ANGULARJS

HTML enhanced for web apps!

Prepared By

Ankita Mithaiwala

Computer Engineering Department,

SVNIT,surat

History

- AngularJS version 1.0 was released in 2012.
- Miško Hevery, a Google employee, started to work with AngularJS in 2009.
- The idea turned out very well, and the project is now officially supported by Google.

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.

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)

Key Features of ANGULARJS

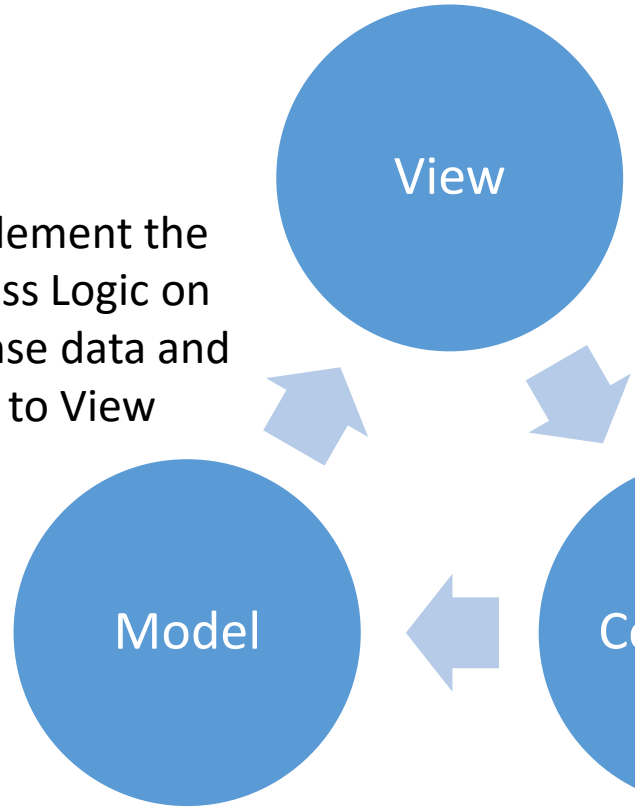
- Declarative HTML approach
- Easy Data Binding : Two way Data Binding
- Reusable Components
- MVC/MVVM Design Pattern
- Dependency Injection
- End to end Integration Testing / Unit Testing
- Routing
- Tempting

Continue..

- Modules
- Services
- Expressions
- Filters
- Directives
- Form Validation
- $scope, $http, $routeProvider…

