UNIT 4

Angular JS


ANGULARJS
by Google

What is AngularJS

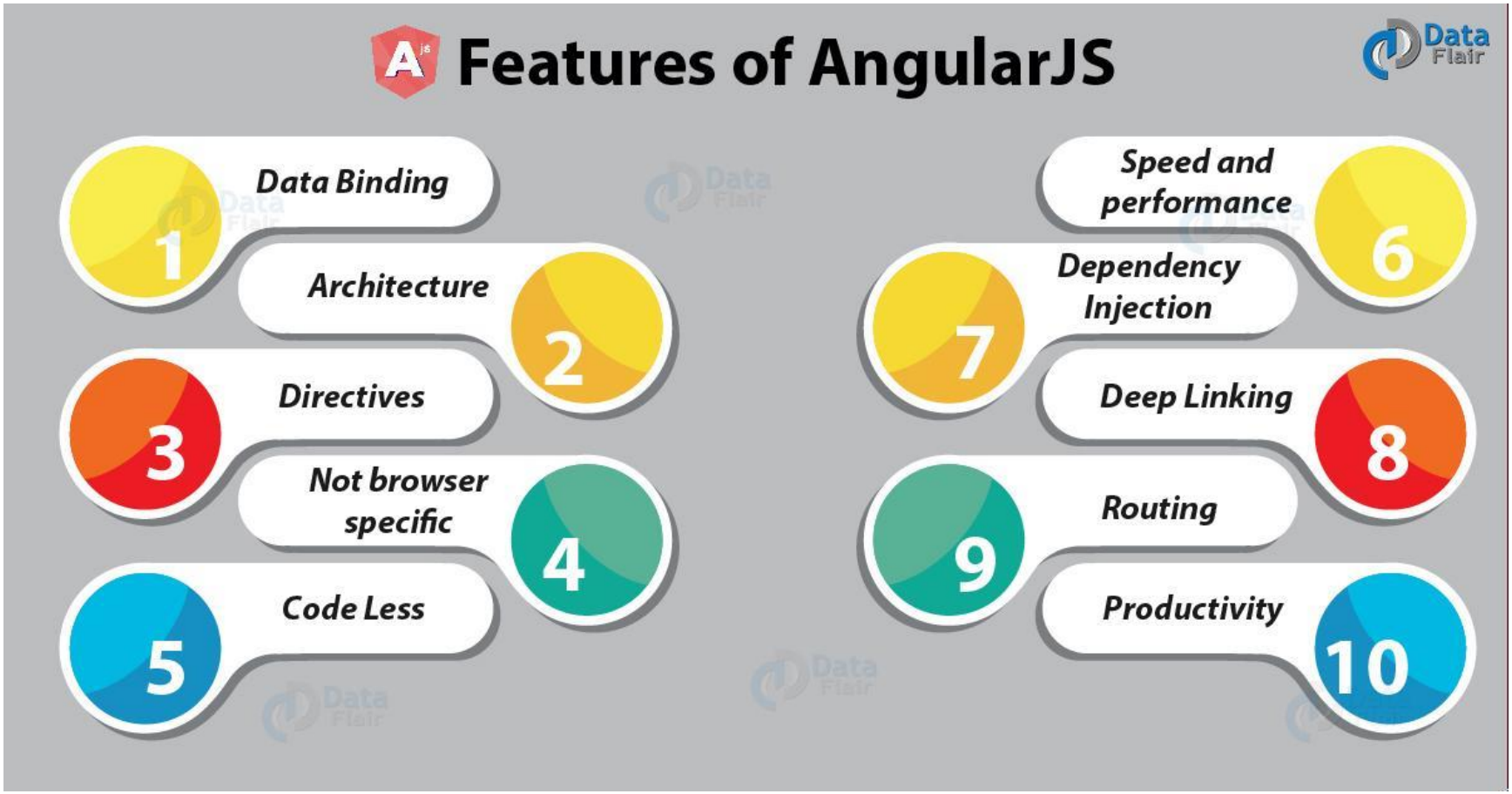
- AngularJS is a **JavaScript framework** written in JavaScript.
- It is a open-source front-end web framework for developing single-page applications.
- It can be added to an HTML page with a <script> tag.
- AngularJS extends HTML attributes with **Directives**, and binds data to HTML with **Expressions**.
- It is based on MVC Javascript Framework by Google for Rich Web Application Development

Why AngularJS

- Structure, Quality and Organization
- Lightweight (< 36KB compressed and minified)
- Free
- Separation of concern
- Modularity
- Extensibility & Maintainability
- Reusable Components

Features of AngularJS

- Two-way Data Binding – Model as single source of truth
- Directives – Extend HTML
- MVC
- Dependency Injection
- Testing
- Deep Linking (Map URL to route Definition)
- Server-Side Communication


Features of AngularJS
Data Flair
1
Data Binding
2
Architecture
3
Directives
4
Not browser specific
5
Code Less
6
Speed and performance
7
Dependency Injection
8
Deep Linking
9
Routing
10
Productivity

General features

- AngularJS is a efficient framework that can create Rich Internet Applications (RIA).
- AngularJS provides developers an options to write client side applications using JavaScript in a clean Model View Controller (MVC) way.
- Applications written in AngularJS are cross-browser compliant.
- AngularJS automatically handles JavaScript code suitable for each browser.
- AngularJS is open source, completely free, and used by thousands of developers around the world. It is licensed under the Apache license version 2.0.
- AngularJS is a framework to build large scale, high-performance, and easyto-maintain web applications.

