

 免费电子书

学习

# AngularJS

Free unaffiliated eBook created from  
**Stack Overflow contributors.**

#angularjs

.....	1
<b>1: AngularJS</b> .....	<b>2</b>
.....	2
.....	2
Examples .....	9
.....	9
Angular .....	10
.....	11
Hello World .....	12
NG- .....	12
.....	13
Angular .....	13
AngularJS .....	14
<b>2: \$ http</b> .....	<b>17</b>
Examples .....	17
\$ http .....	17
\$ http .....	17
\$ http .....	18
<b>3: Angular \$</b> .....	<b>20</b>
.....	20
Examples .....	20
\$ scope .....	20
.....	20
.....	21
\$ scope .....	21
\$ scope .....	22
.....	23
<b>4: angularjs</b> .....	<b>24</b>
.....	24
Examples .....	24
Angularjs .....	24

<b>5: AngularJS `=` `@`</b>	<b>25</b>
.....	25
Examples	25
@	25
=	25
.....	25
.....	26
.....	26
<b>6: AngularJS</b>	<b>27</b>
Examples	27
.....	27
.....	27
html5Mode	27
7 AngularJS	29
<b>7: Angular\$ q</b>	<b>32</b>
Examples	32
\$ q.allpromise	32
\$ q.promise	32
\$ q.defer	33
\$ q	34
Promise	34
.....	35
\$ q.whenpromise	36
\$ q.when\$ q.resolve	36
\$ q	36
.....	36
<b>8: HTTP</b>	<b>38</b>
.....	38
Examples	38
.....	38
httpInterceptor	38
httpFlash	39

.....	39
.....	39
.....	40
<b>9: ng-class</b> .....	<b>41</b>
Examples.....	41
ng-ng.....	41
<b>1.</b> .....	<b>41</b>
<b>2.</b> .....	<b>41</b>
<b>3.</b> .....	<b>41</b>
<b>10: NG-</b> .....	<b>43</b>
.....	43
Examples.....	43
NG-.....	43
.....	43
<b>11: NG-</b> .....	<b>45</b>
.....	45
.....	45
.....	45
.....	45
Examples.....	45
.....	45
.....	45
ng-repeat-start + ng-repeat-end.....	46
<b>12: NG-</b> .....	<b>47</b>
.....	47
.....	47
Examples.....	47
ng.....	47
<b>13: SignalRAngularJs</b> .....	<b>48</b>
.....	48
Examples.....	48

SignalRAngularJs [ChatProject].....	48
<b>14: UI.....</b>	<b>52</b>
.....	52
Examples.....	52
.....	52
.....	53
resolve.....	54
/.....	55
<b>15: .....</b>	<b>57</b>
Examples.....	57
angularjs.....	57
.....	57
.....	57
<b>16: ES6.....</b>	<b>58</b>
Examples.....	58
FileSizeES6.....	58
<b>17: ngModelController.....</b>	<b>60</b>
Examples.....	60
.....	60
.....	61
<b>18: ngRoute.....</b>	<b>65</b>
.....	65
Examples.....	65
.....	65
.....	66
.....	67
<b>19: .....</b>	<b>69</b>
Examples.....	69
/HTML.....	69
<b>20: .....</b>	<b>70</b>
.....	70
.....	70

Examples.....	70
.....	70
.....	70
.....	71
.....	71
.....	72
<b>21:</b> .....	<b>73</b>
.....	73
.....	73
Examples.....	73
.....	73
.....	73
\$ inject.....	74
vanilla JavaScriptAngularJS.....	74
<b>22:</b> .....	<b>75</b>
.....	75
Examples.....	75
ngStorage.....	75
.....	76
<b>23:</b> .....	<b>77</b>
Examples.....	77
- .....	77
ngRepeat.....	77
ngShowngHide.....	80
ngOptions.....	81
ngModel.....	83
ngClass.....	83
ngIf.....	83
<b>JavaScript.....</b>	<b>83</b>
.....	84
<b>DOMcurrentUser.....</b>	<b>84</b>

DOMcurrentUser	84
.....	84
ngMouseenterngMouseleave	85
ngDisabled	85
ngDbclick	85
.....	86
ngClick	87
ngRequired	88
NG-	88
ngCloak	89
ngInclude	89
ngSrc	90
ngPattern	90
ngValue	90
ngCopy	90
.....	91
ngPaste	91
ngHref	91
ngList	91
<b>24:</b>	<b>93</b>
Examples	93
angular.equals	93
angular.isString	93
angular.isArray	93
angular.merge	94
angular.isDefinedangular.isUndefined	94
angular.isDate	94
angular.isNumber	94
angular.isFunction	95
angular.toJson	95
angular.fromJson	95
angular.noop	96

angular.isObject.....	96
angular.isElement.....	97
angular.copy.....	97
angular.identity.....	97
angular.forEach.....	98
<b>25: - .....</b>	<b>99</b>
Examples.....	99
.....	99
.....	100
<b>26: .....</b>	<b>102</b>
Examples.....	102
7.....	102
1ng-repeat.....	102
2.....	102
3.....	103
4.....	103
5ng-if / ng-show.....	104
6.....	104
7.....	105
.....	105
.....	105
.....	106
.....	106
ng-if vs ng-show.....	107
<b>NG-.....</b>	<b>107</b>
<b>NG-.....</b>	<b>107</b>
.....	108
.....	108
.....	108
.....	108
<b>27: .....</b>	<b>110</b>



