**Домашнее задание**по теме: «*JS framework.   
Сравнение Vue.js, Angular.js и React.js   
на примере приложения ToDo-list*»

Подготовили:  
Дмитриева Ирина  
Журба Алина  
АС-17-04

Оглавление

[JavaScript 1](#_Toc43129198)

[JS FrameWorks 1](#_Toc43129199)

[Преимущества построения приложения на JS-фреймворке: 3](#_Toc43129200)

[Framework Vue.js 3](#_Toc43129201)

[Концепции Vue.js 3](#_Toc43129202)

[Подключение Vue.js 5](#_Toc43129203)

[Framework Angular.js 6](#_Toc43129204)

[Основные понятия Angular.js 6](#_Toc43129205)

[Пример 9](#_Toc43129206)

[Настройка Angular.js 9](#_Toc43129207)

[Библиотека React.js 10](#_Toc43129208)

[Подключение React.js 11](#_Toc43129209)

[Сравнение Angular.js | Vue.js | React.js 12](#_Toc43129210)

[Angular.js 13](#_Toc43129211)

[React.js 14](#_Toc43129212)

[Vue.js 14](#_Toc43129213)

[Выводы 15](#_Toc43129214)

# JavaScript

JavaScript это высокоуровневый, интерпретируемый язык программирования. Это динамический, прототипный и мультипарадигмальный язык. JavaScript это одна из трех основных технологий, используемых в веб-разработке, наряду с CSS и HTML. Используется он для придания веб-страницам интерактивности.

Большинство сайтов используют JavaScript. Он поддерживается большинством современных браузеров, благодаря встроенным в них JavaScript-движкам. Этот язык поддерживает функциональный, объектно-ориентированный и прототипный стили программирования. Новички часто путают JavaScript и Java, несмотря на то, что это два принципиально разных языка программирования.

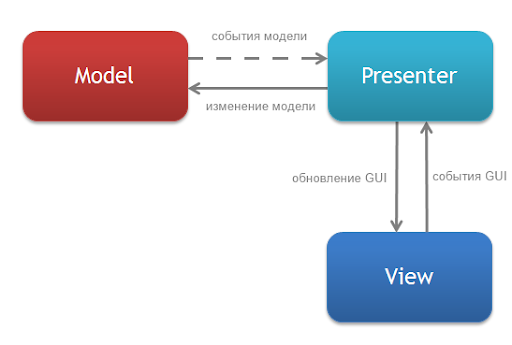
# JS FrameWorks

Фреймворк – это платформа, которая предоставляет программистам основу для разработки приложений. Он содержит заранее определенные и реализованные классы и функции.

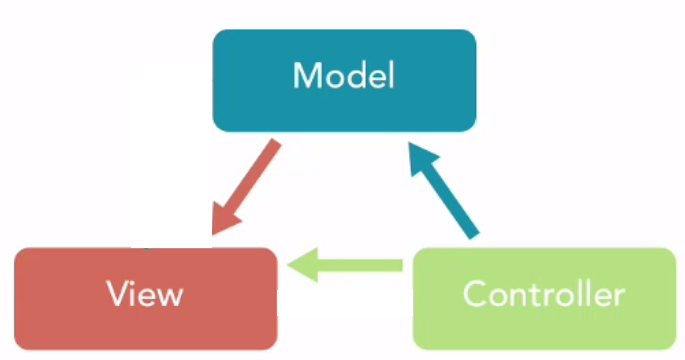
В подавляющем большинстве случаев, фреймворки используются для написания, так называемых, Single Page Applications. Т.е. все, что происходит на сайте, происходит на одной страничке, без прямого перехода с нее. С их помощью можно разрабатывать как полноценные сайты, так и функциональные модули (различные онлайн-инструменты). Конечно, полноценные фреймворки лучше подходят для первой задачи, а для второй рекомендуется использовать более легковесные фреймворки или библиотеки.

Они предоставляют четкую структуру приложения и реализуются с использованием так называемых «паттернов проектирования» (понятие, перекочевавшее больше из бекенда, но явно имеющее место во фронтенд разработке с использованием js-фреймворков). Наиболее широко распространены следующие паттерны: MVC (Model-View-Controller), MVP (Model-View-Presenter) и MVVM (Model-View-ViewModel).

