AngularJS

AngularJS Filters

Lesson Objectives

- **Built-In Filters**
- **Creating Custom Filters**

Filters

- A filter formats the value of an expression for display to the user.
- Filters can be invoked in HTML with the | (pipe) character inside the template.
- We can also use filters from within JavaScript by using the $filter service.
- To pass an argument to a filter in the HTML form, we pass it with a colon after the filter name (for multiple arguments, we can simply append a colon after each argument)
- Angular gives us several built-in filters as well as an easy way to create our own.

Built-In Filters

- **uppercase**
  - Converts string to uppercase.
  - {{'igate'|uppercase}} <!-- Displays: IGATE -->

```
angular.module('filterApp',[])
.controller('FilterController', function($scope,$filter) {
        $scope.companyName = $filter('uppercase')('igate');
});
```

- **lowercase**
  - Converts string to lowercase.
  - {{'IGATE'|lowercase}} <!-- Displays: igate -->
- **number**
  - Formats a number as text. If the input is not a number an empty string is returned.
  - {{ 123.456789 | number:2 }} <!-- Displays: 123.46 -->

Built-In Filters

- **currency**
  - The currency filter formats a number as currency. Currency gives us the option of displaying a currency symbol or identifier.
  - The default currency option is that of the current locale we can also pass in a currency to display.
  - {{ 30 | currency}} <!-- Displays: $30.00 -->
  - {{ 30 | currency : "Rs."}} <!-- Displays: Rs.30.00 -->
- **json**
  - Convert s JavaScript object into JSON string.
  - This filter is mostly useful for debugging
  - {{{ 'Id':714709,'Name':'Karthik'} | json }}

<!-- Displays: { "Id": 714709, "Name": "Karthik"} -->

Built-In Filters

date

- Formats date to a string based on the requested format.
- Date to be formatted can be either a Date object or milliseconds (string or number)
- The date formatter provides us several built-in options. If no date format is passed, then it set to the default mediumDate.

```
angular.module('filterApp',[])
.controller('FilterController', function($scope,$filter) {
        $scope.today = new Date();
        $scope.formattedDate = $filter('date')(new Date(),'EEEE dd, MMMM yyyy');
});
```

- {{today | date:'medium'}} <!-- Displays: Jul 30, 2014 5:13:19 PM -->
- {{today | date:'short'}} <!-- Displays: 7/30/14 5:13 PM -->
- {{today | date:‘fullDate'}} <!-- Displays: Wednesday, July 30, 2014-->
- {{today | date:‘longDate'}} <!-- Displays: July 30, 2014-->

Built-In Filters

- {{today | date:'mediumDate'}} <!-- Displays: Jul 30, 2014-->
- {{today | date:‘shortDate'}} <!-- Displays: 7/30/14 -->
- {{today | date:‘mediumTime'}} <!-- Displays: 5:22:21 PM -->
- {{today | date:‘mediumTime'}} <!-- Displays: 5:23 PM -->
- {{today | date:'d-M-y'}} <!-- Displays: 30-7-2014 -->
- {{today | date:'d-M-yyyy'}} <!-- Displays: 30-7-2014 -->
- {{today | date:'dd-MM-yy'}} <!-- Displays: 30-07-14 -->
- {{today | date:'EEEE dd, MMMM yyyy'}} <!-- Displays: Wednesday 30, July 2014 -->
- {{today | date:'EEE dd MMM yyyy'}} <!-- Displays: Wed 30 Jul 2014 -->
- {{ today | date:'hh:mm:ss.sss a' }} <!-- Displays: 05:35:31.951 PM -->
- {{ today | date:'hh:mm:ss a' }} <!-- Displays: 05:35:31 PM -->
- {{ today | date:'H:m:s a' }} <!-- Displays: 17:35:31 PM -->
- {{ today | date:'Z' }} <!-- Displays: +0530 -->

Built-In Filters

filter

- filter selects a subset of items from an array of items and returns a new array.
- The filter method takes a string, object, or function that it will run to select or reject array elements.

