**AngularJS Cookie**

<!DOCTYPE html>

<html ng-app="myApp">

<head>

<script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.0-beta.1/angular.min.js"></script>

<script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.0-beta.1/angular-cookies.min.js"></script>

<meta charset="utf-8">

<title>AngularJS Cookies Example</title>

</head>

<body ng-controller="MyController">

{{platformCookie}} {{myFruit}}

<script>

var myApp = angular.module('myApp', ['ngCookies']);

myApp.controller('MyController', ['$scope', '$cookies', '$cookieStore', '$window', function($scope, $cookies, $cookieStore, $window) {

$cookies.userName = 'Sandeep';

$scope.platformCookie = $cookies.userName;

$cookieStore.put('fruit', 'Apple');

$cookieStore.put('flower', 'Rose');

$scope.myFruit = $cookieStore.get('fruit');

}]);

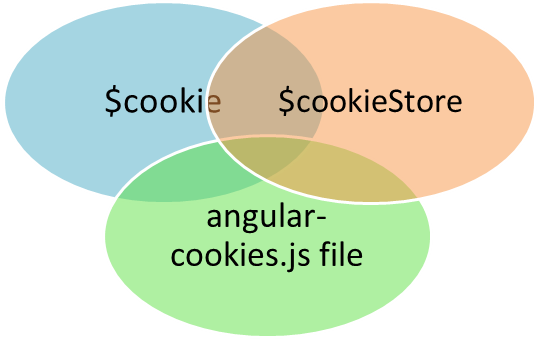
</script>

</body>

</html>

🡺

Sandeep Apple



# AngularJS Message Module



<!DOCTYPE html>

<html ng-app="myApp">

<head>

<script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.0-beta.1/angular.min.js"></script>

<script src="//code.angularjs.org/1.3.0-beta.10/angular-messages.min.js">

</script>

<meta charset="utf-8">

<title>ngMessage Example in AngularJS</title>

</head>

<body>

<form name="studentForm">

<input type="email" name="studentEmail" ng-model="email" required placeholder="Enter email" />

<div ng-messages="studentForm.studentEmail.$error">

<div ng-message="required">Email field is required</div>

<div ng-message="email">Email address is invalid</div>

</div>

</form>

<script>

var appModule = angular.module('myApp', ['ngMessages']);

</script>

</body>

</html>

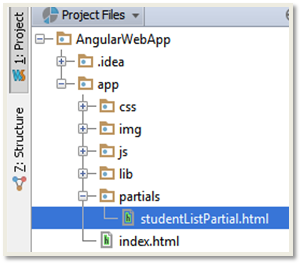
🡺



# Understanding Partials In AngularJS

# "PARTIALS" are Similar to a Angular template.The only difference is there they can be a part of a template.

* The project **structure** will look like,

[](http://www.tutorialsavvy.com/wp-content/uploads/2013/08/structure.png)

* Before Going in Details Let's Discuss some **inbuilt** angular Directives,

**ng-repeat** : This is similar to a **for each loop** which is capable of iterating object present inside an Array.Here in AngularJS template We can use this directive to iterate and construct an array of elements in UI.

**ng-view:**This directive is the key for AngularJS partial rendering.It is responsible for displaying content of a **partial**.

* In This **Demo** We are doing following **steps**:-

***step1***: A **BUTTON** is created inside the  scope of "**StudentController**".

***step2***: a **ng-view** directive element is created inside the  scope of "**StudentController**".

***step3***:BUTTON is linked to CLICK event through **ng-click** directive Which is tied up with a method/function '**showStudentTable**' inside the  scope of "**StudentController**".

***step4***:  '**showStudentTable**' on trigger get the current URL through **$location** global angular variable and [changes](http://www.tutorialsavvy.com/angularjs-require-attribute-directive-development) its path with [an appending](http://www.tutorialsavvy.com/introducing-batarang-angularjs-debugger) string ''**studentsList**".

***step5***: In **app.js** the application router checks the path and **matches** with the supplied condition and founds a match which suggest the PARTIAL file .

***step6***: After this this the **PARTIAL** file is **compiled** and the values of student list array is replaced and send backs to browser which get linked and render in the **ng-view** space.