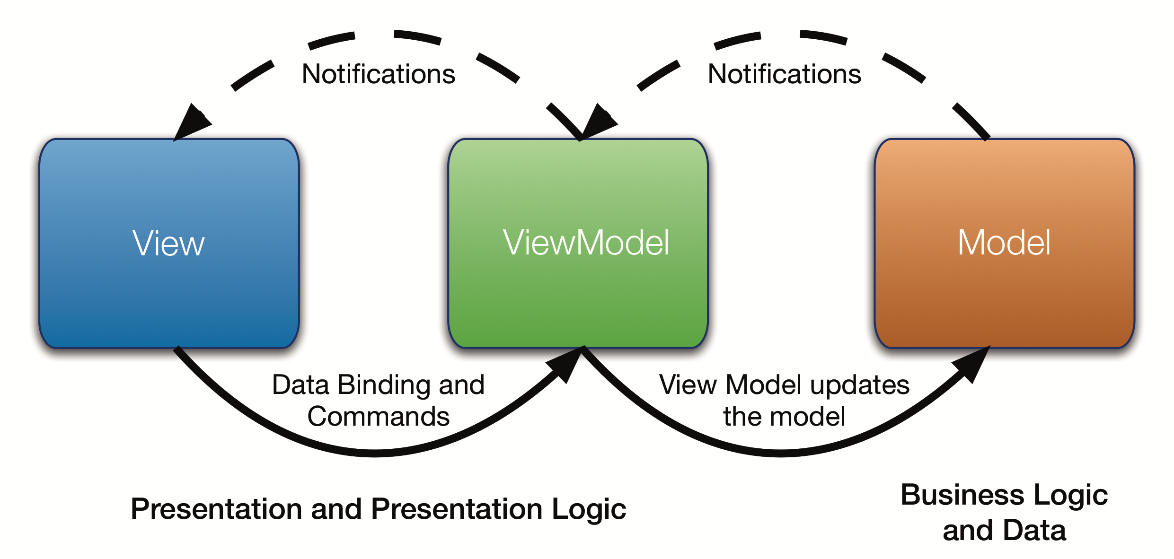
**MVP**



**MVC**



**MVVM**



## Преимущества построения приложения на JS-фреймворке:

* Легко реализовать SPA (Single Page Application)
* Код становится более структурированным
* Более чистый код
* Модульность приложения дает возможность проще работать над приложением нескольким разработчикам одновременно

Из существенных недостатков можно выделить только временно неполную поддержку поисковыми системами, но эта задача редко совпадает с задачей по реализации SPA (Single Page Application), тем более, что ведущие поисковые системы (как минимум, Google), уже практически полностью решили эту проблему.

Самые популярные JS-фреймворки:

1. Angular.js
2. React.js
3. Ext.js
4. Ember.js
5. Backbone.js
6. GWT
7. Knockout
8. Vue.js
9. Dojo
10. Polymer

# Framework Vue.js

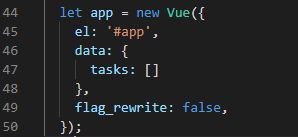
Vue (произносится /vjuː/, примерно как view) — это прогрессивный фреймворк для создания пользовательских интерфейсов. В отличие от фреймворков-монолитов, Vue создан пригодным для постепенного внедрения. Его ядро в первую очередь решает задачи уровня представления (view), что упрощает интеграцию с другими библиотеками и существующими проектами. С другой стороны, Vue полностью подходит и для создания сложных одностраничных приложений (SPA, Single-Page Applications), если использовать его совместно с современными инструментами и дополнительными библиотеками.

## Концепции Vue.js

Основными концепциями Vue являются:

* Конструктор
* Компоненты
* Директивы
* Переходы

Работа с Vue.js начинается с создания нового new Vue. Рассмотрим это на примере нашего кода:

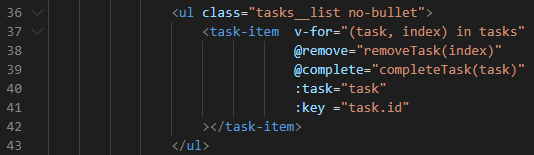


В el у нас элемент, за которым следит Vue. В data хранится текущее состояние.

