МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра информатики и программирования

Разработка библиотеки для построения адаптивных графиков

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 441 группы
направления 02.03.03 Математическое обеспечение и администрирование
информационных систем
факультета компьютерных наук и информационных технологий
Роганова Давида Людвиковича

Научный руководитель:
старший преподаватель _____ (подпись, дата) М. С. Портенко

Консультант:
ведущий программист ЦОПП _____ (подпись, дата) Ю. Ю. Соломина

Зав. кафедрой:
к.ф-м.н. доцент _____ (подпись, дата) М. В. Огнева

Саратов 2019

ВВЕДЕНИЕ

Актуальность темы. Информация может быть представлена в различном виде – текстовом, графическом, звуковом и т. д. В зависимости от задачи и природы самих данных применяется те или иные виды. Для представления большого объёма данных и проведения анализа часто используются диаграммы – графическое представление с помощью геометрических фигур.

На сегодняшний день современные устройства и Интернет не только привносят новые способы взаимодействия с данными, перенося их с бумаги в виртуальное пространство (будь то монитор, дисплей мобильного устройства, виртуальная или дополненная реальность), но также и значительно увеличивают объём создаваемой информации. И, в то время как графики остаются одним из наиболее эффективных средств для визуализации и анализа данных, современные технологии предоставляют новые виды диаграмм и дополнительные возможности по работе с ними:

- Трёхмерные графики, открывающие ранее недоступный опыт взаимодействия с данными;
- Динамическое масштабирование;
- Динамическое изменение типа диаграммы для одного и того же набора данных, позволяющее адаптировать визуализацию под различные задачи;
- Возможность работы на различных устройствах;
- Путешествие по истории данных;
- Адаптация работы для людей с ограниченными возможностями;
- Одновременная работа с большим количеством параметров.

Одно из наиболее популярных и активно развивающихся направлений в информационных технологиях на сегодняшний день – это веб-разработка. В первую очередь это обусловлено следующими факторами:

- Регулярное использование пользователями браузеров;
- Более ограниченный набор технологий для разработки, чем для настольных приложений, если учитывать поддержку разных платформ;

- Кроссплатформенность – браузеры есть почти на всех устройствах, следовательно, приложение может быть запущено на любом из них;
- Значительное расширение за последнее время возможностей веб-приложений, которые приближают их по функциональности к настольным: например, доступ к API некоторых физических устройств (акселерометр, камера, проверка уровня освещения), интеграция с бесконтактными средства оплаты, возможность работы при отсутствии подключения сети («в оффлайне») и др.

В связи с такой популярностью веб-приложений естественным образом появляется также и задача визуализации и реализации интерактивного взаимодействия с данными в браузерах, при чём не только настольных («десктопных»), но и мобильных, так как доля пользователей носимых устройств растёт.

Цель бакалаврской работы – разработка библиотеки на языке JavaScript для отображения высокопроизводительных адаптивных графиков с поддержкой обновления в реальном времени.

Поставленная цель определила **следующие задачи**:

1. Проанализировать готовые решения: их объективные плюсы и минусы, соответствие поставленным требованиям в контексте работы;
2. Оценить рациональность разработки собственного решения вместо использования готовых;
3. Рассмотреть и выбрать вспомогательные инструменты для разработки;
4. Спроектировать основные компоненты и архитектуру библиотеки;
5. Разработать программную реализацию;
6. Оптимизировать разработанное решение для возможности интеграции в реальные приложения и настроить для использования в сборщике проектов.

Практическая значимость бакалаврской работы заключается в разработке библиотеки, которая выполняет актуальные задачи по визуализации данных и удовлетворяет требованиям для применения в специфичной области (например, для отображения геофизических данных).

Структура и объём работы. Бакалаврская работа состоит из введения, пяти разделов, заключения, списка использованных источников и трёх приложений. Общий объем работы – 67 страниц, из них 42 страницы – основное содержание, включая 10 рисунков и 1 таблицу, цифровой носитель в качестве приложения, список использованных источников информации – 20 наименований.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

Первый раздел «Обзор готовых библиотек для построения графиков» посвящён рассмотрению уже существующих популярных решений для отображения графиков. Так же приводится их сравнительная характеристика по следующим критериям:

- возможность бесплатного использования,
- размер библиотеки,
- поддержка вертикальной ориентации графиков,
- производительность на большом объёме данных,
- прокрутка по истории значений,
- простота изменения внешнего вида,
- поддержка мобильных устройств,
- работа с параллельными колонками,
- возможность использования в библиотеке React,
- поддержка логарифмической шкалы,
- поддержка трёхмерных графиков.

В результате сравнения возможностей готовых решений был сделан вывод, что ни одно из них полностью не удовлетворяет поставленным требованиям.

