AngularJS Modules

[« Previous](http://www.w3schools.com/angular/angular_expressions.asp)

[Next Chapter »](http://www.w3schools.com/angular/angular_directives.asp)

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

<div ng-app="myApp">...</div>  
  
<script>

var app = angular.module(**"**myApp**"**, []); 

</script>

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:

Example

<div ng-app="**myApp**" ng-controller=**"myCtrl"**>  
{{ firstName + " " + lastName }}  
</div>  
  
<script>

var app = angular.module(**"myApp"**, []);  
  
app.controller(**"myCtrl"**, function($scope) {  
    $scope.firstName = "John";  
    $scope.lastName = "Doe";  
});

</script>

[Try it Yourself »](http://www.w3schools.com/angular/tryit.asp?filename=try_ng_module)

You will learn more about controllers later in this tutorial.

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](http://www.w3schools.com/angular/angular_ref_directives.asp).

In addition you can use the module to add your own directives to your applications:

Example

<div ng-app="myApp" w3-test-directive></div>  
  
<script>

var app = angular.module("myApp", []);  
  
app.directive("w3TestDirective", function() {  
    return {  
        template : "I was made in a directive constructor!"  
    };  
});

</script>

[Try it Yourself »](http://www.w3schools.com/angular/tryit.asp?filename=try_ng_directive)

You will learn more about directives later in this tutorial.

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:

Example

<!DOCTYPE html>  
<html>  
<script src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script>  
<body>  
  
<div ng-app="**myApp**" ng-controller="**myCtrl**">  
{{ firstName + " " + lastName }}  
</div>  
  
<script src="**myApp.js**"></script>  
<script src="**myCtrl.js**"></script>  
  
</body>  
</html>

[Try it Yourself »](http://www.w3schools.com/angular/tryit.asp?filename=try_ng_module_files)

myApp.js

var app = angular.module(**"myApp"**, []);

|  |  |
| --- | --- |
| **Note** | The [] parameter in the module definition can be used to define dependent modules. |

|  |  |
| --- | --- |
| **Note** | Without the [] parameter, you are not *creating* a new module, but *retrieving* an existing one. |

myCtrl.js

app.controller(**"myCtrl"**, function($scope) {  
    $scope.firstName = "John";  
    $scope.lastName= "Doe";  
});

Functions can Pollute the Global Namespace

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.

Example

<!DOCTYPE html>  
<html>  
<body>  
<script src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script>  
  
<div ng-app="myApp" ng-controller="myCtrl">  
{{ firstName + " " + lastName }}  
</div>  
  
<script>  
var app = angular.module("myApp", []);  
app.controller("myCtrl", function($scope) {  
    $scope.firstName = "John";  
    $scope.lastName = "Doe";  
});  
</script>  
  
</body>  
</html>

[Try it Yourself »](http://www.w3schools.com/angular/tryit.asp?filename=try_ng_module_body)