Директивы – специальные атрибуты для добавления элементам html дополнительной функциональности. Рассмотрим некоторые встроенные директивы:

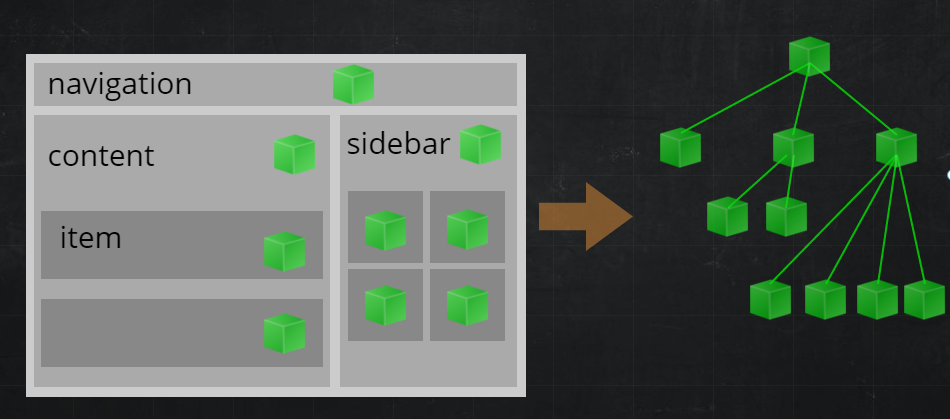
* V-bind — динамически связывается с одним или несколькими атрибутами.
* V-cloak — прячет “усатые” выражения, пока не подтянулись данные
* v-if — условие для рендера элемента
* V-else — обозначает “else блок” для v-if
* V-for — циклично проходит массив объектов
* V-model — связывает состояние с input элементом
* V-on — связывает слушателя события с элементом
* V-once — рендерит элемент только вначале и больше не следит за ним
* V-pre — не компилирует элемент и его дочерние элементы
* V-show — переключает видимость элемента, изменяя свойство CSS display
* V-text — обновляет textContent элемента

В нашем html коде мы используем директиву V-for:

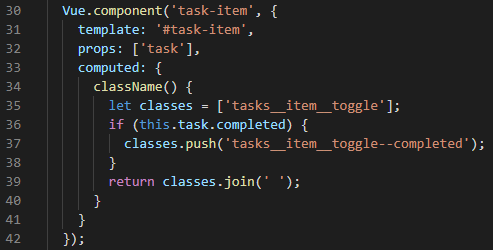


Все Vue-директивы имеют префикс “v-”. В директиву передается какое-то значение состояния, а в качестве аргументов могут быть атрибуты html или события.

Компоненты помогают расширить основные html элементы и внедрить переиспользуемый код. По сути, компоненты — повторно используемые части UI. На этапе проектирования мы разбиваем наше приложение на независимые части и получаем древовидную структуру компонентов.



Новый компонент объявляется с помощью Vue.component, и в первый аргумент мы передаем имя нового тега:



В template выбран элемент, куда Vue будет рендерить.

Vue предоставляет различные способы применения анимационных эффектов, когда элементы отрисованы, обновлены или удалены из DOM. Они включают в себя инструменты для:

* Автоматического применения классов для CSS-переходов и анимации
* Интеграции сторонних библиотек для CSS-анимации
* Использование JavaScript для манипуляции DOM-ом
* Интеграции сторонних библиотек JavaScript для анимации.

## Подключение Vue.js

Проще всего установить Vue.js, создав файл html и подключив к нему Vue:  
<script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>

# Framework Angular.js

AngularJS - это фреймворк с открытым исходным кодом для создания фронтенда веб-приложений. Он нацелен на решение нескольких задач, с которыми сталкивается разработчик при построении одностраничных приложений.

Этот фреймворк упрощает разработку и тестирование приложений. Он реализует подход Model-View-Controller (MVC) и Model-View-ViewModel (MVVM).

