Latest AngularJS interview questions - Part 3

With continuation of AngularJS interview question series, after Part 1 and Part 2, here is list of 3rd set of 10 AngularJS interview questions and their answers.

Latest AngularJS interview questions - Part 1
Latest AngularJS interview questions - Part 2
Latest AngularJS interview questions - Part 4

Q21. What is Angular Expression?
Ans: Angular expressions are JavaScript-like code snippets that are usually placed in bindings such as {{ expression }}. For example, these are valid expressions in Angular:

- {{ 3+4 }}
- {{ a+b }}
- {{ user.name }}
- {{ items[index] }}

Q22. How Angular expressions are different from JavaScript expressions?

Ans: Angular expressions are like JavaScript expressions but there are few differences.

- **Context :** In Angular, the expressions are evaluated against a scope object, while the JavaScript expressions are evaluated against the global window object.
- **Forgiving:** In Angular expression evaluation is forgiving to null and undefined, while in JavaScript undefined properties generates TypeError or ReferenceError.
- **No Control Flow Statements:** Loops, conditionals or exceptions cannot be used in an Angular expression.
- **No Comma And Void Operators:** You cannot use , (comma) or void in an Angular expression. And You cannot create regular expressions in an Angular expression.

Q23. What is compilation process in Angular?

Ans: Once you have the markup, the AngularJS needs to attach the functionality. This process is called "**compilation**" in Angular. Compiling includes rendering of markup, replacing directives, attaching events to directives and creating a scope. The AngularJS has compiler service which traverses the DOM looking for attributes. The compilation process happens in two phases.

- **Compilation :** traverse the DOM and collect all of the directives and creation of the linking function.
- **Linking:** combine the directives with a scope and produce a live view. The linking function allows for the attaching of events and handling of scope. Any changes in the scope model are reflected in the view, and any user interactions with the view are reflected in the scope model.

When you create a new directive, you can write compile and/or linking functions for it to attach your custom behavior.

To understand the compilation process of Angular, must read "**The nitty-gritty of compile and link functions inside AngularJS directives**".

Q24. What is scope in AngularJS?

Ans: A scope is an object that ties a view (a DOM element) to the controller. In the MVC framework, scope object is your model. In other words, it is just a JavaScript object, which is used for communication between controller and view.

Q25. What is $rootscope in AngularJS?

Ans: The $rootScope is the top-most scope. An app can have only one $rootScope which will be shared among all the components of an app. Hence it acts like a global variable. All other $scopes are children of the $rootScope. Since $rootScope is a global, which means that anything you add here, automatically becomes available in $scope in all controller. To add something in $rootScope, you need to use app.run function which ensures that it will run prior to the rest of the app. You may say that "run" function is like "main" method of angular app.

```
app.run(function($rootScope) {
  $rootScope.name = "AngularJS";
});
```

And then you can use in your view. (via expression)

```
<body ng-app="myApp">
  <h1>Hello {{ name }}!</h1>
</body>
```

Q26. What are controllers in AngularJS?

Ans: In Angular, a Controller is a JavaScript constructor function. When a Controller is attached to the DOM via the ng-controller directive, Angular will instantiate a new Controller object, using the specified Controller's constructor function. The job of a controller is to pass data from the model, to the view or the view can asks for something from the controller, and the controller turns to the model and takes that thing, and hands it back to the view.

```
var myApp = angular.module('myApp', []);
myApp.controller('MyController', ['$scope', function($scope) {
    $scope.website = 'jquerybyexample.net';
  }
]);
```

And then in your HTML using ng-controller directive,

```
<div ng-controller="MyController">
  <h1>My website address is {{ website }}!</h1>
</div>
```

Q27. What is the difference between $scope and scope in AngularJS?

Ans: $scope is used while defining a controller (see previous question) where scope is used while creating a link function for custom directive. The common part is that they both refers to scope object in AngularJS, but the difference is that $scope uses dependency injection where scope doesn't. When the arguments are passed-in via dependency injection, their position doesn't matter. So for example, a controller defined as ($scope as first parameter)

```
myApp.controller('MyController', ['$scope', function($scope, $http) {
```

OR ($scope is second parameter)

```
myApp.controller('MyController', ['$scope', function($http, $scope) {
```

In both the case, the postion of $scope doesn't matter to AngularJS. Because AngularJS uses the argument name to get something out of the dependency-injection container and later use it.