Examples.....	110
VS.....	110
<b>28:</b> .....	<b>112</b>
.....	112
Examples.....	112
.....	112
1.5+.....	112
.....	113
.....	114
.....	114
<b>29:</b> .....	<b>116</b>
Examples.....	116
.....	116
<b>30:</b> .....	<b>119</b>
Examples.....	119
.....	119
example.js.....	119
example.html.....	119
.....	119
.....	119
.....	119
<b>31: AngularJSTypeScript</b> .....	<b>121</b>
.....	121
Examples.....	121
.....	121
ControllerControllerAs.....	122
/.....	122
ControllerAs.....	123
.....	<b>123</b>
<b>ControllerAs</b> .....	<b>124</b>
.....	<b>124</b>

<b>32: ES6</b> .....	<b>125</b>
Examples.....	125
.....	125
<b>33:</b> .....	<b>126</b>
.....	126
Examples.....	126
.....	126
.....	126
<b>34:</b> .....	<b>128</b>
.....	128
Examples.....	128
.....	128
.....	128
.....	128
UI.....	128
ngRoute.....	129
.....	129
.....	129
<b>35:</b> .....	<b>130</b>
Examples.....	130
.....	130
<b>36:</b> .....	<b>132</b>
.....	132
Examples.....	132
.....	132
<b>37:</b> .....	<b>134</b>
.....	134
Examples.....	134
.....	134
.....	135
Minification.....	135

.....	136
Angular JSControllerAs .....	136
Minification-Safe .....	137
.....	138
<b>38: SelfThis .....</b>	<b>139</b>
.....	139
Examples .....	139
.....	139
<b>39: .....</b>	<b>141</b>
.....	141
Examples .....	141
.....	141
<b>40: .....</b>	<b>143</b>
Examples .....	143
.....	143
.....	143
angular.factory .....	143
\$ sce - .....	144
" .....	144
.....	145
.....	145
<b>41: .....</b>	<b>149</b>
Examples .....	149
.....	149
.....	149
<b>42: .....</b>	<b>151</b>
.....	151
Examples .....	151
.....	151
<b>\$\$ .....</b>	<b>151</b>
<b>\$\$ .....</b>	<b>151</b>

.....	152
<b>AngularJS</b> .....	<b>152</b>
.....	153
\$ destroy \$ rootScope \$.....	154
<b>43:</b> .....	<b>155</b>
.....	155
Examples.....	155
.....	155
\$ digest \$ watch.....	155
\$ scope.....	156
<b>44:</b> .....	<b>158</b>
.....	158
.....	158
Examples.....	158
LifeCycle.....	158
.....	<b>158</b>
.....	159
.....	159
“require”.....	159
JS.....	160
<b>45:</b> .....	<b>161</b>
.....	161
.....	161
Examples.....	162
.....	162
.....	163
.....	164
.....	165
.....	166
.....	167
.....	168

.....	169
<b>46:</b> .....	<b>170</b>
Examples.....	170
.....	170
.....	170
CSS.....	171
ngMessages.....	171
.....	<b>171</b>
.....	<b>171</b>
.....	172
.....	172
.....	173
<b>47:</b> .....	<b>174</b>
.....	174
.....	174
Examples.....	174
.....	174
.....	174
.....	175
<b>48: MVC</b> .....	<b>177</b>
.....	177
Examples.....	177
.....	177
<b>mvc</b> .....	<b>177</b>
.....	177
.....	177
<b>49: -</b> .....	<b>178</b>
Examples.....	178
.....	178
.....	178
.....	178

<b>50:</b>	<b>180</b>
Examples	180
.....	180
ng-inspect chrome	180
.....	184
<b>51: Angular 2+</b>	<b>186</b>
.....	186
Examples	186
AngularJS	186
.....	186
.....	187
.....	187
.....	188
WebpackES6	188
<b>52:</b>	<b>189</b>
Examples	189
.....	189
Javascript	189
HTML	189
.....	190
.....	190
.....	190
.....	191
ng-repeat	192
.....	193

---

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: [angularjs](#)

It is an unofficial and free AngularJS ebook created for educational purposes. All the content is extracted from [Stack Overflow Documentation](#), which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official AngularJS.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to [info@zzzprojects.com](mailto:info@zzzprojects.com)

# 1: AngularJS

AngularJSWeb: [Angular 1.x](#) [Angular 2](#) [Angular 2](#) [Stack Overflow](#)

