REACTJS*, JAVASCRIPT*, ANGULAR*

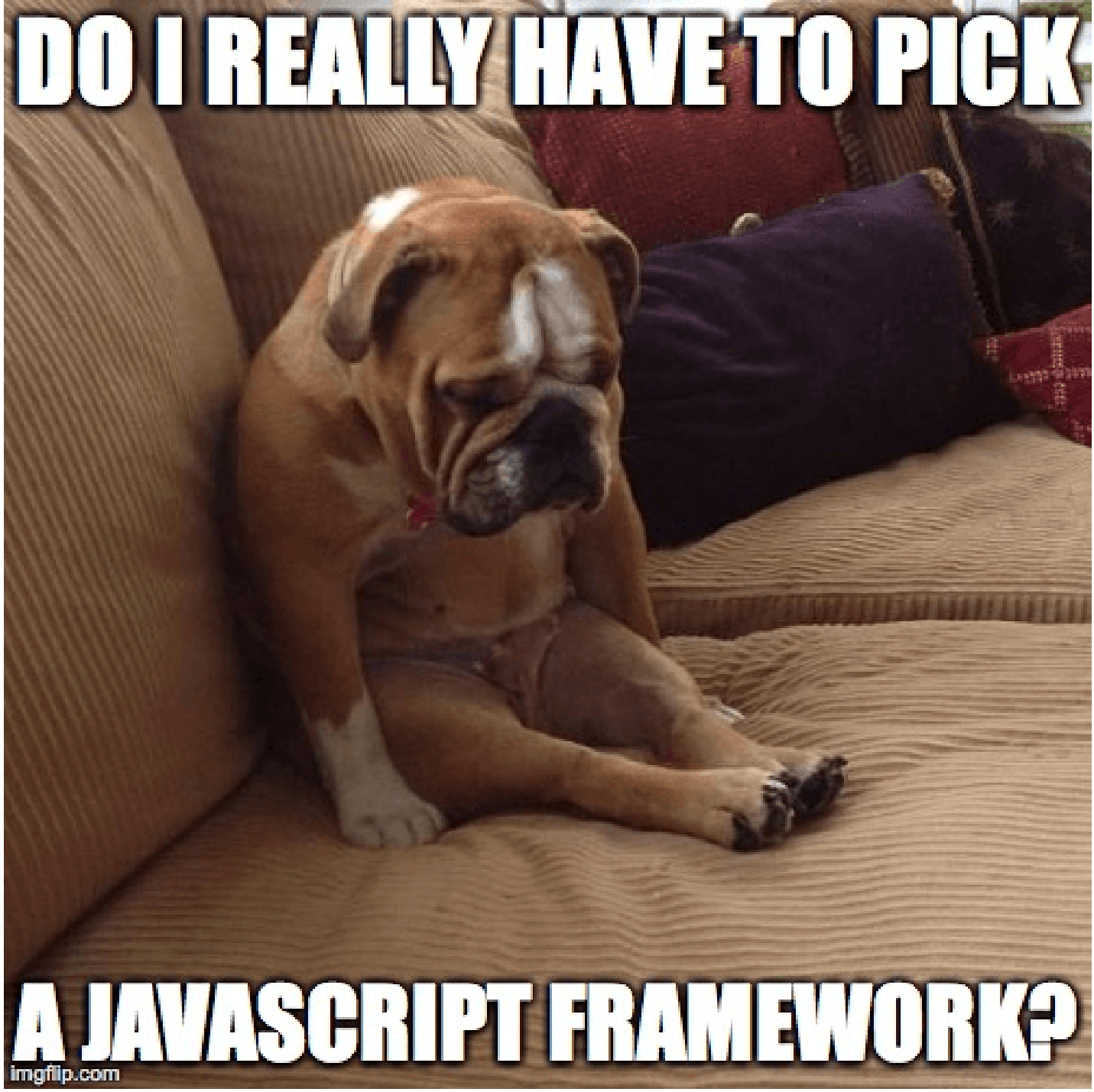
Angular vs. React vs. Vue: Сравнение 2017

ПЕРЕВОД

Synoptic 18 сентября 2017 в 22:25 82,4k

Оригинал: Jens Neuhaus

Выбор JavaScript-фреймворка для вашего веб-приложения может быть невыносим. В настоящее время очень популярны Angular и React, и есть также выскочка, получающий много внимания в последнее время: VueJS. Кроме них, лишь эти несколько новичков.



Итак, как же нам выбрать? Список плюсов и минусов никогда не повредит. Проделаем это в стиле моей предыдущей статьи, “9 шагов: выбор технологического стэка для вашего веб-приложения”.

Прежде чем начнем — SPA или нет?

Сперва вы должны принять четкое решение, нужно ли вам одностраничное веб-приложение(SPA), или вы предпочли бы

многостраничный подход. Больше на эту тему в моем посте “Одностраничные веб-приложения(SPA) против Многостраничных веб-приложений(MPA)”(скоро выйдет, следите за обновлениями в Twitter).

Участники сегодняшнего состязания: Angular, React and Vue

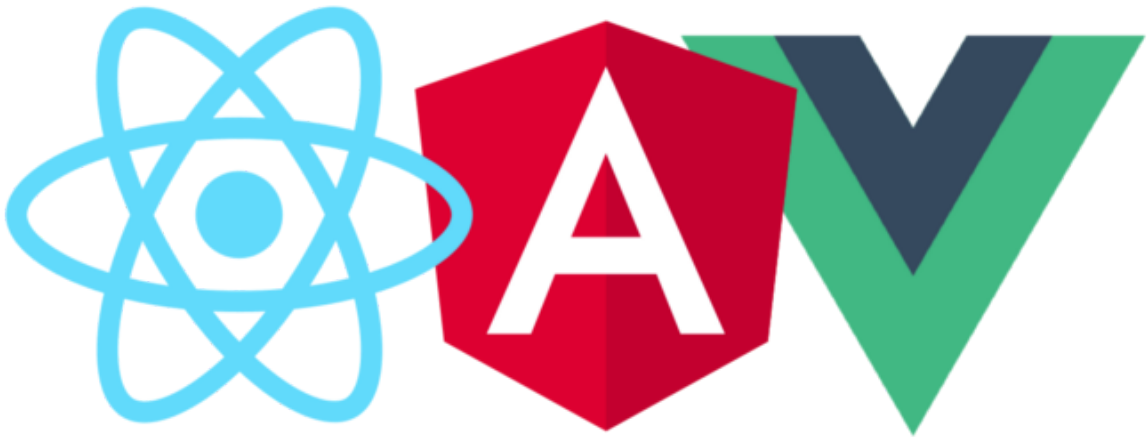
Сначала я хотел бы обсудить **жизненный цикл и стратегические соображения**. Затем мы перейдем к **возможностям и идеям** всех трех JavaScript-фреймворков. Наконец, мы придем к **выводам**.

Вопросы, которые мы сегодня рассмотрим:

- Насколько зрелые эти фреймворки/библиотеки?
- Будут ли эти фреймворки существовать еще какое-то время?
- Насколько велики и полезны сообщества вокруг них?
- Насколько легко найти разработчиков под каждый из этих фреймворков?
- Каковы основные идеи этих фреймворков?
- Насколько просто использовать эти фреймворки для небольших и крупных веб-приложений?
- Какова кривая обучения для каждого из фреймворков?
- Какую производительность вы ожидаете от этих фреймворков?
- Где вы можете подробнее узнать, что у них под капотом?
- Как начать разрабатывать с выбранным фреймворком?

