

# AngularJS



JavaScript  
Courses

[vk.com/js.courses](https://vk.com/js.courses)

[js.courses.dp.ua/files](https://js.courses.dp.ua/files)

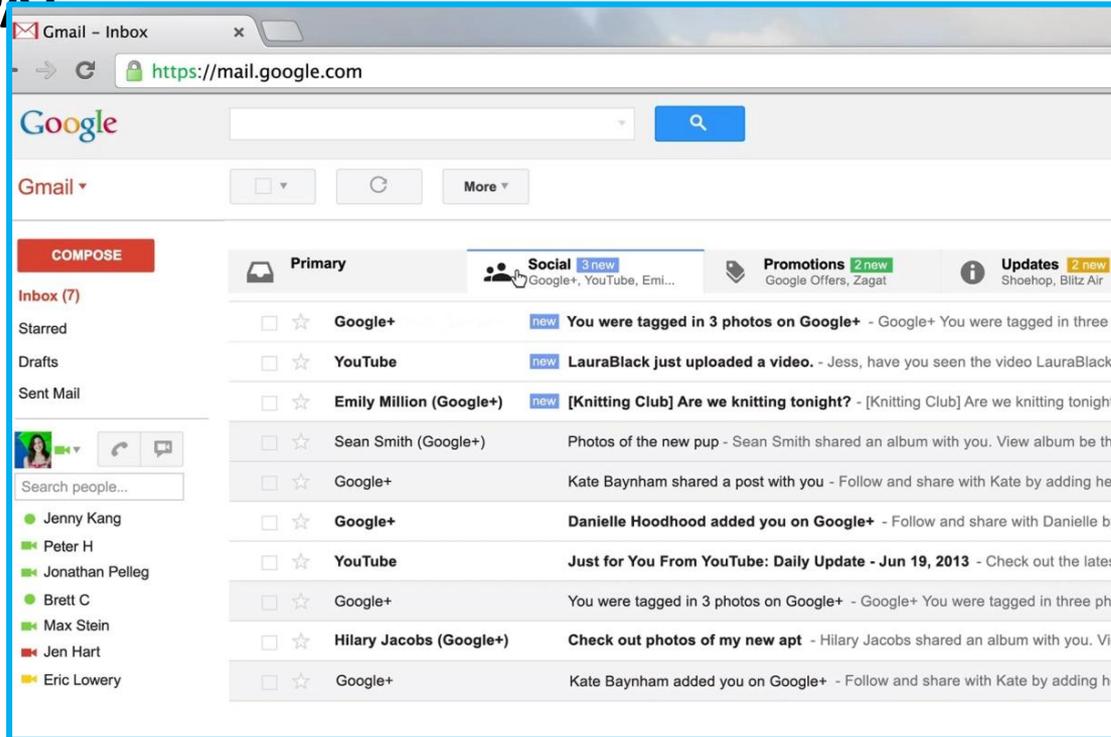
# Еще раз про прототипы, или как библиотеки обрачивают объекты

```
1 <html>
2 <head>
3   <script>
4     function $$ (element) {
5       element.make_red = function () {
6         this.style.color = "red";
7         return this;
8       };
9       return element;
10    }
11
12    window.onload = function () {
13      var h1 = document.querySelector("h1");
14      var div = document.querySelector("div");
15
16      $$ (h1).make_red ();
17      div.make_red ();
18    }
19  </script>
20 </head>
21 <body>
22   <h1>Header</h1>
23   <div>Text</div>
24 </body>
25 </html>
```



Как js-библиотеки обрачивает своими функциями объекты

# Одностраничное приложение (single page application - SPA)



*Gmail*  
типичный  
пример SPA

*SPA – это веб-приложение, размещенное на одной странице, которая для обеспечения работы загружает все JS и CSS файлы вместе с загрузкой самой страницы (или по ходу работы AJAX-ом).*

[https://ru.wikipedia.org/wiki/Одностраничное приложение](https://ru.wikipedia.org/wiki/Одностраничное_приложение)

ие

## Императивный подход в программирование и декларативный

**Императивный подход** – состоит в описании того как достигнуть требуемый результат, т.е. необходимо написать последовательность действий, выполнив которую будет получен результат. **JavaScript** – императивный язык.

