

### Filters

#### amount | currency[:symbol]

Formats a number as a currency (ie \$1,234.56).

#### date | date[:format]

#### array | filter:expression

Selects a subset of items from array.  
Expression takes *string|Object|function()*

#### data | json

Convert a JavaScript object into JSON string.

#### array | limitTo:limit

Creates a new array containing only a specified number of elements in an array.

#### text | linky<sup>1</sup>

Finds links in text input and turns them into html links.

#### string | lowercase

Converts string to lowercase.

#### number | number[:fractionSize]

Formats a number as text. If the input is not a number an empty string is returned.

#### array | orderBy:predicate[:reverse]

Predicate is function(\*)|string|Array. Reverse is boolean

#### string | uppercase

Converts string to uppercase.

You can inject the \$filter service and do `$filter('filterName')(value[, :optionalParam][, :optionalParam])` in use it in your javascript.

<sup>1</sup> Requires ngSanitize Module

### Services

#### \$anchorScroll

#### \$cacheFactory

compiledHtml = `$compile(html)(scope)`

#### \$controller

#### \$cookieStore

#### \$document

#### \$exceptionHandler(exception[, cause])

#### \$filter(name)

#### \$http[[options]]

#### \$httpBackend

#### \$injector

#### \$interpolate(text[, mustHaveExpression])

#### \$locale

#### \$location

#### \$log

#### \$parse(expression)

#### \$provide

#### \$q

#### \$resource(url[, paramDefaults][, actions])

#### \$rootScope

#### \$rootScope

#### \$route

#### \$routeParams

#### \$routeProvider

#### \$sanitize(html)

#### \$scope See \$rootScope

#### \$templateCache

#### \$timeout(fn[, delay][, invokeApply])

#### \$window

### Directive Definition Object

#### name {string}

Name of the current scope. Optional defaults to the name at registration.

#### priority {integer}

Specifies order multiple directives apply on single DOM element (higher = first)

#### terminal {true}

Current *priority* will be last set of directives to execute

#### scope {true | object}

*True* - create child scope. *Undefined|false* - use parent scope. *{}* - isolate scope (with specified attributes/scope variables passed):  
*@* or *@attr* - bind local model to value of DOM attribute (string), *=* or *=attr* - bi-directional binding between local model and the parent scope, *&* or *&attr* - execute an expression in context of parent. Reference *attr* OR assumes model of same name

#### controller function(\$scope, \$element, \$attrs, \$transclude)

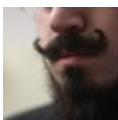
Controller constructor function instantiated before pre-linking phase and shared with other directives if requested by name

#### require {string | array[string]}

Require another controller (*ngModel*).  
Prefixes: *?* - Don't raise error. *^* - Look on parent elements too

#### restrict {string: 'EACM'}

**E - Element:** `<my-directive />`. **A - Attribute** (default): `<div my-directive="exp" />`. **C - Class:** `<div class="my-directive: exp;" />`. **M - Comment:** `<!-- directive: my-directive exp -->`



By ProLoser

[cheatography.com/proloser/](http://cheatography.com/proloser/)

[www.DeanSofer.com](http://www.DeanSofer.com)

Published 9th August, 2012.

Last updated 31st July, 2014.

Page 1 of 4.

Sponsored by **Readability-Score.com**

Measure your website readability!

<https://readability-score.com>

### Directive Definition Object (cont)

#### template {string}

Replace current element with contents and migrates all attributes / classes

#### templateUrl {string}

Same as *template* but the template is loaded from the specified URL

#### replace {boolean}

*true*: template replaces element instead of appending

#### transclude {boolean}

Compiles contents on parent (pre-isolate) scope. Usually used with `ngTransclude` & templates.

#### compile function(tElement, tAttrs, fn

*transclude*(function(scope, cloneLinkingFn) ) returns link()

For transforming the template (rare, run once per template instance).

#### link function(scope, iElement, iAttrs, controller)

Executed after template is cloned (run once per clone). Contains most logic (DOM listeners, etc). *Controller* can be an array.