filter based on a string :

```
{{ ['Anil', 'Latha', 'Mahima', 'Sachin', 'Veena'] | filter:'e' }}
<!-- returns array ["Veena"]  'e' returns other than 'Veena'  -->
```

filter based on a function:

```
angular.module('filterApp',[])
.controller('FilterController', function($scope,$filter) {
        var pattern = /^\d{6}$/;
        $scope.getSixDigitsPattern = function(item) {
                return pattern.test(item);
        }
});
```

Built-In Filters

```
{{ ['714709', '562A', '044-235', '801234','ABC'] | filter:getSixDigitsPattern }}
<!-- return array ["714709","801234"] -->
```

filter based on a object:

```
{{
    [
        {"Id":1,"Name":"Anil","Location":"Mumbai"},
        {"Id":2,"Name":"Latha","Location":"Bangalore"},
        {"Id":3,"Name":"Mahima","Location":"Pune"},
        {"Id":4,"Name":"Sachin","Location":"Mumbai"},
        {"Id":5,"Name":"Veena","Location":"Pune"}
    ] | filter:{"Location":"Mumbai"}
}}
<!-- returns array
[
    {"Id":1,"Name":"Anil","Location":"Mumbai"},
    {"Id":4,"Name":"Sachin","Location":"Mumbai"}
]
-->
```

Built-In Filters

limitTo

- limitTo filter creates a new array or string that contains only the specified number of elements, either taken from the beginning or end, depending on whether the value is positive or negative.

```
{{ 'IGATE GLOBAL SOLUTIONS' | limitTo:5 }}
<!-- returns  first 5 characters: IGATE -->

{{ 'IGATE GLOBAL SOLUTIONS' | limitTo:-9 }}
<!-- returns  last 9 characters: SOLUTIONS -->

{{
        ['Bangalore','Chennai','Hyderabad','Gandhinagar','Mumabai','Noida','Pune']
       | limitTo:2
}}
<!--  returns  first 2 array elements :  ["Bangalore","Chennai"] -->
```

Built-In Filters

orderBy

- The orderBy filter orders the specific array using an expression.
- It is ordered alphabetically for strings and numerically for numbers.
- It takes 2 parameters. First parameter is the predicate used to determine the order of the sorted array. Second parameter(optional) is a boolean value, if it is true it will sort the data in reverse order.

```
{{['Chennai','Bangalore','Pune','Mumbai'] | orderBy:'toString()'}}
<!-- returns ["Bangalore","Chennai","Mumbai","Pune"] -->


{{[{'Id':1,'Location':'Bangalore'},{'Id':2,'Location':'Chennai'}] | orderBy:'Id':true}}
<!-- returns [{"Id":2,"Location":"Chennai"},{"Id":1,"Location":"Bangalore"}] -->
```

Custom Filters

- We can easily create custom filters in AngularJS.

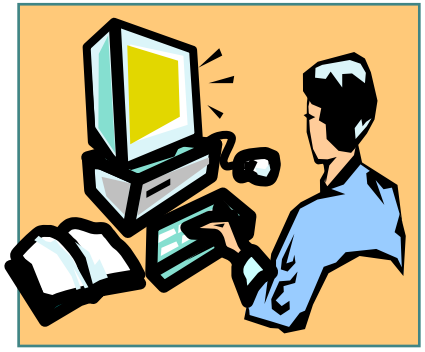
```
angular.module('filterApp',[])
.filter('getIntegers',function(){
        return function(numberArray){
                var size = numberArray.length;
                var evenNumbers = [];
                for(counter=0;counter<size;counter++)
                {
                        if(numberArray[counter]%2==0)
                            evenNumbers.push(numberArray[counter]);
                }
                return evenNumbers;
        }
})

{{[1,2,3,4,5,6,7,8,9,10] | getIntegers}}        <!- - returns [2,4,6,8,10]  - ->
```

- Filters are just functions to which we pass input. In the function above, we simply take the input as the array on which we are calling the filter.

Demo

- **FiltersDemo**



Summary

- Filters are just functions to which we pass input.
- Filters can be invoked in HTML with the | (pipe) character inside the template.
- We can indicate a parameter to a filter using a colon (:)
- We can limit a filter to only search in a specific field
- We can specify custom date formats using date filter