**Декларативный подход** – состоит в описании того что необходимо получить, оставив порядок выполнения и способ достижения результата «на усмотрение» компьютера. **HTML&CSS** – представители декларативного подхода.

# AngularJS

## ПОПЫТКА ВНЕСТИ В JS ДЕКЛАРАТИВНЫЕ

```
<script src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js">
```

*AngularJS – Javascript библиотека, и как любая JS-библиотека не привносит ничего такого, чего нельзя написать на чистом JS, но как и все библиотеки направлена на уменьшение объема кода который пишет разработчик*



v.1

<https://angularjs.org/>

v.2 Beta

# Директивы – основа

## AngularJS

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <script src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script>
5 </head>
6 <body ng-app="">
7   Ваше имя: <input type="text" ng-model="user_name">
8   <h1>Привет {{user_name}}!</h1>
9 </body>
10 </html>
```

???

Ваше имя:

**Привет Елена!**

```
Elements Console Sources Network Timeline Profiles Resources Security
<!DOCTYPE html>
<html>
  <head>...</head>
  <body ng-app class="ng-scope">
    "
    Ваше имя: "
    <input type="text" ng-model="user_name" class="ng-valid ng-dirty ng-valid-
    parse ng-touched">
    <h1 class="ng-binding">Привет Елена!</h1>
  </body>
</html>
```

Директива **ng-app** – определяет тег в рамках которого будет «крутиться» Angular приложение.

Директива **ng-model** – определяет имя под которым будет доступного значение из поля ввода (модель).

**{{x}}** – директива вставляет данные из модели в тело документа.

# Связка данных (data

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <script src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js">
  </script>
5 </head>
6 <body ng-app="" ng-init="price=75">
7   Выберите цену: <input type="range" ng-model="price">
8   <h1>Цена: {{price}}$</h1>
9 </body>
10 </html>
```



Выберите цену:

**Цена: 38\$**

Elements Console Sources Network Timeline Profiles Resources Security

```
<!DOCTYPE html>
<html>
  <head>...</head>
  <body ng-app ng-init="price=75" class="ng-scope">
    "
    Выберите цену: "
    <input type="range" ng-model="price" class="ng-untouched ng-valid ng-dirty
ng-valid-parse">
    <h1 class="ng-binding">Цена: 38$</h1>
  </body>
</html>
```

**ng-init** – директива задающая стартовые значения для модели.

# Связка данных (data)

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <script src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js">
  </script>
5 </head>
6 <body ng-app="" ng-init="price=75; quantity=1">
7   Цена: <input type="range" ng-model="price"><span> {{price}}$</span>
8   <br>
9   Кол-во: <input type="range" ng-model="quantity"> <span>{{quantity}} ед.</span>
10  <h1>Итого: {{price*quantity}}$</h1>
11 </body>
12 </html>
```

Цена:  57\$  
Кол-во:  25 ед.

**Итого: 1425\$**

Elements Console Sources Network Timeline Profiles Resources Security Audits

```
<!DOCTYPE html>
<html>
  <head>...</head>
  <body ng-app ng-init="price=75; quantity=1" class="ng-scope">
    "
    Цена: "
    <input type="range" ng-model="price" class="ng-valid ng-dirty ng-valid-parse ng-touched">
    <span class="ng-binding"> 57$</span>
    <br>
    "
    Кол-во: "
    <input type="range" ng-model="quantity" class="ng-valid ng-dirty ng-valid-parse ng-touched">
    <span class="ng-binding">25 ед.</span>
    <h1 class="ng-binding">Итого: 1425$</h1>
  </body>
```



# Связка данных (data

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <style>
5     div {
6       width: 100px;
7       height: 100px;
8       background: rgb(0, 0, 0)
9     }
10  </style>
11  <script src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js" ></script>
12 </head>
13 <body ng-app="" ng-init="color_r='0'; color_g='0'; color_b='0'">
14
15   R<input type="range" min="0" max="255" ng-model="color_r" >
16   G<input type="range" min="0" max="255" ng-model="color_g" >
17   B<input type="range" min="0" max="255" ng-model="color_b" >
18
19 <div style="background: rgb({{color_r}},{{color_g}}, {{color_b}})"></div>
20 </body>
21 </html>
```