Core Features

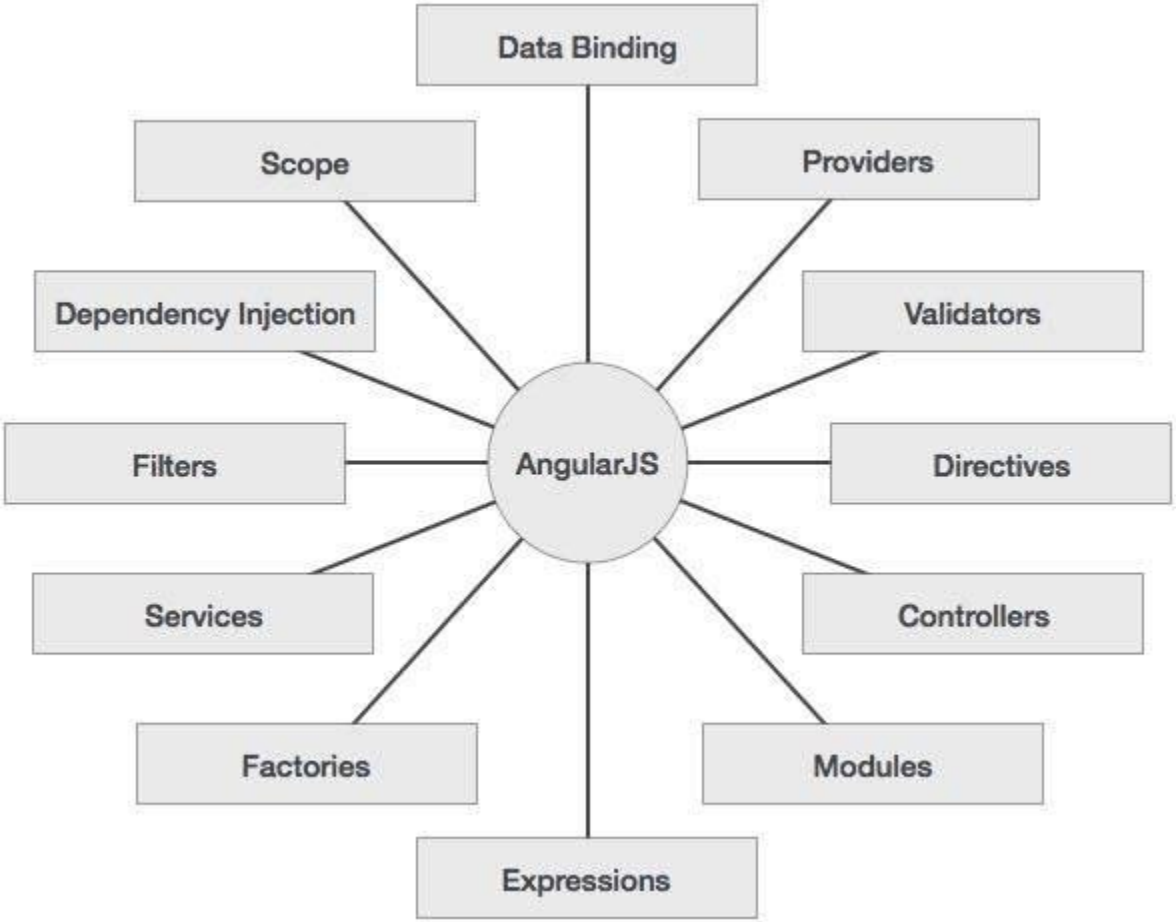
The core features of AngularJS are as follows −

- **Data-binding** − It is the automatic synchronization of data between model and view components.
- **Scope** − These are objects that refer to the model. They act as a glue between controller and view.
- **Controller** − These are JavaScript functions bound to a particular scope.
- **Services** − AngularJS comes with several built-in services such as $http to make a XMLHttpRequests. These are singleton objects which are instantiated only once in app.
- **Filters** − These select a subset of items from an array and returns a new array.
- **Directives** − Directives are markers on DOM elements such as elements, attributes, css, and more. These can be used to create custom HTML tags that serve as new, custom widgets. AngularJS has built-in directives such as ngBind, ngModel, etc.

- **Templates** − These are the rendered view with information from the controller and model. These can be a single file (such as index.html) or multiple views in one page using *partials*.
- **Routing** − It is concept of switching views.
- **Model View Whatever** − MVW is a design pattern for dividing an application into different parts called Model, View, and Controller, each with distinct responsibilities. AngularJS does not implement MVC in the traditional sense, but rather something closer to MVVM (Model-View-ViewModel). The Angular JS team refers it humorously as Model View Whatever.
- **Deep Linking** − Deep linking allows to encode the state of application in the URL so that it can be bookmarked. The application can then be restored from the URL to the same state.
- **Dependency Injection** − AngularJS has a built-in dependency injection subsystem that helps the developer to create, understand, and test the applications easily.

Concepts

Important parts of AngularJS



Advantages of AngularJS

- The It provides the capability to create Single Page Application in a very clean and maintainable way.
- It provides data binding capability to HTML. Thus, it gives user a rich and responsive experience.
- AngularJS code is unit testable.
- AngularJS uses dependency injection and make use of separation of concerns.