1.6.5	201773
1.6.4	2017331
1.6.3	201738
1.6.2	201727
1.5.11	2017113
1.6.1	20161223
1.5.10	○
1.6.0	2016128
<i>1.6.0-rc.2</i>	20161124
1.5.9	20161124
<i>1.6.0-RC.1</i>	20161121
<i>1.6.0-rc.0</i>	20161026
1.2.32	20161011
1.4.13	20161010
1.2.31	20161010
1.5.8	2016722
1.2.30	2016721
1.5.7	2016615
1.4.12	2016615
1.5.6	2016527
1.4.11	2016527
1.5.5	2016418



1.5.4	2016414
1.5.3	2016325
1.5.2	2016319
1.4.10	2016316
1.5.1	2016316
1.5.0	201625
<i>1.5.0 rc.2</i>	2016128
1.4.9	2016121
<i>1.5.0 RC.1</i>	2016116
<i>1.5.0 rc.0</i>	2015129
1.4.8	
<i>1.5.0 beta.2</i>	
1.4.7	2015930
1.3.20	2015930
1.2.29	2015930
<i>1.5.0 beta.1</i>	2015930
<i>1.5.0 beta.0</i>	2015917
1.4.6	2015917
1.3.19	2015917
1.4.5	2015828
1.3.18	2015819
1.4.4	2015813
1.4.3	2015715
1.3.17	201577
1.4.2	201577
1.4.1	2015616

1.3.16	201566
1.4.0	2015527
1.4.0-rc.2	2015512
1.4.0-RC.1	2015424
1.4.0-rc.0	2015410
1.3.15	2015317
1.4.0-beta.6	2015317
1.4.0-beta.5	2015-02-24
1.3.14	2015-02-24
1.4.0-beta.4	201529
1.3.13	201529
1.3.12	201523
1.4.0-beta.3	201523
1.3.11	2015127
1.4.0-beta.2	2015127
1.4.0-beta.1	2015120
1.3.10	2015120
1.3.9	2015115
1.4.0-beta.0	2015114
1.3.8	
1.2.28	
1.3.7	
1.3.6	2014129
1.3.5	2014122
1.3.4	
1.2.27	20141121

1.3.3	
1.3.2	2014117
1.3.1	20141031
1.3.0	20141014
<i>1.3.0 rc.5</i>	2014109
1.2.26	2014103
<i>1.3.0 rc.4</i>	2014102
<i>1.3.0 rc.3</i>	2014924
1.2.25	2014917
<i>1.3.0 rc.2</i>	2014917
1.2.24	2014910
<i>1.3.0 RC.1</i>	2014910
<i>1.3.0 rc.0</i>	2014830
1.2.23	2014823
<i>1.3.0 beta.19</i>	2014823
1.2.22	2014812
<i>1.3.0 beta.18</i>	2014812
1.2.21	2014-07-25
<i>1.3.0 beta.17</i>	2014-07-25
<i>1.3.0 beta.16</i>	2014718
1.2.20	2014711
<i>1.3.0 beta.15</i>	2014711
1.2.19	201471
<i>1.3.0 beta.14</i>	201471
<i>1.3.0 beta.13</i>	2014616
<i>1.3.0 beta.12</i>	2014614

1.2.18	2014614
1.3.0 beta.11	201466
1.2.17	201466
1.3.0 beta.10	2014524
1.3.0 beta.9	2014517
1.3.0 beta.8	201459
1.3.0 beta.7	2014426
1.3.0 beta.6	2014422
1.2.16	201444
1.3.0 beta.5	201444
1.3.0 beta.4	2014328
1.2.15	2014322
1.3.0 beta.3	2014321
1.3.0 beta.2	2014-03-15
1.3.0 beta.1	201438
1.2.14	201431
1.2.13	2014215
1.2.12	201428
1.2.11	201423
1.2.10	2014125
1.2.9	2014115
1.2.8	2014-01-10
1.2.7	201413
1.2.6	○
1.2.5	20131213
1.2.4	2013126

1.2.3	20131127
1.2.2	20131122
1.2.1	20131115
1.2.0	2013118
<i>1.2.0-rc.3</i>	
<i>1.2.0-rc.2</i>	201394
1.0.8	2013822
<i>1.2.0rc1</i>	2013813
1.0.7	2013522
1.1.5	2013522
1.0.6	201344
1.1.4	201344
1.0.5	2013220
1.1.3	2013220
1.0.4	2013123
1.1.2	2013123
1.1.1	20121127
1.0.3	20121127
1.1.0	201294
1.0.2	201294
1.0.1	2012-06-25
1.0.0	2012-06-14
<i>v1.0.0rc12</i>	2012-06-12
<i>v1.0.0rc11</i>	2012-06-11
<i>v1.0.0rc10</i>	2012-05-24
<i>v1.0.0rc9</i>	2012-05-15

