



Introduction to Angular JS

...

Framework By Google

What we will cover?

Why JS?

What is AngularJS?

Why use MV* or MVW?

AngularJS

- Architecture

- Features

- Quick Start

TypeScript - Brief

Demo

Comparison - “Big 4”

Why JS?

These day's most actions on webpage go back and forth to webserver.
Using JS in such case would provide:

- More interactive and responsive UI

- Save bandwidth

- Save CPU consumption on the web server

What?

Misko Hevery and Adam Abrons

Maintained by Google and community

Initial release: 2009

Current stable release: June 15, 2016, v1.5.7

Latest release candidate: Angular 2



What?

MVW framework to help build client apps in HTML and JS (or Dart, TypeScript).

Consists of some cooperating libraries some core and some optional.

Write applications by composing HTML templates with Angularized-markup.

Write component classes to manage those templates.

Add application logic in services, and handing the top root component to Angular's bootstrapper.

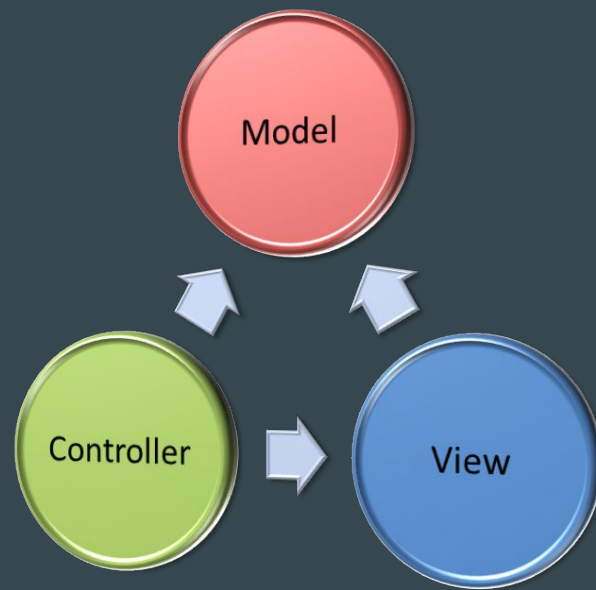
What is MVW? And Why use it?

Model-View-**Whatever**

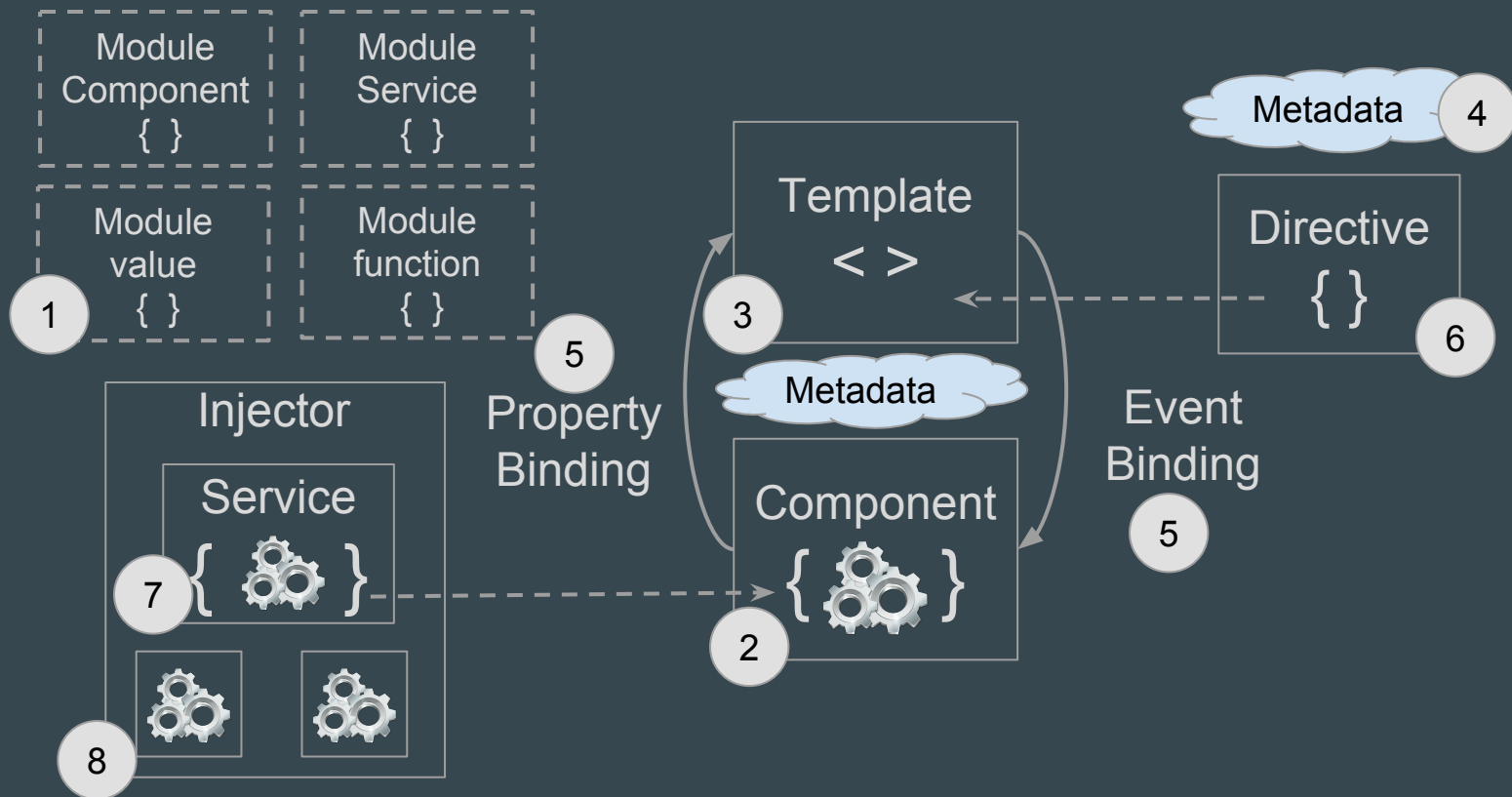
Separate presentation logic from business logic

