Столыпинский
вестник

Научная статья
Original article
УДК 004

НЕСТАНДАРТНЫЕ АРХИТЕКТУРА В НАПИСАНИЕ ВЕБ ПРИЛОЖЕНИЙ

NON-STANDARD ARCHITECTURE IN WRITING WEB APPLICATIONS

Яровая Екатерина Владимировна, Магистранта Гродненского государственного университета им Я.Купалы, Беларусь, г.Гродно

Yaravaya Katsiaryna Vladimiravna, Master's student Yanka Kupala State University of Grodno, Belarus, Grodno

Аннотация

В данной статье будут рассмотрены подходы, которые значительно отличаются от наиболее популярных фреймворков или библиотек, используемых для написания современных приложений. Рассмотрим, как можно компилировать язык программирования JavaScript, какие в этом подходе есть плюсы и минусы. Так же затронем какие технические решения приводят к оптимизации кода или затрудняют разработку. Обсудим тенденцию направления развития фреймворков и их пользу для заказчиков и исполнителей окончательного продукта.

Annotation

This article will consider approaches that differ significantly from the most popular frameworks or libraries used to write modern applications. Consider how you can compile the programming language JavaScript, which in this approach has pros and cons. As will touch on what technical solution leads to code optimization or complicate development. We will discuss the trend in the direction of frameworks and their usefulness for customers and executors of the final product.

Ключевые слова: Svelte, Ember js, веб разработка, фреймворки, библиотеки, компилятор JavaScript

Keywords: Svelte, Ember js, web development, frameworks, libraries, JavaScript compiler

В мире разработки клиентских приложений есть технологии, которые прочно укоренились в умы, в портфолио, и в общем заняли свою нишу. Мы обсудим в этой статье совершенно новый подход, который использует не библиотеку и не фреймворк, а использует компилятор. И мы узнаем, как с помощью калькулятора можно писать клиентскую часть приложения. Ангулар себя позиционирует, как полноценный фреймворк с помощью которого можно написать любое веб приложение без подключения дополнительных библиотек. React – это библиотека, которая позволяет сделать наше приложение очень отзывчивым, но обработки запросов или хранение данных нужно подключать дополнительно, конечно можно использовать useContest для хранения состояния, но это не самая лучшая идея для больших интерпрайс проектов. Vue – позиционирует себя, как прогрессивный фрэймворк, на него можно переходить из “легаси” технологии, достаточно прост в понимании и удобен в написании. Svelte – о котором сегодня и пойдет речь позиционирует себя как компилятор.

Первая версия Svelte появилась в 29 ноября 2016 года, имя для технологии было выбрано Ричем Харрисом и его коллегами, вторая версия 19

апреля 2018, в ней исправлен ряд ошибок. 21 апреля 2019 года третья версия, где переосмыслили подход к реактивности за счет компилятора.

Какие же проблемы, которые Svelte берется решить – снижение размера бандлов, разработчики считают, что мы отправляем пользователю слишком много кода. Производительность, минимизация, абстракция и приближения настолько близко к нативному коду, насколько это возможно. Совместимость компонентов написана на одном фреймворке, не могут быть использованы на другом. Есть технологи, которые могут позволить решить проблемы, перечисленные выше – это code splitting and tree shaking, но это не самый удобный подход в современном мире. Многие фреймворки несут в себе функциональность, которую вы никогда не будет использовать, вы не можете взять только то, что нужно нам. Так вот основная идея в том, что svelte существует только тогда, когда мы пишем код, во время компиляции работает статический анализатор, и после написания компилирует в низкоуровневый высокоэффективный код.

