуется подбирать стек из разных технологий. React можно использовать практически с любым стеком.

Это одна из причин, по которой React иногда называют библиотекой, а не фреймворком: по одному из определений фреймворк должен позволять полностью реализовать ту или иную модель.

### **Трудности с SEO-оптимизацией**

React-компоненты намного сложнее оптимизировать в рамках SEO. SEO — это поисковая оптимизация, проработка сайта таким образом, чтобы он соответствовал требованиям поисковых систем. Если сайт находится на первой странице поиска по тематическим запросам, его посетят больше пользователей.

Чтобы поисковая система «заметила» сайт, страницы должен проиндексировать поисковый робот. Роботу проще индексировать HTML. React-компоненты, написанные на JavaScript, индексируются тяжелее и медленнее, поэтому страницы сложнее оптимизировать.

## **Как начать пользоваться React**

Для начала новичку нужно на достаточном уровне изучить JS. Кроме «чистого» JavaScript, React поддерживает TypeScript. Но для работы с React знать TypeScript необязательно.

После изучения JavaScript новичок может начинать работу с проектом. У официального сайта есть русская версия, на ней подробно описаны способы, как это можно сделать. Можно воспользоваться онлайн-песочницей или подключить документ с библиотекой к страницам сайта — для тренировки и обучения подойдет любой вариант.

На ранних этапах начинающему разработчику может хватить новых версий документации на русском языке. Чтобы получить полноценную практику и глубоко изучить библиотеку, потребуется помощь наставника.