But in case of link function, the position of scope does matter because it doesn't uses DI. The very first parameter has to be scope and that's what AngularJS expects.

```
app.directive("myDirective", function() {
  return {
    scope: {};
    link: function(scope, element, attrs) {
      // code goes here.
    }
  };
});
```

However you can name it anything of your choice. In below code, foo is your scope object.

```
link: function(foo, bar, baz) {
  // code goes here.
}
```

Q28. What is MVC Architecture in AngularJS?

Ans: In AngularJS, scope objects are treated as **M**odel. The **V**iew is display of model that is your data. And the model gets initialized within a JavaScript constructor function, called **C**ontroller in AngularJS. Let take a look at below code to understand it better.

```
<!DOCTYPE html>
<html>

<head>
  <script data-require="angular.js@*" data-semver="1.3.6"
src="https://code.angularjs.org/1.3.6/angular.js"></script>
  <link rel="stylesheet" href="style.css" />
  <script>
    var myApp = angular.module('myApp', []);
    myApp.controller('MyController', ['$scope',
      function($scope) {
        $scope.website = 'jquerybyexample.net';
      }
    ]);
  </script>
</head>

<body ng-app="myApp">
  <div ng-controller="MyController">
    <h1>My website address is {{ website }}</h1>;
  </div>
</body>
</html>
```

Here MyController is a Controller and $scope (Model) is populated within Controller. And later on in div element $scope object data is displayed (View).

Q29. Can we have nested controllers in AngularJS?

Ans: **YES**. We can create nested controllers in AngularJS. Nested controller are defined in hierarchical manner while using in View. Take a look at below code. Here the hierarchy is "MainCtrl -> SubCtrl -> SubCtrl1".

```
<div ng-controller="MainCtrl">
  <p>{{message}} {{name}}!</p>

  <div ng-controller="SubCtrl">
    <p>Hello {{name}}!</p>

    <div ng-controller="SubCtrl2">
      <p>{{message}} {{name}}! Your username is  {{username}}.</p>
    </div>
  </div>
</div>
```

Q30. In case of nested controllers, does the $scope object shared across all controllers?

Ans: **YES**. The $scope object is shared across all controllers and it happens due to scope inheritance. Since the ng-controller directive creates a new child scope, we get a hierarchy of scopes that inherit from each other. So if we define a property on a parent scope, the child scope can manipulate the property. And if the property is not present in child scope, then parent scope property's value is used. Lets consider, the previous question HTML where there are 3 controllers. And here is the AngularJS code to define these controllers.

```
var myApp = angular.module('myApp', []);
myApp.controller('MainCtrl', ['$scope',
  function($scope) {
    $scope.message = 'Welcome';
    $scope.name = 'Jon';
  }
]);
myApp.controller('SubCtrl', ['$scope',
  function($scope) {
    $scope.name = 'Adams';
  }
]);
myApp.controller('SubCtrl2', ['$scope',
  function($scope) {
    $scope.name = 'Ema';
    $scope.username = 'ema123';
  }
]);
```

You will see that the controller "SubCtrl2" doesn't have "message" property define but it is used in HTML. So in this case, the immediate parent scope property will be used. But immediate parent scope which is "SubCtrl" in this case, also doesn't have "message" property. So it again goes one level up and finds the property is present so it uses parent scope value for "message" property and displays it.

That completes Part-3 as well. If you have not read **Part 1** and **Part 2** then do read and keep visiting for upcoming set of interview questions and answers. You can also subscribe to RSS feed, follow us on Facebook and twitter to get the updates about this blog.

Also read,
Latest AngularJS interview questions - Part 1
Latest AngularJS interview questions - Part 2
Latest AngularJS interview questions - Part 4

You can also Download **jQuery Interview Questions** free eBook.