**ColorPicker** с тонкой настройкой без единой строчки JS-кода.

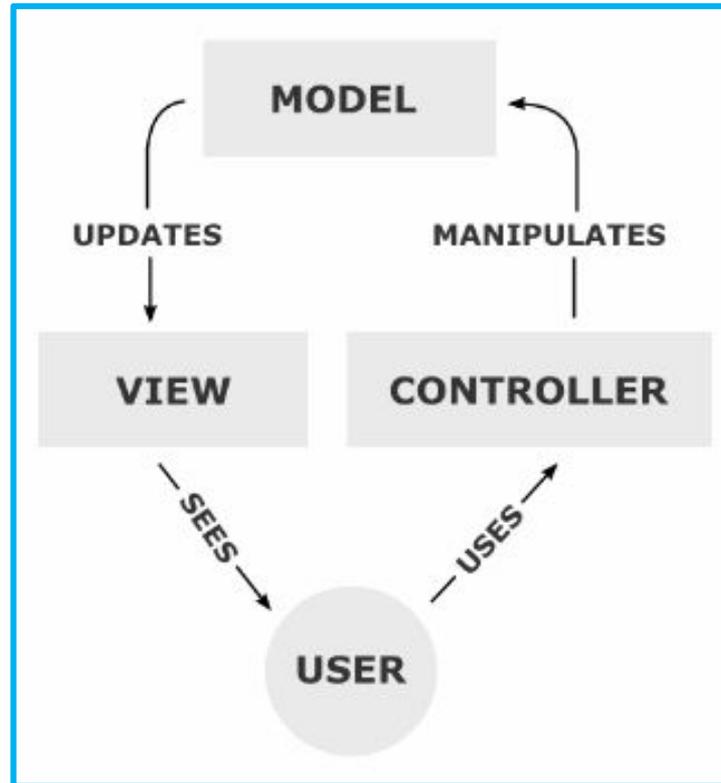
# Связка данных (data

The screenshot shows a web browser window with a single tab titled 'cube.html'. The address bar shows the file path: 'file:///D:/OneDrive/Courses/JS\_Feb\_2016/Angular/cube.html'. The browser's developer tools are open, displaying the 'Elements' panel. The selected element is the body of the HTML document, which contains three range inputs for color selection (R, G, B) and a div with a light green background. The 'Styles' panel on the right shows the default user agent styles for the body element, including 'display: block;' and 'margin: 8px;'. A visual representation of the box model is shown at the bottom of the styles panel, indicating a 790 x 118 pixel area with 8px margins.

```
<!DOCTYPE html>
<html>
  <head>...</head>
  <body ng-app ng-init="color_r='0'; color_g='0'; color_b='0'" class="ng-scope">
    "
    R"
    <input type="range" min="0" max="255" ng-model="color_r" class="ng-valid ng-dirty ng-
    valid-parse ng-touched">
    "
    G"
    <input type="range" min="0" max="255" ng-model="color_g" class="ng-valid ng-dirty ng-
    valid-parse ng-touched">
    "
    B"
    <input type="range" min="0" max="255" ng-model="color_b" class="ng-valid ng-dirty ng-
    valid-parse ng-touched">
    <div style="background: rgb(181,225, 105)"></div>
  </body>
</html>
```

ColorPicker с тонкой настройкой без единой строки JS-кода.

# Model-view-controller (MVC, «модель-представление-контроллер», «модель-вид-контроллер»).



*Основная цель применения этой концепции состоит в отделении бизнес-логики (модели) от её визуализации (представления, вида).*

