



FRONT-END FRAMEWORKS

BÀI 7: BỘ LỘC VÀ DỊCH VỤ

- ⊙ Sử dụng và tạo bộ lọc
- ⊙ Sử dụng và tạo dịch vụ



- 📖 Sử dụng các bộ lọc dựng sẵn
- 📖 Tạo bộ lọc mới
- 📖 Sử dụng các dịch vụ dựng sẵn
- 📖 Tạo dịch vụ mới



- ❑ Bộ lọc trong AngularJS được sử dụng với với mục đích chính
 - ❖ Định dạng dữ liệu
 - ❖ Lọc dữ liệu theo điều kiện
- ❑ AngularJS cung cấp các bộ lọc dựng sẵn đồng thời hướng dẫn cách để bạn tạo ra bộ lọc cho mục đích của riêng mình
- ❑ Ví dụ sau đây họ và tên được định dạng là in HOA

```
<div ng-app="myApp" ng-controller="personCtrl">  
  <p>Họ và tên {{ fullname | uppercase }}</p>  
</div>
```

- ☐ number
 - ❖ Định dạng số
- ☐ currency
 - ❖ Định dạng tiền tệ.
- ☐ date
 - ❖ Định dạng thời gian.
- ☐ lowercase
 - ❖ Định dạng chuỗi in thường.
- ☐ uppercase
 - ❖ Định dạng chuỗi in hoa.
- ☐ json
 - ❖ Định dạng một đối tượng thành chuỗi JSON.
- ☐ limitTo
 - ❖ Giới hạn số phần tử của mảng hoặc số ký tự của chuỗi.
- ☐ orderBy
 - ❖ Sắp xếp các phần tử mảng.
- ☐ filter
 - ❖ Lấy tập con của mảng.

❑ Bộ lọc number được sử dụng để định dạng số

❑ Cú pháp

❖ `{{ value | number : fractionsize }}`

- value: giá trị cần định dạng
- number: bộ lọc định dạng số
- fractionsize: số chữ số thập phân

❑ Bộ lọc currency được sử dụng để định dạng tiền tệ

❑ Cú pháp

❖ `{{ value | currency : symbol : fractionsize }}`

- value: số cần định dạng
- currency: bộ lọc định dạng tiền tệ
- Symbol: đơn vị tiền tệ
- fractionsize: số chữ số thập phân

- Số: 1234.5678
- Định dạng số: 1,234.57
- Định dạng tiền tệ: \$1,234.568



```
<div ng-app="myApp" ng-controller="myCtrl">
  <ul>
    <li>Số: {{mynum}}</li>
    <li>Định dạng số: {{mynum | number : 2}}</li>
    <li>Định dạng tiền tệ: {{mynum | currency : '$' : 3}}</li>
  </ul>
</div>
<script>
  var app = angular.module('myApp', []);
  app.controller('myCtrl', function ($scope) {
    $scope.mynum = "1234.5678";
  });
</script>
```

- ❑ Bộ lọc date được sử dụng để định dạng thời gian
- ❑ Cú pháp
 - ❖ `{{ value | date : format : timezone }}`
 - Value: giá trị cần định dạng
 - Date: bộ lọc định dạng
 - Format: chuỗi định dạng
 - Timezone: múi giờ

- Tự nhiên: "2017-03-02T04:56:39.507Z"
- Định dạng mặc định: Mar 2, 2017
- Định dạng short: 3/2/17 11:56 AM
- Định dạng dd-MM-yyyy: 02-03-2017
- Định dạng HH:mm:ss a: 11:56:39 AM



```
<div ng-app="myApp" ng-controller="myCtrl">
  <ul>
    <li>Tự nhiên: {{now}}</li>
    <li>Định dạng mặc định: {{now | date}}</li>
    <li>Định dạng short: {{now | date : 'short'}}</li>
    <li>Định dạng dd-MM-yyyy: {{now | date : 'dd-MM-yyyy'}}</li>
    <li>Định dạng HH:mm:ss a: {{now | date : 'HH:mm:ss a'}}</li>
  </ul>
</div>
<script>
  var app = angular.module('myApp', []);
  app.controller('myCtrl', function ($scope) {
    $scope.now = new Date();
  });
</script>
```

