

Hello World Example

Code:

```
<!doctype html>
<html xmlns:ng="http://angularjs.org">
<script src="http://code.angularjs.org/angular-0.9.19.min.js" ng:autobind></script>
<body>
Your name: <input type="text" name="yourname" value="World"/>
<hr/>
Hello {{yourname}}!
</body></html>
```

Result:

Widgets

ng:non-bindable	Blocks angular from processing an HTML element.
ng:repeat	Creates a collection of cloned HTML elements
ng:required	Verifies presence of user input.
ng:validate	Validates content of user input
ng:view	Works with \$route to "include" partial templates.
ng:switch	Conditionally changes DOM structure.
ng:include	Includes an external HTML fragment.
ng:options	Dynamically generates a list of <option> elements for a <select> element using an array or an object.

Directives

ng:submit	Binds angular expressions to <i>onsubmit</i> events.
ng:change	Runs an expression when an input widget's value changes.
ng:class	Conditionally set CSS class on an element.

ng:click	Executes custom behavior when element is clicked.
ng:controller	Creates a controller and links it to the DOM.
Filters	
<pre>{{expression <filter-name>: <param1>:<param2>:...}}</pre>	
currency	Formats a number as a currency.
date	Formats a number as a date. E.g. {{1288323623006 date:'yyyy-MM-dd HH:mm:ss Z'}}
html	Prevents the input from getting escaped by angular.
json	Converts a JavaScript object into JSON string.
linky	Turns links into html links.
lowercase	Lowercases a string.
number	Formats a number as text Usage: {{number_expression number[fractionSize]}}
Custom Filter	
<pre>angular.filter(<name>, function(input,additional params) { return uiValue; });</pre>	
Formatters	
ng:format="<formatter-name>"	
boolean	Formats user input as Boolean.
json	Formats user input as JSON.
list	Formats user input strings as an array.
number	Formats user input string as number.
trim	Trims leading and trailing spaces from user input.
Custom Formatter	
<pre>angular.formatter(<name>, { parse: function(value) { ... }, format: function(value { ... } });</pre>	
Validators	
ng:validate="<validator-name>[:param1][:param2][:...]"	
asynchronous	Provides asynchronous validation via a callback function.

date	Checks if user input is a valid date.
email	Checks if user input is a valid email.
integer	Checks if user input is a valid integer.
Json	Checks if user input is a valid JSON.
number	Checks if user input is a valid number.
phone	Checks if user input is a valid phone number.
regexp	Restricts valid input to a specified regular expression pattern.
url	Validates that user input is a well formed URL.
Custom Validator	
<pre>angular.validator(<name>, function(input [, additional params]) { [your validation code]; if ([validation succeeds]) { return false; } else { return "my error message"; } } }</pre>	
Services	
\$defer({callback[,delay])	Defers function execution.
\$invalidWidgets	Holds references to invalid widgets.
\$updateView	Queues view updates.
\$xhr({method, url[, post], success, error})	Generates an XHR request.
Custom Service	
<pre>angular.service(<name>, function(dep1) { return someService; }, {\$inject: ['dep1']});</pre>	
Controllers	
<pre><div ng:controller="SomeController"> function SomeController(dep1) {...} SomeController.\$inject = ['dep1'];</pre>	
this.\$watch(<watchExpr>, function(value) {...})	Registers listener as a callback to be executed every time a <i>watchExp</i> changes.

jQuery Mobile Angular Adapter

ngm:shared-controller="var1:Ctrl1,..."	Directive to share the same instance of a controller between mobile pages. The shared controller is saved into a variable. E.g. <div ngm:shared-controller="rental:RentalController">
ngm:<event>="action()"	Directives for general event handlers. E.g.
ngm:if	Conditionally changes DOM structure. Similar to ng:switch, but does not need nested elements.
\$waitDialog: - show([msg]) - hide() - waitFor(promise[, msg])	A service to access the jQuery Mobile Wait Dialog. E.g. this.\$waitDialog.show('Please wait');
\$navigate(['transition:'] pageld')	A service to change the current page, optionally using a defined transition. <i>transition="back"</i> : The browser will go back in history to the defined page. <i>pageld="back"</i> : Goes back one step in history.
Navigation Expressions	
<pre>\$navigate(test, 'outcome1[:transition1]:page1',...)</pre>	
<p>Specifies navigation based on outcomes. This will execute the navigation whose <i>outcome</i> equals <i>test</i>. If <i>test</i> is a promise, this will use the result of the promise.</p> <p><i>outcome="failure"</i> Special outcome for the false value or a failed promise.</p> <p><i>outcome="success"</i> Special outcome for all cases where "failure" does not match.</p> <p>This can be used directly in HTML pages, e.g. </p>	

Paging for Lists

Lists can be paged in the sense that only some entries of an array in the controller are shown.

```
<ul data-role="listview">
<li ng:repeat="item in list.$paged()">
{{item}}
</li>
<li ngm:if="list.$paged().hasMorePages()">
<a href="#"
ngm:click="list.$paged().loadNextPage()">
Load more
</a>
</li>
</ul>
```



Kirchstraße 6 | 51647 Gummersbach
Telefon 02261 6001-0 | Fax -4200

info@opitz-consulting.com
www.opitz-consulting.com/go_mobile

- www.youtube.com/opitzconsulting
- www.slideshare.net/opitzconsulting
- www.xing.com/net/opitzconsulting
- www.twitter.com/oc_wire