AngularJS представляет собой комплексный фреймворк. В стандартной поставке он предоставляет следующие возможности:

* Все, что вам нужно для создания CRUD-приложений: data-binding, базовые директивы для шаблонов, валидация форм, роутинг, deep linking, повторное использование компонентов, dependency injection, инструменты для взаимодействия с серверными (RESTful) источниками данных.
* Все, что вам нужно для тестирования: средства для модульного тестирование, end-to-end тестирования, mock-и.
* Шаблон типового приложения, включающего в себя структуру каталогов и тестовые скрипты.

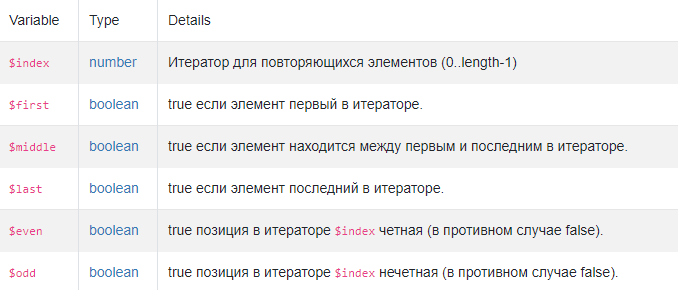
## Основные понятия Angular.js

*Директивы*

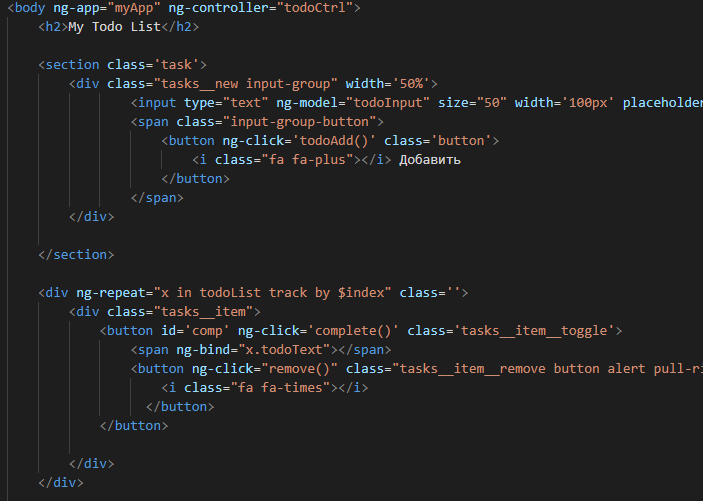
На директивах держится практически вся декларативная часть AngularJS. Именно они используются для обогащения синтаксиса HTML. В процессе компиляции DOM директивы берутся из HTML и исполняются. Директивы могут добавить какое-то новое поведение и/или модифицировать DOM. В стандартную поставку входит достаточно большое количество директив для построения веб приложений. Но ключевой особенностью является возможность разработки своих директив, за счет чего HTML может быть превращен в DSL.

Пример использования директив в нашем проекте:

1. ng-app: это директива, сообщающая AngularJS: «здесь находится приложение». DOM, окружённый элементом с этой директивой будет работать под управлением AngularJS.
2. ng-controller: директива, сообщающая, что тут подключен контроллер
3. ng-model: связывает модель с переменной
4. ng-click: срабатывание по клику и выполнение выражения в атрибуте
5. ng-repeat: позволяет пройтись по коллекции элементов от начала и до конца. Директива используется вместе с следующими специальными свойствами из локальной области видимости.



1. ng-bind: связывает значение какой-либо переменной с компонентом DOM



*Scope-ы*

Scope — это объект, имеющий отношение к модели в приложении. Он является контекстом выполнения для выражений. Scope-ы выстраиваются в иерархическую структуру, похожую на DOM. При этом они наследуют свойства от своих родительских scope-ов.

Scope-ы являются как бы «клеем» между контроллером и представлением. В процессе выполнения фазы связывания шаблона директивы устанавливают наблюдение ($watch) за выражениями в рамках scope. $watch дает директивам возможность реагировать на изменения для отображения обновленного значения или каких-либо других действий. И контроллеры, и директивы имеют ссылку на scope, но не имеют ссылок друг на друга. Так контроллеры изолируются от директив и от DOM-а. За счет этого возрастают возможности по тестированию приложения.

