3 of 31

AngularJS Interview Question #1

What are the basic steps to unit test an AngularJS filter?

(Question provided by [Daniel Lamb](#))

1. Inject the module that contains the filter.
2. Provide any mocks that the filter relies on.
3. Get an instance of the filter using `$filter('yourFilterName')`.
4. Assert your expectations.

Dependency injection is a powerful software design pattern that Angular employs to compose responsibilities through an intrinsic interface. However, for those new to the process, it can be puzzling where you need to configure and mock these dependencies when creating your isolated unit tests. The open-source project “Angular Test Patterns” is a free resource that is focused on dispelling such confusion through high-quality examples.

This question is useful since it can give you a feel for how familiar the candidate is with automated testing (TDD, BDD, E2E), as well as open up a conversation about approaches to code quality.

Source:
https://github.com/daniellmb/angular-test-patterns/blob/master/patterns/filter.md
https://docs.angularjs.org/guide/unit-testing

AngularJS Interview Question #2

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

(Question provided by [Daniel Lamb](#))

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

Receive New Tutorials
GET IT FREE

Jank happens when your application cannot keep up with the screen refresh rate. To achieve 60 frames-per-second, you only have about 16 milliseconds for your code to execute. It is crucial that the scope digest cycles are as short as possible for your application to be responsive and smooth. Memory use and digest cycle performance are directly affected by the number of active watches. Therefore, it is best to keep the number of watches below 2,000. The open-source utility `ng-stats` gives developers insight into the number of watches Angular is managing, as well as the frequency and duration of digest cycles over time.

Caution: Be wary of relying on a “single magic metric” as the golden rule to follow. You must take the context of your application into account. The number of watches is simply a basic health signal. If you have many thousands of watches, or worse, if you see that number continue to grow as you interact with your page. Those are strong indications that you should look under the hood and review your code.

This question is valuable as it gives insight into how the candidate debugs runtime issues while creating a discussion about performance and optimization.

Sources:
https://github.com/kentcdodds/ng-stats
http://jankfree.org

Daniel has over 15 years experience specializing in large-scale front-end architecture and web development. If you want to practice your interview with Daniel and get feedback on your performance, feel free to schedule a mock interview with Daniel!

AngularJS Interview Question #3

How do you share data between controllers?

(Question provided by Tome Pejoski)

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.
Here is a good video explanation on egghead.io.

AngularJS Interview Question #4

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

(Question provided by Tome Pejoski)

`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.

AngularJS Interview Question #5

What is a digest cycle in AngularJS?

(Question provided by Tome Pejoski)

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.

For more information, take a look in the ng-book explanation: The Digest Loop and $apply

AngularJS Interview Question #6

Where should we implement the DOM manipulation in AngularJS?

(Question provided by Tome Pejoski)

Receive New Tutorials
GET IT FREE

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

Here is a detailed explanation

AngularJS Interview Question #7

Is it a good or bad practice to use AngularJS together with jQuery?

(Question provided by [Tome Pejoski](#))

It is definitely a bad practice. We need to stay away from jQuery and try to realize the solution with an AngularJS approach. jQuery takes a traditional imperative approach to manipulating the DOM, and in an imperative approach, it is up to the programmer to express the individual steps leading up to the desired outcome.

AngularJS, however, takes a declarative approach to DOM manipulation. Here, instead of worrying about all of the step by step details regarding how to do the desired outcome, we are just declaring what we want and AngularJS worries about the rest, taking care of everything for us.
Here is [a detailed explanation](#)

Tome is a senior software engineer at Netcetera and has worked on various AngularJS, 2 of which are enterprise-level. Feel free to [schedule a mock interview with Tome](#)!

AngularJS Interview Question #8

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

(Question provided by [Jad Salhani](#))

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

AngularJS Interview Question #9

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

(Question provided by [Jad Salhani](#))

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

AngularJS Interview Question #10

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

(Question provided by [Jad Salhani](#))

– 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

AngularJS Interview Question #11

Explain how `$scope.$apply()` works

(Question provided by [Jad Salhani](#))

`$scope.$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
Source: https://docs.angularjs.org/api/ng/type/$rootScope.Scope

AngularJS Interview Question #12

What directive would you use to hide elements from the HTML DOM by removing them from that DOM not changing their styling?

(Question provided by [Jad Salhani](#))

The `ngIf` Directive, when applied to an element, will remove that element from the DOM if it's condition is false.