- ❑ Year
 - ❖ "yyyy"(2016), "yy"(16), "y"(2016)
- ❑ Month
 - ❖ "MMMM"(January), "MMM"(Jan), "MM"(01), "M"(1)
- ❑ Day of month
 - ❖ "dd"(06), "d"(6)
- ❑ Day of week
 - ❖ "EEEE"(Tuesday), "EEE"(Tue)
- ❑ Hour(24)
 - ❖ "HH"(09), "H"(9)
- ❑ Hour(12)
 - ❖ "hh"(09), "h"(9)
- ❑ Minute
 - ❖ "mm"(05), "m"(5)
- ❑ Second
 - ❖ "ss"(05), "s"(5)
- ❑ Millisecond
 - ❖ "sss"(035)
- ❑ AM/PM:
 - ❖ "a"

- ❑ "short" = "M/d/yy h:mm a"
- ❑ "medium" = "MMM d, y h:mm:ss a"
- ❑ "shortDate" = "M/d/yy"
- ❑ "mediumDate" = "MMM d, y"
- ❑ "longDate" = "MMMM d, y"
- ❑ "fullDate" = "EEEE, MMMM d, y"
- ❑ "shortTime" = "h:mm a"
- ❑ "mediumTime" = "h:mm:ss a"

❑ Bộ lọc uppercase, lowercase được sử dụng để định dạng chuỗi in hoa, in thường

❑ Cú pháp

❖ {{ string | uppercase}}, {{ string | lowercase}}

➤ String: giá trị cần định dạng

➤ Uppercase & lowercase: bộ lọc định dạng

❑ Bộ lọc json được sử dụng để định dạng đối tượng thành chuỗi json

❑ Cú pháp

❖ {{ object | json : spacing }}

➤ Object: đối tượng cần định dạng

➤ Json: bộ lọc định dạng

➤ Spacing: số ký tự trống giữa các thuộc tính, mặc định là 2

- Định dạng uppercase: NGUYỄN VĂN TÈO
- Định dạng lowercase: nguyên văn tèo
- Định dạng json: { "id": "TeoNV", "name": "Nguyễn Văn Tèo" }



```
<div ng-app="myApp" ng-controller="myCtrl">
  <ul>
    <li>Định dạng uppercase: {{user.name | uppercase}}</li>
    <li>Định dạng lowercase: {{user.name | lowercase}}</li>
    <li>Định dạng json: {{user | json}}</li>
  </ul>
</div>
<script>
  var app = angular.module('myApp', []);
  app.controller('myCtrl', function ($scope) {
    $scope.user = {
      id: "TeoNV",
      name: "Nguyễn Văn Tèo"
    };
  });
</script>
```



DEMO

Hiển thị thông tin cá nhân

+ Họ và tên: in hoa

+ Ngày sinh: ngay-thang-năm

+ Lương: 3 số lẻ, VNĐ



- ❑ Bộ lọc này dùng để giới hạn số phần tử trong mảng hoặc số ký tự trong chuỗi
- ❑ Cú pháp:
 - ❖ `{{ object | limitTo : limit : begin }}`
 - Object: mảng hoặc chuỗi
 - limitTo: bộ lọc giới hạn số lượng
 - Limit: số phần tử cần lấy.
 - Begin: vị trí bắt đầu lấy. Nếu là số âm thì tính từ cuối về đầu, mặc định tính từ đầu.
- ❑ Ví dụ
 - ❖ `<div ng-repeat="mang | limitTo : 3">`

Phương ở vị trí số 2 (tính từ 0)

- Phương
- Hạnh
- Hoa
- N
- g
- u

```
<div ng-app="myApp" ng-controller="myCtrl">
  <ul>
    <li ng-repeat="item in mang | limitTo : 3 : 2">{{item}}</li>
  </ul>
  <ul>
    <li ng-repeat="item in chuoi | limitTo : 3">{{item}}</li>
  </ul>
</div>
<script>
  var app = angular.module('myApp', []);
  app.controller('myCtrl', function ($scope) {
    $scope.mang = ["Tuấn", "Cường", "Phương", "Hạnh", "Hoa", "Hồng"];
    $scope.chuoi = "Nguyễn Văn Tèo";
  });
</script>
```