MVC : Model View Controller

View :
- Renders the Model data
- Send User actions/events to controller
- UI



3. Implement the Business Logic on response data and Bind it to View

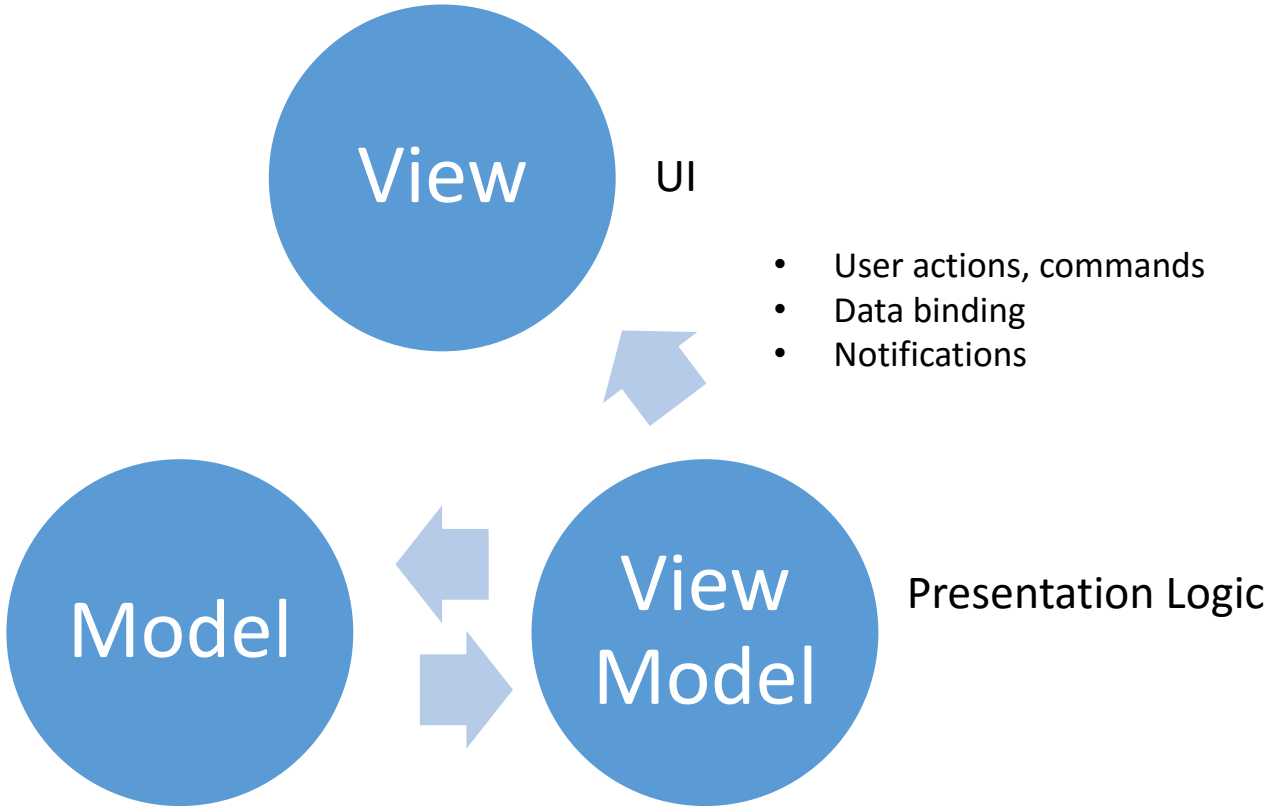
1. Event or User Action or View Load

Model:
- Business Logic
- Notify view changes
- Application Functionality
- Data in general

Controller:
- Define Application Behavior
- Maps user actions to Model
- Select view for response

2. Maps to particular Model after fetching the data

MVVM: Model View View Model



SPA

- **Single-Page Applications** (SPAs) are Web **apps** that load a**single** HTML **page** and dynamically update that **page** as the user interacts with the **app**. SPAs use AJAX and HTML5 to create fluid and responsive Web **apps**, without constant **page** reloads. However, this means much of the work happens on the client side, in JavaScript.
- Web browser JavaScript **frameworks**, such as AngularJS, Ember.js, Meteor.js, ExtJS and React have adopted **SPA** principles.
- AngularJS is a fully client-side **framework**. AngularJS's templating is based on bidirectional UI data binding.

- AngularJS is a JavaScript Framework
- AngularJS is a JavaScript framework. It is a library written in JavaScript.
- AngularJS is distributed as a JavaScript file, and can be added to a web page with a script tag:
- <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script>

- AngularJS Extends HTML
- AngularJS extends HTML with **ng-directives**.
- The **ng-app** directive defines an AngularJS application.
- The **ng-model** directive binds the value of HTML controls (input, select, textarea) to application data.
- The **ng-bind** directive binds application data to the HTML view.

Example:

- AngularJS starts automatically when the web page has loaded.
- The **ng-app** directive tells AngularJS that the element is the "owner" of an AngularJS **application**.
- The **ng-model** directive binds the value of the input field to the application variable **name**.
- The **ng-bind** directive binds the **innerHTML** of the <p> element to the application variable **name**.

AngularJS Directives

- AngularJS lets you extend HTML with new attributes called **Directives**.
- AngularJS has a set of built-in directives which offers functionality to your applications.
- AngularJS also lets you define your own directives.
- AngularJS directives are HTML attributes with an **ng** prefix.

AngularJS Directives

- AngularJS directives are extended HTML attributes with the prefix ng-.
- The ng-app directive initializes an AngularJS application.
- The ng-init directive initializes application data.
- The ng-model directive binds the value of HTML controls (input, select, text area) to application data.
- **https://www.w3schools.com/angular/angular_ref_directives.asp**

ng-app

- AngularJS is perfect for database CRUD (Create Read Update Delete) applications.
- The ng-app directive defines the **root element** of an AngularJS application.
- The ng-app directive will **auto-bootstrap** (automatically initialize) the application when a web page is loaded.

