**Angular**

**Книги**

Angular для профессионалов стр.418

Angular и TypeScript. Сайтостроение для профессионалов стр.22

**Сайты**

Введение в Angular: что это за фреймворк и зачем его использовать

<https://webformyself.com/vvedenie-v-angular-chto-eto-za-frejmvork-i-zachem-ego-ispolzovat/>

Просто Angular

<https://habr.com/ru/post/341688/#s1_1>

<https://angular.io/>

Руководство по Angular 7

<https://metanit.com/web/angular2/>

<https://angular.io/docs>

Удивительный Angular

<https://habr.com/ru/post/348818/#s1>

**Обоснование выбора Angular.**

Современные сайты сильно полагаются на JS, поэтому необходимость правильной организации и тестирования кода возросла. Невозможно создать сложный, эффективный и удобный в обслуживании интерфейс на чистом JavaScript. В JavaScript тяжело поддерживать синхронизацию интерфейса и состояние приложения. Поэтому клиентские фреймворки стали популярны, и на данный момент их, как минимум, десяток.

Angular позволяет из "коробки" создавать большие и сложные по части бизнес-логики приложения. В Angular уже есть много инструментов для создания приложения.

Angular подготовлен к мобильным устройствам и десктопу – один фреймворк под множество платформ.

Angular представляет не только инструменты, но и шаблоны дизайна для создания обслуживаемого проекта. В Angular представление и логика приложения не связаны, что сильно очищает и упрощает разметку, т.е. код удобно структурирован.

Angular построен на TypeScript, который, в свою очередь, полагается на ES6. Angular убрал жесткую связь между различными компонентами приложения. Инъекция проходит подобно NodeJS, что позволяет легко заменять компоненты.

Angular активно обслуживается и имеет большое сообщество и экосистему. По фреймворку можно найти много материалом и полезных сторонних инструментов.

Angular тщательно протестирован и поддерживает юнит тесты и сквозное тестирование с помощью инструментов типа Jasmine и Protractor.

* **Детальное описание технологии.**

Angular (версия 2 и выше) — это открытая и свободная платформа для разработки веб-приложений, написанная на языке TypeScript, разрабатываемая командой из компании Google, а также сообществом разработчиков из различных компаний. Angular — это полностью переписанный фреймворк от той же команды, которая написала AngularJS.

Изначально создавался как вторая версия AngularJS. Angular 2 был переписан с нуля на TypeScript, обладает иной архитектурой и не является обратно совместимым с AngularJS, в связи с чем для предотвращения путаницы было решено развивать его как отдельный фреймворк, нумерация версий которого начинается с 2.

Необходимо знать множество дополнительных технологий для работы с Angular:

Рекомендуется писать приложения Angular на TypeScript. Необходимо писать код на современном JS (ES6).

Для ускорения процесса разработки, нужно использовать Angular CLI. Angular CLI — это интерфейс командной строки, который позволяет быстро создавать проекты, добавлять файлы и выполнять множество определенных задач, такие как тестирование, сборка и развертывание. Для установки Angular и других компонентов активно используется npm или yarn.

**Angular приспособления для облегчения разработки UI:**

* ***Компоненты***

Компонент управляет отображением представления на экране, в ее основе используется Shadow DOM по умолчанию (для создания инкапсулированного визуального поведения). Как правило, компоненты используются для создания простого виджета в пользовательском интерфейсе, в то же время они могут представлять из себя набор еще более простых компонентов внутри себя (для увеличения абстракции и создания простых функциональных виджетов внутри приложения).

Каждый компонент имеет свой собственный детектор изменений, который гарантирует проверку привязок данных, определенных шаблоне.

* ***Шаблоны***

Шаблон — это html-разметка, в которой вы можете описывать ваши взаимодействия с DOM на основе модели данных и событий вашего класса компонента.

* ***Внедрение зависимостей***

Внедрение зависимостей (англ. Dependency Injection) — это композиция структурных шаблонов проектирования, при которой за каждую функцию приложения отвечает один, условно независимый объект (сервис), который может иметь необходимость использовать другие объекты (зависимости), известные ему интерфейсами. Зависимости передаются (внедряются) сервису в момент его создания.

* ***Директивы***

Директивы позволяют получать прямой доступ к DOM элементов. Они бывают двух видов: структурные и атрибутные. Благодаря директивам которые определяют набор инструкций, которые применяются при рендеринге html-кода т.е. HTML элементы могут вести себя динамически. Структурные директивы изменяют структуру DOM с помощью добавления или удаления html-элементов. Существует минимум три встроенных структурных директивы: ngIf, ngSwitch и ngFor.

* ***Пайпы***

Пайп (pipe) представляет собой особый обработчик, который позволяет форматировать отображаемые значения. Каждое приложение начинается с простой задачи: получить данные, преобразовать их и показать пользователям. Получение данных может быть таким же простым, как создание локальной переменной, или таким же сложным, как потоковая передача данных.

Как только данные поступают, можно поместить их необработанные значения toString непосредственно в представление, но это редко дает хороший пользовательский опыт. В большинстве случаев пользователи предпочитают видеть дату в простом формате, таком как 15 апреля 1988 года, а не в формате необработанной строки Пт 15 апреля 1988 00:00:00 GMT-0700 (тихоокеанское летнее время).

Очевидно, что некоторые значения выигрывают от небольшого редактирования. Можно применить их в своих шаблонах HTML, как и стили. Пайп предоставляет возможность писать преобразования отображаемых значений, которые можно объявить в своем HTML.

* ***Класс HttpClient***

Для взаимодействия с сервером и отправки запросов по протоколу http применяется класс HttpClient. Этот класс определяет ряд методов для отправки различного рода запросов: GET, POST, PUT, DELETE. Данный класс построен поверх стандартного объекта в JavaScript - XMLHttpRequest. Angular HttpClient довольно прост. Необходимо вызвать метод get и передать ему url. Данный метод get возвращает объект Observable. Этот класс является частью библиотеки RxJS ("Reactive Extensions"), которая используется во многих местах Angular'а.

* ***Angular-CLI***

Angular CLI облегчает создание приложения, которое уже работает, прямо из коробки. Генерация компонентов, маршрутов, сервисов и каналов с помощью простой команды. CLI также создаёт простые тестовые оболочки вышеописанного. Легко тестировать приложение локально во время разработки.

Встроенный поток данных, безопасность типов и модульный интерфейс командной строки (CLI) являются отличительными признаками того, что делает Angular такой самодостаточной средой. Однако, учитывая, что это структура, которая помогает даже крупнейшим компаниям управлять своими корпоративными приложениями, в сочетании с обязательством Google не вносить более серьезных изменений, Angular будет только развиваться.