Q. Name the key features of AngularJS ?

The key features of AngularJS are:

* Scope
* Controller
* Model
* View
* Services
* Data Binding
* Directives
* Filters
* Testable

Q.What is the difference between link and compile in Angular.js?

* Compile function is used for template DOM Manipulation and to collect all the directives.
* Link function is used for registering DOM listeners as well as instance DOM manipulation and is executed once the template has been cloned.

Q. What are the advantages of using Angular.js framework?

Angular.js framework has the following advantages:

* Supports two way data-binding
* Supports MVC pattern
* Support static template and angular template
* Can add custom directive
* Supports REST full services
* Supports form validations
* Support both client and server communication
* Support dependency injection
* Applying Animations
* Event Handlers

Q. Explain what is injector in AngularJS?

An injector is a service locator, used to retrieve object instance as defined by provider, instantiate types, invoke methods, and load modules.

Q. What is factory method in AngularJS?

Factory method is used for creating a directive.  It is invoked when the compiler matches the directive for the first time. We can invoke the factory method using $injector.invoke.

Syntax: module.factory('factoryName',function);  
Result: When declaring factoryName as an injectable argument you will be provided with the value that is returned by invoking the function reference passed to module.factory.

Q. Can AngularJS have multiple ng-app directives in a single page?

No. Only one AngularJS application can be auto-bootstrapped per HTML document. The first ngApp found in the document will be used to define the root element to auto-bootstrap as an application. If another ng-app directive has been placed then it will not be processed by AngularJS and we will need to manually bootstrap the second app, instead of using second ng-app directive.

Q. What is $rootscope in AngularJS?

Every application has a single root scope. All other scopes are descendant scopes of the root scope. Scopes provide separation between the model and the view, via a mechanism for watching the model for changes. They also provide event emission/broadcast and subscription facility.

Q. What is bootstrapping in AngularJS?

Bootstrapping in AngularJS is nothing but initializing, or starting the Angular app. AngularJS supports automatic and manual bootstrapping.

* Automatic Bootstrapping: this is done by adding ng-app directive to the root of the application, typically on the tag or tag if you want angular to bootstrap your application automatically. When angularJS finds ng-app directive, it loads the module associated with it and then compiles the DOM.
* Manual Bootstrapping:Manual bootstrapping provides you more control on how and when to initialize your angular App. It is useful where you want to perform any other operation before Angular wakes up and compile the page.

Q. How to implement routing in AngularJS?

It is a five-step process:

* Step 1: – Add the “Angular-route.js” file to your view.
* Step 2: – Inject “ngroute” functionality while creating Angular app object.
* Step 3: – Configure the route provider.
* Step 4: – Define hyperlinks.
* Step 5: – Define sections where to load the view.

Q. What is the difference between one-way binding and two-way binding?

– One way binding implies that the scope variable in the html will be set to the first value its model is bound to (i.e. assigned to)

– Two way binding implies that the scope variable will change it’s value everytime its model is assigned to a different value.

Q. Explain what is a $scope in AngularJS?

Scope is an object that refers to the application model. It is an execution context for expressions. Scopes are arranged in hierarchical structure which mimics the DOM structure of the application. Scopes can watch expressions and propagate events. Scopes are objects that refer to the model. They act as glue between controller and view.

Q. What are Directives?

Directives are markers on a DOM element (such as an attribute, element name, comment or CSS class) that tell AngularJS’s HTML compiler ($compile) to attach a specified behavior to that DOM element (e.g. via event listeners), or even to transform the DOM element and its children. Angular comes with a set of these directives built-in, like ngBind, ngModel, and ngClass. Much like you create controllers and services, you can create your own directives for Angular to use. When Angular bootstraps your application, the HTML compiler traverses the DOM matching directives against the DOM elements.

Q. What should be the maximum number of concurrent “watches”? Bonus: How would you keep an eye on that number?

To reduce memory consumption and improve performance it is a good idea to limit the number of watches on a page to 2,000. Utility called ng-stats can help track your watch count and digest cycles.

Q. How do you share data between controllers?

Create an AngularJS service that will hold the data and inject it inside of the controllers. Using a service is the cleanest, fastest and easiest way to test.

However, there are couple of other ways to implement data sharing between controllers, like:

– Using events

– Using $parent, nextSibling, controllerAs, etc. to directly access the controllers

– Using the $rootScope to add the data on (not a good practice)

The methods above are all correct, but are not the most efficient and easy to test.

Q. What is the difference between ng-show/ng-hide and ng-if directives?

ng-show/ng-hide will always insert the DOM element, but will display/hide it based on the condition. ng-if will not insert the DOM element until the condition is not fulfilled.

ng-if is better when we needed the DOM to be loaded conditionally, as it will help load page bit faster compared to ng-show/ng-hide.