- AngularJS provides reusable components.
- With AngularJS, the developers can achieve more functionality with short code.
- In AngularJS, views are pure html pages, and controllers written in JavaScript do the business processing.
- AngularJS applications can run on all major browsers and smart phones, including Android and iOS based phones/tablets.

Disadvantages of AngularJS

- **Not Secure** – Being JavaScript only framework, application written in AngularJS are not safe. Server side authentication and authorization is must to keep an application secure.
- **Not degradable** – If the user of your application disables JavaScript, then nothing would be visible, except the basic page.

Data Flair

Angular

Advanatges & Disadvantages

Advanatges

- Open source
- Easy to extend
- Easy to test
- Great MVC
- Google supported
- No Pre-requisite knowledge
- Easy to customize
- Single page application (SPA)

Disadvantages

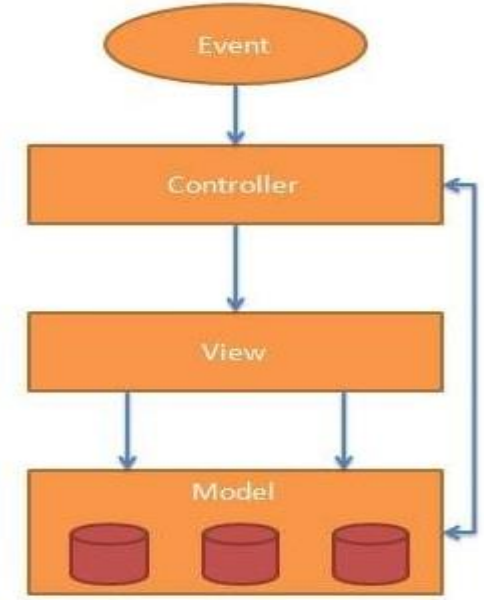
- Less Secure
- Memory Leakage
- No specific way
- Not Supported Everywhere
- Java Script Based

Model View Controller(MVC)

- MVC stands for Model View Controller.
- It is a software design pattern for developing web applications.
- It is very popular because it isolates the application logic from the user interface layer and supports separation of concerns.

Model

- It is responsible for managing application data.
- It responds to the requests from view and to the instructions from controller to update itself.



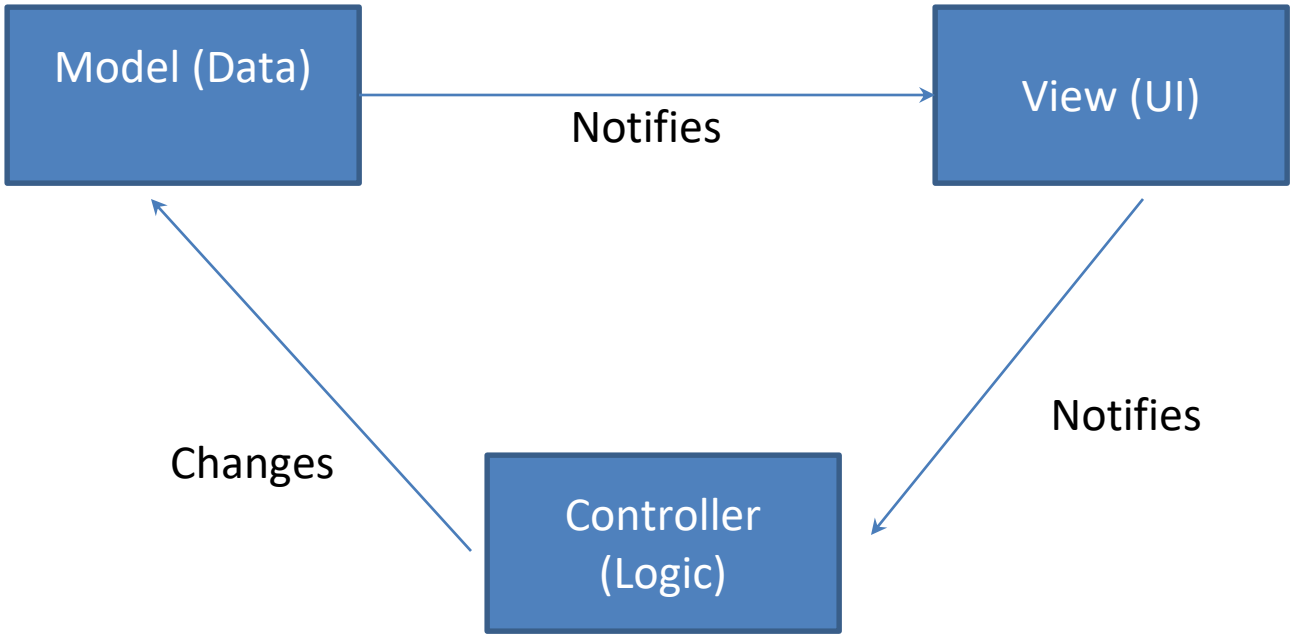
View

- It is responsible for displaying all data or only a portion of data to the users.
- It also specifies the data in a particular format triggered by the controller's decision to present the data.
- They are script-based template systems such as JSP, ASP, PHP and very easy to integrate with AJAX technology.

