Selects a subset of items from array and returns it as a new array.

Usage

In HTML Template Binding

{{ filter\_expression | filter : expression : comparator}}

In JavaScript

$filter('filter')(array, expression, comparator)

Arguments

| **Param** | **Type** | **Details** |
| --- | --- | --- |
| array | [**Array**](https://docs.angularjs.org/) | The source array. |
| expression | [**string**](https://docs.angularjs.org/) [**Object**](https://docs.angularjs.org/) [**function()**](https://docs.angularjs.org/) | The predicate to be used for selecting items from array.  Can be one of:   * string: The string is used for matching against the contents of the array. All strings or objects with string properties in array that match this string will be returned. This also applies to nested object properties. The predicate can be negated by prefixing the string with !. * Object: A pattern object can be used to filter specific properties on objects contained by array. For example {name:"M", phone:"1"} predicate will return an array of items which have property name containing "M" and property phone containing "1". A special property name $ can be used (as in {$:"text"}) to accept a match against any property of the object or its nested object properties. That's equivalent to the simple substring match with a string as described above. The predicate can be negated by prefixing the string with !. For example {name: "!M"} predicate will return an array of items which have property name not containing "M".   Note that a named property will match properties on the same level only, while the special$ property will match properties on the same level or deeper. E.g. an array item like{name: {first: 'John', last: 'Doe'}} will **not** be matched by {name: 'John'}, but **will** be matched by {$: 'John'}.   * function(value, index, array): A predicate function can be used to write arbitrary filters. The function is called for each element of the array, with the element, its index, and the entire array itself as arguments.   The final result is an array of those elements that the predicate returned true for. |
| comparator | [**function(actual, expected)**](https://docs.angularjs.org/) [**true**](https://docs.angularjs.org/) [**undefined**](https://docs.angularjs.org/) | Comparator which is used in determining if the expected value (from the filter expression) and actual value (from the object in the array) should be considered a match.  Can be one of:   * function(actual, expected): The function will be given the object value and the predicate value to compare and should return true if both values should be considered equal. * true: A shorthand forfunction(actual, expected) { return angular.equals(actual, expected)}. This is essentially strict comparison of expected and actual. * false|undefined: A short hand for a function which will look for a substring match in case insensitive way.   Primitive values are converted to strings. Objects are not compared against primitives, unless they have a custom toString method (e.g. Date objects). |