Receive New Tutorials
[GET IT FREE](#)

AngularJS Interview Question #13

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

(Question provided by [Jad Salhani](#))

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

Sources:
– https://docs.angularjs.org/api/ng/function/angular.copy
– https://en.wikipedia.org/wiki/Object_copying

AngularJS Interview Question #14

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

(Question provided by Jad Salhani)

To add promise functionality to a service, we inject the "$q" dependency in the service, and then use it like so:

```
angular.factory('testService', function($q){
        return {
                getName: function(){
                        var deferred = $q.defer();

                        //API call here that returns data
                        testAPI.getName().then(function(name){
                                deferred.resolve(name)
                        })

                        return deferred.promise;
                }
        }
})
```

The `$q` library is a helper provider that implements promises and deferred objects to enable asynchronous functionality

Source: https://docs.angularjs.org/api/ng/service/$q

Jad is a 5-star rated Codementor who has helped mentor many AngularJS users. If you want to practice your interview with Jad and understand AngularJS concepts better, feel free to schedule a mock interview with Jad!

AngularJS Interview Quesion #15

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

(Question provided by Harikishore Tadigotla)

– 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 may to evolve reusable API within and AngularJS app
Receive New Tutorials
Overview: GET IT FREE

- AngularJS Services help create reusable components.
- A Service can be created either using the `service()` method or the `factory()` method.
- A typical service can be injected into another service or into an AngularJS Controller.

Source:
– https://docs.angularjs.org/guide/services

– http://www.tutorialspoint.com/angularjs/angularjs_services.htm

Harikishore is a former lead developer for Nokia and has been a senior systems analyst for JP Morgan before. If you want to practice your interview with Harikishore and get feedback, feel free to schedule a mock interview with Harikishore!

AngularJS Interview Question #16

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?

(Question provided by Nuno Brites*)*

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

For more information, feel free to check out the AngularJS documentation.

AngularJS Interview Question #17

When should you use an attribute versus an element?

(Question provided by Nuno Brites*)*

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.

Sources: https://docs.angularjs.org/api/ng/service/$compile#directive-definition-object

Nuno is a full stack software engineer with 6 years of experience developing enterprise apps. If you want to practice your interview with Nuno, feel free to schedule a mock interview with Nuno!

AngularJS Interview Question #18

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

(Question provided by Fouad Kada*)*

To reset a `timeout` and/or `$interval`, assign the result of the function to a variable and then call the `.cancel()` function.

```
var customTimeout = $timeout(function () {
  // arbitrary code
}, 55);

$timeout.cancel(customTimeout);
```

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

```
var deregisterWatchFn = $scope.$on('$destroy', function () {
    // we invoke that deregistration function, to disable the watch
    deregisterWatchFn();
});
```

Fouad is a software engineer with 3 years of experience with AngularJS. If you want to practice your interview with Fouad, feel free to schedule a mock interview with Fouad!

AngularJS Interview Question #19

Explain what is a `$scope` in AngularJS

(Question provided by Ashish Kumar*)*

Scope is an object that refers to the application model. It is an execution context for expressions. Scopes are arranged in hierarchical structure which mimic 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.

This question is important as it will judge a persons knowledge about a $scope object, and it is one of the most important concepts in AngularJS. Scope acts like a bridge between view and model.

Source: https://docs.angularjs.org/guide/scope

AngularJS Interview Question #20

What are Directives?

(Question provided by Ashish Kumar*)*

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.

This question is important because directives define the UI while defining a single page app. You need to be very clear about how to create a new custom directive or use the existing ones already pre-build in AngularJS.

Source: https://docs.angularjs.org/guide/directive

AngularJS Interview Question #21

What is DDO Directive Definition Object?

(Question provided by Ashish Kumar*)*

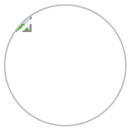
"DDO is an object used while creating a custome directive. A standard DDO object has following parameters.

```
var directiveDefinitionObject = {
    priority: 0,
    template: '<div></div>', // or // function(tElement, tAttrs) { ... },
    // or
    // templateUrl: 'directive.html', // or // function(tElement, tAttrs) { ... },
    transclude: false,
    restrict: 'A',
    templateNamespace: 'html',
    scope: false,
    controller: function($scope, $element, $attrs, $transclude, otherInjectables) { ... },
    controllerAs: 'stringIdentifier',
    bindToController: false,
    require: 'siblingDirectiveName', // or // ['^parentDirectiveName', '?optionalDirectiveName', '?^optionalParent'],
    compile: function compile(tElement, tAttrs, transclude) {
      return {
        pre: function preLink(scope, iElement, iAttrs, controller) { ... },
        post: function postLink(scope, iElement, iAttrs, controller) { ... }
      }
      // or
      // return function postLink( ... ) { ... }
    },
    // or
    // link: {
```

```
    //  pre: function preLink(scope, iElement, iAttrs, controller) { ... },
    //  post: function postLink(scope, iElement, iAttrs, controller) { ... }
    // }
    // or
    // link: function postLink( ... ) { ... }
  };"
```

This question mainly judges whether candidate knows about creating custom directives.

