**JavaScript Developer Questions (C) Time 1 Hour**

1. Angular 2 can detect when component data changes, and then automatically re-render the view to reflect that change?
2. True
3. In Angular 2.0, the template compilation process is ?
4. asynchronous
5. There are two ways to build forms in Angular 2, namely . . nd . . .
6. model-driven b) template-driven
7. We can use the NgFor and NgIf directives using the . . . . . . . symbol to indicate we're dealing with a directive that creates a template

b) \*

1. The three types of directives in Angular 2 are ...?
2. attribute directives b) structural directives c) components d)all
3. Both the following examples involving the directives are identical?
4. true
5. Explain window.onload and onDocumentReady?

The onload function is not run until all the information on the page is loaded. This leads to a substantial delay before any code is executed.

onDocumentReady loads the code just after the DOM is loaded. This allows early manipulation of the code.

1. What is the difference between .call() and .apply()?

- The function .call() and .apply() are very similar in their usage except a little difference. .call() is used when the number of the function’s arguments are known to the programmer, as they have to be mentioned as arguments in the call statement. On the other hand, .apply() is used when the number is not known. The function .apply() expects the argument to be an array.

The basic difference between .call() and .apply() is in the way arguments are passed to the function. Their usage can be illustrated by the given example.

var someObject = {

myProperty : 'Foo',

myMethod : function(prefix, postfix) {

alert(prefix + this.myProperty + postfix);

}

};

someObject.myMethod('<', '>'); // alerts '<Foo>'

var someOtherObject = {

myProperty : 'Bar'

};

someObject.myMethod.call(someOtherObject, '<', '>'); // alerts '<Bar>'

someObject.myMethod.apply(someOtherObject, ['<', '>']); // alerts '<Bar>'

1. Explain the role of deferred scripts in JavaScript?

parsing of the HTML code, during page loading, is paused until the script has not stopped executing. It means, if the server is slow or the script is particularly heavy, then the webpage is displayed with a delay. While using Deferred, scripts delays execution of the script till the time HTML parser is running. This reduces the loading time of web pages and they get displayed faster.

1. Write about the errors shown in JavaScript?

Load-time errors: The errors shown at the time of the page loading are counted under Load-time errors. These errors are encountered by the use of improper syntax, and thus are detected while the page is getting loaded.

Run-time errors: This is the error that comes up while the program is running. It is caused by illegal operations, for example, division of a number by zero, or trying to access a non-existent area of the memory.

Logic errors: It is caused by the use of syntactically correct code, which does not fulfill the required task. For example, an infinite loop.

1. What are the decodeURI() and encodeURI()?

EncodeURl() is used to convert URL into their hex coding. And DecodeURI() is used to convert the encoded URL back to normal.

var uri="my test.asp?name=ståle&car=saab";

document.write(encodeURI(uri)+ "<br>");

document.write(decodeURI(uri));

1. How to Set Http Request Header in Angular?

The HTTP Interceptors are used to Set Http Headers Request in Angular 4 using the import from “@angular/common/http”. The HTTP Interceptors are available in Angular 4.x versions.

The HTTP Interceptors are not supported in Angular 2. We are creating the HttpClient Injectable class to achieve this. You can see the below examples for set http headers request with and without HTTP interceptors.

1. How to apply the CSS class binding in AngularJs?

The ng-class angularjs binding are used to bind the CSS classes in HTML controls with JS binding functions.

<div ng-class="{'background': color.back}">

1. How to use filter nested properties in angularjs object?

- <div ng-controller="ngController">

<input type="text" ng-model="Selected\_CityId" />

<div ng-repeat="country in countries | filter : search">

Country : {{country.name}}

Locations : {{ country.locations }}

</div>

</div>

1. How to configuring routes in angularjs?

Declaring a Dependency in ngRoute Route module

Configuring the $routeProvider

The ng-view directive

$routeParams used to pass the route parameters.

var app = angular.module('ConfigurRoutes', ['ngRoute']);

app.config(function ($routeProvider) {

$routeProvider

.when('/Home', {

templateUrl: 'Views/Home/index.cshtml',

controller: 'HomeCtrl',

controllerAs: 'Home'

})

.when('/Home/Email/:EmailId', {

templateUrl: 'Views/Home/email.cshtml',

controller: 'EmailCtrl',

controllerAs: 'Email'

})

.when('/TermsnCondition/', {

templateUrl: 'Views/Terms/TermsnCondition.cshtml',

controller: 'TermsnConditionCtrl',

controllerAs: 'Terms'

})

.otherwise({

redirectTo: '/Account'

});

});

1. $scope vs scope in angularjs ?

- This is interesting question and little bit confusion about $scope and scope and how to differentiate to in controller and directive both case in Angularjs.

In the case of controller, When you are using $scope in controller.

The injection inject scope based on matching valuable name ($scope). In case when we using scope as name its not work

app.controller("MyController", ['$scope', '$http', function (i, j) {

console.log("$scope : " + i);

console.log("scope : " + j);

}

]);

000000000000000

app.directive("MyDirective", function () {

return {

link: function ($scope, element, attrs, controller) {

console.log($scope);

console.log(element);

}

}

});

1. What is use of ngInit and ngOnInit()?   
   In Angularjs, ngInit is called when template is re-rendered. In other words “ng-init” is called, when I take turns back to a page

- In Angular2, there is no “ng-init” but we can create a ways like this using the directive and ngOnInit class. Angular 2 provides life cycle hook ngOnInit by default

The ngOnInit is invoked when the component is initialized and invoked only once when the directive is instantiated. It is a best practice to implement these life-cycle interfaces.

1. What is use of ngOnInit and constructor ?  
   - The constructor is a default method runs when component is being constructed.

- The constructor is a typescript feature and it is used only for a class instantiations and nothing to do with Angular 2.

- The constructor called first time before the ngOnInit().

-The ngOnInit event is an Angular 2 life-cycle event method that is called after the first ngOnChanges and the ngOnInit method is use to parameters defined with @Input otherwise the constructor is OK.

-The ngOnInit is called after the constructor and ngOnInit is called after the first ngOnChanges.

-The ngOnChanges is called when an input or output binding value changes.

constructor(){

}

ngOnInit(){

}

1. How to create prototype in JavaScript?

The prototype is a fundamental concept of JavaScript and its must to known JavaScript developers and all the JavaScript objects have an object and its property called prototype and its used to add and the custom functions and property.

var employee = function () {

//This is a constructor function.

}

var empInstance = new employee();

empInstance.deportment = "IT";

The example with prototype as given below.

var employee = function () { //This is a constructor function.}

employee.prototype.deportment = "OB";

//Crate the instance of above constructor function and assign in a variable

var empInstance = new employee();

empInstance.deportment = "ROOM";

1. What are Event Emitters and how it works in Angular 2?  
     
   - Angular 2 doesn’t have bi-directional digest cycle, unlike angular 1. In angular 2, any change occurred in the component always gets propagated from the current component to all its children in hierarchy. If the change from one component needs to be reflected to any of its parent component in hierarchy, we can emit the event by using Event Emitter api.

In short, EventEmitter is class defined in @angular/core module which can be used by components and directives to emit custom events.

@output() somethingChanged = new EventEmitter();

We use somethingChanged.emit(value) method to emit the event. This is usually done in setter when the value is being changed in the class.

This event emit can be subscribed by any component of the module by using subscribe method.

myObj.somethingChanged.subscribe(val) => this.myLocalMethod(val));