The partial file **studentListPartial.html** content is as below,

|  |
| --- |
| <table class="table">  <thead>  <tr>  <th>NAME</th>  <th>ROLL</th>  <th>SUBJECT</th>  <th>MARK</th>  <th>AGE</th>  <th>COUNTRY</th>  </tr>  </thead>  <tbody>  <tr ng-repeat="aStudent in studentList">  <td>{{aStudent.name }}</td>  <td>{{aStudent.roll }}</td>  <td>{{aStudent.subject }}</td>  <td>{{aStudent.mark }}</td>  <td>{{aStudent.age }}</td>  <td>{{aStudent.country }}</td>  </tr>  </tbody>  </table> |

The **index.html** file contents are as below**,**

|  |
| --- |
| <!doctype html>  <html lang="en" ng-app="studentDetailApp">  <head>  <title>Student Details App</title>  <link rel="stylesheet" href="css/bootstrap-combined.min.css"/>  <link rel="stylesheet" href="css/app.css"/>  </head>  <body class="student-container">  <div class="row-fluid" ng-controller="StudentController">  <button class='btn btn-info' ng-click="showStudentTable('/studentsList')">Show Student Table</button>  <div ng-view="studenttable"></div>  </div>  <!--Required JS Files List :start -->  <script src="lib/angular/angular.js"></script>  <script src="lib/bootstrap/ui-bootstrap-tpls-0.5.0.min.js"></script>  <script src="js/controllers.js"></script>  <script src="js/app.js"></script>  <!--Required JS Files List :end -->  </body>  </html> |

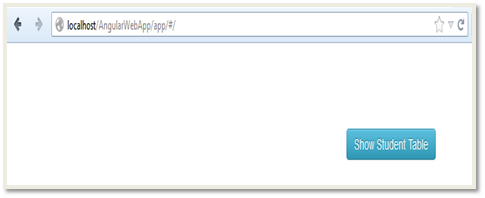
The **controller.js** file contents,

|  |
| --- |
| 'use strict';  /\* Controllers Module for studentDetailApp application\*/  var studentControllerModule = angular.module('studentDetailApp.controllers', ['ui.bootstrap']);  /\*StudentController: controller for students\*/  studentControllerModule.controller('StudentController', function ($rootScope, $scope, $location, $routeParams) {  $scope.studentList = [  {name: "Sandeep", roll: 4, subject: 'Mathematics', mark: 25, age: 23, country: 'India'},  {name: "Hari", roll: 5, subject: 'Geograph', mark: 35, age: 23, country: 'India'},  {name: "Ram", roll: 3, subject: 'History ', mark: 45, age: 23, country: 'India'},  {name: "John", roll: 2, subject: 'Mathematics', mark: 15, age: 25, country: 'UK'},  {name: "Jim", roll: 1, subject: 'Mathematics', mark: 33, age: 23, country: 'UK'},  {name: "Kelly", roll: 6, subject: 'Mathematics', mark: 23, age: 23, country: 'US'}  ];  $scope.showStudentTable = function(pathurl){  console.log(pathurl)  $location.path(pathurl)  }  }); |

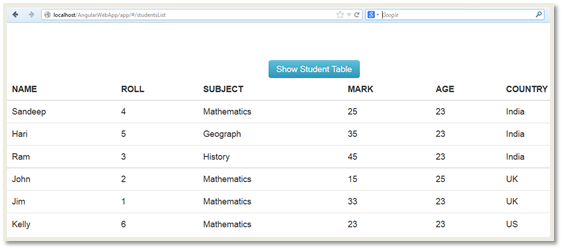
**Finally** We have **app.js** file which have all the **routing** information,

|  |
| --- |
| 'use strict';  angular.module('studentDetailApp', ['studentDetailApp.controllers']).  config(['$routeProvider', function($routeProvider, StudentController) {  $routeProvider.when('/studentsList', {templateUrl: 'partials/studentListPartial.html', controller: 'StudentController'});  $routeProvider.otherwise({redirectTo: '/'});  }]); |

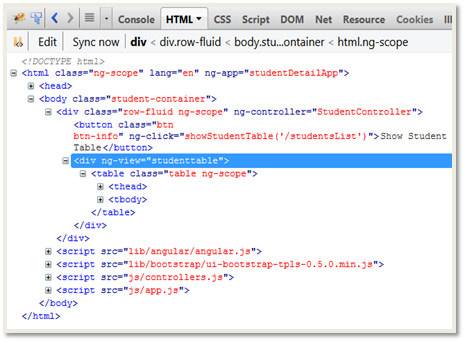
* The **initial outpu**t in browser,

[](http://www.tutorialsavvy.com/wp-content/uploads/2013/08/angular-output-root.png)

* After **Clicking** the **button** the **output** in **Browser** look like,

[](http://www.tutorialsavvy.com/wp-content/uploads/2013/08/angular-output-click.png)

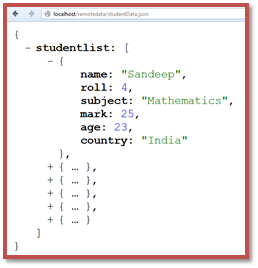
* The **Firebug console** show you the rendered **HTML**,

[](http://www.tutorialsavvy.com/wp-content/uploads/2013/08/angular-firebug-console.png)

# Using Resources In AngularJS

* **AngularJS** provides **$resource** module for handling **remote data**.This module supports **REST** data interaction.
* In the **core** of [AngularJS](http://www.tutorialsavvy.com/angularjs-built-in-filter-example) **$http** module provides core functionality for remote data interaction.