We only need to keep in mind what the difference between these directives is, so deciding which one to use totally depends on the task requirements.

Q. What is a digest cycle in AngularJS?

In each digest cycle Angular compares the old and the new version of the scope model values. The digest cycle is triggered automatically. We can also use $apply() if we want to trigger the digest cycle manually.

Q. Where should we implement the DOM manipulation in AngularJS?

In the directives. DOM Manipulations should not exist in controllers, services or anywhere else but in directives.

Q. If you were to migrate from Angular 1.4 to Angular 1.5, what is the main thing that would need refactoring?

Changing .directive to .component to adapt to the new Angular 1.5 components

Q. How would you specify that a scope variable should have one-time binding only?

By using “::” in front of it. This allows the check if the candidate is aware of the available variable bindings in AngularJS.

Q. Explain how $scope.$apply() works?

$scope.$apply re-evaluates all the declared ng-models and applies the change to any that have been altered (i.e. assigned to a new value)

Explanation: $scope.$apply() is one of the core angular functions that should never be used explicitly, it forces the angular engine to run on all the watched variables and all external variables and apply the changes on their values.

Q. What makes the angular.copy() method so powerful?

It creates a deep copy of the variable. A deep copy of a variable means it doesn’t point to the same memory reference as that variable. Usually assigning one variable to another creates a “shallow copy”, which makes the two variables point to the same memory reference. Therefore if we change one, the other changes as well.

Q. How would you make an Angular service return a promise? Write a code snippet as an example

To add promise functionality to a service, we inject the “$q” dependency in the service. The $q library is a helper provider that implements promises and deferred objects to enable asynchronous functionality.

Q. What is the role of services in AngularJS and name any services made available by default?

– AngularJS Services are objects that provide separation of concerns to an AngularJS app.

– AngularJS Services can be created using a factory method or a service method.

– Services are singleton components. All components of the application (into which the service is injected) will work with single instance of the service.

– An AngularJS service allows developing of business logic without depending on the View logic which will work with it.

Few of the inbuilt services in AngularJS are:

– the $http service: The $http service is a core Angular service that facilitates communication with the remote HTTP servers via the browser’s XMLHttpRequest object or via JSONP

– the $log service: Simple service for logging. Default implementation safely writes the message into the browser’s console

– the $anchorScroll: it scrolls to the element related to the specified hash or (if omitted) to the current value of $location.hash()

Why should one know about AngularJS Services, you may ask. Well, understanding the purpose of AngularJS Services helps bring modularity to AngularJS code.

Services are the best way to evolve reusable API within and AngularJS app.

Q. When creating a directive, it can be used in several different ways in the view. Which ways for using a directive do you know? How do you define the way your directive will be used?

When you create a directive, it can be used as an attribute, element or class name. To define which way to use, you need to set the restrict option in your directive declaration.The restrict option is typically set to:

‘A’ – only matches attribute name

‘E’ – only matches element name

‘C’ – only matches class name

These restrictions can all be combined as needed:

‘AEC’ – matches either attribute or element or class name

Q. When should you use an attribute versus an element?

Use an element when you are creating a component that is in control of the template. Use an attribute when you are decorating an existing element with new functionality.

This topic is important so developers can understand the several ways a directive can be used inside a view and when to use each way.

Q. How do you reset a $timeout, $interval(), and disable a $watch()?

To reset a timeout and/or $interval, assign the result of the function to a variable and then call the .cancel() function. to disable $watch(), we call its deregistration function. $watch() then returns a deregistration function that we store to a variable and that will be called for cleanup

Q. How would you validate a text input field for a twitter username, including the @ symbol?

You would use the ngPattern directive to perform a regex match that matches Twitter usernames. The same principal can be applied to validating phone numbers, serial numbers, barcodes, zip codes and any other text input.

Q. How would you implement application-wide exception handling in your Angular app?

Angular has a built-in error handler service called $exceptionHandler which can easily be overridden. This is very useful for sending errors to third party error logging services or helpdesk applications. Errors trapped inside of event callbacks are not propagated to this handler, but can manually be relayed to this handler by calling $exceptionHandler(e) from within a try catch block.

Q. How would you react on model changes to trigger some further action? For instance, say you have an input text field called email and you want to trigger or execute some code as soon as a user starts to type in their email.

We can achieve this using $watch function in our controller.

function MyCtrl($scope) {

$scope.email = "";

$scope.$watch("email", function(newValue, oldValue) {

if ($scope.email.length > 0) {

console.log("User has started writing into email");

}

});

}