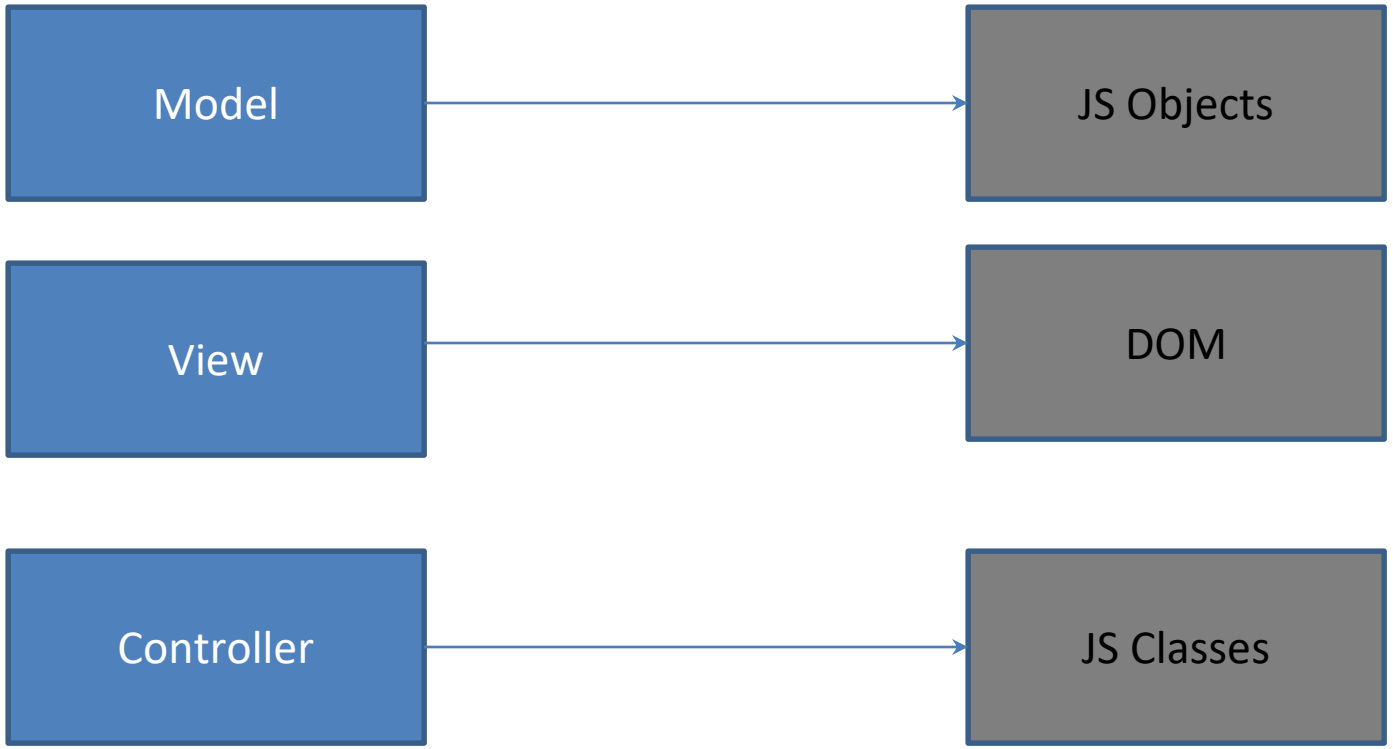
Controller

- It is responsible to control the relation between models and views.
- It responds to user input and performs interactions on the data model objects.
- The controller receives input, validates it, and then performs business operations that modify the state of the data model.