-- For this demo **studentData.json** file is created in a **remotedata** folder and deployed it to our local **WAMP** server.  
-- This **File** can be called by using this **URL**,  
   **http://localhost/remotedata/studentData.json**

[](http://www.tutorialsavvy.com/wp-content/uploads/2013/08/remote-json-file.png)

* "**studentDetailApp.services**" is the module name for all the resources.**"StudentService"** is the **resource** that points to our remote data **URL**.We are [going](http://www.tutorialsavvy.com/angularjs-uigrid-module-example) to use this resource in our controller to get the data.

--- A new resource is created by **factory()** method.In the below code a resource '**StudentService**' is created under the module '**studentDetailApp.services**'.  
--- **$resource** provides different method like **get**, **query**, **save**, **remove** and **delete**.  
---The **services.js** file contains all the code,

|  |
| --- |
| 'use strict';  /\* Services Module for studentDetailApp application \*/ var studentControllerModule = angular.module('studentDetailApp.services', ['ngResource']);  studentControllerModule.factory('StudentService', function($resource){  return $resource('http://localhost/remotedata/studentData.json');  }); |

* The main router module '**studentDetailApp**'  is provided with the **'studentDetailApp.services'** which holds all the resources. **--showStudentTable()** method called by **click** which modifies the **URL** with **/studentsList.** **--routeProvider** determines the appropriate **template** for the **action**.

The **app.js** file contains all the code for it,

|  |
| --- |
| 'use strict';  angular.module('studentDetailApp', ['studentDetailApp.controllers', 'studentDetailApp.services']).   config(['$routeProvider', function($routeProvider, StudentController) {   $routeProvider.when('/studentsList', {  templateUrl: 'partials/studentListPartial.html',  controller: 'StudentController'   });   $routeProvider.otherwise({redirectTo: '/'});   }]); |

* As the service module is present in **app context** the **StudentService** is available for access.**showStudentTable()** method is declared under the scope of the **controller** and called on the **click** of the button.**get()** method is called and when response get available the '**studentlist**' property of the response which hold the array of student.**$scope.studentList** hols the data array from the response and get applied to **view**.

          --- On call of **get()** method it **immediately** return an **empty array [ ]**.  
          ---On **available** of **Asynchronous** response the scope variable got updated with a **full array**.

         The **controller.js** file contains all the code,

|  |
| --- |
| 'use strict';  /\* Controllers Module for studentDetailApp application\*/ var studentControllerModule = angular.module('studentDetailApp.controllers', ['ui.bootstrap']);  /\*StudentController: controller for students\*/ studentControllerModule.controller('StudentController', function ($rootScope, $scope, $location, $routeParams,StudentService) {   $scope.showStudentTable = function(pathurl){  $location.path(pathurl);  StudentService.get(function(result) {  $scope.studentList=result.studentlist;  });  } }); |

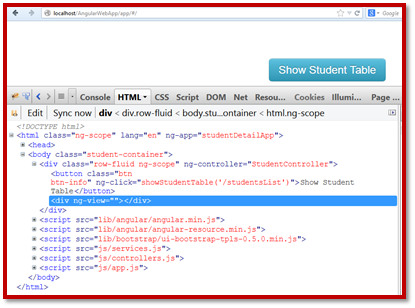
On available of **$scope.studentList** with data the **compiled partial** renders the **data** in table format. The **studentListPartial.html** file contains below code for rendering Student Data ,

|  |
| --- |
| <table class="table">  <thead> <tr>  <th>NAME</th>  <th>ROLL</th>  <th>SUBJECT</th>  <th>MARK</th>  <th>AGE</th>  <th>COUNTRY</th>  </tr> </thead>   <tbody> <tr ng-repeat="aStudent in studentList">  <td>{{aStudent.name }}</td>  <td>{{aStudent.roll }}</td>  <td>{{aStudent.subject }}</td>  <td>{{aStudent.mark }}</td>  <td>{{aStudent.age }}</td>  <td>{{aStudent.country }}</td>  </tr> </tbody> </table> |