Example

- <!DOCTYPE html>
- <html>
- <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script>
- <body>
-
- The name is
-
- </body>
- </html>
- The ng-app directive also tells AngularJS that the element is the "owner" of the AngularJS application.

Data Binding (ng-model)

- The {{ firstName }} expression, in the example above, is an AngularJS data binding expression.
- Data binding in AngularJS binds AngularJS expressions with AngularJS data.
- {{ firstName }} is bound with ng-model="firstName".
- The ng-model directive binds the value of HTML controls (input, select, textarea) to application data.
- The ng-model directive can also:
- Provide type validation for application data (number, email, required).
- Provide status for application data (invalid, dirty, touched, error).
- Provide CSS classes for HTML elements.
- Bind HTML elements to HTML forms.
- Show Example4_data_bind.

Two-Way Binding

- The binding goes both ways. If the user changes the value inside the input field, the AngularJS property will also change its value:
- Data binding in AngularJS is the synchronization between the model and the view.
- When data in the *model* changes, the *view* reflects the change, and when data in the *view* changes, the *model* is updated as well. This happens immediately and automatically, which makes sure that the model and the view is updated at all times.
- Example 5

AngularJS Controller

- AngularJS controllers **control the data** of AngularJS applications.
- AngularJS controllers are regular **JavaScript Objects**.
- AngularJS applications are controlled by controllers.
- The **ng-controller** directive defines the application controller.
- A controller is a **JavaScript Object**, created by a standard JavaScript **object constructor**.
- Example 6

Example:

- The AngularJS application is defined by **ng-app="myApp"**. The application runs inside the .
- The **ng-controller="myCtrl"** attribute is an AngularJS directive. It defines a controller.
- The **myCtrl** function is a JavaScript function.
- AngularJS will invoke the controller with a **$scope** object.
- In AngularJS, $scope is the application object (the owner of application variables and functions).
- The controller creates two properties (variables) in the scope (**firstName** and **lastName**).
- The **ng-model** directives bind the input fields to the controller properties (firstName and lastName).

Controllers In External Files

- In larger applications, it is common to store controllers in external files.
- Example 7
- Example 8 for repeat and object value

AngularJS Scope

- The scope is the binding part between the HTML (view) and the JavaScript (controller).
- The scope is an object with the available properties and methods.
- The scope is available for both the view and the controller.
- How to Use the Scope?
- When you make a controller in AngularJS, you pass the $scope object as an argument.

If we consider an AngularJS application to consist of:

- View, which is the HTML.
- Model, which is the data available for the current view.
- Controller, which is the JavaScript function that makes/changes/removes/controls the data.
- Then the scope is the Model.
- The scope is a JavaScript object with properties and methods, which are available for both the view and the controller.

AngularJS Filters

- Filters can be added in AngularJS to format data.

AngularJS provides filters to transform data:

- currency Format a number to a currency format.
- date Format a date to a specified format.
- filter Select a subset of items from an array.
- json Format an object to a JSON string.
- limitTo Limits an array/string, into a specified number of elements/characters.
- lowercase Format a string to lower case.
- number Format a number to a string.
- orderBy Orders an array by an expression.
- uppercase Format a string to upper case.

Adding Filters to Expressions

- Filters can be added to expressions by using the pipe character |, followed by a filter.
- Example 9
- The lowercase filter format strings to lower case.

