AngularJS

AngularJS is a JavaScript framework. I

Angular JS is an extended HTML for web apps.

Angular JS is especially for Single Page Application, which results in extraordinarily expressive, readable, and quick to develop.

Angular JS is easy to learn. In this tutorial, we will explain all the basic concepts of AngularJS with examples.

Here , AngularJS is organized in modules which contains various components of an AngularJS application.

[ng (core module)](https://docs.angularjs.org/api/ng)

This module is provided by default and contains the core components of AngularJS.

|  |  |
| --- | --- |
| [Directives](https://docs.angularjs.org/api/ng#directive) | This is the core collection of directives you would use in your template code to build an AngularJS application.  Some examples include: [ngClick](https://docs.angularjs.org/api/ng/directive/ngClick), [ngInclude](https://docs.angularjs.org/api/ng/directive/ngInclude), [ngRepeat](https://docs.angularjs.org/api/ng/directive/ngRepeat), etc… |
| [Services / Factories](https://docs.angularjs.org/api/ng#service) | This is the core collection of services which are used within the DI of your application.  Some examples include: [$compile](https://docs.angularjs.org/api/ng/service/$compile), [$http](https://docs.angularjs.org/api/ng/service/$http), [$location](https://docs.angularjs.org/api/ng/service/$location), etc… |
| [Filters](https://docs.angularjs.org/api/ng#filter) | The core filters available in the ng module are used to transform template data before it is rendered within directives and expressions.  Some examples include: [filter](https://docs.angularjs.org/api/ng/filter/filter), [date](https://docs.angularjs.org/api/ng/filter/date), [currency](https://docs.angularjs.org/api/ng/filter/currency), [lowercase](https://docs.angularjs.org/api/ng/filter/lowercase), [uppercase](https://docs.angularjs.org/api/ng/filter/uppercase), etc... |
| [Global APIs](https://docs.angularjs.org/api/ng#function) | The core global API functions are attached to the angular object. These core functions are useful for low level JavaScript operations within your application.  Some examples include: [angular.copy()](https://docs.angularjs.org/api/ng/function/angular.copy), [angular.equals()](https://docs.angularjs.org/api/ng/function/angular.equals), [angular.element()](https://docs.angularjs.org/api/ng/function/angular.element), etc... |

[ngRoute](https://docs.angularjs.org/api/ngRoute)

Use ngRoute to enable URL routing to your application. The ngRoute module supports URL management via both hashbang and HTML5 pushState.

Include the **angular-route.js** file and set **ngRoute** as a dependency for this to work in your application.

|  |  |
| --- | --- |
| [Services / Factories](https://docs.angularjs.org/api/ngRoute#service) | The following services are used for route management:   * [$routeParams](https://docs.angularjs.org/api/ngRoute/service/$routeParams) is used to access the querystring values present in the URL. * [$route](https://docs.angularjs.org/api/ngRoute/service/$route) is used to access the details of the route that is currently being accessed. * [$routeProvider](https://docs.angularjs.org/api/ngRoute/provider/$routeProvider) is used to register routes for the application. |
| [Directives](https://docs.angularjs.org/api/ngRoute#directive) | The [ngView](https://docs.angularjs.org/api/ngRoute/directive/ngView) directive will display the template of the current route within the page. |

[ngAnimate](https://docs.angularjs.org/api/ngAnimate)

Use ngAnimate to enable animation features within your application. Various core ng directives will provide animation hooks into your application when ngAnimate is included. Animations are defined by using CSS transitions/animations or JavaScript callbacks.

Include the **angular-animate.js** file and set **ngAnimate** as a dependency for this to work in your application.

|  |  |
| --- | --- |
| [Services / Factories](https://docs.angularjs.org/api/ngAnimate#service) | Use [$animate](https://docs.angularjs.org/api/ng/service/$animate) to trigger animation operations within your directive code. |
| [CSS-based animations](https://docs.angularjs.org/api/ngAnimate) | Follow ngAnimate’s CSS naming structure to reference CSS transitions / keyframe animations in AngularJS. Once defined, the animation can be triggered by referencing the CSS class within the HTML template code. |
| [JS-based animations](https://docs.angularjs.org/api/ngAnimate) | Use [module.animation()](https://docs.angularjs.org/api/ng/type/angular.Module" \l "animation) to register a JavaScript animation. Once registered, the animation can be triggered by referencing the CSS class within the HTML template code. |

[ngAria](https://docs.angularjs.org/api/ngAria)

Use ngAria to inject common accessibility attributes into directives and improve the experience for users with disabilities.

