УДК 004.021

ПРОЕКТИРОВАНИЕ МАСШТАБИРУЕМОГО ГРАФИЧЕСКОГО ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА НА ОСНОВЕ REDUX

М. А. Щербаков
Студент БГУИР КСиС, инженер-программист, Центр информатизации и инновационных разработок

С. Н. Нестеренков
Кандидат технических наук, начальник отдела информационных технологий центра информатизации и инновационных разработок БГУИР, доцент кафедры Программного обеспечения информационных технологий

Центр информатизации и инновационных разработок Белорусского государственного университета информатики и радиоэлектроники, Республика Беларусь.
Белорусский государственный университет информатики и радиоэлектроники, Республика Беларусь.
E-mail: scherbakov_ma_1@mail.ru, s.nesterenkov@bsuir.by.

М. А. Щербаков.

Является студентом 4-ого курса кафедры программного обеспечения информационных технологий. Проходит производственную практику в Центре информатизации и инновационных разработок в Белорусском государственном университете информатики и радиоэлектроники. Проводит научные исследования в области использования Redux шаблона для проектирования пользовательского интерфейса.

С. Н. Нестеренков.

Кандидат технических наук, начальник отдела информационных технологий центра информатизации и инновационных разработок Белорусского государственного университета информатики и радиоэлектроники, доцент кафедры Программного обеспечения информационных технологий. Многократный автор публикаций на тему машинного обучения, алгоритмов принятия решений, искусственных нейронных сетей и автоматизации.

Аннотация. В данном докладе будет рассмотрено использование перспектив использования шаблона дизайна Redux в настольных графических приложениях. Также будут освещены преимущества использования данного подхода к проектированию прикладных приложений в условиях современной вычислительной многопоточной техники.

Ключевые слова: графические пользовательские интерфейсы, Redux, разбиение приложения для многопоточного выполнения, масштабируемые графические приложения.

Введение.

Первые графические пользовательские интерфейсы появились на свет в 70-х годах прошлого века. Возможности вычислительной техники на тот момент ограничивались одним вычислительным процессором, на котором выполнялась как бизнес-логика программного обеспечения, так и отрисовка пользовательского графического интерфейса. Прогресс не стоял на месте, стали появляться всё более сложные приложения, а вычислительная мощность выросла до нескольких процессоров, способных выполнять несколько операций одновременно. В связи с этим возникли проблемы масштабируемости и связности данных, хранимых для отображения в пользовательском интерфейсе и необходимых для работы с бизнес-логикой приложения. Также

одной из больших проблем в проектировании приложений стала задача распараллеливания потоков выполнения и отделение графического потока от потока бизнес-логики.

Redux – шаблон дизайна.

Одним из подходов к решению многих проблем, в частности распараллеливанию задач, вынесению отрисовки графических элементов в отдельный поток, является Redux. Redux – это шаблон дизайна, который предполагает единый источник правды (SSoT) для бизнес-логики и графических компонентов, называемый «Store», доступ на чтение к которому имеют как графические компоненты, так и бизнес-логика, а доступ на запись отсутствует вовсе. Вместо записи в Store напрямую используется система событий и их обработчиков. События называются «Actions», а обработчики называются «Reducers». Actions представляют собой обычные объекты с полями, одним из обязательных полей которых является поле «type», т. е. поле типа события. Reducers – это обычные чистые функции, которые принимают Store (оно же глобальное состояние) в качестве первого аргумента, а в качестве второго принимают Actions. Единственное предназначение Reducers заключается в изменении состояния Store в соответствии с Actions.

Использование Redux графическим интерфейсом.

При проектировании графических компонент, нужно предусмотреть некоторый общий стандартный интерфейс, который будет использоваться для уведомления компонент об изменениях состояний Store. При изменении состояния Store вызывается специальный метод у графических компонент, который сообщает, что состояние было изменено. Компоненты же в свою очередь имеют свободный доступ к этому состоянию, могут считать его и выполнить отрисовку нового содержимого при необходимости. Подход чтения состояния и отсутствие возможности записи напрямую позволяет избежать синхронизации между потоками, которые отрисовывают пользовательский интерфейс, и потоками, в которых выполняются Reducers, бизнес-логика и другая полезная логика программы, что позволяет добиться минимальных подтормаживаний при отрисовке.

Существуют такие важные события, как нажатия кнопок, ввод текста, движение курсора мышки и т. д. Для обработки таких событий существуют Actions. Для каждого типа заводиться свой собственный Action, который содержит помимо обязательного поля type еще поля с полезной нагрузкой, например, координатами мышки, зажатыми клавишами и т. д. После этого такое событие ставится в очередь. Этот процесс постановки в очередь выполняется максимально быстро и требует минимальной синхронизации, что позволяет добиться больших скоростей обработки.

Использование Redux бизнес-логикой.

