What is ANGULARJS?

•It's not a JavaScript library (As they say). There are no functions which we can directly call and use.

•It is not a DOM manipulation library like jQuery. But it uses subset of jQuery for DOM manipulation (called jqLite).

•Focus more on HTML side of web apps.

•For MVC/MVVM design pattern

•AngularJS is a Javascript MVC framework created by Google to build properly architectured and maintenable web applications.

Philosophy

“ANGULARJS is what HTML could have been if it had been designed for web application development.”

“”ANGULARJS is built around the philosophy that declarative code is better than imperative code while building UIs and wiring different components of web application together.”

```
<!doctype html>
<html ng-app>
  <head>
    <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.0.7/angular.min.js"></script>
  </head>
  <body>
    <div>
      <label>Name:</label>
      <input type="text" ng-model="yourName" placeholder="Enter a name here">
      <hr>
      <h1>Hello {{yourName}}!</h1>
    </div>
  </body>
</html>
```

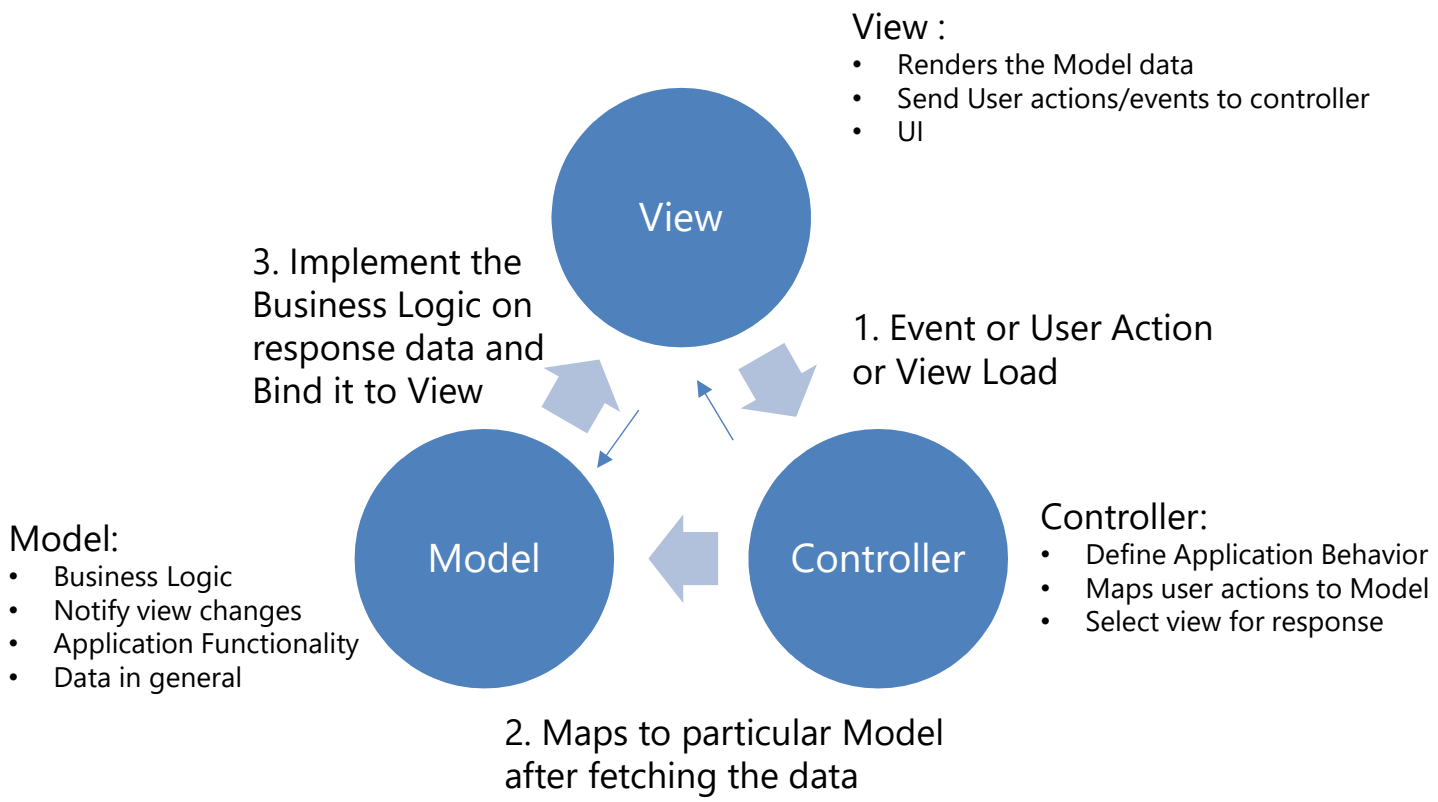
Why ANGULARJS?