Adding Filters to Directives

- Filters are added to directives, like ng-repeat, by using the pipe character |, followed by a filter:
- The orderBy filter sorts an array.
- Example 10

Filter an Array Based on User Input

- By setting the ng-model directive on an input field, we can use the value of the input field as an expression in a filter.
- Example 11

AngularJS Services

- In AngularJS you can make your own service, or use one of the many built-in services.
- In AngularJS, a service is a function, or object, that is available for, and limited to, your AngularJS application.
- The **$location** service has methods which return information about the location of the current web page.
- Example 12

The $http Service

- The $http service is one of the most common used services in AngularJS applications. The service makes a request to the server, and lets your application handle the response.
- Use the $http service to request data from the server.
- Example 13

The $timeout Service

- The $timeout service is AngularJS' version of the window.setTimeout function.
- Example 14

Methods

- The example above uses the .get method of the $http service.
- The .get method is a shortcut method of the $http service. There are several shortcut methods:
- .delete()
- .get()
- .head()
- .jsonp()
- .patch()
- .post()
- .put() Example 15

AngularJS SQL

- AngularJS is perfect for displaying data from a Database. Just make sure the data is in JSON format.
- Fetching Data From a PHP Server Running MySQL.
- Example 16
- Server Code Examples
- The following section is a listing of the server code used to fetch SQL data.
- Using PHP and MySQL. Returning JSON.
- Using PHP and MS Access. Returning JSON.
- Using ASP.NET, VB, and MS Access. Returning JSON.
- Using ASP.NET, Razor, and SQL Lite. Returning JSON.

AngularJS Events

AngularJS has its own HTML events directives.You can add AngularJS event listeners to your HTML elements by using one or more of these directives:

- ng-blur
- ng-change
- ng-click
- ng-copy
- ng-cut
- ng-dblclick
- ng-focus
- ng-keydown
- ng-keypress
- ng-keyup
- ng-mousedown
- ng-mouseenter
- ng-mouseleave
- ng-mousemove
- ng-mouseover
- ng-mouseup
- ng-paste

The event directives allows us to run AngularJS functions at certain user events.
An AngularJS event will not overwrite an HTML event, both events will be executed

Mouse Events

- Mouse events occur when the cursor moves over an element, in this order:
- ng-mouseenter
- ng-mouseover
- ng-mousemove
- ng-mouseleave
- Or when a mouse button is clicked on an element, in this order:
- ng-mousedown
- ng-mouseup
- ng-click
- You can add mouse events on any HTML element. Example 17

AngularJS Animations

- AngularJS provides animated transitions, with help from CSS.
- An animation is when the transformation of an HTML element gives you an illusion of motion.
- Example:
- Check the checkbox to hide the DIV:
- Example 18

AngularJS Expressions

- AngularJS expressions are written inside double braces: **{{ expression }}**.
- AngularJS binds data to HTML using **Expressions**.
- AngularJS will "output" data exactly where the expression is written:
- Show Example.
- AngularJS expressions bind AngularJS data to HTML the same way as the **ng-bind** directive.
- Example 3.

Continue..

- AngularJS expressions can be written inside double braces: {{ *expression* }}.
- AngularJS expressions can also be written inside a directive: ng-bind="*expression*".
- AngularJS will resolve the expression, and return the result exactly where the expression is written.
- **AngularJS expressions** are much like **JavaScript expressions:** They can contain literals, operators, and variables.
- Example {{ 5 + 5 }} or {{ firstName + " " + lastName }}
- If you remove the ng-app directive, HTML will display the expression as it is, without solving it:

Example without ng-app

```
<!DOCTYPE html>
<html>
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script>
<body>
<p>Without the ng-app directive, HTML will display the expression as it is, without solving it.</p>
<div>
<p>My first expression: {{ 5 + 5 }}</p>
</div>

</body>
</html>
```

Output:

Without the ng-app directive, HTML will display the expression as it is, without solving it.

My first expression: {{ 5 + 5 }}

AngularJS Numbers

- AngularJS numbers are like JavaScript numbers:
- <!DOCTYPE html>
- <html>
- <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script>
- <body>
-
- Total in dollar: {{ quantity * cost }}
-
- </body>
- </html>

AngularJS Expressions vs. JavaScript Expressions

- Like JavaScript expressions, AngularJS expressions can contain literals, operators, and variables.
- Unlike JavaScript expressions, AngularJS expressions can be written inside HTML.
- AngularJS expressions do not support conditionals, loops, and exceptions, while JavaScript expressions do.
- AngularJS expressions support filters, while JavaScript expressions do not.