*Сервисы*

Сервисы — singleton-ы, выполняющие какую-либо конкретную задачу, которая является общей для всех или конкретного веб-приложения. Например, $http сервис, который является оберткой над XMLHttpRequest. Несколько примеров других сервисов (полный список смотрите в документации):

* $compile — компиляция HTML-строки или части DOM-а в шаблон, связывание шаблона с конкретным scope-ом;
* $cookies — предоставляет доступ на чтение/запись к cookies.
* $location — работа с адресной строкой
* $resource — фабрика по созданию ресурсных объектов, предназначенных для взаимодействия с серверными (RESTful) источниками данных;

Для использования сервиса необходимо указать его как зависимость для контроллера, другого сервиса, директивы и т.п. AngularJS позаботится обо всем остальном — создании, разрешении зависимостей и т.п.

*Фильтры*

Фильтры предназначены для форматирования данных перед отображением их пользователю, а также фильтрации элементов в коллекциях. Примеры фильтров (полный список можно посмотреть в документации): currency, date, orderBy, uppercase. Использование фильтров достаточно традиционно: {{ expression | filter1 | filter2 }}

*Модули*

Приложения в AngularJS не имеют основного исполняемого метода. Вместо этого модуль выполняет роль декларативного описания того, как приложение должно быть загружено. Благодаря этому, например, при написании сценариев тестирования можно подгрузить дополнительные модули, которые переопределят какие-то настройки, облегчая тем самым комплексное (end-to-end) тестирование.

## Пример

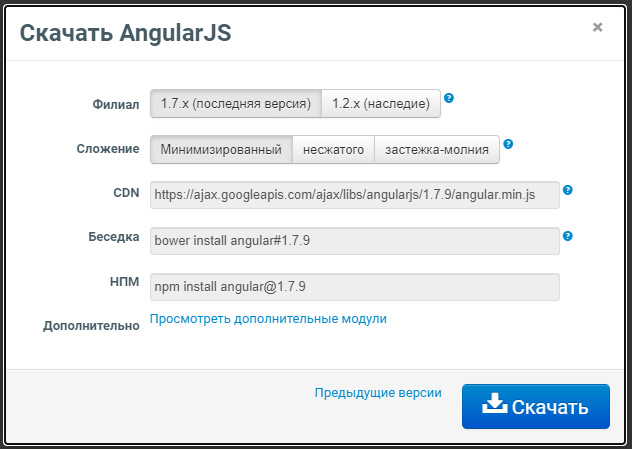
Ниже представлен фрагмент кода, где мы подключаем Ангулар, создаем контроллер:



## Настройка Angular.js

Чтобы упростить задачу, было решено установить не angular-cli, а воспользоваться доступными файлами по ссылкам:

<https://angularjs.org/>



# Библиотека React.js

React.js – это библиотека JavaScript и фреймворком не является. Он используется для построения пользовательских интерфейсов. React поддерживается Facebook, Instagram и другими компаниями. В шаблоне Model-View-Controller (MVC) он соответствует Представлению (View).

React может использоваться для разработки одностраничных и мобильных приложений. Его цель — предоставить высокую скорость, простоту и масштабируемость. В качестве библиотеки для разработки пользовательских интерфейсов React часто используется с другими библиотеками, такими как MobX, Redux и GraphQL.

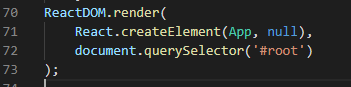
React значительно облегчает создание интерфейсов благодаря разбиению каждой страницы на небольшие фрагменты. Мы называем эти фрагменты компонентами.



Компонент React — это, если по-простому, участок кода, который представляет часть веб-страницы. Каждый компонент — это JavaScript-функция, которая возвращает кусок кода, представляющего фрагмент страницы.

Для формирования страницы мы вызываем эти функции в определённом порядке, собираем вместе результаты вызовов и показываем их пользователю.

Мы можем использовать так называемый ReactDOM для вывода того, что представляет этот код, на страницу:





На последнем скрине мы видим класс App. В React.js - это класс компонентов. Классы компонентов обязательно должны содержать функцию render(). Эта функция возвращает JSX-код компонента.

