What is angularjs?

**AngularJS** (commonly referred to as "**Angular**" or "**Angular.js**") is a complete JavaScript-based [open-source](https://en.wikipedia.org/wiki/Open-source_software) front-end [web application framework](https://en.wikipedia.org/wiki/Web_application_framework) mainly maintained by [Google](https://en.wikipedia.org/wiki/Google) and by a community of individuals and corporations to address many of the challenges encountered in developing [single-page applications](https://en.wikipedia.org/wiki/Single-page_application).

AngularJS is perfect for Single Page Applications (SPAs).

AngularJS is a structural framework for dynamic web apps. It lets you extend HTML's syntax to express your application's components clearly and succinctly.

Spa?

A **single**-**page application** (SPA) is a web **application** or web site that fits on a **single** web **page** with the goal of providing a user experience similar to that of a desktop **application**.

**Definition of: desktop application(1)** An application that runs stand alone in a desktop or laptop computer.

## **Features**

* AngularJS is a powerful JavaScript based development framework to create RICH Internet Application(RIA).
* AngularJS provides developers options to write client side application (using JavaScript) in a clean MVC(Model View Controller) way.
* Application written in AngularJS is cross-browser compliant. AngularJS automatically handles JavaScript code suitable for each browser.

Model view controller:

* A *Model* , which represents the underlying, logical structure of data in a software application and the high-level class associated with it. This object model does not contain any information about the user interface.
* A *View* , which is a collection of classes representing the elements in the user interface (all of the things the user can see and respond to on the screen, such as buttons, display boxes, and so forth)
* A *Controller* , which represents the classes connecting the model and the view, and is used to communicate between classes in the model and view.

Add angularjs to your page;

**library** –:

**(1)** A collection of [files](http://www.webopedia.com/TERM/F/file.html).

**(2)** In [programming](http://www.webopedia.com/TERM/P/program.html), a library is a collection of pre[compiled](http://www.webopedia.com/TERM/C/compile.html) [routines](http://www.webopedia.com/TERM/R/routine.html) that a program can use. The routines, sometimes called [*modules*](http://www.webopedia.com/TERM/M/module.html), are [stored](http://www.webopedia.com/TERM/S/store.html) in [object](http://www.webopedia.com/TERM/O/object.html) [format](http://www.webopedia.com/TERM/F/format.html). Libraries are particularly useful for storing frequently used routines because you do not need to explicitly [link](http://www.webopedia.com/TERM/L/link.html) them to every program that uses them. The [linker](http://www.webopedia.com/TERM/L/linker.html) automatically looks in libraries for routines that it does not find elsewhere.

((linker- a linker is a [program](http://www.webopedia.com/TERM/P/program.html) that combines object [modules](http://www.webopedia.com/TERM/M/module.html) to form an executable program)).

## 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:

## **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.

<https://ajax.googleapis.com/ajax/libs/angularjs/1.6.1/angular.min.js>

<https://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js>

directices:

directives are used in AngularJS to extend the functionality of HTML. directives are defined using "directive" function. A custom directive simply replaces the element for which it is activated.

* **Element directives** − Directive activates when a matching element is encountered.
* **Attribute** − Directive activates when a matching attribute is encountered.
* **CSS** − Directive activates when a matching css style is encountered.
* **Comment** − Directive activates when a matching comment is encountered.

 ng-app directive initializes an AngularJS application.

 ng-init directive initializes application data.ng-init directive defines **initial values** for an AngularJS application.

ng-model directive binds the value of HTML controls (input, select, textarea) to application data.

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

The ng-bind directive tells AngularJS to replace the content of an HTML element with the value of a given variable, or expression.

Data Binding

Data-binding in Angular apps is the automatic synchronization of data between the model and view components. The way that Angular implements data-binding lets you treat the model as the single-source-of-truth in your application. The view is a projection of the model at all times. When the model changes, the view reflects the change, and vice versa.

filter:

Filters can be added in AngularJS to format data.