# Вывод данных (директива)

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Phone Filter.Angular.JS</title>
5   <script src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script>
6   <script>
7     var phones_arr = [
8       { "name": "iPhone 3Gs", "price": "582$" },
9       { "name": "Lenovo", "price": "997$" },
10      { "name": "HTC", "price": "698$" },
11      { "name": "Samsung", "price": "242$" },
12      { "name": "Fly", "price": "793$" },
13      { "name": "Asus", "price": "771$" },
14      { "name": "Nokia", "price": "108$" },
15      { "name": "LG", "price": "214$" },
16      { "name": "IPhone 5s", "price": "216$" },
17      { "name": "IPhone 6+", "price": "326$" },
18      { "name": "IPhone 6s+", "price": "799$" }
19    ];
20
21    var app = angular.module('PhoneFilter', []);
22    app.controller('mainController', function($scope){
23      $scope.phones = phones_arr;
24    });
25  </script>
26 </head>
27 <body ng-app="PhoneFilter" ng-controller="mainController">
28   <h1>Каталог телефонов:</h1>
29   <hr>
30   <div>
31     <div ng-repeat="phone in phones">{{phone.name}} <span>{{phone.price}}</span></div>
32   </div>
33 </body>
34 </html>
```

A

???

Возьмите шаблон с:

[http://js.courses.dp.ua/files/angular/ex03\\_clear.html](http://js.courses.dp.ua/files/angular/ex03_clear.html)

# Вывод данных (директива)

The screenshot shows a web browser window with the title "Каталог телефонов:". The page content lists several phone models and their prices: iPhone 3Gs 582\$, Lenovo 997\$, HTC 698\$, Samsung 242\$, Fly 793\$, and Asus 771\$. Below the page content, the browser's developer tools are open to the "Elements" panel. The DOM tree shows the following structure:

```
<!DOCTYPE html>
<html>
  <head>...</head>
  <body ng-app="PhoneFilter" ng-controller="mainController" class="ng-scope">
    <h1>Каталог телефонов:</h1>
    <hr>
    <div>
      <!-- ngRepeat: phone in phones -->
      <div ng-repeat="phone in phones" class="ng-binding ng-scope">...</div>
      <!-- end ngRepeat: phone in phones -->
      <div ng-repeat="phone in phones" class="ng-binding ng-scope">...</div>
      <!-- end ngRepeat: phone in phones -->
      <div ng-repeat="phone in phones" class="ng-binding ng-scope">...</div>
      <!-- end ngRepeat: phone in phones -->
      <div ng-repeat="phone in phones" class="ng-binding ng-scope">...</div>
      <!-- end ngRepeat: phone in phones -->
      <div ng-repeat="phone in phones" class="ng-binding ng-scope">...</div>
      <!-- end ngRepeat: phone in phones -->
      <div ng-repeat="phone in phones" class="ng-binding ng-scope">...</div>
      <!-- end ngRepeat: phone in phones -->
    </div>
  </body>
</html>
```

The "Styles" panel on the right shows the computed style for the selected element, including:

```
element.style {
}
user agent stylesheet
body {
  display: block;
  margin: 8px;
}
```

A visual representation of the margin and padding is shown below the styles, with a margin of 8px and a padding of 8px. The element's dimensions are 775 x 266.438px.

*Директива **ng-repeat** позволяет многократно повторить вывод фрагмента разметки, вставляя в неё данные из указанного массива.*

# Асинхронная загрузка и вывод

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Phones Filter.Angular.JS</title>
5   <script src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script>
6   <script>
7     → var json_url = 'http://js.courses.dp.ua/files/angular/get_phones_json.php';
8       var app = angular.module('PhoneFilter', []);
9     → app.controller('mainController', function($scope, $http){
10      → $http.get(json_url).then(function(res) {
11        → $scope.phones = res.data;
12      });
13    });
14  </script>
15 </head>
16 <body ng-app="PhoneFilter" ng-controller="mainController">
17   <h1>Каталог телефонов:</h1>
18   <hr>
19   <div>
20     <div ng-repeat="phone in phones">{{phone.name}} <span>{{phone.price}}</span></div>
21   </div>
22 </body>
23 </html>
```

В

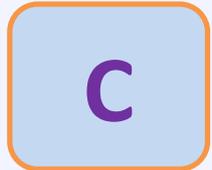
???

**\$http** – объект для выполнения ajax запросов.

[http://js.courses.dp.ua/files/angular/get\\_phones\\_json.php](http://js.courses.dp.ua/files/angular/get_phones_json.php)