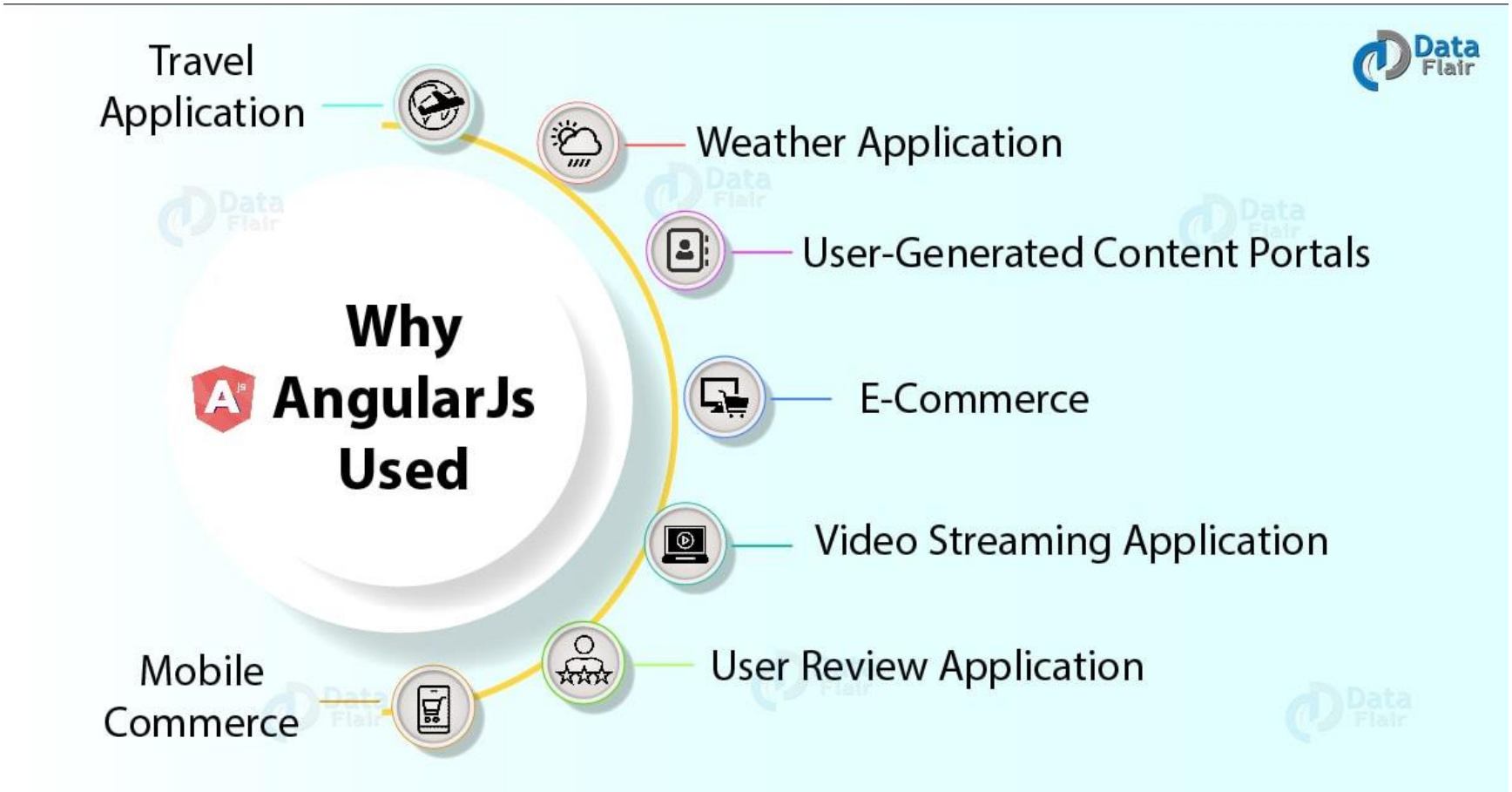
MVC


Model (Data)
Notifies
View (UI)
Changes
Controller
(Logic)
Notifies

MVC



Applications of AngularJS



AngularJS - Environment Setup

- To set up AngularJS library to be used in web application development
- open the link https://angularjs.org/, you will see there are two options to download AngularJS library –


ANGULARJS
Home
Learn
Develop
Discuss
Search
ANGULARJS
by Google
HTML enhanced for web apps!
Download AngularJS 1
(1.5.2 / 1.2.29)
Try the new Angular 2
View on GitHub
Design Docs & Notes


Download AngularJS
Branch
1.5.x (stable)
1.2.x (legacy)
Build
Minified
Uncompressed
Zip
CDN
https://ajax.googleapis.com/ajax/libs/angularjs/1.5.2/angular.min.js
Bower
bower install angular#1.5.2
npm
npm install angular@1.5.2
Extras
Browse additional modules
Previous Versions
Download

- AngularJS is distributed as a JavaScript file, and can be added to a web page with a script tag.
- Use the CDN link or the local file

```
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script>
```

or

```
<script src="angular.min.js"></script>
```

AngularJS - First Application to display Hello message

- **Step 1**: Create a New Folder
- **Step 2**: Open Notepad

  Step 3 : Write the code using notepad
- **Step 4**: Save with a name test.html
- **Step 5:** Open the location you saved the file. Right-click and open the file using the browser.

First Program

```
<!doctype html>
<html ng-app>
  <head>
    <title>ISE</title>
    <script src="angular.min.js"></script>
  </head>
  <body>
My First Program in AngularJS
<h1>{{ 10 + 2 }}</h1>
  </body>
</html>
```

AngularJS Directives

- The AngularJS framework can be divided into three major parts –
- **ng-app** – This directive defines and links an AngularJS application to HTML.
- **ng-model** – This directive binds the values of AngularJS application data to HTML input controls.
- **ng-bind** – This directive binds the AngularJS application data to HTML tags.

- Step 1: Load framework
- Being a pure JavaScript framework, it can be added using <Script> tag.

```
<script src =
"https://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/angular.min.js"
> </script>
```

- Step 2: Define AngularJS application using ng-app directive

```
<div ng-app = ""> ... </div>
```

- Step 3: Define a model name using ng-model directive

```
<p>Enter your Name: <input type = "text" ng-model = "name"></p>
```

- Step 4: Bind the value of above model defined using ng-bind directive

```
<p>Hello <span ng-bind = "name"></span>!</p>
```

```
<html>
  <head>
    <title>AngularJS First Application</title>
  </head>

  <body>
    <h1>Sample Application</h1>

    <div ng-app = "">
      <p>Enter your Name: <input type = "text" ng-model = "name"></p>
      <p>Hello <span ng-bind = "name"></span>!</p>
    </div>

    <script src =
"https://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/angular.min.js">
    </script>

  </body>
</html>
```

How AngularJS integrates with HTML

- The ng-app directive indicates the start of AngularJS application.
- The ng-model directive creates a model variable named name, which can be used with the HTML page and within the div having ng-app directive.
- The ng-bind then uses the name model to be displayed in the HTML tag whenever user enters input in the text box.
- Closing tag indicates the end of AngularJS application


Data Flair
AngularJS
Directives
Syntax
Example
Various Built-in Directives
Types

AngularJS Directives

APPLICATION

ng-app
ng-controller

FORMS

ng-pattern
ng-minlength
ng-maxlength
ng-required
ng-list
ng-true-value
ng-false-value
ng-options
ng-submit

TEMPLATE

ng-csp
ng-disabled
ng-hide | show
ng-if
ng-mouse
ng-repeat
ng-switch
ng-transclude
ng-view
ng-include

OPERATION

ng-change
ng-checked
ng-click
ng-href
ng-selected

BINDING

ng-bind
ng-model
ng-init
ng-src
ng-style

- AngularJS directives are used to extend HTML. They are special attributes starting with **ng-** prefix.