<http://docs.angularjs.org/guide/directive>

### Directives

**ng-app**="plaintext"

**ng-bind**[-html-unsafe]="expression"

**ng-bind-**

**template**="string{{expression}}string{{expression}}"

**ng-change**="expression"

**ng-checked**="boolean"

**ng-class**[-even]-odd]="string|object"

**ng-[dbl]click**="expression"

**ng-cloak**="boolean"

**ng-controller**="plaintext"

<html **ng-csp**> (Content Security Policy)

### Directives (cont)

**ng-disabled**="boolean"

<form|**ng-form** name="plaintext"> | **ng-form**="plaintext"

**ng-hide**|**show**="boolean"

**ng-href**="plaintext{{string}}"

**ng-include**="string"|< **ng-include** src="string" onload="expression" autoscroll="expression">

**ng-init**="expression"

<input **ng-pattern**="/regex/" **ng-minlength** **ng-maxlength** **ng-required**

<input **ng-list**="delimiter|regex">

<input type="checkbox" **ng-true-value**="plaintext" **ng-false-value**="plaintext">

**ng-model**="expression"

**ng-mousedown**="expression"

**ng-mouseenter**="expression"

**ng-mouseleave**="expression"

**ng-mousemove**="expression"

**ng-mouseover**="expression"

**ng-mouseup**="expression"

<select **ng-multiple**>

**ng-non-bindable**

**ng-options**="select [as label] [group by group] for ([key.] value) in object|array"

**ng-pluralize**< **ng-pluralize** count="number" when="object" offset="number">

**ng-readonly**="expression"

**ng-repeat**="([key.] value) in object|array"

<option **ng-selected**="boolean">

**ng-src**="string"

**ng-style**="string|object"

**ng-submit**="expression"

**ng-switch**="expression"|< **ng-switch** on="expression">

**ng-switch-when**="plaintext"

**ng-switch-default**

**ng-transclude** templates

**ng-view**|< **ng-view**>

### Directives (cont)

**ng-bind-html**="expression"

**Bold** means the actual directive

*Italics* mean optional

Pipes mean either|or

Plaintext means no string encapsulation

Superscript means notes or context

<Brackets> mean tag compatibility

Lack of <brackets> means the attribute can apply to any tag

### Module

#### config(configFn)

Use this method to register work which needs to be performed on module loading.

#### constant(name, object)

Because the constant are fixed, they get applied before other provide methods.

#### controller(name, constructor)

#### directive(name, directiveFactory)

#### factory(name, providerFunction)

#### filter(name, filterFactory)

#### provider(name, providerType)

#### run(initializationFn)

Use this method to register work which needs to be performed when the injector with the current module is finished loading.

#### service(name, constructor)

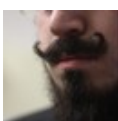
value(name, object)

#### name

Name of the module.

Holds the list of modules which the injector will load before the current module is loaded.

<http://docs.angularjs.org/api/angular.Module>



### Scope Properties and Methods

#### **\$root** or **\$rootScope**

Move to the top-most \$scope (ng-app)

#### **\$parent**

Move to the immediate parent of the current \$scope

#### **\$id**

Auto generated Unique ID

#### **\$destroy** (event)

Broadcasted when a scope and its children are being destroyed

#### **\$apply(exp)**

Executes logic within the AngularJS context and refreshes all models checks.

#### **\$broadcast(name, args)**

Dispatches an event name downwards to all child scopes

#### **\$destroy()**

Removes the current scope (and all of its children) from the parent scope

#### **\$digest()**

Process all of the watchers of the current scope and its children. Since watchers can change models, they will continue firing until all changes stop. **BEWARE OF RECURSIVE CODE**

#### **\$emit(name, args)**

Dispatches an event name upwards through the scope hierarchy

#### **\$eval(expression)**

Executes the expression on the current scope and returns the result

### Scope Properties and Methods (cont)

#### **\$evalAsync(expression)**

Executes the expression on the current scope at a later point in time

#### **\$new(isolate)**

Creates a new child scope

#### **\$on(name, listener)**

Listens on events of a given type

#### **\$watch(watchExp, listener(newVal, oldVal, scope), objectEquality)**

Watch a model (exp) for changes and fires the listener callback. Pass *true* as a third argument to watch an object's properties too.

The following directives create child scopes: *ngInclude*, *ngSwitch*, *ngRepeat*, *ngController*, *uiIf*. Calls to the same *ngController* will create multiple instances and **do not** share scopes. Remember to traverse up the tree to affect *primitives* on the intended scope: *ng-click="\$parent.showPage=true"*

### Global Functions

#### **angular.bind(self, fn, args)**

Returns a function which calls function *fn* bound to *self* (*self* becomes the *this* for *fn*).

#### **angular.bootstrap(element[, modules])**

Use this function to manually start up angular application.