# Фильтрация

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Phones Filter.Angular.JS</title>
5   <script src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script>
6   <script>
7     var json_url = 'http://js.courses.dp.ua/files/angular/get_phones_json.php';
8     var app = angular.module('PhoneFilter', []);
9     app.controller('mainController', function($scope, $http){
10
11     }
12     {
13       $scope.min_price_range = 0;
14       $scope.max_price_range = 1000;
15
16     }
17     {
18       $http.get(json_url).then(function(res) {
19         $scope.phones = res.data;
20       });
21
22     }
23     {
24       $scope.priceFilter = function(data) {
25         return (parseInt(data.price) >= parseInt($scope.min_price_range))
26         && (parseInt(data.price) <= parseInt($scope.max_price_range));
27       };
28     }
29   };
30 </script>
31 </head>
32 <body ng-app="PhoneFilter" ng-controller="mainController">
33   <h1>Каталог телефонов:</h1>
34   <input type="range" min="0" max="1000" ng-model="min_price_range">
35   <input type="range" min="0" max="1000" ng-model="max_price_range">
36   <div>
37     фильтр цен, от <span ng-bind="min_price_range">0</span>$
38     до <span ng-bind="max_price_range">1000</span>$.
39   </div>
40   <hr>
41   <div>
42     <div ng-repeat="phone in phones filter:priceFilter">{{phone.name}} <span>{{phone.price}}</span></div>
43   </div>
44 </body>
45 </html>
```



Директива **filter** позволяет фильтровать выводимы данные, по каким-либо правилам.

# Фильтрация

The screenshot shows a web browser window with the title "Phones Filter.Angular.JS". The address bar shows the file path: `file:///D:/OneDrive/Courses/JS_Feb_2016/Angular/ex03_c.html`. The page content includes a heading "Каталог телефонов:" and a price range filter: "Фильтр цен, от 242\$ до 682\$". Below the filter, two phone models are listed: "iPhone 5s 314\$" and "iPhone 6s. 252\$".

The developer console is open, showing the HTML structure. The selected element is a `div` with the class `ng-binding ng-scope`. The HTML code includes two range input fields for price filtering and a list of phone items generated by `ngRepeat` using the `priceFilter` directive. The console also shows the computed styles for the selected element, including `display: block;` and a box model diagram with dimensions `790 x 18`.

*Директива **filter** позволяет фильтровать выводимы данные, по каким-либо правилам.*

# Сортировка

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Phones Filter.Angular.JS</title>
5   <script src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script>
6   <script>
7     var json_url = 'http://js.courses.dp.ua/files/angular/get_phones_json.php';
8     var app = angular.module('PhoneFilter', []);
9     app.controller('mainController', function($scope, $http){
10
11       $scope.min_price_range = 0;
12       $scope.max_price_range = 1000;
13
14       $http.get(json_url).then(function(res){
15         $scope.phones = res.data;
16       });
17
18       $scope.priceFilter = function(data){
19         return (parseInt(data.price) >= parseInt($scope.min_price_range))
20           && (parseInt(data.price) <= parseInt($scope.max_price_range));
21       }
22     });
23   </script>
24 </head>
25 <body ng-app="PhoneFilter" ng-controller="mainController">
26   <h1>Каталог телефонов:</h1>
27   <input type="range" min="0" max="1000" ng-model="min_price_range">
28   <input type="range" min="0" max="1000" ng-model="max_price_range">
29   <div>
30     Фильтр цен, от <span ng-bind="min_price_range">0</span>$
31     до <span ng-bind="max_price_range">1000</span>$.
32   </div>
33   <hr>
34   <div>
35     <div ng-repeat="phone in phones | orderBy:'price' | filter:priceFilter">{{phone.name}}
36     <span>{{phone.price}}</span>
37   </div>
38 </div>
39 </body>
40 </html>
```

D

???

Директива `orderBy` позволяет указать критерий сортировки

# Сортировка

Добавьте следующие  
стили:

```
<style>
  button:after { content: '\25b2'; }
  button.toggleclass:after { content: '\25bc'; }
</style>
```

E

Добавьте следующий код в

КОН

```
<script>
  var json_url = 'http://js.courses.dp.ua/files/angular/get_phones_json.php';
  var app = angular.module('PhoneFilter', []);
  app.controller('mainController', function($scope, $http){

    $scope.min_price_range = 0;
    $scope.max_price_range = 1000;

    $http.get(json_url).then(function(res) {
      $scope.phones = res.data;
    });

    $scope.priceFilter = function(data) {
      return (parseInt(data.price) >= parseInt($scope.min_price_range)
        && (parseInt(data.price) <= parseInt($scope.max_price_range));
    }

    $scope.reverse = false;
    $scope.order_reverse = function(){
      $scope.reverse = !$scope.reverse;
    }

  });
</script>
```