Принципы работы Svelte заключаются в том, что первым этапом мы пишем высокоуровневый, декларативный код, как и на остальных популярных фреймворках, далее в действие вступает компилятор, который превращает его в банд с низкоуровневым и императивным кодом, с высокой производительность. Отсутствие виртуального дома, как не странно повышает производительность, так как изменение точечно, и мы точно знаем какой DOM узел изменился. В runtime остается только готовый самодостаточный код, по этой причине svelte называют исчезающим фреймворком, после сборки почти никаких следов не остается, что в свою очередь приводит к крайне небольшому конечному банду. К примеру, вендор старте svelte весить 3 килобайт, а у Preact это почти 9 килобайт значительная разница не так ли, а Preact, который является облегченной версией React. Это действительно новый подход к написанию веб приложений. Хранилище состояния в Svelte,

это всего лишь обсервебал (объекты), на которые подписывает приложение и обновляет данные при подписке и все это подключается на этапе компиляции.

Svelte как не хотелось бы думать или надеяться, не идеален и тоже имеет ряд проблем, которые перекладываются на хрупкие плечи команды разработки. Первое что хотелось бы упомянуть, это поддержка редакторами, VS code прекрасно работает с ним, а вот в Webstorm могут быть проблемы. Даже при установке плагинов иногда некоторые элементы подсвечиваются красным, хотя это вполне легальный и рабочий синтаксис в реалиях Svelte. Также если у вас есть небольшое домашний сайт, то возникнут проблемы с нахождение библиотек компонентов, но это весьма поверхностная проблема, в любом случае, если вы хотите чего-то очень креативного, все придется писать самостоятельно.

Есть еще один интересный и может быть, весьма перспективный инструмент. Ember.js – является второй SproutCore, который был переименован в декабре 2011. Из плюсов можно отметить, это фреймворк и у него экосистема достаточно широкая, чтобы вы могли написать полноценное приложение, не прибегая к другим библиотекам. Есть возможность взаимодействовать с операционной системой компьютера. Стабильность без застоя, это означает что процесс обновления Ember.js будет проходить максимально комфортно, и не нанесет вред приложениям на старых версиях, максимальная совместимость. Огромное сообщество, с помощью которого можно найти любое решение для проблемы. Частые обновления, команда Ember планирует выпускать новые мажорные версии каждый год.

Как и любое решение Ember имеет ряд минусов. Он достаточно сложный в освоении и новичку или молодому разработчику придется потратить значительное время на освоение этого фреймворка. Он не прощает ошибок, любое неправильное использование может привести к сбоям в работе. Не набирает популярность, хоть и есть достаточно большое комьюнити, кто

отдает свое предпочтение Ember, но у него застой в популярности, он не привлекает новых разработчиков.

Все фреймворки, компиляторы или библиотеки, которые мы затронули в этой небольшой статье могут прекрасно подойти для решения каких-либо задач на ваших проектах. Советы на тему, что нужно использовать в ваших проектах и стоит ли изучать ту или иную технологию, зависит только от вас и от ваших задач. Каждая из этих технологий может прекрасно подойти для отдельных задач, как говорится все зависит от того, что конкретно для вашего проекта будет хорошо.

Литература

1. Шкляр Л., Архитектура веб-приложений (2010)
2. Н. Мациевский, Реактивные веб-сайты (2020)
3. С. Пьюривал, Основы разработки веб-приложений (2015)
4. Modern Compiler Implementation in C: Basic Techniques (1997)
5. Альфред Ахо, Рави Сети, Джеффри Дэвид Ульман, Моника Лам, Компиляторы: принципы, технологии и инструменты(1986)

Literature

1. Shklar, L., Web Application Architecture (2010)
2. N. Maciejewski, Reactive Websites (2020)
3. S. Purewal, Fundamentals of Web Application Development (2015)
4. Modern Compiler Implementation in C: Basic Techniques (1997)
5. Alfred Aho, Ravi Sethi, Jeffrey David Ullman, Monica Lam, Compilers: Principles, Technology, and Tools(1986)

Для цитирования: Яровая Е.В. НЕСТАНДАРТНЫЕ АРХИТЕКТУРА В НАПИСАНИЕ ВЕБ ПРИЛОЖЕНИЙ// Научный сетевой журнал «Столыпинский вестник» №5/2022