Особенность создаваемых компонентов:

Компоненты, основанные на классах, могут хранить информацию о текущей ситуации. Эта информация называется состоянием (state), она хранится в JS-объекте.



Особенность механизма привязки данных

В React используется односторонняя привязка данных. Это — большой плюс данной библиотеки, так как выражается это в том, что программист всегда точно знает о том, что привело к изменению состояния приложения. Подобный подход к привязке данных значительно упрощает отладку приложений.

## Подключение React.js

Чтобы подключить React.js к html, нужно добавить ссылку на библиотеку:

 <script src="https://cdnjs.cloudflare.com/ajax/libs/react/16.12.0/umd/react.production.min.js"></script>

 <script src="https://cdnjs.cloudflare.com/ajax/libs/react-dom/16.11.0/umd/react-dom.production.min.js"></script>

# Сравнение Angular.js | Vue.js | React.js

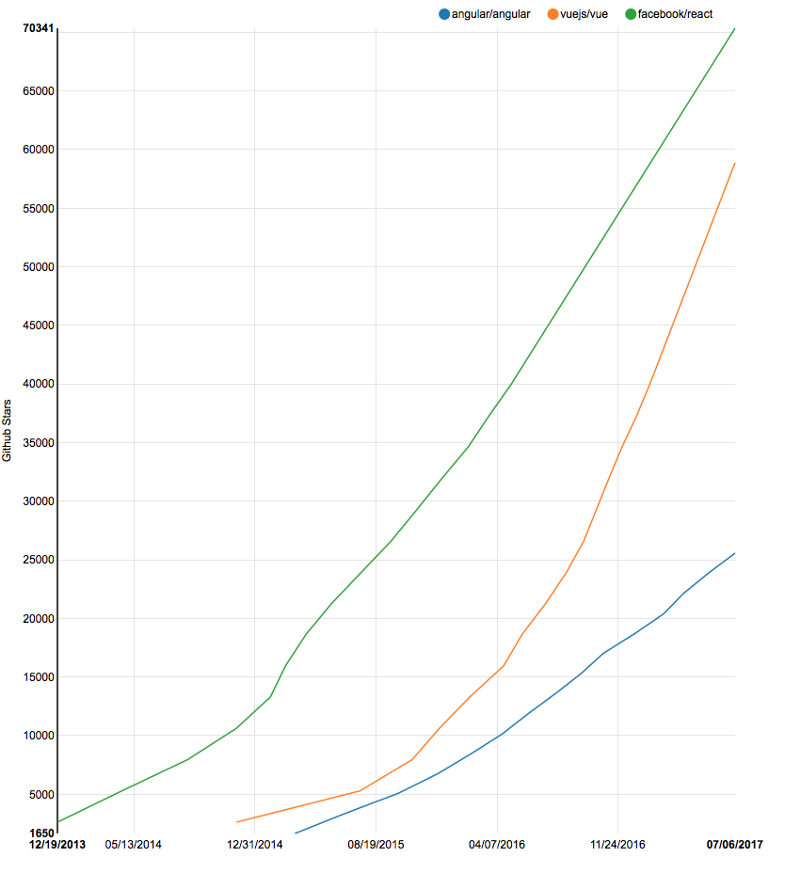
Angular – разработан и поддерживается компанией Google. Над проектам работают 36 человек, React – разработан и поддерживается Facebook, Vue – создан группой разработчиков (16 человек).

Как уже отмечалось ранее, Angular и React активно используются и имеют поддержку больших компаний: Facebook, Instagram и Whatsapp. Также они активно используются компанией Google для своих проектов (например, с помощью Angular & Dart было реализовано множество вещей для нового Adwords UI). С другой стороны, Vue был создан группой разработчиков, которые получали поддержку только благодаря Patreon, а также других спонсоров.