Read more at https://docs.angularjs.org/guide/directive



Ashish has 7 years of experience in software development and currently holds a 5-star rating on Codementor.io. If you want to practice your interview with Ashish, feel free to schedule a mock interview with Ashish!

AngularJS Interview Question #22

(Question provided by Jon Oyanguren López)

What is a singleton pattern and where we can find it in Angularjs?

Is a great pattern that restricts the use of a class more than once. We can find singleton pattern in angular in dependency injection and in the services.

In a sense, if you do 2 times '`new Object()`' without this pattern, you will be alocating 2 pieces of memory for the same object. With singleton pattern, if the object exists, you reuse it.

Source: http://joelhooks.com/blog/2013/05/01/when-is-a-singleton-not-a-singleton/

AngularJS Interview Question #23

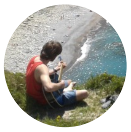
(Question provided by Jon Oyanguren López)

What is an interceptor? What are common uses of it?

An interceptor is a middleware code where all the `$http` requests go through.

The interceptor is a factory that are registered in `$httpProvider`. You have 2 types of requests that go through the interceptor, request and response (with `requestError` and `responseError` respectively). This piece of code is very useful for error handling, authentication or middleware in all the requests/responses.

Source: https://docs.angularjs.org/api/ng/service/$http



Jon works as a full stack developer in Angularjs, Javascript, and .Net. He is the CTO of iDriveYourCar and innovUp, and a teacher at CMU University. If you want to practice your interview with Jon, feel free to schedule a mock interview with Jon!

Angular Interview Question #24

How would you programatically change or adapt the template of a directive before it is executed and transformed?

(Question provided by Johann de Swardt)

You would use the compile function. The compile function gives you access to the directive's template before transclusion occurs and templates are transformed, so changes can safely be made to DOM elements. This is very useful for cases where the DOM needs to be constructed based on runtime directive parameters.

Receive New Tutorials
GET IT FREE

Read more about it here.

Angular Interview Question #25

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

(Question provided by [Johann de Swardt](#))

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.

The official documentation can be found [here](#).

Angular Interview Question #26

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

(Question provided by [Johann de Swardt](#))

Angular has a built-in error handler service called `$exceptionHandler` which can easily be overriden as seen below:

```
myApp.factory('$exceptionHandler', function($log, ErrorService) {
    return function(exception, cause) {

        if (console) {
            $log.error(exception);
            $log.error(cause);
        }

        ErrorService.send(exception, cause);
    };
});
```

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.

Johann has 13 years of professional programming experience, and he's a full-stack Javascript developer specialising in NodeJS, AngularJS and Ionic apps. He is also currently the CTO of a consulting firm. If you want to practice your interview with Johann, feel free to [schedule a mock interview Johann](#)!

AngularJS Interview Question #27

How do you hide an HTML element via a button click in AngularJS?

(Question provided by [Nishant Kumar](#))

You can do this by using the `ng-hide` directive in conjunction with a controller we can hide an HTML element on button click.

```
<div ng-controller="MyCtrl">
        <button ng-click="hide()">Hide element</button>
        <p ng-hide="isHide">Hello World!</p>
</div>

function MyCtrl($scope){
        $scope.isHide = false;
        $scope.hide = function(){
                $scope.isHide = true;
        }
}
```

AngularJS Interview Question #28

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.

(Question provided by [Nishant Kumar](#))

Receive New Tutorials
[GET IT FREE](#)

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");
                }
        });
}
```

AngularJS Interview Question #29

How do you disable a button depending on a checkbox's state?

(Question provided by [Nishant Kumar](#))

We can use the `ng-disabled` directive and bind its condition to the checkbox's state.

```
<body ng-app>
        <label><input type="checkbox" ng-model="checked"/>Disable Button</label>
        <button ng-disabled="checked">Select me</button>
</body>
```

Nishant is the UI Engineering Lead at a Cloud Platform Startup, and a tech contributor at Mozila. If you want to practice your interview with Nishant, feel free to [schedule a mock interview Nishant](#)!

Conclusion

If you're still uncertain about your AngularJS knowledge, feel free to [schedule a session](#) with [AngularJS experts](#) at Codementor and ask them to help review your code, help you understand concepts you have trouble grasping, and more. We're always here for you, and good luck on acing that interview!