УДК 004.9 **Информационные технологии**

***Макаров Олег Сергеевич**, студент магистратуры,*
ФГБОУ ВО «Национальный исследовательский Мордовский государственный университет им. Н. П. Огарёва», Россия, г. Саранск
***Щенникова Елена Владимировна**, доцент, доктор физ.-мат. наук,*
профессор кафедры фундаментальной информатики
факультета математики и информационных технологий
ФГБОУ ВО «Национальный исследовательский Мордовский государственный университет им. Н. П. Огарёва», Россия, г. Саранск

СРАВНИТЕЛЬНЫЙ АНАЛИЗ ТЕХНОЛОГИИ ASP.NET CORE

Аннотация: В статье представлен сравнительный анализ новой технологии ASP.NET Core с фреймворком ASP.NET MVC, описаны основные различия, проанализированы преимущества и недостатки с точки зрения веб-разработчиков программного обеспечения.

Ключевые слова: ASP.NET Core, ASP.NET MVC, веб-приложение, кроссплатформенность, внедрение зависимостей, Blazor.

Abstract: The article presents a comparative analysis of the new ASP.NET Core technology with the ASP.NET MVC framework, describes the main differences, analyzes the advantages and disadvantages from the point of view of web software developers.

Key words: ASP.NET Core, ASP.NET MVC, web application, cross-platform, dependency injection, Blazor.

Введение

Платформа ASP.NET Core от корпорации Microsoft, которая появилась сравнительно недавно [3], продолжает набирать свою популярность, однако

программисты все еще с недоверием относятся к новой и непроверенной технологии, предпочитая предшествующий ей фреймворк ASP.NET MVC. Проанализируем преимущества и недостатки новой технологии в сравнении со старой.

Преимущество 1. Единый набор технологий для шаблонов MVC и API.

В ASP.NET MVC на начальном этапе предлагается выбрать шаблон проектирования, от которого будет сильно зависеть архитектура приложения, поскольку шаблоны требуют различных зависимостей. В ASP.NET Core же нет различий между построением проекта на фреймворках MVC или API, программист может использовать одни и те же компоненты в любых целях, что существенно ускоряет и упрощает работу над программным обеспечением.

Преимущество 2. Изменения структуры проекта (решения)

В ASP.NET MVC обязательно использование файлов Global.asax и web.config. Также есть множество других директорий, которые заимствованы еще от технологии WebForms. Структура проекта на ASP.NET Core более лаконична и не требует большинства файлов, которые программисту могут и не потребоваться. Приложение на ASP.NET Core аналогично обычному консольному приложению по своей структуре: имеется единая точка входа – файл Program со статическим методом Main, что позволяет отлаживать программу в едином стиле.

Также больше не требуется использовать файлы конфигурации, формат которых диктует ASP.NET MVC. В ASP.NET Core можно гибко работать с конфигурацией приложения в любом формате (JSON, XML, CSV) и даже определять свои форматы обработки данных.

Также добавилась директория wwwroot, которая представляет собой корень приложения при работе веб-сервера, структуру и название которой определяет сам программист. Больше не требуется писать специальные правила для игнорирования конфиденциальных файлов, которые требуются при разработке. Дополнительно облегчается работа с клиентским кодом:

значительно упрощаются задачи минимизации и объединения статических файлов, например, с помощью технологий gulp или webpack.

Преимущество 3. Кроссплатформенность

Приложения на ASP.NET MVC требуют для своего запуска среду исполнения .NET, что позволяет запускать такие приложения на Unix-подобных операционных системах только с использованием специальных сторонних средств, например, Mono. С технологией ASP.NET Core это не обязательно, потому что технология является полностью кроссплатформенной и не требует зависимостей от ядра систем на базе Windows [2, с. 133].

Преимущество 4. Встроенная поддержка внедрения зависимостей.

Внедрение зависимостей или Dependency Injection (DI) – это подход в программировании, позволяющий гибко управлять реализациями определенных интерфейсов во всем приложении [4]. Такой механизм позволяет писать слабо связанный код, очень удобный для модульного и интеграционного тестирования. Данный подход стал практически общепринятым при написании большинства программных приложений независимо от применяемых технологий. К сожалению, в ASP.NET MVC такой механизм доступен только с использованием сторонних технологий, таких как Unity, Windsor Castle, StructureMap и др. От проекта к проекту использование подобных пакетов отличается, что заставляет программистов изучать дополнительную литературу для сопровождения кода и исправления дефектов. В ASP.NET Core используется гибкая встроенная поддержка DI, не требующая подключения дополнительных программных библиотек. Однако по желанию программист может использовать привычные ему средства DI или даже реализовать свои.

Преимущество 5. Сквозной проброс зависимостей и единый пакет SDK

В процессе длительной работы над программным обеспечением, ошибки и уязвимости сторонних зависимостей устраняются, старые версии перестают поддерживаться, поэтому некоторые компоненты приложения следует постоянно обновлять для обеспечения безопасности [5].