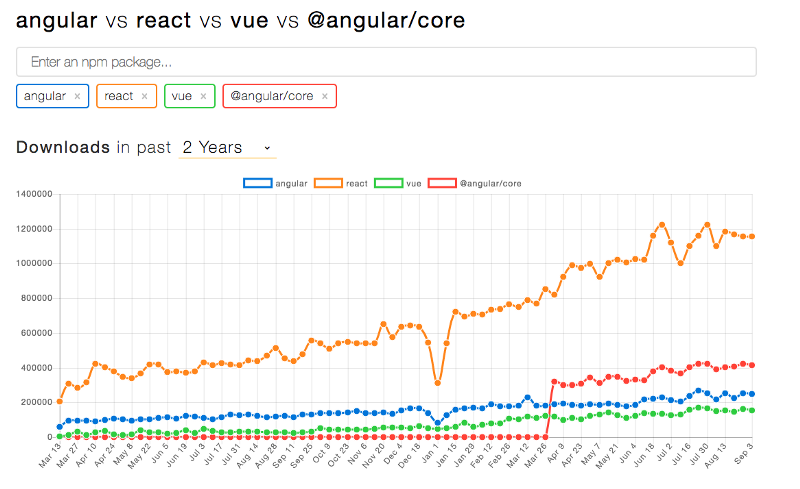
Немного статистики:

На Github Angular имеет более 25000 звезд и 463 последователя, React – 70000 звезд и более чем 1000 последователей, а Vue умудрился собрать 60000 звезд и только 120 последователей.

Кажется, что Vue довольно быстро набирает обороты: например, в течение последних трех месяцев Angular 2 ежедневно в среднем собирал 31 звезду, React – 74 звезды, а Vue.JS – 107 звезд.



Также можно отследить количество загрузок определенных пакетов npm:



## Angular.js

|  |  |
| --- | --- |
| **Преимущества** | **Недостатки** |
| Использует HTML & CSS хорошо известным способом. | Изучение TS и ключевых функций, таких как DI |
| Подробная документация, позволяющая получить всю необходимую информацию. | Проблемы с миграцией, которые могут возникнуть при переходе от старой версии к последней. |
| MVVM (Model-View-ViewModel) | Требуется больше времени на обучение. |
| Новые функции, такие как улучшенный RXJS, более быстрая компиляция (менее чем за 3 секунды), новый запуск HttpClient. |  |
| Двусторонняя привязка данных |  |
| Маршрутизация, управление состоянием |  |
| Goolge Долгосрочная поддержка. |  |

## React.js

|  |  |
| --- | --- |
| **Преимущества** | **Недостатки** |
| Простота с точки зрения синтаксиса. | Отсутствие официальной документации |
| Высокий уровень гибкости и максимальная отзывчивость. | React не принят, что означает, что разработчики иногда имеют слишком большой выбор |
| Можно добавить в существующие веб-приложения / представления. |  |
| Легко работает с высокой нагрузкой. |  |
| 100% библиотека JavaScript с открытым исходным кодом |  |
| Нисходящая привязка данных |  |
| Данные со стороны пользователя легко представляются на стороне сервера. |  |
| Миграция между версиями очень проста. |  |
| Виртуальный DOM (объектная модель документа) |  |

## Vue.js

|  |  |
| --- | --- |
| **Преимущества** | **Недостатки** |
| Простота с точки зрения синтаксиса. | Нехватка ресурсов |
| Легкий и простой в освоении. | Некоторые ключевые функции для более сложных приложений должны быть добавлены вручную. |
| Основная функция, предоставляемая основной командой. | Быть слишком гибким может быть проблематично. |
| Хорошо использует HTML и CSS |  |
| Подробная документация. |  |
| Простая интеграция. |  |
| Большое масштабирование. |  |
| Маленький размер. |  |

# Выводы

Мы знаем плюсы и минусы каждого фреймворка и можем сделать выводы, какой лучше подойдет для тех или иных целей:

Angular.js:

* Если любите кодировать на TypeScript
* Если увлекаетесь ООП
* Если хотите сосредоточиться на крупномасштабных, многофункциональных приложениях
* Если хотите разрабатывать нативные приложения, гибридные или веб-приложения (долгосрочные или тяжелые инвестиционные проекты)

React.js:

* Если нравится гибкость больше, чем другие функции
* Если любите JavaScript
* Если хотите разрабатывать кроссплатформенные приложения, SPA или мобильные приложения

Vue.js:

* Если являетесь поклонником чистого кода
* Если вы новичок
* Если собираетесь разрабатывать небольшой проект
* Если работаете в одиночку или небольшой команде
* Если нужен фреймворк, который меньше весит