Let us discuss the following directives –

- **ng-app** – This directive starts an AngularJS Application.
- **ng-init** – This directive initializes application data.
- **ng-model** – This directive defines the model that is variable to be used in AngularJS.
- **ng-repeat** – This directive repeats HTML elements for each item in a collection.

ng-app directive

- The ng-app directive starts an AngularJS Application.
- It defines the root element.
- It automatically initializes or bootstraps the application when the web page containing AngularJS Application is loaded.
- It is also used to load various AngularJS modules in AngularJS Application.
- In the following example, we define a default AngularJS application using ng-app attribute of a element.

```
<div ng-app = "">
   ...
</div>
```

ng-init directive

- The ng-init directive initializes an AngularJS Application data.
- It is used to assign values to the variables.
- In the following example, we initialize an array of countries. We use JSON syntax to define the array of countries.

```
<div ng-app = "" ng-init = "countries = [{locale:'en-US',name:'United States'},
  {locale:'en-GB',name:'United Kingdom'}, {locale:'en-FR',name:'France'}]">
  ...
</div>
```

ng-model directive

The ng-model directive defines the model/variable to be used in AngularJS Application. In the following example, we define a model named name.

```
<div ng-app = "">
   ...
   <p>Enter your Name: <input type = "text" ng-model = "name"></p>
</div>
```

ng-repeat directive

The ng-repeat directive repeats HTML elements for each item in a collection. In the following example, we iterate over the array of countries.

```
<div ng-app = "">
   ...
   <p>List of Countries with locale:</p>
      <ol>
      <li ng-repeat = "country in countries">
         {{ 'Country: ' + country.name + ', Locale: ' + country.locale }}
      </li>
   </ol>
</div>
```

- Program to demonstrate the usage of directives

AngularJS - Expressions

- Expressions are used to bind application data to HTML.
- Expressions are written inside double curly braces such as in {{ expression}}.
- Expressions behave similar to ng-bind directives.
- AngularJS expressions are pure JavaScript expressions and output the data where they are used.

- Using numbers

```
<p>Expense on Books : {{cost * quantity}} Rs</p>
```

- Using Strings

```
<p>Hello {{student.firstname + " " + student.lastname}}!</p>
```

- Using Object

```
<p>Roll No: {{student.rollno}}</p>
```

- Using Array

```
<p>Marks(Math): {{marks[3]}}</p>
```

- Program to demonstrate the usage of directives Expressions
- **e**xpressions.html

Ex.: Product of Two Numbers in AngularJS

```
<div ng-app="">
   <p>First Number:
      <input type="text" ng-model="a" />
   </p>
   <p>Second Number:
      <input type="text" ng-model="b" />
   </p>
   <p>Product: {{ a * b }}</p>
```

- The `ng-model` directive binds the value of the input field to the application variable name.
- So, the variable `"a"` will contain first `TextBox` Value and `"b"` will contain second `TextBox` value.
- **{{ expression }}** is used to dynamically bind the result of the expression.
- Thus, while typing on the `TextBoxes`, this place will populate with the result of the expression, which is our expected summation.