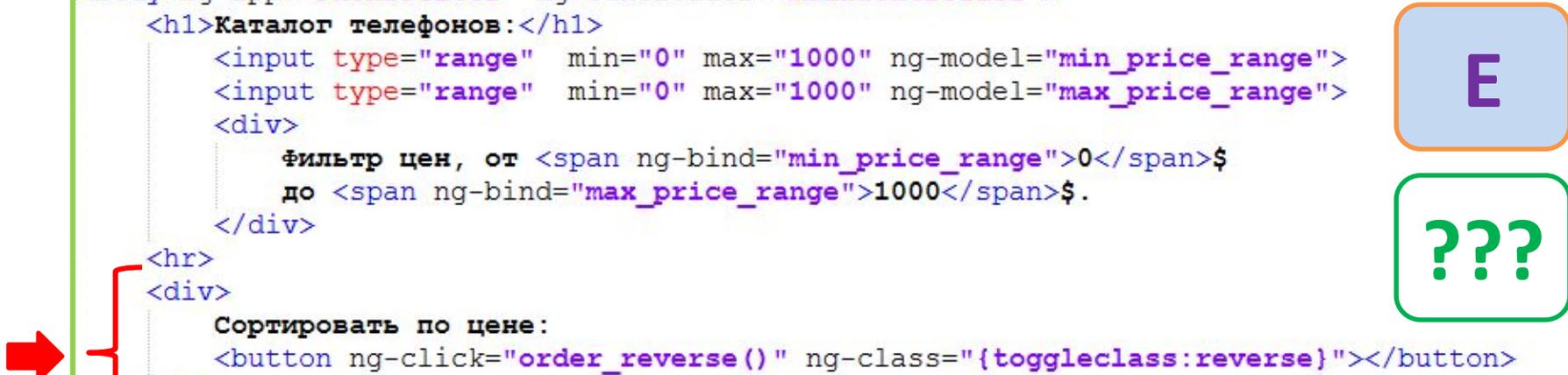
???

Директива `orderBy` позволяет указать критерий сортировки. И порядок сортировки можно динамически

# Сортировка

Добавьте следующую разметку в **данных**

```
<body ng-app="PhoneFilter" ng-controller="mainController">
  <h1>Каталог телефонов:</h1>
  <input type="range" min="0" max="1000" ng-model="min_price_range">
  <input type="range" min="0" max="1000" ng-model="max_price_range">
  <div>
    Фильтр цен, от <span ng-bind="min_price_range">0</span>$
    до <span ng-bind="max_price_range">1000</span>$.
  </div>
  <hr>
  <div>
    Сортировать по цене:
    <button ng-click="order_reverse()" ng-class="{toggleclass:reverse}"></button>
  </div>
  <hr>
  <div>
    <div ng-repeat="phone in phones | orderBy:'price':reverse | filter:priceFilter ">
      {{phone.name}} <span>{{phone.price}}</span>
    </div>
  </div>
</body>
</html>
```



Директива `orderBy` позволяет указать критерий сортировки. И порядок сортировки можно динамически менять.