❑ | limitTo : 3 : -4

- ❖ Lấy 3 phần tử, bắt đầu phần tử thứ 4 tính từ cuối mảng

❑ | limitTo : 3 : 2

- ❖ Lấy 3 phần tử, bắt đầu phần tử thứ 3

- Phương
- Hạnh
- Hoa

- Phương
- Hạnh
- Hoa

```
<div ng-app="myApp" ng-controller="myCtrl">
  <ul>
    <li ng-repeat="item in mang | limitTo : 3 : -4">{{item}}</li>
  </ul>
  <ul>
    <li ng-repeat="item in mang | limitTo : 3 : 2">{{item}}</li>
  </ul>
</div>
<script>
  var app = angular.module('myApp', []);
  app.controller('myCtrl', function ($scope) {
    $scope.mang = ["Tuấn", "Cường", "Phương", "Hạnh", "Hoa", "Hồng"];
  });
</script>
```



DEMO

Tạo mảng 10 nhân viên

+ Họ và tên

+ Ngày sinh

+ Lương

Hiển thị các nhân viên từ 3 đến 7



❑ Bộ lọc orderBy được sử dụng để sắp xếp các phần tử mảng khi xuất ra

❑ Cú pháp:

❖ `{{ array | orderBy : expression : reverse }}`

- Array: mảng cần sắp xếp
- orderBy: bộ lọc sắp xếp
- Expression: biểu thức sắp xếp
- Reverse: sắp xếp tăng (true, mặc định), giảm (false)

❑ Ví dụ:

❖ `<div ng-repeat="mang | orderBy">`

Các phần tử đã được
sắp xếp tăng dần

- Cường
- Hoa
- Hạnh
- Hồng
- Phương
- Tuấn

```
<div ng-app="myApp" ng-controller="myCtrl">
  <ul>
    <li ng-repeat="item in mang | orderBy">{{item}}</li>
  </ul>
</div>
<script>
  var app = angular.module('myApp', []);
  app.controller('myCtrl', function ($scope) {
    $scope.mang = ["Tuấn", "Cường", "Phương", "Hạnh", "Hoa", "Hồng"];
  });
</script>
```

```
<div ng-app="myApp" ng-controller="myCtrl">
  <table border="1">
    <tr>
      <th>Name</th>
      <th>Mark</th>
    </tr>
    <tr ng-repeat="sv in students | orderBy : '-mark'">
      <td>{{sv.name}}</td>
      <td>{{sv.mark}}</td>
    </tr>
  </table>
</div>
<script>
  var app = angular.module('myApp', []);
  app.controller('myCtrl', function ($scope) {
    $scope.students = [
      { name: "Tuấn", mark: 7 },
      { name: "Cường", mark: 9 }
    ]
  });
</script>
```

Giảm dần theo thuộc tính mark

Name	Mark
Cường	9
Tuấn	7



DEMO

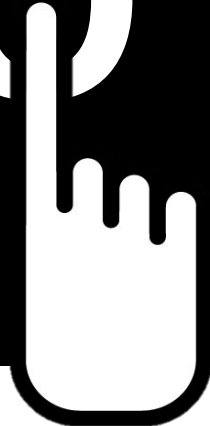
Tạo mảng 10 nhân viên

+ Họ và tên

+ Ngày sinh

+ Lương

Hiển thị các nhân viên giảm theo lương





FRONT-END FRAMEWORKS

BÀI 7 (PHẦN 2)

❑ Bộ lọc này được sử dụng để lọc các phần tử so khớp với biểu thức lọc

❑ Cú pháp

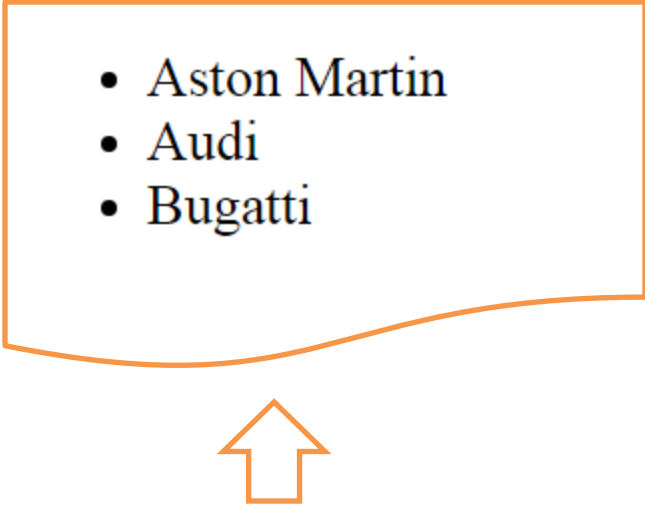
❖ `{{ array | filter : expression : comparator }}`

- Array: mảng cần lọc
- Filter: bộ lọc
- Expression: biểu thức lọc
- Comparator: hình thức so sánh, true là chính xác, false là chứa