- Defines numerous ways to organize web application at client side.
- Enhances HTML by attaching directives, custom tags, attributes, expressions, templates within HTML.
- Encourage TDD
- Encourage MVC/MVVM design pattern
- Code Reuse
- Good for Single Page Apps (SPA)
- Cool Features -> Next Slide

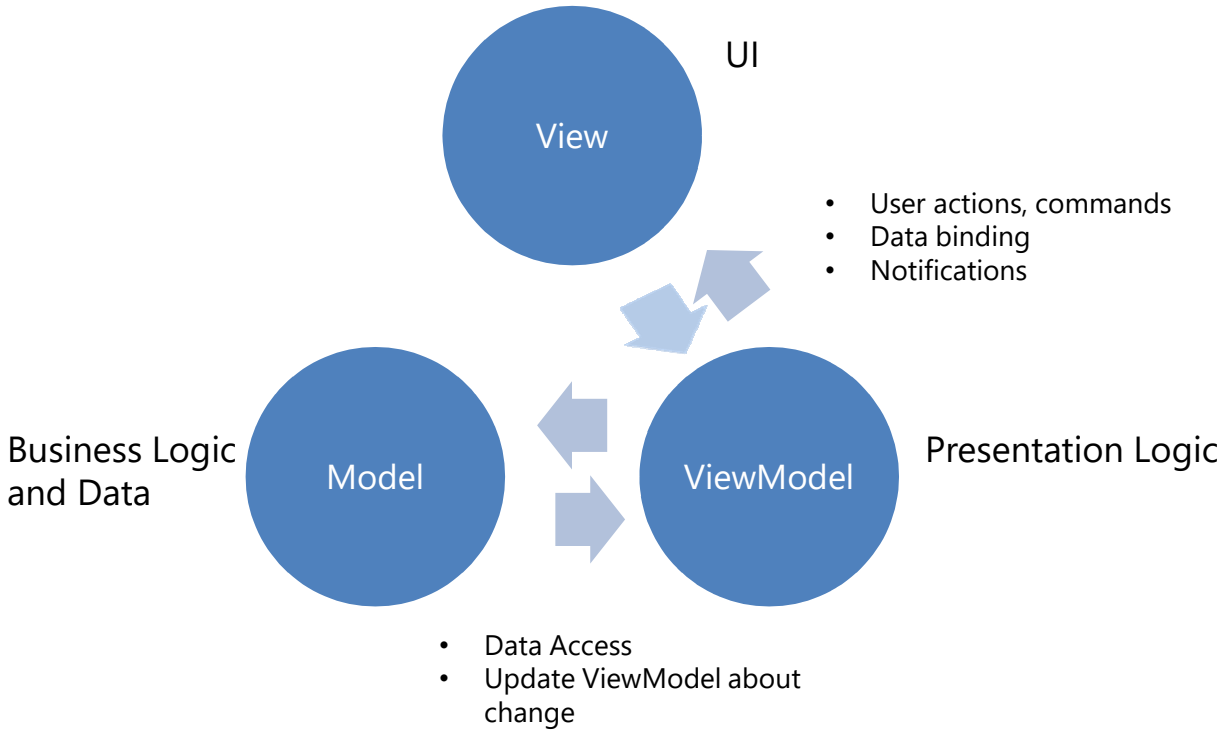
Key Features of ANGULARJS

- Declarative HTML approach
- Easy Data Binding : Two way Data Binding
- Reusable Components
- MVC/MVVM Design Pattern
- Dependency Injection
- End to end Integration Testing / Unit Testing
- Routing
- Templating
- Modules
- Services
- Expressions
- Filters
- Directives
- Form Validation
- $scope, $http, $routeProvider…

MVC : Model View Controller



MVVM: Model View ViewModel



ng-app

Use this directive to auto-bootstrap an application.

Only one ng-app directive can be used per HTML document

```
<html ng-app>
```

HTML Compiler

Angular's HTML compiler allows the developer to teach the browser new HTML syntax.

The compiler allows you to attach behavior to any HTML element or attribute and even create new HTML elements or attributes with custom behavior.

Angular calls these behavior extensions directives.
Compiler is an angular service which traverses the DOM looking for attributes.
The compilation process happens in two phases.

Compile: traverse the DOM and collect all of the directives. The result is a linking function.

Link: combine the directives with a scope and produce a live view.
Any changes in the scope model are reflected in the view, and any user interactions with the view are reflected in the scope model.
This makes the scope model the single source of truth.

Directive

The directives can be placed in element names, attributes, class names, as well as comments.

Directives are a way to teach HTML new tricks.

A directive is just a function which executes when the compiler encounters it in the DOM

<input **ng-model='name'**>
Custom Defined Directives
<span **draggable**>Drag ME

Expression

Expressions are JavaScript-like code snippets that are usually placed in bindings such as {{ expression }}

```
<body>1+2={{1+2}}</body>
```

Forms

Form and controls provide validation services, so that the user can be notified of invalid input.

This provides a better user experience, because the user gets instant feedback on how to correct the error.

<input type="text" ng-model="user.name" name="uName" **required** />
<button ng-click="update(user)“ ng-disabled="**form.$invalid ||** isUnchanged(user)">
SAVE</button>

Module

Modules declaratively specify how an application should be bootstrapped.