Поскольку Reducers используются лишь с одной целью – выполнить изменение состояния в соответствии с подготовленными данными, выполнять в них бизнес-логику приложения нельзя. Для выполнения бизнес-логики приложения используется прослойка между очередью поступивших Actions и соответствующими для Actions обработчиками – «Middleware». Такая прослойка позволяет подписываться на происходящие события и выполнять тяжелые вычисления асинхронно. Достигается это за счет поглощения исходного Actions, постановки тяжеловесной задачи в очередь на выполнение, например, в пул потоков. Обработка продолжается дальше до тех пор, пока не будет выполнено асинхронное или тяжелое вычисление. Как только вычисление будет выполнено, оно заново породит Action со всеми необходимым для Reducer данными.

Область применения при разработке.

Рассмотренный шаблон дизайна может быть применен при проектировании приложений с графическим интерфейсом с использование широкого спектра технологий. Одной из уже реализовавших данный шаблон дизайна является ReactJS. Это веб-фреймворк, позволяющий проектировать графические компоненты для динамических сайтов. Также, данный шаблон может быть реализован с использованием таких языков и технологий, как C++, C#, Ruby, Python и других языках.

Заключение.

Redux – мощный шаблон дизайна, позволяющий расширять и пополнять библиотеку графических компонентов быстро и надежно, поскольку ни один из компонентов не изменяет

состояние программы напрямую, а лишь имеет доступ на чтение к нему. Также, использование данного шаблона позволяет разделять потоки выполнения графических компонент, изменения глобального состояния, выполнения бизнес-логики.

Список литературы

[1] Гращенко Л. А. Обобщенная модель угроз информационной безопасности визуальных интерфейсов пользователя // Известия Орловского государственного технического университета. Серия: Информационные системы и технологии. – 2016. – №. 1. – С. 41-45.

[2] Kunle Olukotun. Chip Multiprocessor Architecture - Techniques to Improve Throughput and Latency. – Morgan and Claypool Publishers, 2017. – 154 p. – ISBN 159829122X.

[3] Mario Nemirovsky, Dean M. Tullsen. Multithreading Architecture. – Morgan and Claypool Publishers, 2013. – 1608458555 p. – ISBN 1608458555.

[4] Гэри Неббет. Справочник по базовым функциям API Windows NT/2000 = Windows NT/2000 Native API Reference. – М.: «Вильямс», 2002. – С. 528. – ISBN 1-57870-199-6.

[5] Алекс Бэнкс. React и Redux: функциональная веб-разработка. – СПб.: Питер, 2017. – 336 с. – (Бестселлеры O'Reilly). – ISBN 9785446106684.

[6] Zandstra M. PHP Objects, Patterns, and Practice. – 5th Edition. – «Apress», 2016. – С. 583. – ISBN 978-1-4842-1995-9.

[7] Зандстра М. PHP. Объекты, шаблоны и методики программирования. – 5-е изд.. – СПб.: «Диалектика», 2019. – С. 736. – ISBN 978-5-907144-54-5.

[8] Фаулер, Мартин. Рефакторинг кода на JavaScript: улучшение проекта существующего кода. – 2-е изд.. – СПб.: «Диалектика», 2019. – С. 464. – ISBN 978-5-907144-59-0.

[9] Бертош, В.А. Внедрение интерактивных и интернет технологий в образовательный процесс / В.А. Бертош, А.Г. Хачатрян, С.Н. Нестеренков // Проблемы повышения эффективности образовательного процесса на базе информационных технологий = Problems of improving the efficiency of the educational process based on information technology : материалы XII Междунар. науч.-практ. конф., Минск, 25 апреля 2019 г. / Белорус. гос. ун-т информатики и радиоэлектроники ; редкол.: Ю.Е. Кулешов [и др.]. - Минск, 2019. - С. 14-17.

[10] Нестеренков, С. Н. Использование сверточных неиронных сетеи для классификации и анализа тональности текстов / С. Н. Нестеренков, П. А. Федоров, В. А. Денисов // Информационные технологии и системы 2019 (ИТС 2019) : материалы междунар. науч. конф., Минск, 30 окт. 2019 г. / Белорус. гос. ун-т информатики и радиоэлектроники ; редкол.: Л. Ю. Шилин [и др.]. – Минск, 2019. – С. 248-249.

DESIGNING A SCALABLE GRAPHIC USER INTERFACE BASED ON REDUX

M.A. SCHERBAKOV
BSUIR KSiS student, software engineer, Center for Informatization and Innovation Development

S.N. NESTERENKOV,
Candidate of Technical Sciences, Head of the Information Technology Department of the Center for Informatization and Innovative Developments of BSUIR, Associate Professor of the Department of Information Technology Software

Center for Informatization and Innovative Developments of the Belarusian State University of Informatics and Radioelectronics, Republic of Belarus
Belarusian State University of Informatics and Radioelectronics, Republic of Belarus
E-mail: scherbakov_ma_1@mail.ru, s.nesterenkov@bsuir.by

Abstract. This talk will look at the perspectives of using the Redux design pattern in desktop graphical applications. The advantages of using this approach to the design of applied applications in the context of modern computing multithreaded technology will also be highlighted.

Keywords: graphical user interfaces, Redux, application splitting for multithreading, scalable graphical applications.