Filters can be added to expressions by using the pipe character |, followed by a filter.

*Expression:*

AngularJS will "output" data exactly where the expression is written

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.

Example {{ 5 + 5 }} or {{ firstName + " " + lastName }}

## **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.

Controller:

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

AngularJS controllers are regular **JavaScript Objects**.

AngularJS applications are controlled by controllers.

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

A controller is a **JavaScript Object**, created by a standard JavaScript **object constructor**.

# Module:

The module is a container for the different parts of an application.

A module is a collection of services, directives, controllers, filters, and configuration information.

The module is a container for the application controllers.

Controllers always belong to a module.

Scope:

AngularJS Scope. The scope is the binding part between the HTML (view) and the JavaScript (**controller**). The scope is an object with the available properties and methods. The scope is available for both the view and the **controller**.

Array [ ] :

That array is meant to add various module to your current app which is mentioned in your first part of angular.module as string`, You could simply say for injecting various dependency.

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

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

## **What is Routing in AngularJS?**

If you want to navigate to different pages in your application, but you also want the application to be a SPA (Single Page Application), with no page reloading, you can use the ngRoute module.

ngRoute :

The ngRoute module routes your application to different pages without reloading the entire application.

you must add the ngRoute as a dependency in the application module::

Now your application has access to the route module, which provides the $routeProvider.

$routeProvider.:

Use the $routeProvider to configure different routes in your application:

Script:

To make your applications ready for routing, you must include the AngularJS Route module:

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

Ng-view:

Your application needs a container to put the content provided by the routing.

This container is the ng-view directive.

There are three different ways to include the ng-view directive in your application

Config:

Define the $routeProvider using the config method of your application. Work registered in the config method will be performed when the application is loading.

## What is a Service?

In AngularJS, a service is a function, or object, that is available for, and limited to, your AngularJS application.

AngularJS has about 30 built-in services.

Why using the $ symbol before :

'$' Prefix Naming Convention  
As a naming convention, angular's built-in services, Scope methods and a few other angular APIs have a '$' prefix in front of the name. Don't use a '$' prefix when naming your services and models, in order to avoid any possible naming collisions.

Naming conversion:

In [computer programming](https://en.wikipedia.org/wiki/Computer_programming), a [**naming convention**](https://en.wikipedia.org/wiki/Naming_convention) is a set of rules for choosing the character sequence to be used for [identifiers](https://en.wikipedia.org/wiki/Identifier) which denote [variables](https://en.wikipedia.org/wiki/Variable_(computer_science)), [types](https://en.wikipedia.org/wiki/Data_type), [functions](https://en.wikipedia.org/wiki/Subroutine), and other entities in [source code](https://en.wikipedia.org/wiki/Source_code) and [documentation](https://en.wikipedia.org/wiki/Software_documentation).

Json:

JSON is a syntax for storing and exchanging data.

When exchanging data between a browser and a server, the data can only be text.

JSON is text, and we can convert any JavaScript object into JSON, and send JSON to the server.

We can also convert any JSON received from the server into JavaScript objects.

Object prototype:

Every JavaScript object has a prototype. The prototype is also an object.

All JavaScript objects inherit their properties and methods from their prototype.

## The ***this*** Keyword

In JavaScript, the thing called **this**, is the object that "owns" the JavaScript code.

The value of **this**, when used in a function, is the object that "owns" the function.

The value of **this**, when used in an object, is the object itself.

The **this** keyword in an object constructor does not have a value. It is only a substitute for the new object.

The value of **this** will become the new object when the constructor is used to create an object.

Note that **this** is not a variable. It is a keyword. You cannot change the value of **this**.

Mvc:

**M**odel **V**iew **C**ontroller or MVC as it is popularly called, is a software design pattern for developing web applications. A Model View Controller pattern is made up of the following three parts −

* **Model** − It is the lowest level of the pattern responsible for maintaining data.
* **View** − It is responsible for displaying all or a portion of the data to the user.
* **Controller** − It is a software Code that controls the interactions between the Model and View.