Второй радел «Выбор инструментов для разработки» посвящён выбору вспомогательных инструментов, которые будут использоваться в разработке собственной библиотеки. Были рассмотрены решения для следующих задач:

- основа для приложения, абстракция над DOM-деревом,
- работа с векторными изображениями и расчётами для отрисовки графиков,
- работа с трёхмерной графикой.

В качестве основы приложения была выбрана библиотека React по следующим причинам:

- самое популярное на данный момент решение;
- поддерживается крупной компанией;
- обладает большим сообществом и богатой экосистемой;
- имеет достаточно подробную документацию, большое количество обучающих материалов и статей;
- обладает хорошей обратной совместимостью с предыдущими версиями, благодаря чему обновление версии библиотеки минимально затрагивает существующий код;
- построена вокруг идеи функциональной парадигмы программирования, которая хорошо ложится на разработку пользовательских интерфейсов;
- малый размер, незначительно влияющий на скорость загрузки приложения;
- переносимость React-кода на другие платформы.

Для работы с векторной графикой была выбрана библиотеки D3, а с трёхмерной – Three.js. Это обусловлено тем, что они проверены временем, являются одними из наиболее популярных решений в своей области, а также обладают обширным сообществом и большим количеством обучающих материалов.

Третий раздел «Обзор используемых библиотек» посвящён обзору выбранных в предыдущем разделе вспомогательных решений и их возможностей.

React – библиотека для построения пользовательских интерфейсов. Она является ориентированной на разработку с использованием компонентного подхода. В основе так же лежит декларативный подход для описания компонентов в зависимости от их текущего состояния. Кроме того, React-код может быть перенесён на другие платформы, например iOS или Android.

Так же React предоставляет абстракцию над DOM-деревом (в случае веб-приложений), которая называется VirtualDOM и упрощает работу разработчика.

D3 – библиотека для управления и визуализации данных. Она разбита на множество компонентов, которые также можно использовать отдельно. Для разработки собственной библиотеки из них понадобились два:

- d3-scale: применяется для построения шкалы, которая представляет собой функцию, отображающую значения из диапазона данных в диапазон, представляющий визуальную позицию этих данных;
- d3-shape: позволяет строить различные геометрические фигуры на основе данных (линии, области, дуги).

Three.js – это библиотека, упрощающая взаимодействие с WebGL API. Оно позволяет получать доступ к графическому аппаратному обеспечению из JavaScript и рендерить вывод на веб-странице в элемент HTML canvas. Благодаря этой технологии в браузерах возможно полноценно использовать трёхмерную графику без подключения сторонних плагинов (например, Adobe Flash).

Изучив данный набор инструментов, был сделан вывод, что их возможностей достаточно, чтобы выполнить поставленную задачу по разработке собственной библиотеки для работы с графиками.

Четвёртый раздел «Архитектура разработанной библиотеки» посвящён описанию структуры разработанной библиотеки и компонентов, которые доступны для использования.

В основе лежит компонент Chart, который является общим контейнером для графиков и содержит основную логику по расчёту видимой области, прокрутке истории данных, выводу всплывающих подсказок для значений.

Компонент Column – контейнер, позволяющий сгруппировать различные графики в одну колонку. Помимо визуального разделения, колонка также предоставляет общую шкалу, по которой выстраиваются значения графиков внутри.

Группа компонентов Shapes включает в себя непосредственно графики, которые определяют способ визуализации переданных данных – в виде линии, области, текста или области с накоплением.

В основе построения графиков с помощью разработанной библиотеки лежит принцип композиции компонентов, описанных выше. Пример использования библиотеки:

```
<Chart min={1500} max={4100} zoom={15}>
    <Column scaleType=”linear>
        <Shapes.Line data={data1} color=”white”>
        <Shapes.Line data={data2} color=”cyan”>
    </Column>
    <Column scaleType=”log”>
        <Shapes.Area data={data3} color=”#bcae36”>
    </Column>
    <Column>
        <Shapes.Text data={data4} color=”#1f90e7”>
    </Column>
</Chart>
```

Так же в состав компонентов библиотеки входит ThreeDimensionalChart, который позволяет работать с трёхмерными графиками и может быть использован следующим образом:

```
<ThreeDimensionalChart
    data={data}
    colors={colors}
    xAccessor={d => d.x}
    yAccessor={d => d.y}
    zAccessor={d => d.z}
    zeroCrossing
/>
```

В результате подобная архитектура и компонентный подход позволяют декларативно описывать графики, а также дают возможность расширять возможности готовых библиотечных компонентов и создавать новые.

Кроме того, были сделаны следующие оптимизации для повышения производительности:

- Кеширование данных по интервалам при работе с удалёнными источниками;
- Фильтрация точек, позволяющая разредить слишком плотный график, значительно уменьшая количество точек при сохранении общей картины;
- Одновременное отображение только видимых точек.