На старт, внимание, марш!

Жизненный цикл и стратегические соображения



React vs. Angular vs. Vue

1.1 Немного истории

Angular — это JavaScript-фреймворк, основанный на TypeScript. Разработанный и поддерживаемый Google, он описывается как “*Супергеройский JavaScript MVW фреймворк*”. Angular(известный также как “Angular 2+”, “Angular 2” или “ng2”) — переписанный, по большей части несовместимый преемник AngularJS(известный как “Angular.js” или “AngularJS 1.x”). Хотя AngularJS(старый) был впервые выпущен в октябре 2010-го, он до сих пор получает багфиксы и т.д. Новый Angular(без JS) был представлен как версия №2 в сентябре 2016. Последний мажорный релиз — версия 4, так как версию 3 пропустили. Angular используется Google, Wix, weather.com, healthcare.gov и Forbes(в соответствии с madewithangular, stackshare и libscore.com).

React характеризуется как “*JavaScript-библиотека для создания пользовательских интерфейсов*”. Впервые выпущенный в марте 2013-го, React разработан и поддерживается Facebook-ом, который использует React-компоненты на нескольких страницах(однако, не являясь одностраничным веб-приложением). В соответствии с этой статьей Криса Кордла, React в Facebook применяется значительно шире, чем Angular в Google. React также используют в Airbnb, Uber, Netflix, Twitter, Pinterest, Reddit, Udemy, Wix, Paypal, Imgur, Feedly, Stripe, Tumblr, Walmart и других(в соотв. с Facebook, stackshare и libscore.com).

В данный момент Facebook работает над выпуском **React Fiber**. Это изменит React под капотом — в результате, рендеринг должен значительно ускориться — но обратная совместимость сохранится после изменений. В Facebook рассказывали об этих изменениях на их конференции для разработчиков в апреле 2017-го, также была опубликована неофициальная статья о новой архитектуре. Предположительно, React Fiber будет выпущен вместе с React 16.

Vue — один из самых быстроразвивающихся JS-фреймворков в 2016-м. Vue описывает себя как “*Интуитивный, Быстрый и Интегрируемый MVVM для создания интерактивных интерфейсов*”. Впервые он был выпущен в феврале 2014-го бывшим сотрудником Google Эваном Ю(кстати, Эван тогда написал интересный пост про маркетинговую деятельность и цифры в первую неделю после старта). Это был неплохой успех, особенно учитывая, что Vue привлекает столько внимания будучи театром одного актера, без поддержки крупной компании. На данный момент, у Эвана есть команда из дюжины основных

разработчиков. В 2016 была выпущена вторая версия. Vue используют Alibaba, Baidu, Expedia, Nintendo, GitLab. Список более мелких проектов можно найти на madewithvuejs.com.

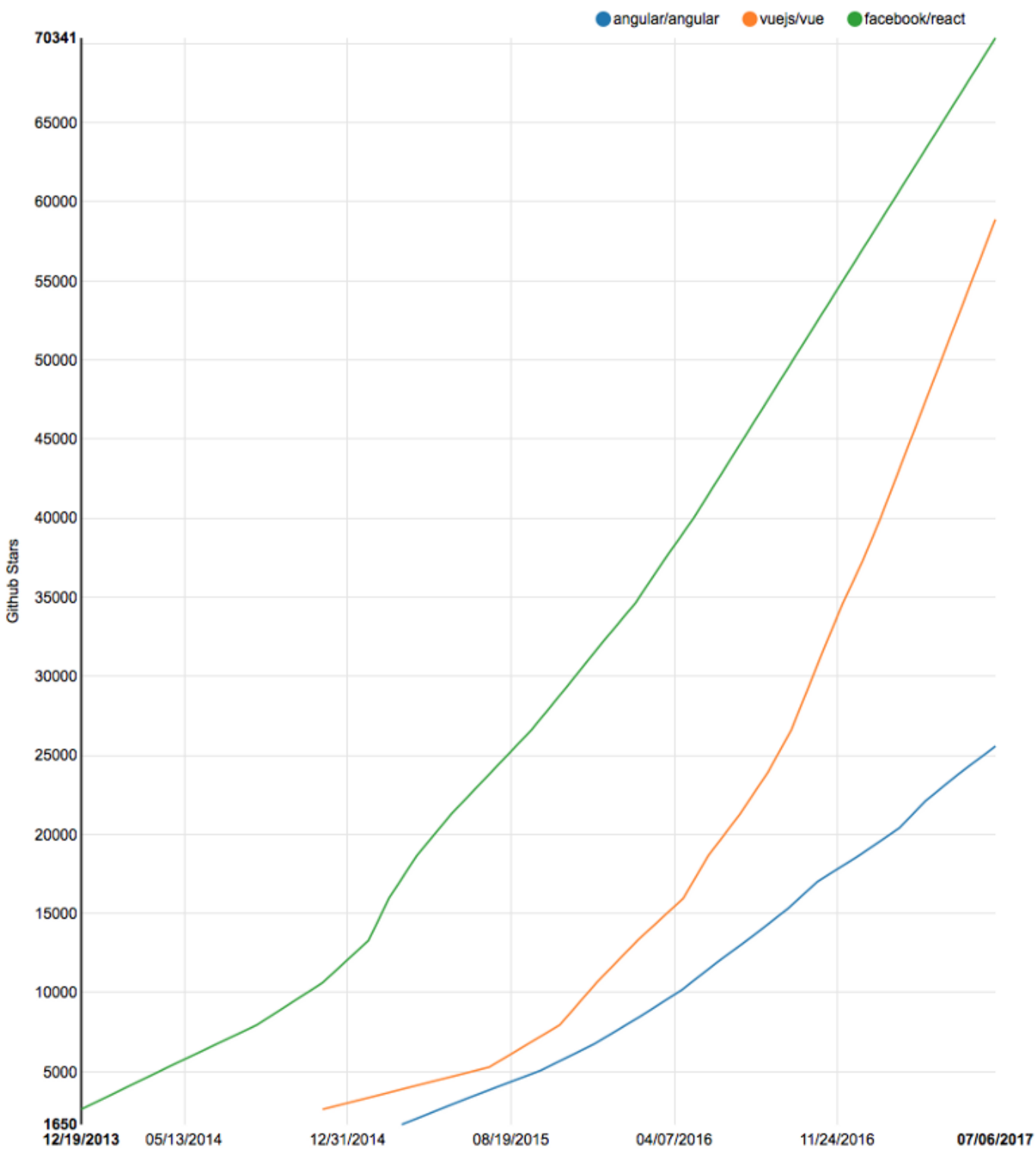
И Angular, и Vue доступны под лицензией **MIT**, в то время как React — под **BSD3-license**. Есть много обсуждений по поводу патентного файла. Джеймс Аид(бывший инженер Facebook) объясняет причины и историю, лежащую за этим файлом: *Патент Facebook касается распространения их кода при сохранении возможности защитить себя от патентных исков*. Файл патента обновлялся единожды и некоторые люди утверждают, что React можно использовать, если ваша компания не собирается подавать в суд на Facebook. Можете ознакомиться с обсуждением вокруг этого Github issue. Я не являюсь адвокатом, поэтому вы сами должны решить, создает ли лицензия React проблемы для вас или вашей компании. Есть еще много статей на эту тему: Дэннис Уолш пишет, почему вам не стоит бояться. Рауль Крипалани предостерегает от использования в стартапах, у него также есть обзор в формате “изложение мыслей”. Также существует недавнее оригинальное заявление от Facebook на эту тему: “Разъяснение лицензии React”.