<i>v1.0.0rc8</i>	2012-05-07
<i>v1.0.0rc7</i>	2012-05-01
<i>v1.0.0rc6</i>	2012-04-21
<i>v1.0.0rc5</i>	2012-04-12
<i>v1.0.0rc4</i>	2012-04-05
<i>v1.0.0rc3</i>	2012-03-30
<i>v1.0.0rc2</i>	2012-03-21
<i>G3-v1.0.0rc1</i>	2012-03-14
<i>G3-V1.0.0-RC2</i>	2012-03-16
<i>1.0.0rc1</i>	2012-03-14
0.10.6	2012-01-17
0.10.5	2011-11-08
0.10.4	2011-10-23
0.10.3	2011-10-14
0.10.2	2011-10-08
0.10.1	2011-09-09
0.10.0	2011-09-02
0.9.19	2011-08-21
0.9.18	2011-07-30
0.9.17	2011-06-30
0.9.16	2011-06-08
0.9.15	2011-04-12
0.9.14	2011-04-01
0.9.13	2011-03-14
0.9.12	2011-03-04
0.9.11	2011-02-09

0.9.10	2011-01-27
0.9.9	2011-01-14
0.9.7	2010-12-11
0.9.6	2010-12-07
0.9.5	2010-11-25
0.9.4	2010-11-19
0.9.3	2010-11-11
0.9.2	2010-11-03
0.9.1	2010-10-27
0.9.0	2010-10-21

## Examples

### HTML

```

<!DOCTYPE html>
<html ng-app>
<head>
  <title>Hello, Angular</title>
  <script src="https://code.angularjs.org/1.5.8/angular.min.js"></script>
</head>
<body ng-init="name='World'">
  <label>Name</label>
  <input ng-model="name" />
  <span>Hello, {{ name }}!</span>
  <p ng-bind="name"></p>
</body>
</html>

```

Hello, World! ◦ ◦

#### 1. Content Delivery NetworkAngular◦

```
<script src="https://code.angularjs.org/1.5.8/angular.min.js"></script>
```

#### 2. ng-appHTMLAngularng-app

```
<html ng-app>
```

#### 3. ng-initname

```
<body ng-init=" name = 'World' ">
```

*ng-init* ◦

#### 4. HTML ◦ `ng-model``<input>``name`

```
<input ng-model="name" />
```

#### 5. `{{ }}`

```
<span>Hello, {{ name }}</span>
```

#### 6. `ng-bind``"{{ }}"`

```
<span ng-bind="name"></span>
```

◦ ◦

`ng-bind` ◦ Hello, {{name}}`ng-bind` ◦ `ng-cloak` ◦

## Angular

### AngularJS

```
<!DOCTYPE html>
<html ng-app="myDemoApp">
  <head>
    <style>.started { background: gold; }</style>
    <script src="https://code.angularjs.org/1.5.8/angular.min.js"></script>
    <script>
      function MyDataService() {
        return {
          getWorlds: function getWorlds() {
            return ["this world", "another world"];
          }
        };
      }

      function DemoController(worldsService) {
        var vm = this;
        vm.messages = worldsService.getWorlds().map(function(w) {
          return "Hello, " + w + "!";
        });
      }

      function startup($rootScope, $window) {
        $window.alert("Hello, user! Loading worlds...");
        $rootScope.hasStarted = true;
      }

      angular.module("myDemoApp", [/* module dependencies go here */)
        .service("worldsService", [MyDataService])
        .controller("demoController", ["worldsService", DemoController])
        .config(function() {
```



```

        console.log('configuring application');
    })
    .run(["$rootScope", "$window", startup]);
</script>
</head>
<body ng-class="{ 'started': hasStarted }" ng-cloak>
  <div ng-controller="demoController as vm">
    <ul>
      <li ng-repeat="msg in vm.messages">{{ msg }}</li>
    </ul>
  </div>
</body>
</html>

```

1. ng-app="myDemoApp" **ngApp** DOM"myDemoApp"angular.module;
2. <script src="angular.min.js">**AngularJS**;

MyDataService DemoControllerstartup ◦

3. angular.module(...). ◦ ◦ module(...);

4. .service(...) **Angular Service**;

5. .controller(...) **Angular Controller**;

6. .config(...) ◦

7. .run(...) ◦ ◦

- Angularstartup\$rootScope;
- Angularstartup\$window;
- startup;

8. ng-classclass **ngClass** \$rootScopehasStarted

9. ng-cloak**AngularAngular** html“ {{ msg }} ”◦

10. ng-controller **AngularDOM**;

11. ng-repeat**AngularDOM** ;

12. {{ msg }};

AngularHTMLJavascript**ng-app ng-controllerng-if ng-repeat**. Angular\$scope ◦

\$scope ◦ ◦

◦

```

<div ng-app="myApp">
  <h1>Hello {{ name }}</h1>
</div>

```

## Angular\$scope ◦ Javascriptname\$scope

```
angular.module("myApp", [])
  .run(function($rootScope) {
    $rootScope.name = "World!";
  });
```

## Javascript\$scope ◦

### ◦ Angular\$scope ◦ ◦

```
<div ng-app="myApp">
  <div ng-controller="MyController">
    <h1>Hello {{ name }}</h1>
  </div>
</div>
```

