# Introduction to AngularJS

The **ng-app** directive defines the application, the **ng-controller** directive defines the controller.

<div ng-app="**myApp**" ng-controller="**myCtrl**">

AngularJS **controllers** control AngularJS applications.

The **ng-model** directive binds the value of HTML controls (input, select, textarea) to application data.  
  <p>Name: <input type="text" ng-model="name"></p>

The **ng-bind** directive binds application data to the HTML view.  
  <p ng-bind="name"></p>

## Understanding the Scope

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 Modules and Controllers

### AngularJS module

An AngularJS module defines an application. The module is a container for the different parts of an application. The module is a container for the application controllers. Controllers always belong to a module.

A module is created by using the AngularJS function angular.module

<div ng-app="myApp">...</div>  
  
<script>  
**The "myApp" parameter refers to an HTML element in which the application will run.**  
var app = angular.module("myApp", []);   
  
</script>

### AngularJS Controller

AngularJS controllers **control the data** of AngularJS applications.

AngularJS controllers are regular **JavaScript Objects**.

<div ng-app="myApp" ng-controller="myCtrl">  
  
First Name: <input type="text" ng-model="firstName"><br>  
Last Name: <input type="text" ng-model="lastName"><br>  
<br>  
Full Name: {{firstName + " " + lastName}}  
<br>  
Full Name: {{fullName()}}  
</div>  
  
<script>  
var app = angular.module('myApp', []);  
app.controller('myCtrl', function($scope) {  
    $scope.firstName = "John";  
    $scope.lastName = "Doe";

This is how you define a function

    $scope.fullName = function() {  
        return $scope.firstName + " " + $scope.lastName;  
    };  
});

The AngularJS application is defined by  **ng-app="myApp"**. The application runs inside the <div>.

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

<div ng-app="myApp" ng-controller="personCtrl">  
  
First Name: <input type="text" ng-model="firstName"><br>  
Last Name: <input type="text" ng-model="lastName"><br>  
<br>  
Full Name: {{fullName()}}  
  
</div>  
  
<script src="personController.js"></script>

## AngularJS expressions

**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.

<p>My first expression: {{ 5 + 5 }}</p>

</div>

<script>

**AngularJS modules define AngularJS applications.**  
var app = angular.module('**myApp**', []);

**AngularJS controllers control applications:**  
app.controller('**myCtrl**', function($scope) {

    $scope.firstName= "John";  
    $scope.lastName= "Doe";

});  
</script>

### 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.

## AngularJS numbers and strings

AngularJS numbers and strings are like JavaScript numbers:

<div ng-app="" ng-init="quantity=1;cost=5">  
  
<p>Total in dollar: {{ quantity \* cost }}</p>  
  
</div>

<div ng-app="" ng-init="firstName='John';lastName='Doe'">  
  
<p>The name is {{ firstName + " " + lastName }}</p>  
  
</div>

Same example using ng-bind:

<div ng-app="" ng-init="quantity=1;cost=5">  
  
<p>Total in dollar: <span ng-bind="quantity \* cost"></span></p>  
  
</div>

<div ng-app="" ng-init="firstName='John';lastName='Doe'">  
  
<p>The name is <span ng-bind="firstName + ' ' + lastName"></span></p>  
  
</div>

## AngularJS Objects

AngularJS objects are like JavaScript object

<div ng-app="" ng-init="person={firstName:'John',lastName:'Doe'}">  
  
<p>The name is {{ person.lastName }}</p>  
  
</div>

Same example using ng-bind:

<div ng-app="" ng-init="person={firstName:'John',lastName:'Doe'}">  
  
<p>The name is <span ng-bind="person.lastName"></span></p>  
  
</div>

## AngularJS Arrays

<div ng-app="" ng-init="points=[1,15,19,2,40]">  
  
<p>The third result is {{ points[2] }}</p>  
  
</div>

Same example using ng-bind:

<div ng-app="" ng-init="points=[1,15,19,2,40]">  
  
<p>The third result is <span ng-bind="points[2]"></span></p>  
  
</div>