1.2 Core development

Как было отмечено ранее, Angular и React поддерживаются и используются крупными компаниями. Facebook, Instagram и Whatsapp используют их на своих страницах. Google использует их во многих своих проектах: например, новый пользовательский интерфейс Adwords был реализован с помощью Angular и Dart. Опять же, Vue разрабатывается группой лиц, чья работа

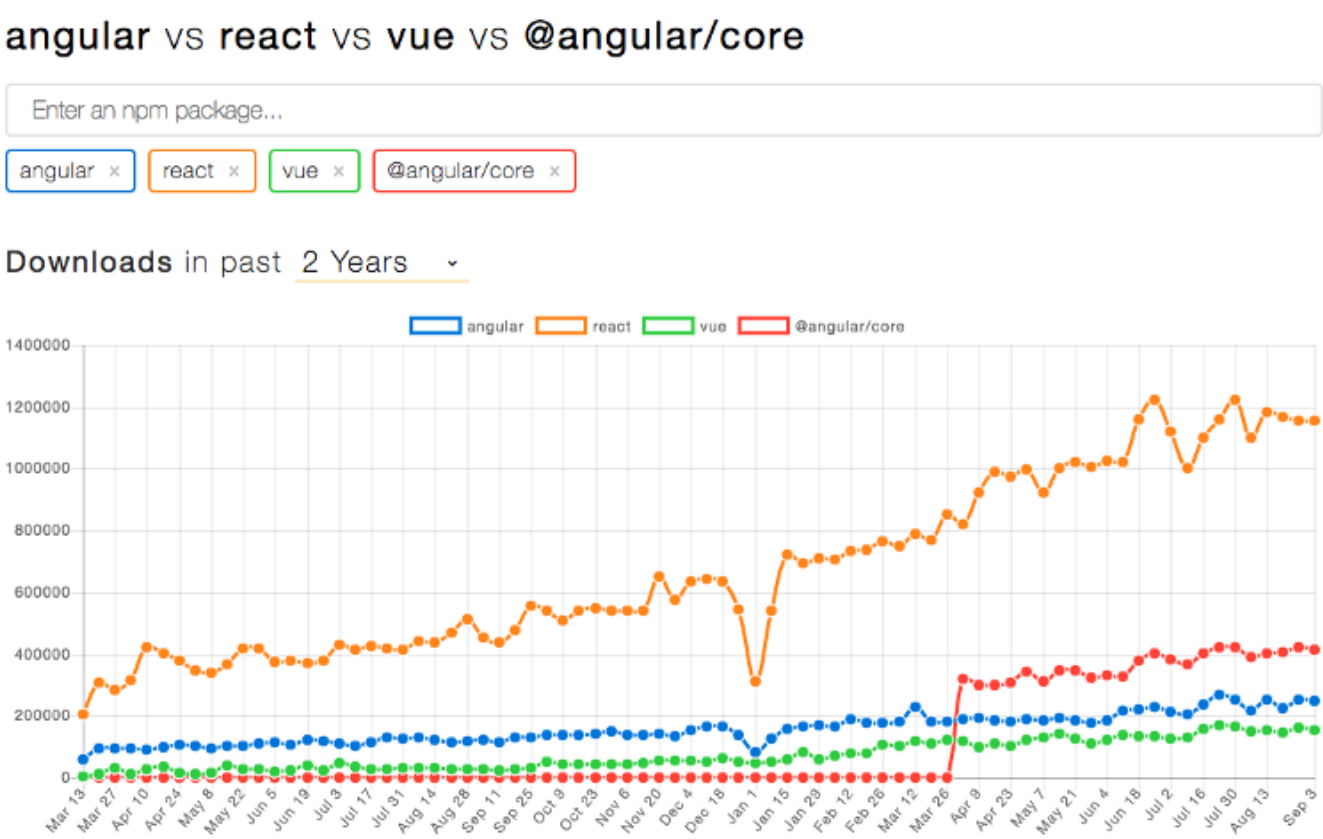
поддерживается через Patreon и другие средства спонсирования. Решайте сами, хорошо это или плохо. Маттиас Гёцке считает, что небольшая команда Vue — это плюс, потому что это ведет к более чистому коду / API, и меньшему оверинженерингу.

Посмотрим на статистику: на странице команды Angular перечислено 36 человек, у Vue — 16 человек, у React страницы команды нет. На GitHub-е у Angular больше 25 000 звезд и 463 контрибьютора, у React — больше 70 000 звезд и 1000 контрибьюторов, и у Vue почти 60 000 звезд и лишь 120 контрибьюторов. Можете также проверить страничку “Github Stars History for Angular, React and Vue”. Опять же, Vue, похоже, очень хорошо развивается. В соответствии с bestof.js, за последние три месяца Angular 2 получал в среднем 31 звезду в день, React — 74 звезды, Vue — 107 звезд.



A Github Stars History для Angular, React и Vue (Источник)

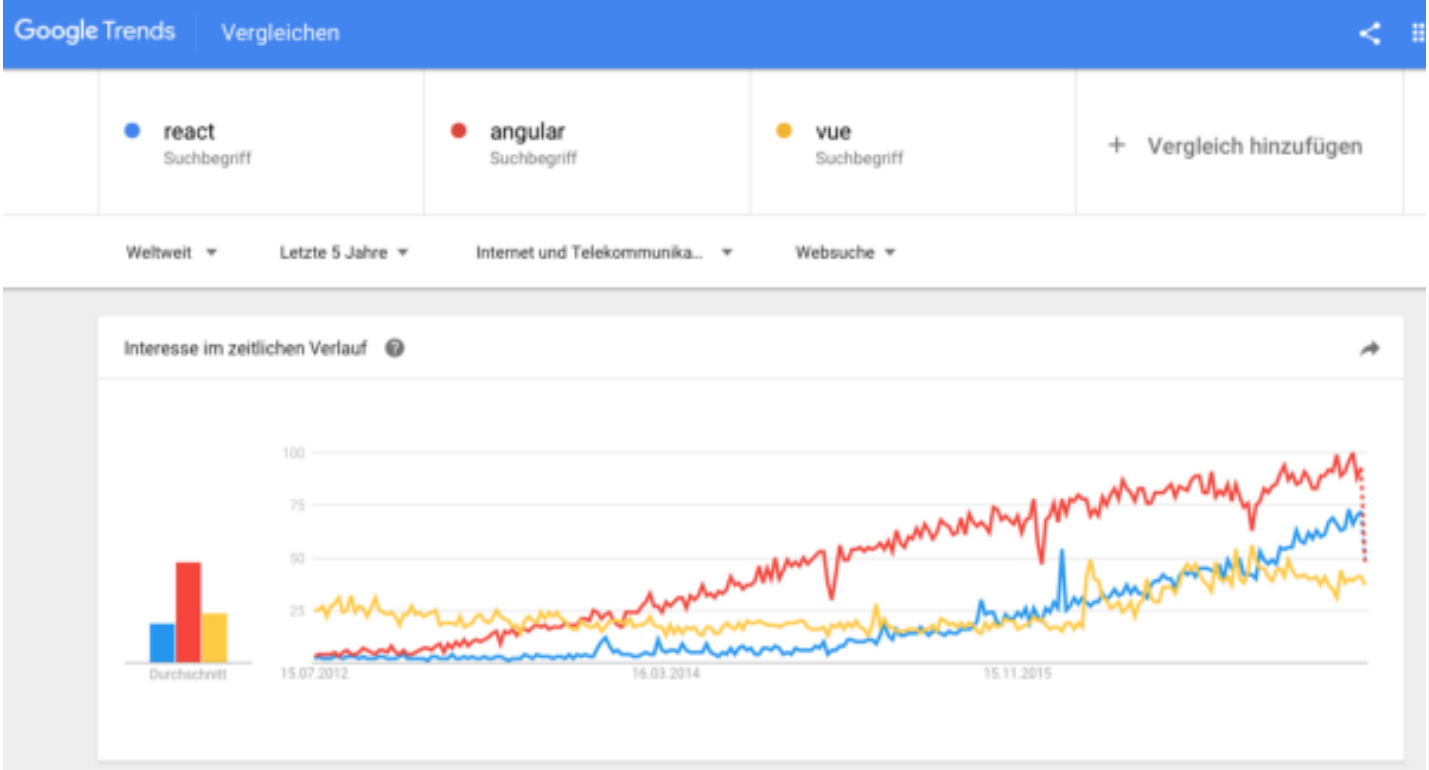
Апдейт: Спасибо Полу Хеншелю за то, что указал на npm trends. Они показывают количество скачиваний для данных npm-пакетов и даже полезнее, как чистый взгляд на звезды GitHub.



