УДК 004.777+004.272

ОСОБЕННОСТИ РАЗРАБОТКИ ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ ПУБЛИКАЦИИ РАССКАЗОВ С ИСПОЛЬЗОВАНИЕМ АРХИТЕКТУРНОГО ШАБЛОНА ПРОЕКТИРОВАНИЯ MVC

Богомаз Д.Л., Ярмолик М.А.

Белорусский государственный университет информатики и радиоэлектроники, г. Минск, Республика Беларусь

Научный руководитель: Горбач А.П. – магистр техн. наук, старший преподаватель

Аннотация. Выявлены главные положительные стороны использования MVC-шаблона на примере приложения на базе ASP.NET Core. Рассмотрено использование фреймворка SignalR на платформе ASP.NET Core на языке программирования TypeScript. Рассмотрены различия динамической и статической типизации на примере языков программирования TypeScript и JavaScript.

Ключевые слова: MVC, веб-приложение, шаблон проектирования, SignalR

Введение. Тенденция последних лет заключается в том, что все больше веб-сайтов представляют собой веб-приложения. Их основной особенностью является связь с сервером. Веб-приложение использует для этого HTTP-запросы. Веб-сайты также не предназначены для взаимодействия с пользователем. По мере совершенствования браузеров веб-приложения становятся более популярными. С ростом приложений количество запросов не изменяется, однако размер запроса со временем только увеличивается [1].

Шаблон проектирования MVC – это предсказуемый шаблон проектирования, который поддерживается большим количеством фреймворков и используется пори создании веб-приложений и мобильных приложений [2].

В данной статье авторами показаны преимущества паттерна MVC и сравнение его с другими архитектурными паттернами на примере его использования в ASP.NET.

Основная часть. Шаблон проектирования MVC предлагает множество преимуществ по сравнению с обычным JavaScript. Он позволяет писать более организованный и, соответственно, легче поддерживаемый код. Этот шаблон использован и протестирован в большом количестве проектов на протяжении нескольких поколений.

MVC состоит из 3-ёх ключевых компонентов: контроллер (controller), модель (model) и вид (view).

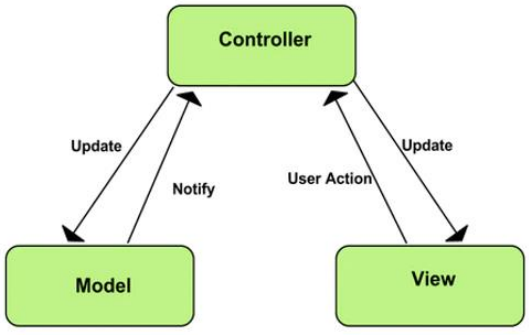

Рисунок 1 – Схема взаимодействия компонентов MVC

Модель – это тот компонент, который хранит информацию или данные приложения. Этот компонент абсолютно независимый, самостоятельный. Обычно, если в модели происходят изменения, она уведомит “слушателей” о них.

Вид – компонент, предоставляемый пользователю для его взаимодействия с приложением.

Контроллер – ядро всего приложения. Он обрабатывает запросы, поступающие от вида, обновляет вид, обеспечивает перенаправление пользователя, добавляет “слушателей событий” и обновляет модель.

Есть несколько вариаций шаблона MVC: MVP (Model-View-Presenter) и MVVP (Model-View-ViewModel).

Таким образом, шаблон MVC делает приложение модульным и позволяет:

- писать легко расширяемый и легко поддерживаемый код;
- разделять обязанности;
- разрабатывать контроллер, вид, модель параллельно, независимо.

Недостатками архитектуры MVC является сложность понимания архитектуры MVC, строгость правил в методах. Недостатки архитектуры являются очень несущественными в сравнении с её положительными сторонами.

В проекте по публикации рассказов каждая часть приложения имеет свой код, как, например, администрирование, изменение рассказов, изменение глав, домашний контроллер для главной страницы, контроллер главной страницы. Это позволяет иметь хорошую модульность всех частей приложения. Каждая логичная сущность являлась моделью. Большая часть моделей хранились в базе данных.

SignalR – библиотека, которая позволяет добавить в своё приложение функциональность в режиме реального времени. Это означает, что сервер может незамедлительно отправлять данные подключённым клиентам.

Программный интерфейс SignalR позволяет серверу вызывать функции JavaScript подключенных клиентов при помощи “remote procedure calls” (RPCs).

SignalR берет на себя ответственность за управление подключениями. Он умеет автоматически масштабироваться в зависимости от количества подключённых клиентов.