Для приложений, написанных на ASP.NET MVC, общие зависимости необходимо подключать в каждый проект решения вследствие особенностей платформы. В будущем это может вызвать определенные проблемы. Например, если корневая сборка содержит зависимость, то все остальные проекты в данном решении должны подключить и скопировать эту зависимость локально. Во-первых, это линейно увеличивает место, занимаемое проектом на диске. Во-вторых, если иные проекты в решении содержат другую версию той же зависимости, могут возникнуть проблемы с совместимостью по всему приложению.

В ASP.NET Core зависимость, используемая в нижележащей сборке, будет пробрасываться до вышележащих сборок вместе со ссылкой на корневую сборку. Это позволяет использовать одну и ту же версию программной библиотеки во всех проектах приложения.

Проекты решения, как правило, содержат одни и те же корневые зависимости для разработки, поэтому программирование на платформе ASP.NET Core зависит от целого набора таких зависимостей, внутри которого поддерживается строгое соблюдение версионности всех пакетов. Это обеспечивает их совместимость, потому что невозможно изменить отдельную зависимость без обновления всего пакета. Дополнительно такие наборы пакетов позволяют сэкономить место на диске, поскольку они расположены в локальной папке пользователя компьютера и не копируются в папки проектов приложения, а представляются ссылкой.

Преимущество 6. Открытый исходный код

В отличие от ASP.NET MVC исходный код платформы ASP.NET Core полностью открыт для всех желающих и располагается по адресу: https://github.com/dotnet/aspnetcore. Проект все так же развивается под руководством Microsoft, однако теперь любой желающий может отслеживать изменения и даже завести задачу в баг-трекере. Сообщество ASP.NET Core постоянно расширяется, программисты всего мира участвуют в устранении дефектов и развитии технологии.

Преимущество 7. Blazor

Большинство веб-приложений создаются командами, численность которых редко превышает 10-15 человек. Как правило, программисты либо делят обязанности по написанию кода серверной и клиентской стороны, либо одновременно пишут на разных языках. С выпуском ASP.NET Core версии 3.0 появилась новая революционная технология Blazor, позволяющая писать код клиентской части на языке программирования C# [1]. Визуальный интерфейс создается все теми же средствами HTML и CSS, заменяется лишь логика пользовательского интерфейса. Таким образом, программисты могут использовать лишь один язык для всего веб-приложения, используя его сильные стороны. Это дополнительно позволяет избежать дублирования кода на серверной и клиентской частях, ведь зачастую они используют смежную логику.

Недостаток 1. Документация

Документация ASP.NET Core пока еще не такая обширная, как у ASP.NET MVC. В настоящий момент Microsoft активно работает над этим, увеличивая количество описаний и примеров использования и привлекая сторонних специалистов. Эта проблема отчасти нивелируется из-за открытого исходного кода, ведь каждый человек теперь может найти любой интересующий его компонент и посмотреть логику работы.

Недостаток 2. Небольшое количество инструментов

Выход новой технологии всегда сопровождается отсутствием инструментов, прикладных пакетов и программных библиотек, которые ее поддерживают. Корпорация Microsoft грамотно подошла к этому и с релизом ASP.NET Core обновила свой инструмент разработки Visual Studio. Поэтому данный недостаток относится только к отсутствию поддержки сторонних широко распространенных пакетов, написанных для .NET Framework. Данная проблема устранится с течением времени после завоевания популярности новой технологии.

Недостаток 3. Поддержка и сопровождение устаревшего кода (legacy source code).

Большинство крупных и средних приложений на ASP.NET MVC невозможно или слишком затратно переводить на новую технологию. Какое-то время пионерами ASP.NET Core будут только новые программные продукты, однако со временем ситуация выровняется.

Заключение

Несмотря на недостатки, ASP.NET Core имеет ряд крупных преимуществ. С уверенностью можно сказать, что ASP.NET Core в скором будущем не только заменит ASP.NET MVC, но и составит конкуренцию иным технологиям для создания веб-приложений.

Библиографический список:

1. Введение в ASP.NET Core Blazor [Электронный ресурс]. – [Б. м. : Б. и.], [20–]. – Режим доступа: https://docs.microsoft.com/ru-ru/aspnet/core/blazor. – Загл. с экрана (дата обращения: 12.07.2020).

2. Макаров О. С. Сравнение платформ .NET Core и .NET Framework / О. С. Макаров // Фундаментальная и прикладная наука: состояние и тенденции развития. – Петрозаводск, 2019. – С. 131–134.

3. ASP.NET Core – Краткое руководство [Электронный ресурс]. – [Б. м. : Б. и.], [20–]. – Режим доступа: https://coderlessons.com/tutorials/microsoft-technologies/izuchite-asp-net-core/asp-net-core-kratkoe-rukovodstvo. – Загл. с экрана (дата обращения: 24.06.2020).

4. Kalkan H. ASP.NET Core Dependency Injection Best Practices, Tips & Tricks [Электронный ресурс] / H. Kalkan. – [Б. м. : Б. и.], [20–]. – Режим доступа: https://medium.com/volosoft/asp-net-core-dependency-injection-best-practices-tips-tricks-c6e9c67f9d96. – Загл. с экрана (дата обращения: 01.07.2020).

5. What is Information Security? [Электронный ресурс]. – [Б. м. : Б. и.], [201–]. – Режим доступа: https://www.geeksforgeeks.org/what-is-information-security/. – Загл. с экрана (дата обращения: 19.06.2020).