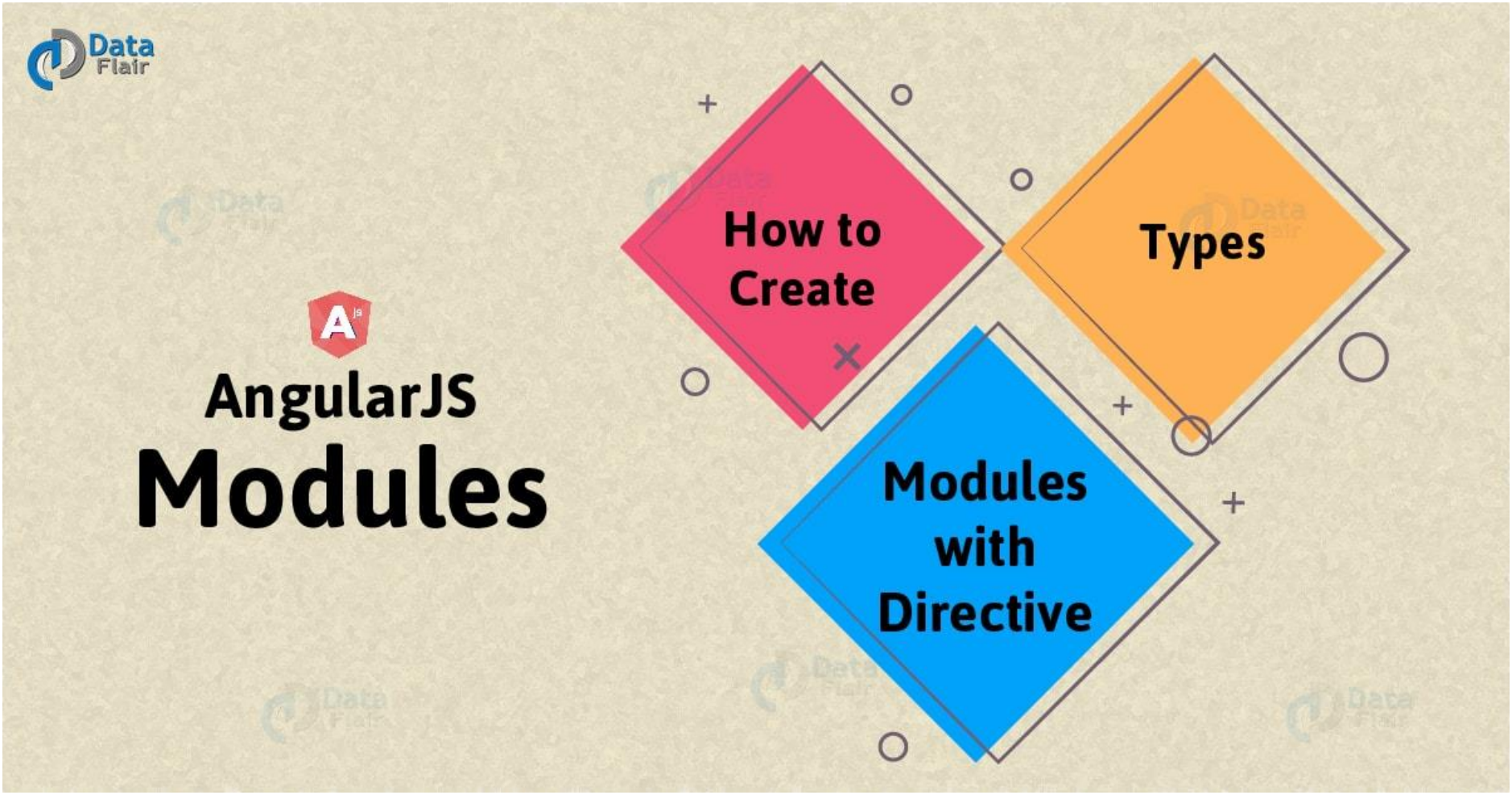
Ex.: Addition of Two Numbers in AngularJS

```
<div ng-app="">
    <p>First Number:
        <input type="text" ng-model="a" />
    </p>
    <p>Second Number:
        <input type="text" ng-model="b" />
    </p>
    <p>Sum: {{ a -- b }}</p>
```

Ex.: Addition of Two Numbers in AngularJS

```
<div ng-app="">
  <p>First Number:
    <input type="number" ng-model="a" />
  </p>
  <p>Second Number:
    <input type="number" ng-model="b" />
  </p>
  <p>Sum: {{ a + b }}</p>
```

AngularJS
Modules and Controllers


Data Flair
AngularJS
Modules
How to Create
Types
Modules with Directive

What is a Module in AngularJS?

- A module in AngularJS is a container that holds various parts of an AngularJS application such as controller, directives, services and filters.
- It serves as the entry point to the application.
- A module specifies how the AngularJS starts.
- AngularJS supports modular approach. Modules are used to separate logic such as services, controllers, application etc. from the code and maintain the code clean.

- What is a module and how to create a Module ?
- What is a Controller and how to create a controller ?
- How to register a controller with the module ?

how to create a Module ?

- Use angular objects module () method
- Application Module - used to initialize an application with controller(s).

```
var myApp= angular.module("MyModule", []);
```

In creating a module, specify two parameters:

- the name of the module
- array of dependencies the module depends on.

What is a Controller in AngularJS?

- Controller Module - used to define the controller.
- A controller in **AngularJS is function** that controls the communication between the application logic and the view.
- For example, a controller would manage retrieving data from the database and displaying it in a html page.
- AngularJS application mainly relies on controllers to control the flow of data in the application.
- A controller is defined using *ng-controller* directive.
- A controller is a JavaScript object that contains attributes/properties, and functions.
- Each controller accepts $scope as a parameter, which refers to the application/module that the controller needs to handle.

How to Create a Controller

- To create a controller, simply create a function and assign it to a variable as shown below

```
var myController = function($scope) {
$scope. message = " Happy New Year";
}
```

- In this controller, we pass a parameter called **$scope** to the controller function.
- **$scope** is used to make variables available in the view.
- So any variable attached to the $scope is accessible from the view using the binding expression **{{ }}.**

How Register a Controller with a Module

- The next step after creating a module and a controller is to register the controller with a module.

  To register a controller with a module, use the syntax below:

-

```
myApp.controller("MyController",
MyController);
```

OR

```
myApp.controller("myController", function
($scope) {
$scope.message=" Happy New Year ";
});
```

- Now That we have created a module and a controller, and registered a controller with a module, we can now move to the next step and that is

- Program to demonstrate the usage of directives **Controllers**
- **5pgm1.html**
- **6pgm2.html**

FILTERS

- Filters are used to modify the data.
- They can be clubbed in expression or directives using pipe (|) character.
- The following list shows commonly used filters.

S.No.	Name	Description
1	uppercase	converts a text to upper case text.
2	lowercase	converts a text to lower case text.
3	currency	formats text in a currency format.
4	filter	filter the array to a subset of it based on provided criteria.
5	orderby	orders the array based on provided criteria.