Количество скачиваний для заданных npm-пакетов в течение двух лет

1.3 Жизненный цикл на рынке

Angular, React и Vue сложно сравнить в Google Trends из-за их разнообразных имен и версий. Одним из способов получить приблизительные значения может быть поиск в категории “Internet & technologies”. Вот результат:



Ну, что ж. Vue до 2014 года не существовало — значит, тут что-то не так. La Vue по-французски — “вид”, “взгляд”, или “мнение”. Может быть, дело в этом. Сравнение “VueJS” с “Angular” или “React” несправедливо, так как у VueJS почти нет результатов, которые можно сравнить с остальными.

В таком случае, попробуем кое-что другое. Technology Radar от ThoughtWorks дает хорошее представление о том, как технологии эволюционируют в течение времени. Redux находится на стадии принятия(принятия в проектах!) и он был бесценен на ряде проектов ThoughtWorks. Vue.js на стадии испытаний(испытайте!). Его описывают, как легковесную и гибкую альтернативу Angular с более низкой кривой обучения. Angular 2 находится на стадии оценки — он успешно используется командами ThoughtWorks, но пока не имеет настоятельных рекомендаций.

В соответствии с последним опросом StackOverflow 2017, React любят 67% опрошенных разработчиков, а AngularJS — 52%. “Отсутствие интересу к продолжению разработки” регистрирует большие значения для AngularJS(48%) в сравнении с React(33%). Vue не входит в первую десятку ни в одном из случаев. Далее, есть опрос statejs.com, сравнивающий “front-end фреймворки”. Самые интересные факты: React и Angular обладают 100%-й известностью, Vue незнаком 23%-м опрошенных людей. Касательно “удовлетворенности”, React набрал 92% для варианта “использовал бы снова”, Vue — 89%, Angular 2 — только 65%.

Как насчет другого опроса об удовлетворенности пользователей? Эрик Элиотт запустил один в октябре 2016-го, чтобы оценить Angular 2 и React. Лишь 38% опрошенных людей использовали бы Angular 2 снова, в то время как 84% использовали бы снова React.

1.4 Долгосрочная поддержка и миграции

API React-а достаточно стабилен, как заявляет об этом Facebook в своих принципах проектирования. Существуют также скрипты, помогающие мигрировать с вашего текущего API на новый: попробуйте react-codemod. Миграции достаточно просты и здесь нет такой вещи(и необходимости в ней), как LTS версии. В этом посте на Reddit люди отмечают, что апгрейд на самом деле никогда не был проблемой. Команда React написала пост об их схеме версионирования. Когда они добавляют предупреждение об устаревании, они оставляют его для остальной части текущей релизной версии до того момента, пока поведение не будет изменено в следующей мажорной версии. Планов выпустить

новую мажорную версию нет — v14 выпущена в октябре 2015-го, v15 опубликована в апреле 2016-го, а у v16 пока нет даты релиза. Обновление не должно стать проблемой, как недавно заметил один из основных разработчиков React.

Что касается Angular, есть пост о версионировании и релизах Angular начиная с релиза v2. Будет одно мажорное обновление каждые шесть месяцев и период устаревания(deprecation period), как минимум шесть месяцев(два мажорных релиза). Существуют экспериментальные API, помеченные в документации более коротким периодом устаревания. Официального анонса нет, но в соответствии с этой статьей, команда Angular анонсировала **LTS версии начиная с Angular 4**. Они будут поддерживаться как минимум год после следующего мажорного релиза. Это значит, что Angular 4 будет поддерживаться багфиксами и важными патчами как минимум до **сентября 2018-го**. В большинстве случаев, обновление со второй до четвертой версии Angular также просто, как обновление зависимостей Angular. Angular также предоставляет руководство с информацией о том, какие понадобятся изменения в дальнейшем.

Процесс обновления с Vue 1.x до 2.0 должен быть простым для небольшого приложения — команда разработчиков утверждает, что 90% API осталось без изменений. Имеется приятный инструмент для диагностики обновления и помощи во время миграции, работающий из консоли. Один из разработчиков отметил, что обновление с v1 до v2 до сих пор не приносит особого удовольствия на больших приложениях. К сожалению,

roadmap-а для следующего мажорного релиза или информации о планах создания LTS версий нет.

Еще кое-что: Angular — это цельный фреймворк, предлагающий кучу вещей в комплекте. React гибче, чем Angular, и вы, вероятно, столкнетесь с использованием более независимых, неустаканенных, быстроразвивающихся библиотек — это означает, что вам придется самостоятельно заботиться о соответствующих обновлениях и миграциях. Это может нанести вред, если определенные пакеты больше не поддерживаются или если какой-то другой пакет в определенный момент становится стандартом де-факто.

1.5 Кадровые ресурсы и найм

Если у вас есть HTML-разработчики, которые не хотят углубляться в JavaScript, вам лучше выбрать Angular или Vue. React повлечет за собой увеличение количества JavaScript-а(позже мы поговорим об этом).

У вас есть дизайнеры, работающие в непосредственной близости с кодом? Пользователь “pier25” отмечает в своем Reddit-посте, что выбирать React имеет смысл, если вы работаете в Facebook, где каждый разработчик — супергерой. В реальном мире, вы не всегда найдете дизайнера, способного модифицировать JSX — по существу, работать с HTML-шаблонами будет намного проще.

Хорошая новость относительно Angular это то, что новый Angular 2-разработчик из другой компании сможет быстро ознакомиться со

всеми необходимыми соглашениями. Каждый проект на React отличается в плане архитектурных решений и разработчикам нужно будет знакомиться с настройками конкретного проекта.