Для поддержки обмена данными в режиме реального времени используются: веб-сокеты, отправляемые сервером события, Long Polling.

Вышеперечисленные компоненты называются транспортами (transports). Транспорты позволяют разработчикам сфокусироваться на поставленных задачах, а не на том, как передаются данные на более низком уровне. SignalR способен динамически определять, какие транспорты поддерживаются и выбирать наиболее подходящий.

После установки соединения, SignalR начинает периодически отправлять сообщения для проверки состояния подключения. В случае, если ответа не последует, будет сгенерировано исключение.

SignalR использует хаб для передачи данных между серверами и клиентами. Хаб позволяет клиенту и серверу вызывать методы друг друга. Хабы не постоянны. Каждый вызов метода хаба происходит на новом объекте. Поэтому хранить данные там нельзя.

В веб-приложении для публикации рассказов с помощью SignalR были созданы комментарии. Это сделано для моментального добавления и удаления комментариев сразу у всех пользователей.

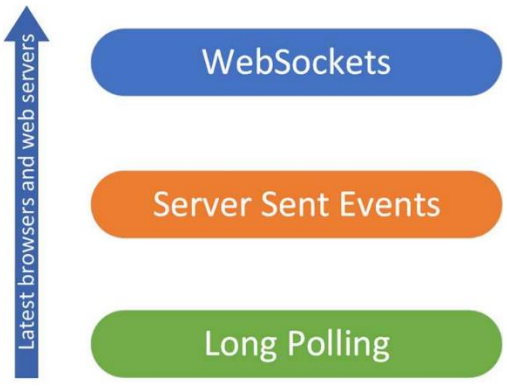

Рисунок 2 – Схема транспортов SignalR

Сейчас язык программирования TypeScript набирает популярность в сравнении с JavaScript. Главным различием языков является типизация. В TypeScript используется статическая типизация, а в JavaScript - динамическая.

Динамически типизированные языки не требуют указывать тип, но и не определяют его сами. Благодаря гибкости кода во время выполнения и интроспекции получается на порядок проще и быстрее писать универсальные алгоритмы и конструкции, реализовывать шаблоны проектирования более гибко. Это сильно упрощает интерфейсы библиотек, что в совокупности ведёт к более простому коду и легкому его пониманию.

Главным преимуществом статической типизации будет являться однозначное определение типа переменной, что позволяет дать компилятору больше информации для оптимизации кода.

Недостатком проектирования архитектуры на динамических языках программирования являются ошибки преобразования типов, отладка которых является тяжелым процессом. Также на этапе поддержки приложения добавление нового функционала и расширение приложения может повлечь за собой ошибки в других участках кода.

Именно по этим причинам большинство фреймворков сейчас пишутся именно на языке TypeScript. Некоторые системы требуют переписать огромные части приложения из-за серьезных ошибок типов.

Заключение. MVC-паттерн является отличным решениием для большинства веб-приложений. Использование его на платформе ASP.NET Core с TypeScript и SignalR позволяет получить максимальную модульность и расширяемость в поддержке. Это сможет быть отличным решением для поддержания проекта в дальнейшем.

Список литературы

1. *MVC Architecture [Электронный ресурс]. - Режим доступа https://developer.chrome.com/docs/apps/app_frameworks/.*

2. *MVC Architecture in 5 minutes: a tutorial for beginners [Электронный ресурс]. - Режим доступа https://www.educative.io/blog/mvc-tutorial.*

3. *MVC Pattern [Электронный ресурс]. - Режим доступа https://medium.com/@anshul.vyas380/mvc-pattern-3b5366e60ce4.*

4. What is SignalR? *[Электронный ресурс]. - Режим доступа https://www.syncfusion.com/succinctly-free-ebooks/real-time-asp-net-core-3-apps-with-signalr-succinctly/what-is-signalr.*

UDC 004.777+004.272

DEVELOPING A WEB APPLICATION FOR PUBLISHING WRITINGS USING MVC PATTERN

Bogomaz D.L., Yarmolik M.A.

Belarusian State University of Informatics and Radioelectronics, Minsk, Republic of Belarus (style T-institution)

Gorbach A.P. – PhD, associate professor

Annotation. Discovered the main positive aspects of using the MVC pattern on the example of ASP.NET Core application. Analyzed the use of SignalR framework with TypeScript on the basis of ASP.NET platform. Analyzed the difference between dynamic typing and static typing on the example of TypeScript and JavaScript programming languages.

Keywords. MVC, SignalR, design pattern, web application.