## \$scope ◦

```
angular.module("myApp", [])
  .controller("MyController", function($scope) {
    $scope.name = "Mr Local!";
  });
```

## \$scope ◦ **controllerAs** ◦

## \$scope JavaScriptAngular\$scope ◦ ◦ \$scope.model ◦

## \$scope.model ◦

## Angular ◦ \$scopeAngular ◦

## Hello World ◦

## Angular 1DOM ◦ HTML ◦

## Angular{{ }} ◦

```
{{ 'Hello' + 'World' }}
```

```
HelloWorld
```

## NG-

## Angularng-app DOM ◦ Angular ◦ ng-app

```
<html>
  <head>
    <script src="/angular.js"></script>
```

```
</head>
<body ng-app>
  {{ 'Hello' + 'World' }}
</body>
</html>
```

body。

Angular DOM。 MisularMiskoAngular

“WebWeb。

HTMLAngular。 ng-app

- ng-click
- ng-hide
- <form> HTML。

Angular100。

AngularHTML。

## Angular

。

```
var app = angular.module('mainApp', []);
app.controller('FirstController', function($scope) {
  $scope.name= 'Hello World !';
});
```

。

```
var app=angular.module("mainApp",[]);app.controller("FirstController",function(e){e.name=
'Hello World !'})
```

minification\$ scope。 \$ scope'e'。 'e'。

```
var app = angular.module('mainApp', []);
app.controller('FirstController', ['$scope', function($scope) {
  $scope.message = 'Hello World !';
}]);
```

## \$ inject

```
FirstController.$inject = ['$scope'];
var FirstController = function($scope) {
  $scope.message = 'Hello World !';
```

```
}  
  
var app = angular.module('mainApp', []);  
app.controller('FirstController', FirstController);
```

```
var  
app=angular.module("mainApp", []);app.controller("FirstController",["$scope",function(a){a.message="Hello  
World !"}]);
```

angular'a\$ scope'Hello World'。

## AngularJS

[egghead.io](http://egghead.io)AngularJS



all



WATCH LUKAS RUEBBELKE'S COURSE

## Using Angular 2 Patterns in Angular 1.x Apps



Implementing modern component-based architecture in your new or existing Angular 1.x web application is a breath of fresh air.

In this course, y...

0 of 13 lessons

WATCH AARON FROST'S COURSE

## Introduction to Angular Material



Angular Material is an Angular native, UI component framework from Google. It is a reference implementation of Google's Material Design and provide...

0 of 7 lessons

WATCH KENT C. DODD'S COURSE

## AngularJS Authentication with JWT



JSON Web Tokens (JWT) are a more modern approach to authentication. As the web moves to a greater separation between the client and server, JWT pro...

0 of 7 lessons

WATCH JOEL HOOK'S COURSE

## Learn Protractor Testing for AngularJS



Protractor is an end-to-end testing framework for AngularJS applications. It allows you to do the browser and test the expected state of your ap...

0 of 10 lessons

- <https://egghead.io/courses/angularjs-application-architecture>
- <https://egghead.io/courses/angular-material-introduction>
- <https://egghead.io/courses/building-an-angular-1-x-ionic-application>
- <https://egghead.io/courses/angular-and-webpack-for-modular-applications>
- <https://egghead.io/courses/angularjs-authentication-with-jwt>
- <https://egghead.io/courses/angularjs-data-modeling>
- <https://egghead.io/courses/angular-automation-with-gulp>
- <https://egghead.io/courses/learn-protractor-testing-for-angularjs>
- <https://egghead.io/courses/ionic-quickstart-for-windows>
- <https://egghead.io/courses/build-angular-1-x-apps-with-redux>
- <https://egghead.io/courses/using-angular-2-patterns-in-angular-1-x-apps>

AngularJS <https://riptutorial.com/zh-CN/angularjs/topic/295/angularjs>

## 2: \$ http

### Examples

#### \$ http

\$httpHTTPpromise。

```
// Simple GET request example:
$http({
  method: 'GET',
  url: '/someUrl'
}).then(function successCallback(response) {
  // this callback will be called asynchronously
  // when the response is available
}, function errorCallback(response) {
  // called asynchronously if an error occurs
  // or server returns response with an error status.
});
```

```
appName.controller('controllerName',
  ['$http', function($http){

    // Simple GET request example:
    $http({
      method: 'GET',
      url: '/someUrl'
    }).then(function successCallback(response) {
      // this callback will be called asynchronously
      // when the response is available
    }, function errorCallback(response) {
      // called asynchronously if an error occurs
      // or server returns response with an error status.
    });
  }])
```

\$http。 [http](#)

```
$http.get('/someUrl', config).then(successCallback, errorCallback);
$http.post('/someUrl', data, config).then(successCallback, errorCallback);
```

- \$ http.get
- \$ http.head
- \$ http.post
- \$ http.put
- \$ http.delete
- \$ http.jsonp
- \$ http.patch