Angular также хорош, если у вас есть разработчики с ООП-бэкграундом или те, кто не любят JavaScript. Чтобы заострить на этом внимание, цитата Манеша Чанда:

> “Я не JavaScript-девелопер. Мой бэкграунд — создание крупномасштабных корпоративных систем с использованием “настоящих” платформ для разработки ПО. Я начинал в 1997-м, разрабатывая приложения на C, C++, Pascal, Ada и Fortran. (...) Я могу точно сказать, что JavaScript для меня просто белиберда. Будучи MVP в Microsoft и экспертом, я хорошо понимаю TypeScript. Я также не рассматриваю Facebook, как компанию-разработчика ПО. Однако, Google и Microsoft уже крупнейшие инноваторы в этой сфере. Я чувствую себя более комфортно, работая с продуктом, у которого есть хорошая поддержка от Google или Microsoft. Также (...) с моим бэкграундом, я знаю, что у Microsoft даже большие планы для TypeScript”

Ну… Видимо, я должен упомянуть, что Манеш Чанд является региональным директором в Microsoft.

Сравнение React, Angular и Vue

2.1 Компоненты

Все обсуждаемые фреймворки основаны на компонентах. Компонент получает что-то на вход и после внутренних

вычислений возвращает отрендеренный шаблон UI(область входа / выхода с сайта или элемент списка to-do) на выходе. Определенные компоненты должно быть легко переиспользовать на веб-странице или в других компонентах. Например, у вас мог бы быть компонент сетки(состоящий из шапки и нескольких компонентов для рядов) с разнообразными свойствами(колонки, информация о шапке, data rows, и т.д.) и возможностью переиспользования этого компонента с разными данными на другой странице. Вот всеобъемлющая статья о компонентах на тот случай, если вам захочется изучить их получше.

И React, и Vue превосходно подходят для работы с “тупыми” компонентами: небольшие функции, не имеющие состояния, которые получают что-то на вход и возвращают элементы на выходе.

2.2 TypeScript vs ES6 vs. ES5

React фокусируется на использовании JavaScript ES6. Vue использует JavaScript ES5 либо ES6.

Angular зависит от **TypeScript**. Это дает бо́льшую консистентность в примерах и в опенсорсных проектах(примеры React можно найти в ES5 или в ES6). Это также вводит такие понятия, как декораторы или статическая типизация. Статическая типизация полезна для инструментов для анализа кода, типа автоматического рефакторинга, перехода к определению и т.д. — они должны уменьшить количество багов в приложении, хотя консенсус по поводу их использования, определенно, не достигнут. Эрик Элиот

не согласен с этим в своей статье “Шокирующий секрет статических типов”. Дэниэл С Вонг говорит, что использование статических типов не вредит и что хорошо иметь разработку через тестирование(TDD) и статическую типизацию *одновременно*.

Вам, вероятно, следует знать о том, что вы можете использовать Flow, чтобы включить проверку типов в React. Это инструмент для статической проверки типов, разработанный Facebook для JavaScript. Flow также можно интегрировать в VueJS.

Если вы пишете код на TypeScript, вы уже не пишете стандартный JavaScript. Несмотря на рост, у TypeScript до сих пор крошечное число пользователей, по сравнению со всем языком JavaScript. Один из рисков может заключаться в том, что вы будете двигаться в неверном направлении, поскольку TypeScript может — хотя и вряд ли — исчезнуть со временем. Помимо того, TypeScript создает приличный оверхед на проектах(на обучение) — можете больше почитать об этом в Сравнении Angular 2 и React от Эрика Элиотта.

Апдейт: Джеймс Рейвенскрофт написал к этой статье комментарий о том, что у TypeScript первоклассная поддержка JSX — компоненты можно легко проверить на соответствие типу. Так что, если вам нравится TypeScript и вы хотите использовать React, это не должно быть проблемой.

2.3 Шаблоны — JSX или HTML

React нарушает устоявшиеся best practices. Десятилетиями разработчики пытались разделить шаблоны и встроенную джаваскриптовую логику, но в JSX они опять перемешаны. Может быть, это звучит ужасно, но вам следует послушать речь Питера Ханта “React: Переосмысление best practices”(от октября 2013-го). Он указывает на то, что разделение шаблонов и логики — это просто разделение технологий, а не ответственности. Вы должны создавать компоненты вместо шаблонов. Компоненты переиспользуемы, интегрируемы и удобны для unit-тестирования.

JSX — это опциональный препроцессор с HTML-подобным синтаксисом, который затем компилируется в JavaScript. Отсюда некоторые странности — например, вам нужно использовать *className* вместо *class*, потому что последний является в JavaScript зарезервированным ключевым словом. JSX — большое преимущество для разработки, так как у вас все будет в одном и том же месте, а также быстрее будут работать автокомплит и проверки на стадии компиляции. Когда вы допускаете ошибку в JSX, React не компилирует код и выводит номер строки, в которой допущена ошибка. Angular 2 тихо падает в рантайме(возможно, этот аргумент некорректен, если вы пользуетесь AOT с Angular).

JSX подразумевает, что все в React = JavaScript — он используется и для JSX-шаблонов, и для логики. Кори Хаус указывает в своей статье от января 2016-го: “*Angular 2 продолжает внедрять ‘JS’ в HTML. React внедряет ‘HTML’ в JS*”. Это хорошо, потому что JavaScript мощнее, чем HTML.

Шаблоны в Angular представляют собой усовершенствованный HTML со специальным языком Angular(штуки, вроде *ngIf* или *ngFor*). В то время, как React требует знания JavaScript, Angular заставляет вас учить специфичный для Angular синтаксис.

Vue предлагает “однофайловые компоненты”. Это похоже на компромисс относительно разделения ответственности — шаблоны, скрипты и стили в одном файле, но в трех различных, упорядоченных секциях. Это значит, что вы получаете подсветку синтаксиса, поддержку CSS и возможность легко использовать препроцессоры типа Jade или SCSS. Я прочитал в других статьях, что JSX проще дебажить, потому что Vue не показывает синтаксические ошибки в HTML. Это не так, поскольку Vue конвертирует HTML в render-функции — поэтому ошибки показываются без проблем(Спасибо Виниciусу Рейзу за комментарий и поправки!).

Примечание: Если вам нравится задумка с JSX и вам хочется использовать его в Vue, вы можете использовать babel-plugin-transform-vue-jsx.

2.4 Фреймворки против библиотек

Angular — больше фреймворк, чем библиотека, так как предоставляет убедительные предложения о том, как ваше приложение должно быть структурировано, а также имеет больше функционала из коробки. Angular — это “цельное решение” — включены “батарейки” и вам предоставляется возможность удобного старта. Вам не нужно проводить анализ библиотек,

решать вопрос с роутингом и т.п. — вы можете просто начать работать.

React и Vue, с другой стороны, универсально гибки. Их библиотеки можно совмещать с любого рода пакетами(их достаточно много для React в npm, но у Vue пакетов меньше, так как он еще достаточно молод). Используя React, вы можете даже заменить саму библиотеку на ее API-совместимую альтернативу, такую как Inferno. С большей гибкостью, однако, появляется большая ответственность — для React не существует правил и и ограничительных рекомендаций. Каждый проект требует принятия решения относительно его архитектуры, и все может легко пойти не так, как планировалось.

С другой стороны, Angular поставляется с запутанным клубком систем для сборки, бойлерплейтов, линтеров и других пожирателей времени, с которыми придется иметь дело. Это верно и для React в случае использования стартер-китов или бойлерплейтов. Конечно, они очень полезны, React работает из коробки, и это, может быть для вас способ его изучить. Иногда разнообразие инструментов, необходимых для работы в JavaScript-окружении называют “усталостью от JavaScript”. Вот статья Эрика Клеммонса, который сказал следующее:

> По-прежнему существует множество инструментов, к которым вы не привыкли, в момент когда вы начинаете работать с фреймворком. Они генерируются, но возможно многие разработчики не понимают, что происходит у них под капотом, либо требуется много времени, чтобы это понять.

