Angular framework.

Client side js framework, developed by google

Goals - separate dom manipulation from application logic,

Separation fo concerns (mvc) models are data models, views are html and directives, controllers are js files

Angular.module(‘sample’,[])

Module contains controllers and models means application data.

[] - it creates new module.

No brackets then angular search for existing module.

[‘…’,’…’] adds depencies to new module.

When angular file is attached, it looks for atleast one ng-app, otherwise it won’t work.

Usually one ngapp per page, but we can have many, don’t try nesting it. It is associated with angular module called root module.

<body ng-app=””

Expression {{}} contains variable, expression but not loops

<body ng-app=””>

{{}}

</body>

Templates – html with some angular markup. Ng-\* attributes (directives), evaluating expression.

Process template --

compiles template for errors, loads template in memory, transforms template with data --. Data binding and evaluation of expressions.

Controllers;

Var app = angular.module(‘app’,[]);

App.controller(‘sample’,[‘$scope’,function($scope){ // direct access of scope in controller

// this is definition of controller not instance of controller.

}]);

<div ng-controller=””>

</div>

This creates a memory area called context which hides the data from outside world.

This is attached to dom element. This context is called scope.

Then it creates controller instance and merge scope object into it. Controller updates modal then it is reflected in the view. Scope acts like glue between view and controller.

Any {{variable}} which is not property of scope in controller, angular just skips it, not producing error or undefined.

Angular.element($0).scope()

This give angular version of selected element. Then gives scope object

Angular.element($0).scope().name = ‘ranjan’

Root scope

Var sample = angular.module(‘sample’,[]).run([‘$rootScope’,function($scopeScope){

$rootScope. Taxpercent = 30;

})]).run()….

<https://www.youtube.com/watch?v=oUXku28ex-M&list=PLvZkOAgBYrsS_ugyamsNpCgLSmtIXZGiz&index=8>

Services

A special unit of code which can be used across different controllers. Can be utilities or business logic units.

Service is a singleton

Instantiated only once.

Same instance is maintained thoughout the life time of application. Unlike services, controllers are created a they are needed and destroyed if they are not in use.

Services are loaded and instantiated lazily.

* Only when a controller referring the respective service is instantiated.

We can have multiple services used by multiple controllers.

Services can work with other services too.

Built in services

* Part of angular framework
* Eg. $http, $log, $location etc.

Custom services

* Developed by us.
* Could be extensions to built in services.
* Could be business (logic) services.

3 ways –

* Factory

Var app = angular.module(“app”,[]);

App.controller(‘emp’, function(‘$scope, calcFactory){

…

CalcFactory.doSum();

});

App.factory(‘calcFactory’, function(){

Var oCalcService = {};

oCalcService.doSum = function(){

…

}

Return oCalcService;

})

* Service

Var app = angular.module(“app”,[]);

App.controller(‘emp’, function(‘$scope, calcFactory){

…

CalcFactory.doSum();

});

App.service(‘calcFactory’, function(){

This.oCalcService.doSum = function(){

…

}

});

All methods attached to instance (this) and returned automatically.

* Providers

App.controller(‘emp’, function($scope, calcService){

….

CalcService.doSum();

…

});

App.provider(‘calcService’, function(){

Var baseUrl = ‘’;

This.config = function(url){

baseUrl = url;

};

This.$get = function(){

Var oCalcService = {};

oCalcService.doSum = function() {

};

Return oCalcService;

}

});

App.config([‘calcServiceProvider’, function(calcServiceProvider){

calcServiceProvider.config(‘http://localhost:4467’);

}]);

**Filters**

Any angular filter usually takes input and gives output.

Can be used to :

* Format data
* Transform data
* Search / filer data

Can be used in

* Templates (views)
* Controllers
* Services

Types of filters

* Built in eg. Number, date, filter, uppercase, lowercase, orderby etc.
* Custom

App.controller(‘emp’,[‘$scope’,’$filter’, function($scope, $filter){

$scope.name = $filter(“uppercase”)(data.name);

}])

Angular filters in docs

**Custom Directives**

Various types

* Components
  + Custom element (also called widget)
  + Contains its own html (templates)
* Decorators

To enhance functionality of existing directive

Eg ng-click

* Templating

Eg ng-repeat