#### \$ http

## HTTPWeb◦

httpRequestsService.js

### httpRequestsService.js

```
appName.service('httpRequestsService', function($q, $http){

    return {
        // function that performs a basic get request
        getName: function(){
            // make sure $http is injected
            return $http.get("/someAPI/names")
                .then(function(response) {
                    // return the result as a promise
                    return response;
                }, function(response) {
                    // defer the promise
                    return $q.reject(response.data);
                });
        },

        // add functions for other requests made by your app
        addName: function(){
            // some code...
        }
    }
})
```

### get◦ ◦

```
appName.controller('controllerName',
    ['httpRequestsService', function(httpRequestsService){

        // we injected httpRequestsService service on this controller
        // that made the getName() function available to use.
        httpRequestsService.getName()
            .then(function(response){
                // success
            }, function(error){
                // do something with the error
            })
    }])
```

### httpRequestsService.js ◦

## \$ http

### \$ http◦ ◦

```
$scope.names = [];

$http({
    method: 'GET',
    url: '/someURL'
}).then(function successCallback(response) {
```



```

        $scope.names = response.data;
    },
    function errorCallback(response) {
        alert(response.status);
    });

    alert("The first name is: " + $scope.names[0]);

```

\$scope.names[0] **http** \$scope.names[0] - °

```

$scope.names = [];

$scope.$watch('names', function(newVal, oldVal) {
    if(!newVal.length == 0) {
        alert("The first name is: " + $scope.names[0]);
    }
});

$http({
    method: 'GET',
    url: '/someURL'
}).then(function successCallback(response) {
    $scope.names = response.data;
},
function errorCallback(response) {
    alert(response.status);
});

```

\$ **watch** \$scope.names° \$scope.names newVal.length0° - \$scope.names°

\$ **http** <https://riptutorial.com/zh-CN/angularjs/topic/3620/--http>

## 3: Angular \$

AngularAngularJS。 [docs.angularjs.org](https://docs.angularjs.org)

**\$rootScope**。 \$scope\$ scopeAngular Services。

### Examples

#### \$ scope

```
angular.module('app', [])
.controller('myController', ['$scope', function($scope){
    $scope.person = { name: 'John Doe' };
}]);

<div ng-app="app" ng-controller="myController">
  <input ng-model="person.name" />
  <div ng-repeat="number in [0,1,2,3]">
    {{person.name}} {{number}}
  </div>
</div>
```

ng-repeat。

myControllerperson。

JavaScript。

StringNumberBooleanSymbol

```
var x = 5;
var y = x;
y = 6;
console.log(y === x, x, y); //false, 5, 6
```

```
var x = { name : 'John Doe' };
var y = x;
y.name = 'Jhon';
console.log(x.name === y.name, x.name, y.name); //true, John, John
```

angular。

```
angular.module('app', [])
.controller('myController', ['$scope', function($scope){
    $scope.person = { name: 'John Doe' }; //non-primitive
    $scope.name = 'Jhon Doe'; //primitive
}])
.controller('myController1', ['$scope', function($scope){}]);

<div ng-app="app" ng-controller="myController">
```

```

binding to input works: {{person.name}}<br/>
binding to input does not work: {{name}}<br/>
<div ng-controller="myController1">
  <input ng-model="person.name" />
  <input ng-model="name" />
</div>
</div>

```

Angular \$scope.\$new() ◦

◦

◦ [Angular -](#) ◦

◦

rootScope ◦ - ◦

```

angular.module('app', [])
.run(['$rootScope', function($rootScope) {
  var messages = []
  $rootScope.addMessage = function(msg) {
    messages.push(msg);
  }
}]);

<div ng-app="app">
  <a ng-click="addMessage('hello world!')">it could be accessed from here</a>
  <div ng-include="inner.html"></div>
</div>

```

inner.html

```

<div>
  <button ng-click="addMessage('page!')">and from here to!</button>
</div>

```

## \$ scope

HTML \$ scopes ◦ \$ scope

```

$scope.$on('my-event', function(event, args) {
  console.log(args); // { custom: 'data' }
});

```

\$ on ◦

```

var unregisterMyEvent = $scope.$on('my-event', function(event, args) {
  console.log(args); // { custom: 'data' }
  unregisterMyEvent();
});

```

## \$ scope \$ broadcast \$ emit ◦ \$ emit

```
$scope.$emit('my-event', { custom: 'data' });
```

my-event **\$rootScope**stopPropagationstopPropagation ◦ **\$emit**stopPropagation

## \$ emit \$ broadcast \$ broadcast ◦

```
$scope.$broadcast('my-event', { custom: 'data' });
```

## \$ broadcast ◦

## \$ scope

\$rootScope \$ scope \$ scope ◦ ◦

```
myApp.controller('myController', ['$scope', function($scope) {
    $scope.myFunction = function () {
        alert("You are in myFunction!");
    };
}]);
```

```
$scope.myfunction();
```

## HTML

```
<div ng-controller="myController">
  <button ng-click="myFunction()"> Click me! </button>
</div>
```

```
myApp.directive('triggerFunction', function() {
    return {
        scope: {
            triggerFunction: '&'
        },
        link: function(scope, element) {
            element.bind('mouseover', function() {
                scope.triggerFunction();
            });
        }
    };
});
```