# Сортировка данных (полный код)

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Phone Filter.Angular.JS</title>
5   <style>
6     button:after { content: '\25b2'; }
7     button.toggleclass:after { content: '\25bc'; }
8   </style>
9   <script src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script>
10  <script>
11    var json_url = 'http://js.courses.dp.ua/files/angular/get_phones_json.php';
12    var app = angular.module('PhoneFilter', []);
13    app.controller('mainController', function($scope, $http){
14      $scope.min_price_range = 0;
15      $scope.max_price_range = 1000;
16
17      $http.get(json_url).then(function(res){
18        $scope.phones = res.data;
19      });
20
21      $scope.priceFilter = function(data){
22        return (parseInt(data.price) >= parseInt($scope.min_price_range))
23          && (parseInt(data.price) <= parseInt($scope.max_price_range));
24      }
25
26      $scope.reverse = false;
27      $scope.order_reverse = function(){
28        $scope.reverse = !$scope.reverse;
29      }
30    });
31  </script>
32 </head>
33 <body ng-app="PhoneFilter" ng-controller="mainController">
34   <h1>Каталог телефонов:</h1>
35   <input type="range" min="0" max="1000" ng-model="min_price_range">
36   <input type="range" min="0" max="1000" ng-model="max_price_range">
37   <div>
38     Фильтр цен, от <span ng-bind="min_price_range">0</span>$
39     до <span ng-bind="max_price_range">1000</span>$.
40   </div>
41   <hr>
42   <div>
43     Сортировать по цене:
44     <button ng-click="order_reverse()" ng-class="{toggleclass:reverse}"></button>
45   </div>
46   <hr>
47   <div>
48     <div ng-repeat="phone in phones | orderBy:'price':reverse | filter:priceFilter ">
49       {{phone.name}} <span>{{phone.price}}</span>
50     </div>
51   </div>
52 </body>
53 </html>
```

E

???

# Сортировка

The screenshot shows a web browser window with the title "Phone Filter.Angular.JS". The address bar shows the file path: `file:///D:/OneDrive/Courses/JS_Feb_2016/Angular/task1_complete_alt.html`. The page content includes a heading "Каталог телефонов:", a price filter "Фильтр цен, от 131\$ до 758\$.", and a sorting dropdown "Сортировать по цене:". Below the dropdown is a list of phones with their prices: Samsung 686\$, iPhone 6s. 648\$, Fly 516\$, Asus 505\$, iPhone 3Gs 477\$, HTC 428\$, LG 405\$, iPhone 5s 330\$, iPhone 6. 205\$, and Nokia 169\$.

The developer console shows the following HTML structure for the sorting button:

```
<button ng-click="order_reverse()" ng-class="{toggleclass:reverse}" class="toggleclass">Сортировать по цене:</button>
```

The console also shows the CSS styles for the button:

```
element.style {
}
user agent stylesheet
input[type="button"] {
  margin: -
  border: 2
  padding: 1
}
```

*Директива `orderBy` позволяет указать критерий сортировки. И порядок сортировки можно динамически менять.*

# Помощь при валидации

```
1 <html>
2 <script src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script>
3 <body ng-app="">
4   <form name="my_form">
5     <p>Name:
6       <input name="user_name" ng-model="user_name" required>
7       <span ng-show="my_form.user_name.$touched && my_form.user_name.$invalid">
8         The name is required.
9       </span>
10    </p><p>Email:
11      <input type="email" name="user_email" ng-model="user_email" required>
12      <span ng-show="my_form.user_email.$touched && my_form.user_email.$invalid">
13        The email is required.
14      </span>
15    </p><p>
16      <input type="submit" ng-disabled="my_form.user_name.$invalid ||
17        my_form.user_email.$invalid">
18    </p>
19  </form>
20 </body>
</html>
```



The screenshot shows a web form with two input fields and a submit button. The first field is labeled 'Name:' and is empty. To its right, the text 'The name is required.' is displayed. The second field is labeled 'Email:' and is also empty. To its right, the text 'The email is required.' is displayed. Below these fields is a submit button with the text 'Отправить' (Submit).

Директива **ng-show** – показывает либо скрывает элемент в зависимости от значения данных в модели. **ng-disable** – выключает либо включает элемент ввода в зависимости от значения данных в модели.

# Помощь при валидации

## данных

AngularJS добавляет к полям формы следующие свойства типа `boolean`:

```
ng-show="my_form.user_email.$touched && my_form.user_email.$invalid">
```

**\$untouched** – поле еще не получало фокус;

**\$touched** – поле минимум 1 раз получало фокус;

**\$pristine** – поле не было изменено (в нём может быть значение по умолчанию);

**\$dirty** – поле было изменено по сравнению с версией по умолчанию;

**\$invalid** – содержимое поля не проходит валидацию;

**\$valid** – содержимое поля проходит валидацию.

# По мнению google лучший способ освоить AngularJS пройти обучалку

The screenshot shows the AngularJS website's tutorial page. At the top, there's a navigation bar with 'Home', 'Learn', 'Develop', and 'Discuss' menus, and a search bar. Below the navigation, the page title is 'v1.5.5-build.4754 (snapshot) / Tutorial'. The main content area is titled 'PhoneCat Tutorial App' and includes a table of contents on the left, a description of the app, and a detailed view of a device (Nexus S) with its specifications.

**Tutorial**

- 0 - Bootstrapping
- 1 - Static Template
- 2 - Angular Templates
- 3 - Filtering Repeaters
- 4 - Two-way Data Binding
- 5 - XHRs & Dependency Injection
- 6 - Templating Links & Images
- 7 - Routing & Multiple Views
- 8 - More Templating
- 9 - Filters
- 10 - Event Handlers
- 11 - REST and Custom Services
- 12 - Applying Animations
- The End

## PhoneCat Tutorial App

A great way to get introduced to AngularJS is to work through this tutorial, which walks you through the construction of an AngularJS web app. The app you will build is a catalog that displays a list of Android devices, lets you filter the list to see only devices that interest you, and then view details for any device.

Search:   
Google:   
Sort by: [Alphabetical](#)

**LG Axis**  
Android Powered, Google Maps Navigation, 3 Customizable Home Screens

**Nexus S**

Nexus S is the next generation of Nexus devices, co-developed by Google and Samsung. The latest Android platform (Gingerbread), paired with a 1 GHz Hummer-based processor and 16GB of memory, makes Nexus S one of the fastest phones on the market. It comes pre-installed with the best of Google apps and enabled with new and popular features like text-to-speech, Wi-Fi hotspot, internet Calling, NFC support, and full web browsing. With this device, users will also be the first to receive software upgrades and new Google mobile apps as soon as they become available. For more details, visit <http://www.google.com/nexus>.

Availability and Networks	Battery	Storage and Memory	Connectivity
<b>Availability</b> MT, G2, Orange, Sprint, SofLab, T-Mobile, Verizon	<b>Type</b> Lithium Ion (Li-Ion) <b>Capacity</b> 1500 mAh <b>Talk Time</b> 6 hours <b>Standby time (max)</b> 428 hours	<b>RAM</b> 512MB <b>Internal Storage</b> 16GB/4GB	<b>Network Support</b> Quad-band GSM: 850, 900, 1800, 1900 TD- SCDMA; HSPA: 900, 2100, 1700; HSPA+; LTE HSPA+ (7.2Mbps) HSPA+ (5.76Mbps) <b>Wi-Fi</b> 802.11 b/g/n <b>Bluetooth</b> Bluetooth 2.1 <b>Infrared</b> x

Follow the tutorial to see how Angular makes browsers smarter — without the use of native extensions or plug-ins:

- See examples of how to use client-side data binding to build dynamic views of data that change immediately in response to user actions.
- See how Angular keeps your views in sync with your data without the need for DOM manipulation.
- Learn a better, easier way to test your web apps, with Karma and Protractor.
- Learn how to use dependency injection and services to make common web tasks, such as getting data into your app, easier.

<https://docs.angularjs.org/tutorial>

# Пора вливаться в

## КОМЬЮНИТИ

События

Архив

События

Днепропетровск

по всем темам



RSS

19 апреля (вторник)



19 апреля Днепропетровск, ул. Баумана, 10, 2-й этаж, конференц-зал DataArt

[IT talk: Четыре вещи, которые IT-профессионалы должны делать чаще](#)

Поговорим про истинные роли инженеров и менеджеров в индустрии, о том, что важно и не очень в IT-карьере, о лидерстве и пользе левостороннего движения.

2 идут карьера, клубные встречи

22 апреля (пятница)



22 апреля Днепропетровск

[Конференция HR lab «Кадры высокого полета: тенденции, технологии и практики развития талантов»](#)

Ивент будет посвящен актуальной на сегодня теме развития талантов. Цель команды HR lab – познакомить участников мероприятия со всем разнообразием возможностей

### Популярные

17 апреля, Киев

[JavaScript Frameworks Day 2016](#)

20 апреля, Киев

[iForum 2016](#)

23 апреля, Львов

[Lviv PM Weekend Conference](#)

20 – 21 мая, Киев

[JEEConf 2016](#)

4 – 9 июля, Киев

[The 11th Open International Student Olympiad in programming KPI-OPEN 2016](#)

9 июля, Одесса

[Eastern European Computer Vision Conference](#)

<http://dou.ua/calendar/city/Днепропетровск>

Если вы по описанию не понимаете о чём мероприятие – это верный признак того, что вам нужно на него сходить!!!