Пятый раздел «Сборка и оптимизация кода» посвящён адаптации кодовой базы разработанной библиотеки для использования в современных сборщиках веб-приложений. Здесь рассмотрены такие способы оптимизации, как:

- Code splitting. Заключается в разбиении кода и последующей загрузке только тех фрагментов, которые в данный момент нужны пользователю. Этот способ хорошо проявляет себя в приложениях с навигацией, где

каждой страницы соответствует свой скрипт, который подгружается при переходе на неё;

- Tree shacking – подход, позволяющий удалять неиспользуемый код и работающий благодаря модульной системе из стандарта EcmaScript. В процессе построения дерева зависимостей сборщик отслеживает, какой код был импортирован/экспортирован, но ни разу не использован, и помечает его. В дальнейшем отмеченные классы, функции, переменные удаляются из финальной сборки;
- Минификация кода – способ, состоящий в удалении незначащих символов (в основном пробельных), заменой названий идентификаторов на более короткие, сокращение языковых конструкций. Данный способ позволяет уменьшить размер файлов с кодом, чтобы ускорить их загрузку по сети.

Разработанная библиотека была подготовлена для использования Tree shacking в проектах, в которые она будет подключаться. Это позволит не включать в финальную сборку код неиспользуемых компонентов, сокращая размер загружаемых и исполняемых файлов.

ЗАКЛЮЧЕНИЕ

В данной работе были рассмотрены различные библиотеки для построения графиков в веб-приложениях, как платные, так и бесплатные. В ходе их анализа был сделан вывод, что они не полностью удовлетворяют требованиям к производительности и адаптивности (оптимизации работы на мобильных устройствах).

В результате была разработана собственная библиотека для работы с диаграммами. В ходе разработки были исследованы и выбраны базовые инструменты, проведена работа по оптимизации быстродействия на уровне исполнения кода, а также на этапе сборке проекта.

Использование в основе библиотеки React также позволяет ей естественным образом интегрироваться в единую экосистему других готовых компонентов и большого сообщества, которое за ней стоит.

Благодаря декомпозиции на отдельные компоненты возможно также описывать графики декларативно, подобно HTML-документам. Другая возможность, которая следует отсюда – это расширяемость функциональности. Пользователь библиотеки не только может стилизовать графики по своему вкусу, но также имеет возможность разработать собственные компоненты для всплывающей подсказки, линии или нового типа графиков. При соблюдении внутреннего программного интерфейса они могут быть использованы вместо стандартных компонентов, расширяя функциональность под потребности пользователя.

Основные источники информации:

1. Techjury [Электронный ресурс]: сайт. URL: https://techjury.net/stats-about/mobile-vs-desktop-usage/ (дата обращения: 22.05.2019). Загл. с экрана. Яз. англ.
2. JavaScript Charts & Maps – amCharts [Электронный ресурс]: сайт. URL: https://www.amcharts.com/ (дата обращения: 22.05.2019). Загл. с экрана. Яз. англ.
3. Interactive JavaScript Charts for your web page [Электронный ресурс]: сайт. URL: https://www.highcharts.com/ (дата обращения: 22.05.2019). Загл. с экрана. Яз. англ.
4. Open source HTML5 Charts for your website [Электронный ресурс]: сайт. URL: chartjs.org/ (дата обращения: 22.05.2019). Загл. с экрана. Яз. англ.
5. Recharts [Электронный ресурс]: сайт. URL: http://recharts.org/ (дата обращения: 22.05.2019). Загл. с экрана. Яз. англ.

6. React – A JavaScript Library for building user interfaces [Электронный ресурс]: сайт. URL: https://reactjs.org/ (дата обращения: 22.05.2019). Загл. с экрана. Яз. англ.
7. VirtualDOM and Internals – React [Электронный ресурс]: сайт. URL: https://reactjs.org/docs/faq-internals.html#what-is-the-virtual-dom (дата обращения: 22.05.2019). Загл. с экрана. Яз. англ.
8. Reconciliation – React [Электронный ресурс]: сайт. URL: https://reactjs.org/docs/reconciliation.html (дата обращения: 22.05.2019). Загл. с экрана. Яз. англ.
9. WebGL Specification [Электронный ресурс]: сайт. URL: https://www.khronos.org/registry/webgl/specs/latest/1.0/ (дата обращения: 22.05.2019). Загл. с экрана. Яз. англ.
10. Code splitting [Электронный ресурс]: сайт. URL: https://webpack.js.org/guides/code-splitting/ (дата обращения: 23.05.2019). Загл. с экрана. Яз. англ.
11. Tree shacking [Электронный ресурс]: сайт. URL: https://webpack.js.org/guides/tree-shaking/ (дата обращения: 22.05.2019). Загл. с экрана. Яз. англ.