Vue кажется самым чистым и легким из трех фреймворков. У GitLab есть пост о принятии решения в пользу Vue.js(октябрь 2016-го):

> Vue.js прекрасно поставляется отлично сбалансированным в плане того, что он может сделать для вас и того, что вам нужно делать самостоятельно. (...) Vue.js всегда находится в пределах досягаемости; прочная, но гибкая сетка готова помочь сохранить эффективность вашего программирования и свести связанные с DOM страдания к минимуму.

Им нравится простота и легкость использования — исходный код очень читабелен и документация или внешние библиотеки не нужны. Все достаточно просто. Vue.js “не делает далекоидущих предположений о большей части чего-либо”. Имеется также подкаст о решении GitLab.

Другой пост о переходе на Vue от Pixeljets. React “*был большим шагом вперед для мира JS в плане state-awareness, и он показал множеству людей реальное функциональное программирование хорошим, практичным способом*”. Одним из серьезных минусов React по сравнению с Vue является проблема разбиения компонентов на более мелкие компоненты из-за ограничений JSX. Вот цитата из статьи:

> Для меня и моей команды важна читабельность кода, но также крайне важно то, чтобы написание кода приносило удовольствие. Нет ничего приятного в том, чтобы создать 6 компонентов, если вы разрабатываете простенький виджет-калькулятор. Во многих ситуациях это также плохо в плане

> поддержки, модификаций или визуальной переработки какого-то виджета, так как вам надо перемещаться по множеству файлов/функций и по-отдельности проверять каждый маленький кусочек HTML. Опять же, я не предлагаю писать монолиты — я предлагаю использовать в повседневной разработке компоненты вместо микрокомпонентов.

На Hacker news и Reddit есть интересные дискуссии об этом посте — в наличии аргументы от недовольных и дальнейших сторонников Vue.

2.5 Управление состоянием и связывание данных

Разрабатывать UI сложно, так как состояния присутствуют везде — данные меняются с течением времени, что влечет за собой увеличение сложности. Определенные способы работы с состоянием оказывают большую помощь, когда приложение разрастается и становится более сложным. Для небольших приложений это может быть перебор, и что-то типа Vanilla JS было бы достаточно.

Как это работает? Компоненты описывают UI в определенный момент времени. Когда данные изменяются, фреймворк перерисовывает UI-компонент целиком — отображаемые данные всегда актуальны. Мы может назвать эту идею “UI как функция”.

React часто работает в паре с Redux. **Redux** описывает себя в трех фундаментальных принципах:

- Единственный источник правды

- Состояние доступно только для чтения
- Изменения делаются с помощью чистых функций

Другими словами: состояние приложения целиком находится в дереве объектов внутри единого хранилища. Это помогает отлаживать приложение и кое-какая функциональность становится проще для реализации. Состояние в read-only режиме и может быть изменено только через “экшены”, чтобы избежать состояния гонки(также полезно для отладки). “Редьюсеры” пишутся, чтобы указать, как “экшены” могут трансформировать состояние.

Большая часть туториалов и бойлерплейтов включают в себя Redux, но вы можете использовать React без него(или **вам может быть вообще не нужен Redux** в вашем проекте). Redux добавляет сложность и достаточно серьезные ограничения для вашего кода. Если вы изучаете React, вам стоит подумать об изучении чистого React перед тем, как вы перейдете к Redux. Вам определенно стоит прочесть “Вам может быть не нужен Redux” Дэна Абрамова.

Некоторые разработчики предлагают использовать **Mobx вместо Redux**. Можете думать о нем, как о “автоматическом Redux”, который упрощает использование и понимание с самого начала. Если хотите посмотреть, вам стоит начать с введения. Вы можете также почитать это полезное сравнение Redux и MobX от Робина. Тот же автор также предлагает информацию о переходе с Redux на MobX. Этот список полезен, если вы хотите проверить другие Flux-библиотеки. И, если вы пришли из мира MVC, вы захотите

прочесть статью “Философия Redux (если вы уже знаете MVC)” Михаила Левковского.

Vue можно использовать с Redux, но он предлагает Vuex как свое собственное решение.

Большое отличие между React и Angular — **одно- или двустороннее связывание**. Двустороннее связывание в Angular меняет модель состояния, когда элемент пользовательского интерфейса(к примеру, поле ввода) обновляется. React идет только одним путем: сначала обновляет модель и затем отрисовывает элемент. Подход Angular чище в коде и проще для реализации разработчиком. Подход React позволяет получить лучшее понимание о том, что происходит с данными, потому что поток данных течет лишь в одном направлении(это делает отладку проще).

Оба подхода имеют плюсы и минусы. Вам нужно понять эти идеи и определить, влияют ли они на ваше решение о выборе фреймворка. И статья “Двустороннее связывание: Angular 2 и React”, и этот вопрос на StackOverflow предоставляют хорошие объяснения. Здесь вы можете найти интерактивные примеры кода(возрастом в 3 года, только для Angular 1 и React). Последнее, но тем не менее важное: Vue поддерживает и одностороннее, и двустороннее связывание(одностороннее по-умолчанию).

Если хотите почитать больше, имеется длинная статья о различных типах состояния и управлении состоянием в Angular-приложениях(от Виктора Савкина).

2.6 Другие концепции программирования

Angular включает в себя dependency injection, паттерн, в котором один объект(сервис) предоставляет зависимости другому объекту(клиенту). Это дает большую гибкость и более чистый код. Статья “Понять dependency injection” в деталях объясняет эту идею.

Паттерн Модель-Вид-Контроллер(MVC) делит проект на три части — модель, вид и контроллер. У Angular, как у MVC-фреймворка, имеется поддержка MVC из коробки. У React есть лишь V — что будет M и C нужно решить вам самим.

2.7 Гибкость и переход к микросервисам

Вы можете работать с React или Vue просто добавляя JavaScript-библиотеку к исходникам. Это невозможно с Angular, потому что он использует TypeScript.

Мы все больше движемся в сторону микросервисов и микроприложений. React и Vue предоставляют вам полный контроль над размером приложения, позволяя включить только те вещи, которые действительно нужны. Они дают больше гибкости при переходе от SPA к микросервисам, используя части бывшего приложения. Angular лучше подходит для SPA, так как для использования в микросервисах он, вероятно, слишком раздут.

Как отмечает Кори Хаус:

> JavaScript быстро развивается, и React позволяет вам заменять небольшие фрагменты вашего приложения более подходящими библиотеками вместо того, чтобы ждать и надеяться на то, что ваш фреймворк обновят. **Философия маленьких, интегрируемых, узкоспециализированных инструментов никогда не выйдет из моды**.

Некоторые люди также используют React для не-SPA вебсайтов(например, для сложных форм или мастеров). Даже Facebook использует React не для главной страницы, а, скорее, для определенных страниц и возможностей.

2.8 Размер и производительность

Обратная сторона функциональности: Angular довольно-таки раздут. Размер сжатого gzip-ом файла — 143кб, по сравнению с 23кб Vue и 43кб React.

И у React, и у Vue есть Virtual DOM, который должен увеличивать производительность. Если вам интересно, можете почитать об отличиях между Virtual DOM и DOM, а также о реальных преимуществах Virtual DOM в React. Один из авторов Virtual DOM также отвечает на вопросы, связанные с производительностью на StackOverflow.