❑ Ví dụ

❖ `<li ng-repeat="x in cars | filter : 'A'">{{x}}`

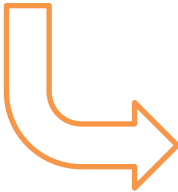
- Hiển thị xe hơi chứa chữ ký tự 'A'

- 
- Aston Martin
 - Audi
 - Bugatti

```
<div ng-app="myApp" ng-controller="myCtrl">
  <ul>
    <li ng-repeat="x in cars | filter : 'A'">{{x}}</li>
  </ul>
</div>
<script>
  var app = angular.module('myApp', []);
  app.controller('myCtrl', function ($scope) {
    $scope.cars = ["Aston Martin", "Audi", "Bentley", "BMW", "Bugatti"];
  });
</script>
```

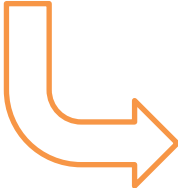
```
<div ng-app="myApp" ng-controller="myCtrl">
  <ul>
    <li ng-repeat="x in customers | filter : 'London' : true">
      {{x.name + ", " + x.city}}
    </li>
  </ul>
</div>
<script>
  var app = angular.module('myApp', []);
  app.controller('myCtrl', function ($scope) {
    $scope.customers = [
      { name: "Food", city: "London" },
      { name: "London", city: "Nework City" },
      { name: "Travel", city: "Heathrow, London" }
    ];
  });
</script>
```

Lọc chính xác với tất cả các thuộc tính

- 
- Food, London
 - London, Nework City

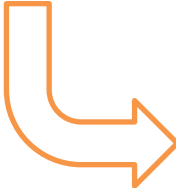
```
<div ng-app="myApp" ng-controller="myCtrl">
  <ul>
    <li ng-repeat="x in customers | filter : 'London' : false">
      {{x.name + ", " + x.city}}
    </li>
  </ul>
</div>
<script>
  var app = angular.module('myApp', []);
  app.controller('myCtrl', function ($scope) {
    $scope.customers = [
      { name: "Food", city: "London" },
      { name: "London", city: "Nework City" },
      { name: "Travel", city: "Heathrow, London" }
    ];
  });
</script>
```

Lọc tương đối với tất cả các thuộc tính

- 
- Food, London
 - London, Nework City
 - Travel, Heathrow, London

```
<div ng-app="myApp" ng-controller="myCtrl">
  <ul>
    <li ng-repeat="x in customers | filter : {name:'O', city:'London'}">
      {{x.name + ", " + x.city}}
    </li>
  </ul>
</div>
<script>
  var app = angular.module('myApp', []);
  app.controller('myCtrl', function ($scope) {
    $scope.customers = [
      { name: "Food", city: "London" },
      { name: "London", city: "Nework City" },
      { name: "Travel", city: "Heathrow, London" }
    ];
  });
</script>
```

Lọc tương đối với
nhiều thuộc tính

- 
- Food, London



DEMO

Tạo mảng 10 nhân viên

+ Họ và tên

+ Ngày sinh

+ Lương

Hiển thị các nhân viên tên Hằng



- ❑ AngularJS cho phép bạn định nghĩa các bộ lọc để thực hiện các định dạng riêng của mình
- ❑ Cú pháp

{expression | myFilter : opt1 : opt2}

Sử dụng

```
var app = angular.module('myApp', []);  
app.filter('myFilter', function () {  
    return function (input, opt1, opt2)  
        // code định dạng  
        return output;  
    };  
});
```

Input: giá trị cần lọc
Output: giá trị đã lọc
Opt1, opt2: tham số bổ sung (không bắt buộc)

Nguyễn Văn Tèo

```
<div ng-app="myApp" ng-controller="myCtrl">
  <h3>{{name | pretty}}</h3>
</div>
<script>
  var app = angular.module('myApp', []);
  app.filter('pretty', function () {
    return function (input) {
      var output = input.split(" ");
      for (i = 0; i < output.length; i++) {
        var s = output[i].trim();
        output[i] = s.charAt(0).toUpperCase()
          + s.substr(1).toLowerCase();
      }
      return output.join(" ");
    };
  });
  app.controller('myCtrl', function ($scope) {
    $scope.name = "nguyễn VĂN tèo";
  });
</script>
```

```

<div ng-app="myApp" ng-controller="myCtrl">
  <ul>
    <li ng-repeat="num in numbers | between : 100 : 150">
      {{num}}
    </li>
  </ul>
</div>
<script>
  var app = angular.module('myApp', []);
  app.filter('between', function () {
    return function (input, min, max) {
      var output = [];
      for (i = 0; i < input.length; i++) {
        if (input[i] >= min && input[i] <= max) {
          output.push(input[i]);
        }
      }
      return output;
    };
  });
  app.controller('myCtrl', function ($scope) {
    $scope.numbers = [100, 5, 200, 250, 150, 107];
  });
</script>