Here, JS provides the model and “logic”

HTML/HTML5 provides the presentation logic



What? - Eight building blocks



AngularJS - Building Blocks

Eight main building blocks

Module
Component
{ }

Cohesive block of code for single purpose.

E.g., export class AppComponent { }

Library
Module
Component
{ }

Some modules act as libraries to other modules

E.g. @angular/core, @angular/common, ..
import { Component } from '@angular/core';

Module
Component
{ }

Contain patch to call a view

Application logic to support the view

Class interacts with view through properties and methods.

AngularJS - Building Blocks

Eight main building blocks

Templates

< >

Template is a form of HTML that tells Angular how to render the component.

Metadata

It tells Angular how to process a class. Decorators are used to attach metadata to “class”.

Data Binding

A mechanism for coordinating parts of a template with parts of a component.

Directives

Templates are dynamic. They transform the DOM according to the directives.

e.g. ngModel (see in Demo)

AngularJS - Building Blocks

Eight main building blocks

Service



Broad category encompassing value, function or feature that app needs. Available for injection.

E.g. logging service, app config, ..

Dependency Injection

Important application design pattern.

Angular provides its own dependency injection framework. Service exposes getter that returns data. @Injectable decorator is important above this Service class

Features

Cross Platform

Progressive Web Apps

Modern web platform capabilities.

High Performance

Native

Build native mobile apps.

Desktop

Apps for Mac, Windows, Linux for web plus ability to access native OS APIs.

Speed & Performance

Code generation

Turn templates into highly optimized JS code

Universal

Near-instant rendering in just HTML and CSS

Code Splitting

Auto code splitting

Only load code required to render the view

Features

Productivity

Templates

Create quickly UI views

Angular CLI

CLI tools - build fast, deploy
instant

IDEs

Full Development Story

Testing

Karma for unit tests

Protractor to make scenario run
fast and stable

Animation

Create high performance
animation with minimal effort

Accessibility

Quick start - 5 min

<https://github.com/manishaluthra/quickstart> (forked ;))

Verify that you are running at least node v5.x.x and npm 3.x.x (package mgr for JS) by running `node -v` and `npm -v` in a terminal/console window.

Use *nvm* to update node,

and update npm by:

```
npm install npm -g
```

Quick start - Steps

Prerequisite: Node.js

Package definition and configuration files

package.json

Npm package configuration

dependencies

devDependencies

Quick start - Dependencies - Npm package configuration

3 package categories

Features

e.g., @angular/core, @angular/common, @angular/compiler, @angular/http, ..

