**Day – 1 (Session - 1)**

1. introduction on Angular JS

**which covers the following**

* What is angular js
* Goals in developing angular.
* Angular features.
* How to add or refer Angular.Js library in your application
* How to create a simple Angular Module
* How to create Angular Application using "ng-app" attribute
* How to attach "ng-app" to angular module
* How to use angular expression

**Highlights.**

* "ng-app" stands for angular application. -It is the root of angular application (starting at the element associated).
* Usually one per web page / app (other workarounds available for multiple "ng-app").
* Cannot be nested.
* Can be attached to any element, but usually to HTML or BODY.
* Usually attached to an angular module (called "root module")
* angular.module = all application logic under one umbrella.
* passing second parameter means, you are creating new module.
* empty square brackets means, the module has no dependencies (to load).
* currently, the "sample" module is attached as "root module" (using ng-app)
* angular.module("sample", []);
* not passing second parameter means, search for and use existing module, which was already created earlier or in another file - angular.module("sample");
* {{ }} = angular expression.
* javascript-like code snippets, evaulated by angular.
* control flow statements, functions etc. NOT allowed

1. Introduction to Controllers.

**which covers the following.**

* How does Angular JS process as Angular Application (life cycle of angular JS).
* What are the steps involved in processing/working with an Angular JS Application
* High-level overview of Angular JS application execution life cycle
* What are templates, compilation, loading, transformation and rendering as part of Angular JS execution process
* What are Angular js controllers and to use them Angular JS
* What is ng-controller attribute and to use it
* What is MVC in Angular JS (Model, View and Controller)
* How to create an Angular module, create a angular controller using a module and finally attach an angular controller to a HTML block
* How to create an instance of angular controller
* How to send information from angular Controller (using a model) to a View
* How to use Evaluating (or binding expressions) in angular View to fetch data from angular Controller.

**Highlights.**

* Nested Controllers
* Sibling controllers.

1. Introduction to Scope

**which covers the following.**

* What is $scope in AngularJS
* How is scope related to controller in Angular JS
* How does controller (work with or) access scope
* How does View/Template access (or evaluate from) scope
* Where does scope stand in AngularJS architecture
* What is difference between "Controler as" and $scope
* Where does scope sit and how does it work like "glue" or mediator between controller and view
* How to access scope in browser developer tools
* How to modify/test/troubleshoot Angular JS scope using developer tools
* How to use methods inside of a Angular JS scope
* How to use multiple controllers in Angular JS
* How to access "Controller as" data using scope in developer tools
* How to use various syntax (or methodologies) to access $scope

**Highlights.**

* Implicit controller instance and using $scope in a controller
* using recommended pattern to use "$scope" (mainly to work even after minimizing js)
* Hybrid syntax oc $scope and "controller as".
* Technically, "controller as" object is automatically made part of $scope

## Scope Inheritance.

**which covers the following.**

* What is Scope Inheritance in Angular JS
* Understanding Scope life-cycle
* How does Inner Scope gets members from Outer Scope in Angular JS
* How does Scope Inheritance works in Angular JS
* Multi-level Scope Inheritance
* Binding members to Outer Scope from Inner Scope (in View)
* Using Outer Scope members in Inner Scope (using Controller)

## Highlights

* Scope relation with controllers and the View (HTML template)
* Use of $0 - to get HTML version of element, angular.element($0) – to get angular version of HTML, , angular.element($0).scope() – to get the scope to visualize
* Use of $digest to manually invoke the digest process

## RootScope.

**which covers the following.**

* What is RootScope in AngularJS
* When is RootScope created in AngularJS
* How does it affect scope inheritance in AngularJS
* How to add members to $rootScope
* How to use $rootScope in Controller (or nested scopes)
* How to use ".run" method as part of module definition

## Highlights

* Rootscope relation with other scopes
* How to share the common properties using rootscope.

1. Data Binding.

**which covers the following.**

* What is Data Binding
* How to use Data Binding features in Angular JS
* Various types of Bidings in Angular JS (One-Way, Two-Way, One-Time)
* How to use "ng-click"
* How to work with "ng-bind" and "ng-model"
* How to use double colon (::) in evaluation expression for one-time binding
* How to push values from Scope to view and vice versa in Angular JS

## Highlights

1. ng-if, ng-show, ng-hide, ng-repeat

**which covers the following.**

* How to do a simple search for records using AJAX together with AngularJS
* How to integrate Bootstrap CSS framework as part of AnguarJS application
* How to use ng-if, ng-show and ng-hide
* Differences between ng-if and ng-show/ng-hide
* How to work with ng-repeat
* How to display multiple records using ng-repeat
* How to work with different collections (Strings, Objects, Maps) for ng-repeat
* How to fetch multiple rows through AJAX and bind using ng-repeat
* How to make using ng-if/ng-show inside ng-repeat
* Inspect various scope object when dealing with ng-if and ng-show

## Highlights

* Scope will be created for ng-if block of HTML, completely remove or render (conditional).
* No scope for ng-show and ng-hide, play with CSS display.
* Scope will be created for each iteration of ng-repeat (looping).

1. Custom Directives

**which covers the following.**

* What are directives.
* Types of directives based on use.
* Use of built in directives (ng-if, ng-show, ng-hide, ng-repeat).
* How to create custom directives using directive method.
* Default restriction of custom directive usage (Element and Attribute).
* Directive Restrictions (Class – C, Element – E, Attribute – A, Comment – M)
* Use of template in directive.
* Use of templateUrl and script tag with type=text/ng-template for directive creation.
* Use of templateUrl and template in other file for directive creation.

## Highlights