* **What is AngularJS and what are some of its advantages?**

**A**

This question might seem basic at first glance, but what you’re really doing is giving the developer a chance to show you what they know about your chosen framework. AngularJS is a powerful JavaScript-based development framework designed to create dynamic single-page applications with fewer lines of code. Some of the key advantages that you’ll want to look for in their response are listed below.

* + Data binding is as easy as writing in your code.
  + AngularJS was made for CRUD applications, which happen to represent the majority of web apps (excluding DOM manipulation-intensive applications like games and GUI editors).
  + It separates DOM manipulation from app logic, making code modular and easy to test.
  + It’s a comprehensive client-side solution in that it decouples the client-side from server-side development effort.
  + It saves months of development time by freeing the developer from having to write repetitive low-level DOM manipulation tasks, manually registering callbacks, and otherwise automating most AJAX application tasks.
  + It’s great for providing a “desktop-like” experience to the end user.

**What is the Model View Controller (MVC)?**

**A**

MVC is a common design pattern used in developing software that consists of three parts: the model, view, and controller. The model is the lowest level of the pattern and is responsible for maintaining data. The view is the part of the application that is presented to the user. The controller is the code that governs all interactions between the model and the view.

**What is data binding in AngularJS? How does it relate to the MVC architecture?**

In most templating systems, data binding is unidirectional. When the model and template components are merged together, it creates a view. However, the developer must write code to constantly synchronize the model and the view. AngularJS uses two-way data binding, where any changes to the view will automatically update the model and vice versa. The view is more or less just a projection of the model, which greatly simplifies things from the programmer’s perspective.

**Explain the concept of scope. How does scope inheritance work in AngularJS?**

Scope is an object that represents the data-model of an AngularJS application—it is the glue between the view and the application controller. Scope inheritance closely mimics the DOM structure of the application. With the exception of isolated scopes created using custom directives, scopes follow prototypal inheritance. The code block below demonstrates typical scope inheritance.

<script>

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

myApp.controller("fruitController", function($scope) {

$scope.message = "This is a Fruit";

$scope.type = "Fruit";

});

myApp.controller("appleController", function($scope) {

$scope.message = "This is an Apple";

});

</script>

The important thing to note in the above example is that values have been set to models in fruitController, and the message in appleController has been overridden.

**Explain the difference between a factory and a service in AngularJS.**

AngularJS encourages developers to store business logic and persistent data separately from controllers and scopes within an application, by providing factories and services. To understand the difference, let’s take a look at the code block below.

// The code below is a service.

app.service('MyService', function () {

this.helloWorld = function () {

console.log('Hello World');

};

});

// The code below is a factory.

app.factory('MyService', function () {

return {

helloWorld: function () {

console.log('Hello World');

};

}

});

While syntactically distinct, the above examples perform the same task of printing “Hello World” to the console. The conceptual difference you’ll want to look for in the applicant’s answer however, is that a service is a constructor function while a factory is a function that must be called. This is why the service can use the “this” keyword while a factory must explicitly return the object that it creates. An even better answer would also reveal that a service is actually calling a predefined factory deep within the Angular.js file loaded into all AngularJS applications. Check out this code snippet from the Angular.js file.

// By calling service() you are also calling a predefined factory() within AngularJs

function service(name, constructor) {

return factory(name, ['$injector', function($injector) {

return $injector.instantiate(constructor);

}]);

}

**Explain why there are two “destroy” events associated with the termination of a scope in AngularJS.**

The first $destroy is an AngularJS event associated with components like controllers or link functions. The second is actually a jqLite/jQuery event associated with the removal of a node, which may occur without a scope teardown.

**What is dependency injection and how does it work?**

AngularJS was designed to highlight the power of dependency injection, a software design pattern that places an emphasis on giving components their dependencies instead of hard coding them within the component. For example, if you had a controller that needed to access a list of customers, you would store the actual list of customers in a service that can be injected into the controller instead of hardcoding the list of customers into the code of the controller itself. In AngularJS you can inject values, factories, services, providers, and constants.

**What are directives? Can you explain the functions of the following directives?**

* ng-app

* ng-model

* ng-bind

* ng-controller

Directives are used by AngularJS to extend the functionality of HTML by adding new attributes with the ng- prefix. The four directives listed above are some of the more important ones every AngularJS developer should know by heart.

* + The ng-app directive can be placed within a element to make it the root element of an AngularJS application.
  + The ng-model directive binds the value of HTML controls to application data.
  + The ng-bind directive binds the content of an HTML element to application data.
  + The ng-controller directive defines the controller object for an application.

**Explain the role of $routeProvider in AngularJS.**

The $routeProvider is used to configure roots within an AngularJS application. It can be used to link a url with a corresponding HTML page or template, and a controller (if applicable).

**How experienced are you with e2e testing? Explain how e2e testing of AngularJS applications works.**

End-to-end (e2e) testing is the practice of testing an application from start to finish to determine whether all the components are working together properly. If unit tests are the first line of defense against bugs within the individual components, e2e testing can be thought of as the safety net that catches issues related to integration and flow within an application. The AngularJS team built Protractor, a Node.js application that can simulate user interactions and help developers test the overall health of their AngularJS applications. It’s a good idea to ask an applicant about past experiences using Protractor to perform e2e testing on AngularJS applications.