## HTML

```
<div ng-controller="myController">
  <button trigger-function="myFunction()"> Hover over me! </button>
</div>
```

ngMouseover ◦ \$ scope

Scope""。 AngularJSrootScope。 rootScope。

o o

- 1.
- 2.
3. {}

o o o o

```
var app = angular.module("test", []);

app.controller("Ctrl1", function($scope) {
    $scope.name = "Prateek";
    $scope.reverseName = function() {
        $scope.name = $scope.name.split('').reverse().join('');
    };
});

app.directive("myDirective", function() {
    return {
        restrict: "EA",
        scope: {},
        template: "<div>Your name is : {{name}}</div>" +
            "Change your name : <input type='text' ng-model='name' />"
    };
});
```

## AngularJS3

1. "@"/
2. "="/
3. ""/

```
<div my-directive
  class="directive"
  name="{{name}}"
  reverse="reverseName()"
  color="color" >
</div>
```

Angular \$ <https://riptutorial.com/zh-CN/angularjs/topic/3157/angular-->

# 4: angularjs

Angularjs

## Examples

### Angularjs

```
<div ng-app="MainApp" ng-controller="SampleController">
  <input ng-model="dishName" id="search" class="form-control" placeholder="Filter text">
  <ul>
    <li dir-paginate="dish in dishes | filter : dishName | itemsPerPage: pageSize"
    pagination-id="flights">{{dish}}</li>
  </ul>
  <dir-pagination-controls boundary-links="true" on-page-
  change="changeHandler(newPageNumber)" pagination-id="flights"></dir-pagination-controls>
</div>
<script type="text/javascript" src="angular.min.js"></script>
<script type="text/javascript" src="pagination.js"></script>
<script type="text/javascript">

var MainApp = angular.module('MainApp', ['angularUtils.directives.dirPagination'])
MainApp.controller('SampleController', ['$scope', '$filter', function ($scope, $filter) {

  $scope.pageSize = 5;

  $scope.dishes = [
    'noodles',
    'sausage',
    'beans on toast',
    'cheeseburger',
    'battered mars bar',
    'crisp butty',
    'yorkshire pudding',
    'wiener schnitzel',
    'sauerkraut mit ei',
    'salad',
    'onion soup',
    'bak choy',
    'avacado maki'
  ];

  $scope.changeHandler = function (newPage) { };
}]);
</script>
```

[angularjs https://riptutorial.com/zh-CN/angularjs/topic/10821/angularjs-](https://riptutorial.com/zh-CN/angularjs/topic/10821/angularjs-)

# 5: AngularJS `=` ``@``

[plunker](#).

## Examples

@.

◦

parens ◦ ◦ ◦

```
<one-way text="Simple text." <!-- 'Simple text.' -->
  simple-value="123" <!-- '123' Note, is actually a string object. -->
  interpolated-value="{{parentScopeValue}}" <!-- Some value from parent scope. You
can't change parent scope value, only child scope value. Note, is actually a string object. --
>
  interpolated-function-value="{{parentScopeFunction()}}" <!-- Executes parent scope
function and takes a value. -->

  <!-- Unexpected usage. -->
  object-item="{{objectItem}}" <!-- Converts object|date to string. Result might be:
'{"a":5, "b":"text"}'. -->
  function-item="{{parentScopeFunction}}" > <!-- Will be an empty string. -->
</one-way>
```

=.

◦ {{...}}.

```
<two-way text="'Simple text.'" <!-- 'Simple text.' -->
  simple-value="123" <!-- 123 Note, is actually a number now. -->
  interpolated-value="parentScopeValue" <!-- Some value from parent scope. You may
change it in one scope and have updated value in another. -->
  object-item="objectItem" <!-- Some object from parent scope. You may change object
properties in one scope and have updated properties in another. -->

  <!-- Unexpected usage. -->
  interpolated-function-value="parentScopeFunction()" <!-- Will raise an error. -->
  function-item="incrementInterpolated" > <!-- Pass the function by reference and you
may use it in child scope. -->
</two-way>
```

◦

◦

◦ ◦ ◦ {{...}} ◦ ◦

```
<expression-binding interpolated-function-value="incrementInterpolated(param)" <!--
```

```

interpolatedFunctionValue({param: 'Hey'}) will call passed function with an argument. -->
    function-item="incrementInterpolated" <!-- functionItem({param: 'Hey'}) ()
will call passed function, but with no possibility set up a parameter. -->
    text="'Simple text.'" <!-- text() == 'Simple text.'-->
    simple-value="123" <!-- simpleValue() == 123 -->
    interpolated-value="parentScopeValue" <!-- interpolatedValue() == Some
value from parent scope. -->
    object-item="objectItem"> <!-- objectItem() == Object item from parent
scope. -->
</expression-binding>

```

◦

```

angular.component("SampleComponent", {
  bindings: {
    title: '@',
    movies: '<',
    reservation: "=",
    processReservation: "&"
  }
});