#### **angular.copy(source[, destination])**

Creates a deep copy of *source*, which should be an object or an array.

#### **angular.element(element)**

Wraps a raw DOM element or HTML string as a jQuery element.

### Global Functions (cont)

#### **angular.equals(o1, o2)**

Determines if two objects or two values are equivalent.

#### **angular.extend(dst, src)**

Extends the destination object *dst* by copying all of the properties from the *src* object(s) to *dst*.

#### **angular.forEach(obj, iterator[, context])**

Invokes the iterator function once for each item in *obj* collection, which can be either an object or an array.

#### **angular.fromJson(json)**

Deserializes a JSON string.

#### **angular.identity()**

A function that returns its first argument. This function is useful when writing code in the functional style.

#### **angular.injector(modules)**

Creates an injector function that can be used for retrieving services as well as for dependency injection.

#### **angular.isArray(value)**

Determines if a reference is an Array.

#### **angular.isDate(value)**

Determines if a value is a date.

#### **angular.isDefined(value)**

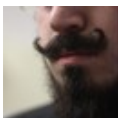
Determines if a reference is defined.

#### **angular.isElement(value)**

Determines if a reference is a DOM element (or wrapped jQuery element).

#### **angular.isFunction(value)**

Determines if a reference is a Function.



By ProLoser

[cheatography.com/proloser/](http://cheatography.com/proloser/)  
[www.DeanSofer.com](http://www.DeanSofer.com)

Published 9th August, 2012.

Last updated 31st July, 2014.

Page 3 of 4.

Sponsored by **Readability-Score.com**

Measure your website readability!

<https://readability-score.com>

### Global Functions (cont)

#### **angular.isNumber(value)**

Determines if a reference is a Number.

#### **angular.isObject(value)**

Determines if a reference is an Object.  
Unlike typeof in JavaScript, nulls are not considered to be objects.

#### **angular.isString(value)**

Determines if a reference is a String.

#### **angular.isUndefined(value)**

Determines if a reference is undefined.

#### **angular.lowercase(string)**

Converts the specified string to lowercase.

#### **angular.mock**

Namespace from 'angular-mocks.js' which contains testing related code.

#### **angular.module(name[, requires], configFn)**

The angular.module is a global place for creating and registering Angular modules. Requires argument always creates a new module.

#### **angular.noop()**

A function that performs no operations.

#### **angular.toJson(obj[, pretty])**

Serializes input into a JSON-formatted string.

#### **angular.uppercase(string)**

Converts the specified string to uppercase.

#### **angular.version**

An object that contains information about the current AngularJS version.

### FormController

\$pristine

\$dirty

\$valid

\$invalid

\$error

<http://docs.angularjs.org/api/ng.directive:form.FormController>

### NgModelController

\$render() Called when the view needs to be updated. It is expected that the user of the ng-model directive will implement this method.

\$setValidity(validationErrorKey, isValid)

\$setViewValue(value)

\$viewValue mixed

\$modelValue mixed

\$parsers array of function after reading val from DOM to sanitize / convert / validate the value

\$formatters array of functions to convert / validate the value

\$error object

\$pristine boolean

\$dirty boolean

\$valid boolean

\$invalid boolean

<http://docs.angularjs.org/api/ng.directive:ngModel.NgModelController>

### Deferred and Promise

#### **\$q.all([array of promises])**

Creates a Deferred object which represents a task which will finish in the future.

#### **\$q.defer()**

Creates a Deferred object which represents a task which will finish in the future.

#### **\$q.reject(reason)**

Creates a promise that is resolved as rejected with the specified reason

#### **\$q.when(value)**

Wraps an object that might be a value or a (3rd party) then-able promise into a \$q promise

#### **Deferred.resolve(value)**

Resolves the derived promise with the value

#### **Deferred.reject(reason)**

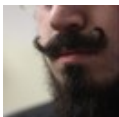
Rejects the derived promise with the reason

#### **Deferred.promise**

Promise object associated with this deferred

#### **Promise.then(successCallback, errorCallback)**

[http://docs.angularjs.org/api/ng.\\$q](http://docs.angularjs.org/api/ng.$q)



By **ProLoser**

[cheatography.com/proloser/](http://cheatography.com/proloser/)  
[www.DeanSofer.com](http://www.DeanSofer.com)

Published 9th August, 2012.

Last updated 31st July, 2014.

Page 4 of 4.

Sponsored by **Readability-Score.com**

Measure your website readability!

<https://readability-score.com>