```



- 100
- 150
- 107



DEMO

Tạo mảng 10 nhân viên

+ Họ và tên

+ Ngày sinh

+ Lương

Hiển thị các nhân viên lương > 6000



- ❑ Trong AngularJS, dịch vụ là một hàm hoặc một đối tượng cung cấp các chức năng xử lý phía hậu trường nhằm hỗ trợ cho lập trình AngularJS
 - ❖ \$location: là một đối tượng
 - ❖ \$interval: là 1 hàm
 - ❖ \$http: là một đối tượng
 - ❖ ...
- ❑ AngularJS cung cấp sẵn rất nhiều dịch vụ đồng thời cũng cung cấp cách thức để bạn có thể xây dựng dịch vụ mới của riêng mình.
- ❑ Để sử dụng một dịch vụ, bạn chỉ cần bổ sung nó vào danh sách đối số của phương thức controller

- ❑ Dịch vụ \$interval là một hàm tương tự setInterval() của window
- ❑ Ví dụ sau sử dụng \$interval để tạo đồng hồ trên trang web. Để sử dụng dịch vụ này trước hết phải truyền thông qua đối số của hàm controller

```
<div ng-app="myApp" ng-controller="myCtrl">
  <h3>{{time | date : 'hh:mm:ss a'}}</h3>
</div>
<script>
  var app = angular.module('myApp', []);
  app.controller('myCtrl', function ($scope, $interval) {
    $scope.time = new Date();
    $interval(function () {
      $scope.time = new Date();
    }, 1000);
  });
</script>
```

05:07:38 PM

URL: http://localhost:50125/Bai7/Service.html

Reload

```
<div ng-app="myApp" ng-controller="myCtrl">
  <h3>URL: {{url}}</h3>
  <button ng-click="reload()">Reload</button>
</div>
<script>
  var app = angular.module('myApp', []);
  app.controller('myCtrl', function ($scope, $location, $window) {
    $scope.url = $location.absUrl();
    $scope.reload = function () {
      $window.location.reload();
    }
  });
</script>
```



DEMO

Sử dụng `$interval`, `$window`
làm cho trang web tự động tải
lại sau mỗi 5 giây



- ❑ Dịch vụ \$http giúp tạo các yêu cầu đọc dữ liệu từ server
- ❑ Phương thức get được sử dụng để đọc dữ liệu từ server
- ❑ Cú pháp

```
$http.get(url).then( tải dữ liệu từ url  
    function (response) {}, thực hiện sau khi tải dữ liệu  
    function (response) {} thực hiện nếu không tải được  
);
```

```
[  
  { "Id": 1000, "Name": "Nokia" },  
  { "Id": 1001, "Name": "Samsung" },  
  { "Id": 1002, "Name": "Apple" },  
  { "Id": 1003, "Name": "Dell" },  
  { "Id": 1004, "Name": "Sony" },  
  { "Id": 1005, "Name": "Canon" },  
  { "Id": 1006, "Name": "Seamen" }  
]
```

Subjects:

- 1000 - Nokia
- 1001 - Samsung
- 1002 - Apple
- 1003 - Dell
- 1004 - Sony
- 1005 - Canon
- 1006 - Seamen

- ❑ Giả sử có file dữ liệu json có cấu trúc như trên đặt trên server (bên trái)
- ❑ Sử dụng dịch vụ \$http để tải dữ liệu này và hiển thị lên trang web (bên phải)

```
<div ng-app="myApp" ng-controller="myCtrl">
  <p>Subjects:</p>
  <ul>
    <li ng-repeat="s in suppliers">
      {{s.Id}} - {{s.Name}}
    </li>
  </ul>
</div>
<script>
  var app = angular.module('myApp', []);
  app.controller('myCtrl', function ($scope, $http) {
    $scope.suppliers = [];
    $http.get("Suppliers.js").then(function (response) {
      $scope.suppliers = response.data;
    }, function (response) {
      alert("Lỗi");
    });
  });
</script>
```