Чтобы проверить производительность, я посмотрел великолепный js-framework-benchmark. Вы можете сами скачать и запустить его или посмотреть на интерактивную таблицу с результатами. Перед тем, как проверить результаты, вам нужно знать, что фреймворки

обманывают бенчмарки — такие замеры производительности не должны быть основой принятия решений.

Name	angular-v4.1.2-keyed	react-v15.5.4-redux-v3.6.0	vue-v2.3.3-keyed
create rows Duration for creating 1000 rows after the page loaded.	193.1 ± 7.9 (1.2)	212.2 ± 14.2 (1.3)	166.7 ± 8.6 (1.0)
replace all rows Duration for updating all 1000 rows of the table (with 5 warmup iterations).	197.4 ± 5.3 (1.2)	206.7 ± 7.3 (1.2)	168.5 ± 5.0 (1.0)
partial update Time to update the text of every 10th row (with 5 warmup iterations).	13.0 ± 4.5 (1.0)	18.0 ± 1.6 (1.1)	17.3 ± 2.9 (1.1)
select row Duration to highlight a row in response to a click on the row. (with 5 warmup iterations).	3.4 ± 2.3 (1.0)	8.7 ± 2.9 (1.0)	9.3 ± 1.7 (1.0)
swap rows Time to swap 2 rows on a 1K table. (with 5 warmup iterations).	13.4 ± 1.0 (1.0)	17.1 ± 1.3 (1.1)	18.3 ± 1.5 (1.1)
remove row Duration to remove a row. (with 5 warmup iterations).	46.1 ± 3.2 (1.0)	52.4 ± 1.7 (1.1)	52.6 ± 2.7 (1.1)
create many rows Duration to create 10,000 rows	1946.0 ± 41.8 (1.2)	1931.7 ± 35.6 (1.2)	1587.5 ± 33.9 (1.0)
append rows to large table Duration for adding 1000 rows on a table of 10,000 rows.	324.6 ± 10.1 (1.0)	366.4 ± 10.9 (1.1)	399.5 ± 11.0 (1.2)
clear rows Duration to clear the table filled with 10.000 rows.	379.9 ± 11.3 (1.5)	410.9 ± 9.9 (1.6)	254.5 ± 5.0 (1.0)
startup time Time for loading, parsing and starting up	84.3 ± 2.6 (1.5)	93.8 ± 6.9 (1.7)	56.6 ± 2.5 (1.0)
slowdown geometric mean	**1.14**	**1.23**	**1.06**

Производительность Angular, React и Vue (Источник)

Name	angular-v4.1.2-keyed	react-v15.5.4-redux-v3.6.0	vue-v2.3.3-keyed
ready memory Memory usage after page load.	4.8 ± 0.0 (1.3)	4.9 ± 0.1 (1.3)	3.8 ± 0.0 (1.0)
run memory Memory usage after adding 1000 rows.	10.9 ± 0.1 (1.4)	10.8 ± 0.1 (1.4)	7.5 ± 0.1 (1.0)

Выделение памяти в Мб (Источник)

Чтобы подытожить: у Vue прекрасная производительность и лучшее распределение памяти, но все фреймворки действительно довольно рядом друг с другом, по сравнению с особенно медленными или особенно быстрыми фреймворками(типа Inferno). Еще раз: тесты производительности следует рассматривать как побочные данные, не для вынесения вердикта.

2.9 Тестирование

Facebook использует Jest, чтобы тестировать свой код на React. Здесь есть сравнение Jest и Mocha, и есть статья о том, как использовать Enzyme с Mocha. Enzyme — библиотека для тестирования, написанная на JavaScript, используемая Airbnb(в сочетании с Jest, Karma и другими тест-раннерами). Если вы хотите почитать побольше, имеются старые статьи про тестирование в React(здесь и здесь).

Далее, существует **Jasmine**, как тестовый фреймворк для Angular 2. В статье Эрика Элиотта говорится о том, что Jasmine “*приводит к миллионам способов написания тестов и утверждений, нуждающихся в тщательном прочтении каждого из них, чтобы понять, что он делает*”. Вывод также раздут и труден для чтения. Есть несколько информативных статей про интеграцию Angular 2 с Karma и Mocha. Также есть старое видео(от 2015-го) о тестовых стратегиях в Angular 2.

У Vue есть недостатки в тестировании, но Эван написал в своем превью от 2017-го, что команда планирует поработать над этим. Они рекомендуют использовать Karma. Vue работает с Jest, также есть тестовая утилита avoriaz.

2.10 Универсальные и нативные приложения

Универсальные приложения внедряются в веб, на десктопы, а также в мир нативных приложений.

И React, и Angular поддерживают нативную разработку. У Angular есть NativeScript(при поддержке Telerik) для нативных приложений и Ionic Framework для гибридных приложений. С React вы можете попробовать react-native-renderer, чтобы разрабатывать кроссплатформенные приложения для iOS или Android, или react-native для нативных приложений. Большое число приложений(включая Facebook; за подробностями на Showcase) сделаны с помощью react-native.

JavaScript-фреймворки рендерят страницы на клиенте. Это плохо для воспринимаемой производительности, пользовательского опыта в целом и SEO. Серверный рендеринг — это плюс. У всех трет фреймворков имеются библиотеки, чтобы помочь с этим. Для React это next.js, у Vue есть nuxt.js, и у Angular есть… Angular Universal.

2.11 Кривая обучения

У Angular, определенно, крутая кривая обучения. Он обладает всеобъемлющей документацией, но иногда это может вас расстраивать, потому что ряд вещей сложнее, чем кажется. Даже если вы глубоко понимаете JavaScript, вам нужно изучить, что происходит под капотом фреймворка. Вначале установка магическая, предлагает большое число встроенных пакетов и кода. Это можно рассматривать, как минус из-за большой, сразу существующей экосистемы, которую вам нужно со временем изучить. С другой стороны, это может и хорошо в определенных ситуациях, поскольку многие решения уже приняты. С React вам, скорее всего, придется принять множество внушительных решений относительно использования third-party библиотек. Существует 16 различных flux-пакетов для управления состоянием, которые можно выбрать для React.

Vue довольно прост для изучения. Компании переходят на Vue из-за того, что он кажется значительно более простым для начинающих разработчиков. Тут вы сможете почитать, как некто описывает переезд своей команды с Angular на Vue. По информации от другого пользователя, приложение на React в его

компании было таким сложным, что новые разработчики не поспевали за сложностью кода. С Vue разрыв между младшим и старшим разработчиком сокращается, и они могут взаимодействовать проще и с меньшим количеством багов, проблем и времени на разработку.

Некоторые люди утверждают, что то, что они сделали на React было бы написано лучше с использованием Vue. Если вы — неопытный JavaScript разработчик — или в последние 10 лет работали в основном с jQuery — вам стоит подумать об использовании Vue. Сдвиг парадигмы более выражен при переходе на React. Vue выглядит больше как чистый JavaScript в тоже время привнося некоторые новые идеи: компоненты, событийная модель и однонаправленный data-flow. Также у него небольшой размер.

Тем временем, у Angular и React свой собственный подход к вещам. Они могут мешать вам, потому что вам *нужно приспосабливаться под них, чтобы заставить их работать*. Это может быть вредно из-за меньшей гибкости и крутой кривой обучения. Это также может быть и полезно, поскольку вас заставляют изучать правильные подходы в процессе изучения технологии. С Vue вы можете работать в стиле старомодного JavaScript. Это может быть проще вначале, но может стать проблемой в долгосрочной перспективе, если все сделать не верно.