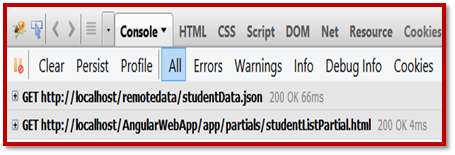
The **index.html** contains below code,

|  |
| --- |
| <!doctype html> <html lang="en" ng-app="studentDetailApp">   <head>  <title>Student Details App</title>  <link rel="stylesheet" href="css/bootstrap-combined.min.css"/>  <link rel="stylesheet" href="css/app.css"/>  </head>   <body class="student-container">   <div class="row-fluid" ng-controller="StudentController">  <button class='btn btn-info' ng-click="showStudentTable('/studentsList')">Show Student Table</button>  <div ng-view></div>  </div>    <!--Required JS Files List :start -->  <script src="lib/angular/angular.min.js"></script>  <script src="lib/angular/angular-resource.min.js"></script>  <script src="lib/bootstrap/ui-bootstrap-tpls-0.5.0.min.js"></script>  <script src="js/services.js"></script>  <script src="js/controllers.js"></script>  <script src="js/app.js"></script>  <!--Required JS Files List :end -->  </body> </html> |

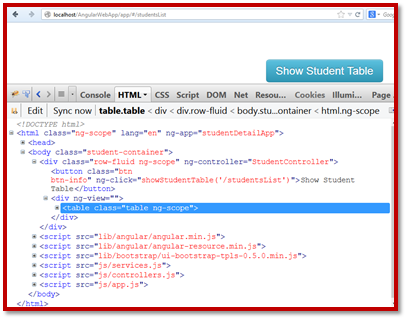
* The  **firebug console** before the click event you can see the **ng-view** is empty.The screenshot is below shows it,

[](http://www.tutorialsavvy.com/wp-content/uploads/2013/08/output-initial-ng-view.png)

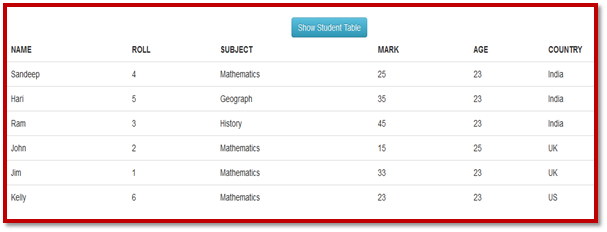
* After the **Button** is **clicked** the remote resource is called in **asynchronous** call.The below screen shot show **two** URL one for remote data and other is for compiled **partial** code in **firebug** console,

[](http://www.tutorialsavvy.com/wp-content/uploads/2013/08/firbug-async-call-console.png)

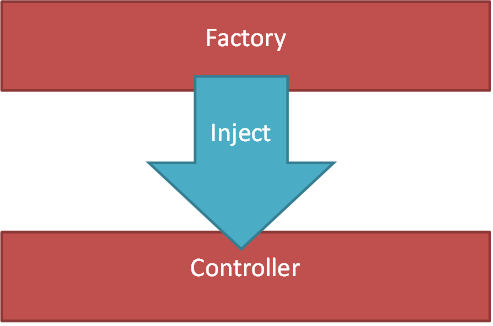
* **After the response** arrived and the data is **available** in scope the view get changed and **ng-view** container gets populated with table rows.Below **screenshot** shows the **ng-view** with the new table element **appended** to it,

[](http://www.tutorialsavvy.com/wp-content/uploads/2013/08/output-populated-ng-view.png)

* The **output** in the **browser** will look  like,

[](http://www.tutorialsavvy.com/wp-content/uploads/2013/08/table-output.png)

# AngularJS Factory Object Development

[](http://www.tutorialsavvy.com/wp-content/uploads/2014/11/angularjs-2Bfactory-2Bmethod.png)

* AngularJS provides **factory()** method to create factory object that can be reused across application.
* A factory object can be **injected** to any module in [AngularJS application](http://www.tutorialsavvy.com/introducing-batarang-angularjs-debugger).

<!DOCTYPE html>

<html ng-app="myApp">

<head>  
<script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.0-beta.1/angular.min.js"></script>  
<meta charset="utf-8">  
<title>JS Bin</title>  
</head>

<body ng-controller="MyController">  
<h3>Original String : {{originalString}}</h3>  
<h3>Reverse String : {{reverseString}}</h3>  
<h3>Character Count : {{characterCount}}</h3>  
<script>  
var myApp = angular.module("myApp", []);