Truyền dịch vụ

Dữ liệu nhận được từ server

- ❑ Bên cạnh `$http.get()` để đọc dữ liệu từ server, dịch vụ `$http` còn cung cấp các phương thức khác để làm việc với Web API.
- ❑ Sau đây là một số phương thức thường dùng khác
 - ❖ `$http.get(url, config)`
 - ❖ `$http.post(url, data, config).then(f1, f2)`
 - ❖ `$http.put(url, data, config) .then(f1, f2)`
 - ❖ `$http.delete(url, config) .then(f1, f2)`
 - Trong đó `f1` là hàm xử lý thành công và `f2` là hàm xử lý lỗi.
 - Cú pháp của `f1` và `f2` là `function(response){}` với `response` chứa thông tin phản hồi từ server
- ❑ Nếu bạn nắm một trong những công nghệ lập trình phía server (PHP, Java, ASP.NET, MVC...) bạn có thể xây dựng các API. Khi đó `$http` sẽ tương tác rất thuận lợi.

❑ Response chứa dữ liệu phản hồi từ server có cấu trúc như sau

❖ data

➤ Dữ liệu chính (chuỗi hoặc json data)

❖ status

➤ Mã trạng thái

❖ statusText

➤ Mô tả trạng thái

❖ headers

➤ Thông tin về header

❖ config

➤ cấu hình của request đã yêu cầu để có response

- ❑ Bên cạnh rất nhiều dịch vụ được dựng sẵn trong AngularJS, bạn cũng có thể tự định nghĩa các dịch vụ cho riêng mình
- ❑ Có 2 cách định nghĩa dịch vụ mới
 - ❖ Sử dụng factory
 - ❖ Sử dụng service

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

app.factory('$tên dịch vụ', function () {
    return function (tham số) {
        // code xử lý
    };
});
```

- ❑ Như vậy bạn chỉ cần đặt tên cho dịch vụ và viết hàm thực hiện chức năng của dịch vụ
- ❑ Ví dụ sau tạo dịch vụ \$add thực hiện tính tổng 2 số

```
var app = angular.module('myApp', []);
app.factory('$add', function () {
    return function (a, b) {
        return parseFloat(a) + parseFloat(b);
    };
});
```

- Sau khi dịch vụ đã được tạo thì được sử dụng như những dịch vụ của AngularJS, nghĩa là truyền dịch vụ vào các hàm controller và sử dụng

```
<div ng-app="myApp" ng-controller="myCtrl">
  <input ng-model="num1" /><input ng-model="num2" />
  <button ng-click="sum()">=</button>
  <input ng-model="total" />
</div>
<script>
  var app = angular.module('myApp', []);
  app.factory('$add', function () {
    return function (a, b) {
      return parseFloat(a) + parseFloat(b);
    };
  });
  app.controller("myCtrl", function ($scope, $add) {
    $scope.sum = function () {
      $scope.total = $add($scope.num1, $scope.num2);
    };
  })
</script>
```

5	7	=	12
---	---	---	----

- ❑ Chỉ việc thay lời gọi phương thức `factory()` bằng lời gọi phương thức `service()`
- ❑ Cách sử dụng dịch vụ là như nhau

```
app.factory('$add', function () {  
    return function (a, b) {  
        return parseFloat(a) + parseFloat(b);  
    };  
});
```



```
app.service('$add', function () {  
    return function (a, b) {  
        return parseFloat(a) + parseFloat(b);  
    };  
});
```

- ❑ Trên đây chúng ta đã tạo một dịch vụ \$add là một hàm thực hiện việc tính tổng 2 số.
- ❑ Đôi khi một dịch vụ không đơn thuần là một hàm mà là một đối tượng nghĩa là nó có thể chứa các trường dữ liệu và nhiều phương thức

```
var app = angular.module('myApp', []);
app.service('$calculator', function () {
    return new function () {
        this.add = function(a, b) {
            return parseFloat(a) + parseFloat(b);
        }
        this.sub = function (a, b) {
            return parseFloat(a) - parseFloat(b);
        }
    };
});
```

Dịch vụ **\$calculator** có 2 phương thức

- **add()**
- **sub()**

- ☑ Sử dụng bộ lọc sẵn có
- ☑ Tạo và sử dụng bộ lọc tùy biến
- ☑ Sử dụng dịch vụ sẵn có
- ☑ Tạo và sử dụng dịch vụ tùy biến





Cảm ơn