Polyfills

e.g., core-js, reflect-metadata, rxjs and zone.js

Other

e.g., angular2-in-memory-web-api, bootstrap

TypeScript - Brief Overview

Developed by Microsoft (<https://github.com/Microsoft/TypeScript>)

First release: October, 2012

Stable release: January 2016, v1.8

JavaScript that scales

Superset of JavaScript that compiles to plain JavaScript

Supports class based object-oriented programming

TypeScript - Features

Interface

```
e.g., interface Person {  
    firstName: string;  
    lastName: string;  
}  
  
function greet(person: Person) {  
    return "Hello, " + person.firstName + " " + person.lastName;  
}  
  
var user = { firstName: "Manisha", lastName: "Luthra"};  
  
document.body.innerHTML = greeter(user);
```

TypeScript - Features

Classes (continued...)

e.g., class Student {

 fullName: string;

 constructor(public firstName, public middleName, public lastName) {

 this.fullName = firstName + " " + middleName + " " + lastName;

 }

}

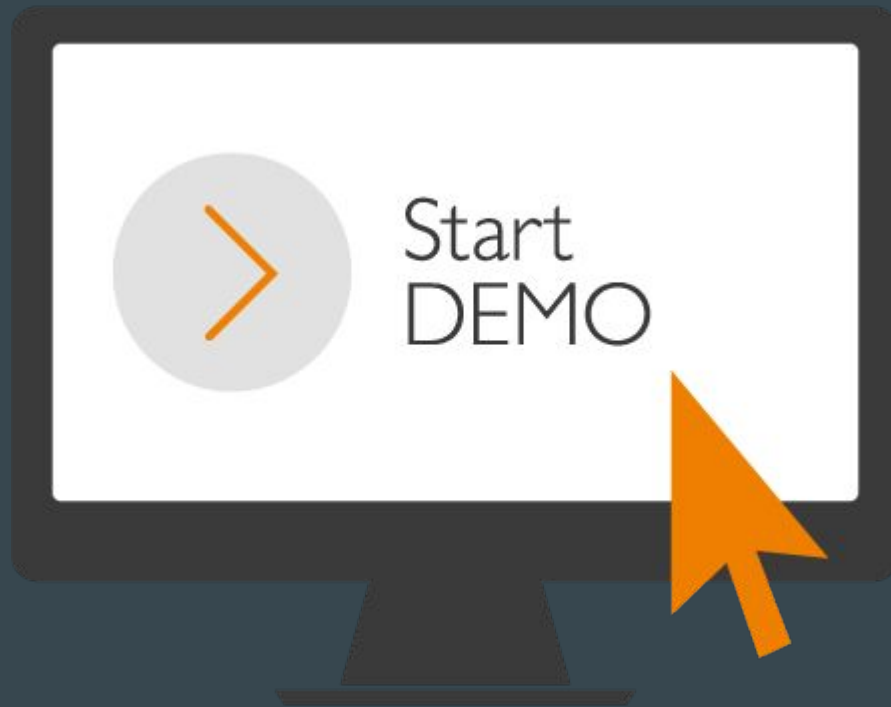
function greet(person: Person) {

}

var user = new Student("Manisha", "K.", "Luthra");

document.body.innerHTML = greeter(user);

Demo



Demo (1/2) - Template and Component

Start from Quick Start or clone

1. `$ npm start` (starts TypeScript compiler, have it watch for changes, and start our server)
2. `update app.component.ts`

```
import { Component } from '@angular/core';
```

```
@Component({  
  selector: 'my-app',  
  template: '<h1>{{title}}</h1><h2>{{hero}}</h2>'  
})  
export class AppComponent {  
  title = 'Angular 2 Tour';  
  hero = 'SAP Hana';  
}
```

Demo (2/2) - Two way binding by directive

```
import { Component } from
 '@angular/core';
```

```
@Component({
  selector: 'my-app',
  template: ` <h1>{{title}}</h1>
               <h2>{{company.name}}
details!</h2>
               <div><label>id: </label>
               {{company.id}}</div>
               <div>
                 <label>name: </label>
                 <input [(ngModel)]=
company.name" placeholder="
name">
               </div>
               `
})
```

```
export class AppComponent {
  title = 'Angular 2 Tour';
  company: Company = {
    id: 1;
    name: 'SAP AG'
  }
}
```

Other Competitive JS Frameworks

“Big 4” JS Frameworks

































Knockout

Backbone

React-Ember

Angular JS

Big 4 - Comparison Matrix

Features	Backbone.js	Ember	Knockout	AngularJS
Open source				
Quick start				
No Dependencies				
Data Binding				
View				
Testing				
Community support				
Third-party libraries				

Helpful Links

Tutorial: <https://docs.angularjs.org/tutorial>

API Docs: <https://docs.angularjs.org/api>

Developer Guide: <https://docs.angularjs.org/guide>

Cheat sheet: <https://angular.io/docs/ts/latest/guide/cheatsheet.html>

References

AngularJS: <https://angular.io/>

JS, MV*: <http://blog.orbitone.com/post/Angularjs-the-Model-View-Whatever-javascript-framework1>

TypeScript: <https://www.typescriptlang.org/index.html>

NgModel Directive AngularJS: <https://angular.io/docs/ts/latest/guide/forms.html#!#ngModel>