Include the **angular-aria.js** file and set ngAria as a dependency for this to work in your application.

|  |  |
| --- | --- |
| [Services](https://docs.angularjs.org/api/ngAria#service) | The [$aria](https://docs.angularjs.org/api/ngAria/service/$aria) service contains helper methods for applying ARIA attributes to HTML.  [$ariaProvider](https://docs.angularjs.org/api/ngAria/provider/$ariaProvider) is used for configuring ARIA attributes. |

[ngResource](https://docs.angularjs.org/api/ngResource)

Use the ngResource module when querying and posting data to a REST API.

Include the **angular-resource.js** file and set **ngResource** as a dependency for this to work in your application.

|  |  |
| --- | --- |
| [Services / Factories](https://docs.angularjs.org/api/ngResource#service) | The [$resource](https://docs.angularjs.org/api/ngResource/service/$resource) service is used to define RESTful objects which communicate with a REST API. |

[ngCookies](https://docs.angularjs.org/api/ngCookies)

Use the ngCookies module to handle cookie management within your application.

Include the **angular-cookies.js** file and set **ngCookies** as a dependency for this to work in your application.

|  |  |
| --- | --- |
| [Services / Factories](https://docs.angularjs.org/api/ngCookies#service) | The following services are used for cookie management:   * The [$cookie](https://docs.angularjs.org/api/ngCookies/service/$cookies) service is a convenient wrapper to store simple data within browser cookies. * [$cookieStore](https://docs.angularjs.org/api/ngCookies/service/$cookieStore) is used to store more complex data using serialization. |

[ngTouch](https://docs.angularjs.org/api/ngTouch)

Use ngTouch when developing for mobile browsers/devices.

Include the **angular-touch.js** file and set **ngTouch** as a dependency for this to work in your application.

|  |  |
| --- | --- |
| [Services / Factories](https://docs.angularjs.org/api/ngTouch#service) | The [$swipe](https://docs.angularjs.org/api/ngTouch/service/$swipe) service is used to register and manage mobile DOM events. |
| [Directives](https://docs.angularjs.org/api/ngTouch#directive) | Various directives are available in ngTouch to emulate mobile DOM events. |

[ngSanitize](https://docs.angularjs.org/api/ngSanitize)

Use ngSanitize to securely parse and manipulate HTML data in your application.

Include the **angular-sanitize.js** file and set **ngSanitize** as a dependency for this to work in your application.

|  |  |
| --- | --- |
| [Services / Factories](https://docs.angularjs.org/api/ngSanitize#service) | The [$sanitize](https://docs.angularjs.org/api/ngSanitize/service/$sanitize) service is used to clean up dangerous HTML code in a quick and convenient way. |
| [Filters](https://docs.angularjs.org/api/ngSanitize#filter) | The [linky filter](https://docs.angularjs.org/api/ngSanitize/filter/linky) is used to turn URLs into HTML links within the provided string. |

[ngMock](https://docs.angularjs.org/api/ngMock)

Use ngMock to inject and mock modules, factories, services and providers within your unit tests.

Include the **angular-mocks.js** file into your test runner for this to work.

|  |  |
| --- | --- |
| [Services / Factories](https://docs.angularjs.org/api/ngMock#service) | ngMock will extend the behavior of various core services to become testing aware and manageable in a synchronous manner.  Some examples include: [$timeout](https://docs.angularjs.org/api/ngMock/service/$timeout), [$interval](https://docs.angularjs.org/api/ngMock/service/$interval), [$log](https://docs.angularjs.org/api/ngMock/service/$log), [$httpBackend](https://docs.angularjs.org/api/ngMock/service/$httpBackend), etc... |
| [Global APIs](https://docs.angularjs.org/api/ngMock#function) | Various helper functions are available to inject and mock modules within unit test code.  Some examples [inject()](https://docs.angularjs.org/api/ngMock/function/angular.mock.inject), [module()](https://docs.angularjs.org/api/ngMock/function/angular.mock.module), [dump()](https://docs.angularjs.org/api/ngMock/function/angular.mock.dump), etc... |

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>

<!DOCTYPE html>

<html>

<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script>

<body>

<div ng-app="">

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

<h1>Hello {{name}}</h1>

</div>

</body>

</html>

## 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.