Когда речь заходит об отладке, плюс React и Vue в меньшем количестве магии. Охота на баги проще, потому что есть лишь

несколько мест, куда нужно смотреть, и у стектрейсов лучше видны отличия между собственным кодом и исходниками библиотеки. Работающие с React сообщают, что им никогда не требовалось читать исходный код библиотеки. Однако, отлаживая ваш код вашего приложения на Angular, вам часто нужно отлаживать внутренности Angular, чтобы понять лежащую в его основе модель. С другой стороны, сообщения об ошибках должны стать более чистыми и информативными начиная с Angular 4.

2.12 Под капотом Angular, React и Vue

Хотите проверить исходники самостоятельно? Хотите увидеть, как все складывается?

Возможно, вам стоит сначала проверить вот эти GitHub-репозитории: React(github.com/facebook/react), Angular(github.com/angular/angular) и Vue(github.com/vuejs/vue).

Как выглядит синтаксис? ValueCoders сравнивают синтаксис Angular, React и Vue.

Хорошо бы также увидеть все на продакшене — вместе с исходным кодом, на котором все основано. На TodoMVC есть список из десятков вариантов одного и того же Todo-приложения, написанного на различных JavaScript-фреймворках — можете сравнить решения на Angular, React и Vue. RealWorld разрабатывает реальное приложение(клон Medium) и у них есть готовые решения для Angular(4+) и React(с Redux). Для Vue работа в процессе.

Также есть некоторые реальные приложения, на которые вы можете посмотреть. Решения на базе React:

- Do (приятное приложение для управления заметками, построенное с помощью React и Redux)
- sound-redux (клиент для Soundcloud на базе React и Redux)
- Brainfock (решение для управления проектами и командами на базе React)
- react-hn и react-news (клоны Hacker news)
- react-native-whatsapp-ui + туториал (клон Whatsapp на React Native)
- phoenix-trello (клон Trello)
- slack-clone + другой туториал (клоны Slack)

Приложения на Angular:

- angular2-hn & hn-ng2 (клоны Hacker News + неплохой туториал о создании еще одного от Эшвина Сурешкумара)
- Redux-and-angular-2 (клон Twitter)

А также решения на Vue:

- vue-hackernews-2.0 и Loopa news (клоны Hacker News)
- vue-soundcloud (демо от Soundcloud)

Заключение

Выберите фреймворк сейчас

И React, и Angular, и Vue достаточно круты, и никого из них нельзя поставить сильно выше остальных. Доверьтесь своему нутру. Эта последний кусочек развлекательного цинизма может помочь вам принять решение:

> Грязный маленький секрет — “современная JavaScript-разработка” не имеет ничего общего с созданием веб-сайтов — это создание пакетов, которые могут использоваться людьми, создающими библиотеки, которые могут использоваться людьми, создающими фреймворки, которым люди, пишущие туториалы и ведущие курсы могут обучать. Не уверен, что кто-то действительно что-то создает для настоящих пользователей.

Это преувеличение, конечно, но в нем возможно есть доля правды. Да, в экосистеме JavaScript *есть* много лишнего шума. Может быть, вы найдете множество более привлекательных альтернатив во время ваших поисков — постарайтесь не быть ослепленным новейшим блестящим фреймворком.

Что я должен выбрать?

- Если вы работаете в Google: **Angular**
- Если вы любите TypeScript: **Angular(или React)**
- Если вам нужно руководство, структура и рука помощи: **Angular**
- Если вы работаете в Facebook: **React**
- Если вам нравится гибкость: **React**
- Если вы любите большие экосистемы: **React**
- Если вам нравится выбирать из десятков пакетов: **React**
- Если вы любиле JS и подход “все-есть-JavaScript”: **React**

- Если вам нравится действительно чистый код: **Vue**
- Если вы хотите простейшая кривую обучения: **Vue**
- Если вы хотите самый легковесный фреймворк: **Vue**
- Если вам хочется разделения ответственности в пределах одного файла: **Vue**
- Если вы работаете один или в небольшой команде: **Vue(или React)**
- Если ваше приложение имеет тенденцию разрастаться: **Angular(или React)**
- Если вы хотите иметь большой пул девелоперов: **Angular или React**
- Если вы работаете с дизайнерами и вам нужны чистые HTML-файлы: **Angular или Vue**
- Если вам нравится Vue, но пугает ограниченная экосистема: **React**
- Если вы не можете решить, изучите сначала **React**, затем **Vue**, затем **Angular**

Итак, вы приняли решение?

Дааа, вы его приняли!

Отлично! Читайте о том, как начать разрабатывать с Angular, React или Vue (скоро, подписывайтесь на меня в Twitter для обновлений).

Дополнительные ресурсы

React JS, Angular & Vue JS—Quickstart & Comparison (восьмичасовое введение и сравнение трех фреймворков)
Angular vs. React (vs. Vue)—the DEAL breaker (короткое, но превосходное сравнение от Доминика Т)
Angular 2 vs. React—the ultimate dance off (неплохое сравнение от Эрика Элиотта)
React vs. Angular vs. Ember vs. Vue.js (сравнение трех фреймворков в форме заметок от Гекана Сари)
React vs. Angular (понятное сравнение двух фреймворков)
Can Vue fight for the Throne with React? (приятное сравнение с большим количеством примеров кода)
10 reasons, why I moved from Angular to React (еще одно неплохое сравнение от Робина Вируча)
All JavaScript frameworks are terrible (большая заметка обо всех основных фреймворках от Мэтта Берджесса)

Похожие публикации

React.js State of the art (интервью с Max Stoiber)

25

m1skam • 21 ноября 2016 в 17:15

React.js: собираем с нуля изоморфное / универсальное приложение. Часть 1: собираем стек

76

ИЗ ПЕСОЧНИЦЫ

yury-dymov • 14 сентября 2016 в 10:45

Какой JavaScript Framework используете вы? Опрос среди JS-разработчиков

21

STEVER • 13 мая 2015 в 12:15

Популярное за сутки

Яндекс открывает Алису для всех разработчиков. Платформа Яндекс.Диалоги (бета)

69

BarakAdama • вчера в 10:52

Почему следует игнорировать истории основателей успешных стартапов

20

ПЕРЕВОД

m1rko • вчера в 10:44

Как получить телефон (почти) любой красотки в Москве, или интересная особенность MT_FREE

24

ИЗ ПЕСОЧНИЦЫ

cab404 • вчера в 20:27

Java и Project Reactor 10

zealot_and_frenzy • вчера в 10:56

Пользовательские агрегатные и оконные функции в PostgreSQL и Oracle 6

erogov • вчера в 12:46

Лучшее на Geektimes

Как фермеры Дикого Запада организовали телефонную сеть на колючей проволоке 31

NAGru • вчера в 10:10

Энтузиаст сделал новую материнскую плату для ThinkPad X200s 49

alizar • вчера в 15:32

Кто-то посылает секс-игрушки с Amazon незнакомцам. Amazon не знает, как их остановить 85

Pochtoycom • вчера в 13:06

Илон Маск продолжает убеждать в необходимости создания колонии людей на Марсе 140

marks • вчера в 14:19

Дела шпионские (часть 1)

TashaFridrih • вчера в 13:16

16