**AngularJS Intro**

AngularJS Directives

* AngularJS is a JavaScript framework. It can be added to an HTML page with a <script> tag.
* AngularJS extends HTML attributes ( <h1>, <a>, … ) with Directives (Dastorol amal) , and binds data to HTML with Expressions. (ebarat ha).
* AngularJS extends HTML with ng-directives:
* The ng-app directive defines an AngularJS application.
* The ng-model directive binds (vasl kardan) the value of HTML controls (input, select, textarea) to application data.
* **The ng-bind** directive binds (vasl kardan) application data to the HTML view.
* The **ng-init**directive initializes AngularJS application variables.
* As you have already seen, AngularJS directives are HTML attributes with an ng prefix.

## AngularJS Expressions

* AngularJS expressions are written inside double braces: {{ expression }}.
* AngularJS will "output" data exactly where the expression is written.

AngularJS Applications

* AngularJS **modules** define AngularJS applications.
* AngularJS **controllers** control AngularJS applications.
* The **ng-app** directive defines the application, the **ng-controller** directive defines the controller.

Read more about this at:

<https://www.w3schools.com/angular/angular_intro.asp>

**AngularJS Expressions**

AngularJS binds data to HTML using Expressions.

AngularJS Expressions

* There are two ways to write expressions in AngualrJS:
  + AngularJS expressions can be written inside double braces: {{ expression }}.
  + AngularJS expressions can also be written inside a directive:

ng-bind="expression".

* **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. (very important)
* Some examples: ng-init="myCol=' lightblue' “ , ng-init="quantity=1; cost=5" .

AngularJS Objects, Arrays

See example: <https://www.w3schools.com/angular/angular_expressions.asp>

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 Modules**

* 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.

Creating a Module

* A module is created by using the AngularJS function angular.module
* The "myApp" parameter refers to an HTML element in which the application will run.
* Now you can add controllers, directives, filters, and more, to your AngularJS application.

Adding a Controller

* Add a controller to your application, and refer to the controller with the

ng-controller directive:

Adding a Directive

* AngularJS has a set of built-in directives which you can use to add functionality to your application.
* For a full reference, visit our [AngularJS directive reference](https://www.w3schools.com/angular/angular_ref_directives.asp).

Modules and Controllers in Files

* It is common in AngularJS applications to put the module and the controllers in JavaScript files.
* In this example, "myApp.js" contains an application module definition, while "myCtrl.js" contains the controller:
* Global functions should be avoided in JavaScript. They can easily be overwritten or destroyed by other scripts.
* AngularJS modules reduces this problem, by keeping all functions local to the module.

When to Load the Library

* While it is common in HTML applications to place scripts at the end of the <body> element, it is recommended that you load the AngularJS library either in the <head> or at the start of the <body>.
* This is because calls to angular.module can only be compiled after the library has been loaded.

**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 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, textarea) to application data.

Repeating HTML Elements

* The ng-repeat directive repeats an HTML element.
* The ng-repeat directive used on an array of objects:

The ng-model Directive

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.

## Create New Directives

* In addition to all the built-in AngularJS directives, you can create your own directives.
* New directives are created by using the .directive function.
* To invoke the new directive, make an HTML element with the same tag name as the new directive.
* When naming a directive, you must use a camel case name, w3TestDirective, but when invoking it, you must use - separated name, w3-test-directive:
* You can invoke a directive by using:
  + Element name
  + Attribute
  + Class
  + Comment

**AngularJS ng-model Directive**