There can be multiple modules in an app

Those could be interdependent too.

```
// declare a module
var myAppModule = angular.module('myApp', [--here goes the dependent Modules--]);
```

Modules are configured with routes, controllers, models etc.

Routing

It Is used for deep-linking URLs to controllers and views (HTML partials).
It watches $location.url() and tries to map the path to an existing route definition.

```
$routeProvider.when('/Book', {
        template: 'examples/book.html',
        controller: BookCntl,
});
 $routeProvider.when('/Book/chapter01', {
        template: 'examples/chapter01.html',
        controller: ChapterCntl,
});
```

Scope

Scope is an object that refers to the application model.

It is an execution context for expressions.

Scopes are arranged in hierarchical structure which mimic the DOM structure of the application.

Scopes can watch expressions and propagate events.

Actually the ViewModel of MVVM.
$scope

Filters

Angular filters format data for display to the user

```
{{ expression [| filter_name[:parameter_value] ...
] }}

{{ uppercase_expression | uppercase }}
 {{ expression | filter1 | filter2 }}
```

Can create custom filters

Why Angular?

to create properly architectured and maintainable web applications

Expressions

To create the views of your applications, you can use expressions within your HTML

- Javascript like code
- Used for small operations in the HTML page

```
<div>1+1 = {{1+1}}</div>
```

1+1 = 2

Expressions are nice for small operations, for real applications, we have something more powerful

Directives

what HTML would have been, had it been designed for building web-apps

Directives

Extends HTML to structure your application

- Declarative
- Use the data available in the scope (more on that later)
- Create the DOM of the fly

Let's have a look at an example: ngRepeat.

```
<div>
    <div ng-repeat="user in users">
        <h3>{{user.name}}</h3>
        <p>{{user.description}}</p>
    </div>
</div>
```

It iterates on a collection in the scope to create the DOM

Directives - ngRepeat

```
<div>
    <div ng-repeat="user in users">
        <h3>{{user.name}}</h3>
        <p>{{user.description}}</p>
    </div>
</div>
```

For each elements in the collection "users" a new has been created with all its children

Directives - ngShow

AngularJS comes with a collection of standard directives that can be combined

```
<div>
    <div ng-repeat="user in users" ng-show="user.gender == 'female'">
        <h3>{{user.name}}</h3>
        <p>{{user.description}}</p>
    </div>
</div>
```

ngShow let you hide elements that do not validate a given predicate
Here, the users that do not have the gender "female" have generated a hidden

Directives - ngSwitch

AngularJS also provides you with complex directives like ngSwitch

```
<div>
    <div ng-repeat="user in users"
         ng-show="user.gender == 'female'"
         ng-switch="user.house">
        <h3>{{user.name}}</h3>
        <p>{{user.description}}</p>
        Sigil:
        <img src="images/targaryen.gif" ng-switch-when="Targaryen">
        <img src="images/stark.gif" ng-switch-when="Stark">
        <img src="images/lannister.gif" ng-switch-when="Lannister">
    </div>
</div>
```

With those directives, you can create the basic structure of your web application easily

Directives

A final word on directives

- All the directives of the AngularJS standard library are named "ngMyAwesomeDirective"
- You can use them with "ng-my-awesome-directive"
- Some directives can be used as attributes, comments, DOM elements name or even CSS classes

```
<div>
    <div ng-my-awesome-directive></div>
    <ng-my-awesome-directive></ng-my-awesome-directive>
    <div class="ng-my-awesome-directive"></div>
    <!-- directive: ng-my-awesome-directive -->
</div>
```

And of course, you can create your own directives (we will create a very basic one later)

Data Binding

connect your models and your views

Data Binding

Angular gives you the ability to define the binding between the data in your scope and your views

- Most directives that are using expressions are creating a bidirectionnal data binding for you
- You can create manually new bindings with the directive ngModel

```
<div>
    <div ng-repeat="user in users">
        <h3>{{user.name}}</h3>
        <p>{{user.description}}</p>
        Edit Description:
        <textarea rows="5" cols="50" ng-model="user.description">
    </div>
</div>
```

The changes are visible in real-time in all the expressions

Filters

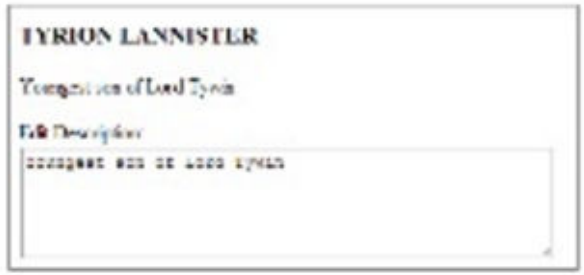
change the way your expressions are displayed

Filters - uppercase

Angular comes with a collection of filters that can change the way your data are displayed

- Usage: {{expression | filter}}

```
<div>
    <div ng-repeat="user in users">
        <h3>{{user.name | uppercase}}</h3>
        <p>{{user.description}}</p>
        Edit Description:
        <textarea rows="5" cols="50" ng-model="user.description">
    </div>
</div>
```



You can also easily create your own filters