**Angular JS**

AngularJS — JavaScript-фреймворк с открытым исходным кодом. Предназначен для разработки одностраничных приложений. Его цель — расширение браузерных приложений на основе MVC шаблона, а также упрощение тестирования и разработки.

Фреймворк работает с HTML, содержащим дополнительные пользовательские атрибуты, которые описываются директивами, и связывает ввод или вывод области страницы с моделью, представляющей собой обычные переменные JavaScript. Значения этих переменных задаются вручную или извлекаются из статических или динамических JSON-данных.

AngularJS был первоначально разработан в 2009 году Мишко Хевери и Адамом Абронсом в Brat Tech LLC, как программное обеспечение позади сервиса хранения JSON-данных, измеряющихся мегабайтами, для облегчения разработки приложений организациями. Сервис располагался на домене «GetAngular.com» и имел нескольких зарегистрированных пользователей, прежде чем они решили отказаться от идеи бизнеса и выпустить Angular как библиотеку с открытым исходным кодом.

Абронс покинул проект, но Хевери, работающий в Google, продолжает развивать и поддерживать библиотеку с другими сотрудниками Google Игорем Минаром и Войта Джином.

**Философия Angular**

AngularJS спроектирован с убеждением, что декларативное программирование лучше всего подходит для построения пользовательских интерфейсов и описания программных компонентов, в то время как императивное программирование отлично подходит для описания бизнес-логики. Фреймворк адаптирует и расширяет традиционный HTML, чтобы обеспечить двустороннюю привязку данных для динамического контента, что позволяет автоматически синхронизировать модель и представление. В результате AngularJS уменьшает роль DOM-манипуляций и улучшает тестируемость.

**Цели разработки**

* Отделение DOM-манипуляции от логики приложения, что улучшает тестируемость кода.
* Отношение к тестированию как к важной части разработки. Сложность тестирования напрямую зависит от структурированности кода.
* Разделение клиентской и серверной стороны, что позволяет вести разработку параллельно.
* Проведение разработчика через весь путь создания приложения: от проектирования пользовательского интерфейса, через написание бизнес-логики, к тестированию.

AngularJS придерживается MVC-шаблона проектирования и поощряет слабую связь между представлением, данными и логикой компонентов. Используя внедрение зависимости, AngularJS переносит на клиентскую сторону такие классические серверные службы, как видозависимые контроллеры. Следовательно, уменьшается нагрузка на сервер и веб-приложение становится легче.

**Angular-директивы**

С помощью директив AngularJS можно создавать пользовательские HTML-теги и атрибуты, чтобы добавить поведение некоторым элементам.

**ng-app**: объявляет элемент корневым для приложения.

**ng-bind**: автоматически заменяет текст HTML-элемента на значение переданного выражения.

**ng-model**: то же что и **ng-bind**, только обеспечивает двустороннее связывание данных. Изменится содержимое элемента, AngularJS изменит и значение модели. Изменится значение модели, AngularJS изменит текст внутри элемента.

**ng-class**: определяет классы для динамической загрузки.

**ng-controller**: определяет JavaScript-контроллер для вычисления HTML-выражений.

**ng-repeat**: создает экземпляр для каждого элемента из коллекции.

**ng-show** и **ng-hide**: показывает или скрывает элемент в зависимости от значения логического выражения.

**ng-switch**: создает экземпляр шаблона из множества вариантов, в зависимости от значения выражения.

**ng-view**: базовая директива, отвечает за обработку маршрутов, которые принимают JSON перед отображением шаблонов, управляемых указанными контроллерами.

**Двустороннее связывание данных**

Двустороннее связывание данных в AngularJS является наиболее примечательной особенностью и уменьшает количество кода, освобождая сервер от работы с шаблонами. Вместо этого, шаблоны отображаются как обычный HTML, наполненный данными, содержащимися в области видимости, определенной в модели. Сервис $scope в AngularJS следит за изменениями в модели и изменяет раздел HTML-выражения в представлении через контроллер. Кроме того, любые изменения в представлении отражаются в модели. Это позволяет обойти необходимость манипулирования DOM и облегчает инициализацию и прототипирование веб-приложений.

**Недостатки AngularJS**

* Двусторонний биндинг означает, что изменив что-либо в своем приложении, это тригерит сотни функций, которые наблюдают за изменениями. И это чудовищно медленная операция, особенно все становится плохо на мобильных платформах. И это фундаментальная часть фреймворка.
* Фреймворк не предназначен для вывода большого количества информации на страницу, т.к. это значительно замедлит работу приложения по причине постоянных проверок хэш сумм моделей.
* Невозможность рендерить страницы на стороне сервера.
* Проблемы с производительностью отдельных ключевых директив.
* Необходимость идти по так называемому "пути AngularJS" - это означает негибкость фреймворка, которая вынуждает разработчика писать код одним способом, которым хочет AngularJS (необязательно единственно верным).
* Сложность устранения ошибок в процессе разработки (большое количество ошибок никак не отображаются в консоли).
* Строго html-ориентированный
* Система маршрутизации ($routeprovider) требует серьезных доработок
* Монолитность, невозможность использовать компоненты отдельно
* Отсутствует возможность отложенной загрузки модулей (lazy-loading)
* Области видимости $scope не совсем прозрачны