UNIT TESTING:

JavaScript is a dynamically typed language which comes with great power of expression, but it also comes with almost no help from the compiler. For this reason we feel very strongly that any code written in JavaScript needs to come with a strong set of tests. We have built many features into AngularJS which make testing your AngularJS applications easy. With AngularJS, there is no excuse for not testing.

1. karma

2. jasmine

### 3. **angular-mocks**

# E2E Testing

As applications grow in size and complexity, it becomes unrealistic to rely on manual testing to verify the correctness of new features, catch bugs and notice regressions. Unit tests are the first line of defense for catching bugs, but sometimes issues come up with integration between components which can't be captured in a unit test. End-to-end tests are made to find these problems.

We have built [Protractor](https://github.com/angular/protractor), an end to end test runner which simulates user interactions that will help you verify the health of your AngularJS application.

# 1.Protractor

[Protractor](http://angular.github.io/protractor) is an end-to-end test framework for [Angular](http://angular.io/) and [AngularJS](http://angularjs.org/) applications. Protractor is a [Node.js](http://nodejs.org/) program built on top of [WebDriverJS](https://github.com/SeleniumHQ/selenium/wiki/WebDriverJs). Protractor runs tests against your application running in a real browser, interacting with it as a user would.

Template?

In AngularJS, templates are written with HTML that contains AngularJS-specific elements and attributes. AngularJS combines the template with information from the model and controller to render the dynamic view that a user sees in the browser.

Run and config method:

* The run method accepts a function, which can be injected with "service", "value" and "constant" components as dependencies. Note that you cannot inject "providers" into run blocks.
* The config method accepts a function, which can be injected with "provider" and "constant" components as dependencies. Note that you cannot inject "service" or "value" components into configuration.

Difference between run and config method:

Configuration block – This block is executed during the provider registration and configuration phase. Only providers and constants can be injected into configuration blocks. This block is used to inject module wise configuration settings to prevent accidental instantiation of services before they have been fully configured. This block is created using config() method.  
  
Run block – This block is executed after the configuration block. It is used to inject instances and constants. This block is created using run() method. This method is like as main method in C or C++. The run block is a great place to put event handlers that need to be executed at the root level for the application. For example, authentication handlers.

((event handlers- An AngularJS event will not overwrite an HTML event, both events will be executed.))

Dependency injection:

AngularJS comes with a built-in dependency injection mechanism. It facilitates you to divide your application into multiple different types of components which can be injected into each other as dependencies.

Dependency Injection is a software design pattern that specifies how components get holds of their dependencies. In this pattern, components are given their dependencies instead of coding them within the component.

* value
* factory
* service
* provider
* constant

## **Value**

In AngularJS, value is a simple object. It can be a number, string or JavaScript object. It is used to pass values in factories, services or controllers during run and config phase.

Factory:

Factory is a function that is used to return value. When a service or controller needs a value injected from the factory, it creates the value on demand. It normally uses a factory function to calculate and return the value.

Services:

In AngularJS, service is a JavaScript object which contains a set of functions to perform certain tasks. Services are created by using service() function on a module and then injected into controllers.

Provider:

In AngularJS, provider is used internally to create services, factory etc. during config phase (phase during which AngularJS bootstraps itself). It is the most flexible form of factory you can create. Provider is a special factory method with a get() function which is used to return the value/service/factory.

Constant:

Bootstrap:

General definition: A technique of loading a program into a computer by means of a few initial instructions which enable the introduction of the rest of the program from an input device.

 The AngularJS initialization process and how you can manually initialize AngularJS if necessary.

Two types:

1. Automatic initialization

2. Manual initialization.

Validation:

AngularJS offers client-side form validation.

AngularJS monitors the state of the form and input fields (input, textarea, select), and lets you notify the user about the current state.

Angularjs also holds information about whether they have been touched, or modified, or not.

You can use standard HTML5 attributes to validate input, or you can make your own validation functions.

Client-side validation cannot alone secure user input. Server side validation is also necessary.