Uppercase Filter

- This adds uppercase filter to an expression using pipe character. Here, we add uppercase filter to print student name in capital letters.

```
Enter first name:<input type="text" ng-model="student.firstName">

Enter last name: <input type="text" ng-model="student.lastName">

Name in Upper Case: {{student.fullName() | uppercase}}
```

Lowercase Filter

- This adds lowercase filter to an expression using pipe character. Here, we add lowercase filter to print student name in small letters.

```
Enter first name:<input type="text" ng-model="student.firstName">

Enter last name: <input type="text" ng-model="student.lastName">

Name in Lower Case: {{student.fullName() | lowercase}}
```

Currency Filter

- This adds currency filter to an expression that returns a number. Here, we add currency filter to print fees using currency format.

```
Enter fees: <input type="text" ng-model="student.fees">

fees: {{student.fees | currency}}
```

Filter Filter

- To display only required subjects, we use subjectName as filter.

```
Enter subject: <input type="text" ng-model="subjectName">

Subject:

<ul>

  <li ng-repeat="subject in student.subjects | filter: subjectName">

    {{ subject.name + ', marks:' + subject.marks }}

  </li>

</ul>
```

Orderby Filter

- To order subjects by marks, we use orderBy marks.

```
Subject:
<ul>
  <li ng-repeat="subject in student.subjects | orderBy:'marks'">
    {{ subject.name + ', marks:' + subject.marks }}
  </li>
</ul>
```

- Program to demonstrate the usage of directives **Filters**
  - **7filters.html**

TABLES

- Table data is generally repeatable. The ng-repeat directive can be used to draw table easily. The following example shows the use of ng-repeat directive to draw a table:

```
<table>
   <tr>
      <th>Name</th>
      <th>Marks</th>
   </tr>
   <tr ng-repeat="subject in student.subjects">
      <td>{{ subject.name }}</td>
      <td>{{ subject.marks }}</td>
   </tr>
</table>
```

- Program to demonstrate the usage of directives **for Tables**
  - **8tables.html**

HTML DOM

- The following directives are used to bind application data to attributes of HTML DOM elements:

S.No.	Name	Description
1	ng-disabled	Disables a given control.
2	ng-show	Shows a given control.
3	ng-hide	Hides a given control.
4	ng-click	Represents a AngularJS click event.

ng-disabled Directive

- Add ng-disabled attribute to an HTML button and pass it a model. Bind the model to a checkbox and see the variation.

```
<input type="checkbox" ng-model="enableDisableButton">Disable Button

<button ng-disabled="enableDisableButton">Click Me!</button>
```

ng-show Directive

Add ng-show attribute to an HTML button and pass it a model. Bind the model to a checkbox and see the variation.

```
<input type="checkbox" ng-model="showHide1">Show Button

<button ng-show="showHide1">Click Me!</button>
```

ng-hide Directive

- Add ng-hide attribute to an HTML button and pass it a model. Bind the model to a checkbox and see the variation.

```
<input type="checkbox" ng-model="showHide2">Hide Button
```

```
<button ng-hide="showHide2">Click Me!</button>
```

ng-click Directive

```
<p>Total click: {{ clickCounter }}</p></td>

<button ng-click="clickCounter = clickCounter + 1">Click Me!</button>
```

- Program to demonstrate the usage of directives
- 9htmldom_ex.html

FORMS

- AngularJS enriches form filling and validation.
- We can use ng-click event to handle the click button and use $dirty and $invalid flags to do the validation in a seamless way.
- Use novalidate with a form declaration to disable any browser-specific validation.
- The form controls make heavy use of AngularJS events.

Forms

- forms in AngularJS provides data-binding and validation of input controls.
- Input Controls :
- Input controls are the HTML input elements:
  - input elements
  - select elements
  - button elements
  - textarea elements

Events

- AngularJS provides multiple events associated with the HTML controls.
- For example, ng-click directive is generally associated with a button.
- AngularJS supports the following events:
  - ng-click
  - ng-dbl-click
  - ng-mousedown
  - ng-mouseup
  - ng-mouseenter
  - ng-mouseleave
  - ng-mousemove
  - ng-mouseover
  - ng-keydown
  - ng-keyup
  - ng-keypress
  - ng-change

- Program to demonstrate the usage of events

forms.html

Ex. TODO List

- After loading angular.min.js we create an Angular JS module called todoApp and a controller called todoController. Inside the controller we set up an empty array called tasks that will hold the todo list. We make it an attribute of the current $scope in order to make it accessible form the HTML.
- We also declare a function called add (also an attribute of the $scope) that takes the value of title (we'll later see this is the name of the input box), and appends it to the list of tasks using push. That's all the JavaScript code we need for a simple TODO list.
- In the HTML part we have a div element that defined the area of the AngularJS Application ng-app and the Angular JS controller ng-controller.
- Inside the controller in the HTML we have two parts. The first part is an input element connected to the $scope.title attribute using ng-model and a button that uses ng-click to launch the $scope.add method when the button is clicked.
- The second part uses the ng-repeat directive to iterate over the elements of the $scope.tasks array and display them one-by-one as list items.

Advanced AngularJS Concept

- Dependency Injection
- Modularity
- Digesting
- Scope
- Handling SEO
- End to End Testing
- Promises
- Localization

Useful Links

- https://angularjs.org/
- http://campus.codeschool.com/courses/shaping-up-with-angular-js/contents
- http://www.toptal.com/angular-js/a-step-by-step-guide-to-your-first-angularjs-app
- https://github.com/raonibr/f1feeder-part1
- https://data-flair.training/blogs/angularjs-controller/
- https://www.tutorialspoint.com/angularjs/angularjs_login_application.htm