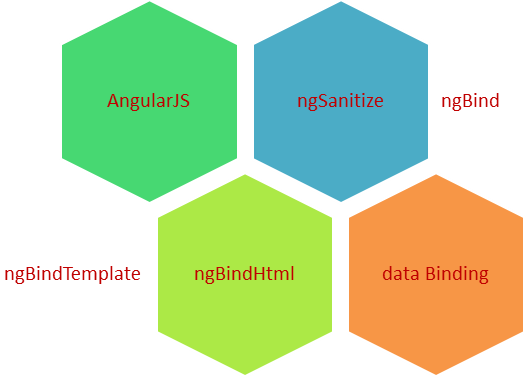
myApp.factory("StringUtil", function() {  
return {  
getReverseString: function(inputString) {  
return inputString.split("").reverse().join("");  
},  
getCharacterCount: function(inputString) {  
return inputString.length;  
}  
};  
});

myApp.controller("MyController", ["$scope", "StringUtil", function($scope, StringUtil) {  
$scope.originalString = "Sandeep Kumar Patel";  
$scope.reverseString = StringUtil.getReverseString($scope.originalString);  
$scope.characterCount = StringUtil.getCharacterCount($scope.originalString);  
}]);  
</script>  
</body>

</html>

|  |
| --- |
| Original String : Sandeep Kumar PatelReverse String : letaP ramuK peednaSCharacter Count : 19 |

# AngularJS ngBind ngBindTemplate ngBindHtml directive Example



* AngularJS provides **ngBind** and **ngBindTemplate** for **TEXT** binding and **ngBindHtml** directive is used for **HTML** binding.
* **ngBindHtml** is present in **ngSanitize m**odule.

<!DOCTYPE html>

<html ng-app="myApp">

<head>

<script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>

<script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular-sanitize.min.js"></script>

<meta charset="utf-8">

<title>AngularJS ngBind ngBindHtml ngBindTemplate Example</title>

</head>

<body ng-controller="MyController">

<h3>ngBind directive Example</h3>

<h4 ng-bind="myName"></h4>

<h3>ngBindTemplate directive Example</h3>

<h4 ng-bind-template="{{myCountry}} {{myCity}}"></h4>

<h3>ngBindHtml directive Example</h3>

<h4 ng-bind-html="myBlog"></h4>

<script>

var myApp = angular.module("myApp", ['ngSanitize']);

myApp.controller("MyController", ["$scope", function($scope) {

$scope.myName = "Sandeep Kumar Patel";

$scope.myBlog = "Chek my Blog <a href='http://www.tutorialsavvy.com'>My Small Tutorial</a>";

$scope.myCity = "Bangalore";

$scope.myCountry = "India";

}]);

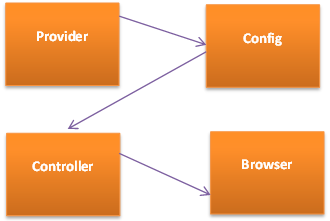
</script>

</body>

</html>

|  |
| --- |
| ngBind directive ExampleSandeep Kumar PatelngBindTemplate directive ExampleIndia BangalorengBindHtml directive ExampleChek my Blog [My Small Tutorial](http://www.tutorialsavvy.com) |

# Understanding AngularJS Provider

[](http://www.tutorialsavvy.com/wp-content/uploads/2014/09/AngularJS-2BProvider.png)

* AngularJS has the feature to create provider which act like a **factory** to provide data to the application.
* [A provider](http://www.tutorialsavvy.com/introducing-batarang-angularjs-debugger) can  be defined using **provider()** function.
* A provider has to implement **$get()** method to be used.

|  |
| --- |
| <!DOCTYPE html>  <html ng-app="studentApp">  <head>  <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.2.14/angular.min.js">  </script>  <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.2.14/angular-route.min.js">  </script>  <meta charset="utf-8">  <title>AngularJS Provider Example</title>  </head>  <body ng-controller="FruitController">  <ol>  <li ng-repeat="fruit in fruits">  {{fruit}}  </li>  </ol>  </body>  </html> |

Below code shows the declaration and use of fruit name provider.

|  |
| --- |
| var studentApp = angular.module("studentApp",[]);  //Declaring a  [provider](http://www.tutorialsavvy.com/beginning-angularjs-animation" \o " provider studentApp)  [studentApp](http://www.tutorialsavvy.com/beginning-angularjs-animation" \o " provider studentApp).provider('fruitNames',function(){  return {  $get: function(){  return {  "list":[  "Apple","Grapes","Orange",  "Banana","Pears","Plum",  "Mango","kiwi","litchi"  ]};  }  };  });  //Injecting the provider to application config  studentApp.config(function(fruitNamesProvider){});  //Using a provider inside controller  studentApp.controller("FruitController",  function ($scope,fruitNames) {  $scope.fruits = fruitNames.list  }); |