```

◦

@ ◦ ◦

< ◦ ◦

= ◦ ◦

◦ ◦

```

bindings: {
  mandatory: '=',
  optional: '=?',
  foo: '=?bar'
}

```

==?bar ◦ (\$compile:nonassign)◦

AngularJS`=@` <https://riptutorial.com/zh-CN/angularjs/topic/6149/angularjs>-----



# 6: AngularJS

## Examples

1. AngularJavaScript ◦
2. ◦ ◦
3. ng-repeat ng-switch ng-view ng-if ng-controller ng-include◦

◦ **“” AngularJS.**

1. ”. ”HTML
2. controllerAs“”
3. \$ parentscope◦ ng-ifng-model="\$parent.foo" ..

---

ngModelgetter / setter◦ getter / setterng-model-options="{ getterSetter: true }"ngModalgetter ◦

```
<div ng-app="myApp" ng-controller="MainCtrl">
  <input type="text" ng-model="foo" ng-model-options="{ getterSetter: true }">
  <div ng-if="truthyValue">
    <!-- I'm a child scope (inside ng-if), but i'm synced with changes from the outside
scope -->
    <input type="text" ng-model="foo">
  </div>
  <div>${scope.foo}: {{ foo() }}</div>
</div>
```

```
angular.module('myApp', []).controller('MainCtrl', ['$scope', function($scope) {
  $scope.truthyValue = true;

  var _foo = 'hello'; // this will be used to cache/represent the value of the 'foo' model

  $scope.foo = function(val) {
    // the function return the the internal '_foo' varibale when called with zero
arguments,
    // and update the internal `_foo` when called with an argument
    return arguments.length ? (_foo = val) : _foo;
  };
}]);
```

Angular ◦

## html5Mode

html5Mode ([mode])

1. index.htmlURL<base href=""> ◦
2. baseUrl◦ - "Resource interpreted as stylesheet but transferred with MIME type text/html" ◦

```

<head>
  <meta charset="utf-8">
  <title>Job Seeker</title>

  <base href="/">

  <link rel="stylesheet" href="bower_components/bootstrap/dist/css/bootstrap.css" />
  <link rel="stylesheet" href="/styles/main.css">
</head>

```

### 3. `base$locationProvider$locationProviderbase.requireBase:false$locationProvider.html5Mode()`

```

$locationProvider.html5Mode({
  enabled: true,
  requireBase: false
});

```

## 4. HTML5 URL。 [AngularJS](#) // \$location

URLindex.html。 `<base>Angularurl。`

### [ui-router](#) HTTP- `html5Mode`。 Apache

```

RewriteEngine on

# Don't rewrite files or directories
RewriteCond %{REQUEST_FILENAME} -f [OR]
RewriteCond %{REQUEST_FILENAME} -d
RewriteRule ^ - [L]

# Rewrite everything else to index.html to allow html5 state links
RewriteRule ^ index.html [L]

```

## nginx

```

server {
    server_name my-app;

    root /path/to/app;

    location / {
        try_files $uri $uri/ /index.html;
    }
}

```

```

var express = require('express');
var app = express();

app.use('/js', express.static(__dirname + '/js'));
app.use('/dist', express.static(__dirname + '/../dist'));
app.use('/css', express.static(__dirname + '/css'));
app.use('/partials', express.static(__dirname + '/partials'));

app.all('*', function(req, res, next) {
    // Just send the index.html for other files to support HTML5Mode

```

```
res.sendFile('index.html', { root: __dirname });
});

app.listen(3006); //the port you want to use
```

## 7 AngularJS

AngularJS。

### 1.DOM

。 。 DOM。 。 。 DOM。 。

```
link: function($scope, element, attrs) {
  //The best place to manipulate DOM
}
```

DOMelement angular.element() Javascript。

### 2.

AngularJS。 。 AngularJS。 。

```
<my-dir>
  <my-transclusion>
  </my-transclusion>
</my-dir>
```

。

```
<my-dir>
  <my-transclusion>
    <input ng-model="name">
  </my-transclusion>
</my-dir>
```

。 transclusionname。 **\$ parent.name**。 。 。

```
$scope.data = {
  name: 'someName'
}
```

'data'

```
<input ng-model="data.name">
```

。

### 3.

- ◦ myDirAmyDirBisoletedmyDirC

```
<input my-dir-a my-dir-c>
```

```
<input my-dir-a my-dir-b>
```

◦

#### 4.\$ emit

\$ emit\$ broadcast\$ on - ◦ ◦ A'someEvent'B.

```
$scope.$emit('someEvent', args);
```

'someEvent'

```
$scope.$on('someEvent', function(){});
```

B. \$ emit.

#### 5.\$ scope. \$ watch

\$ scope. \$ watch. ◦ \$ scope. \$ watch. \$ digest.

```
$scope.$watch('myCtrl.myVariable', function(newVal) {  
    this.myVariable++;  
});
```

myVariablenewVal.

#### 6.

- AngularJS. ◦ ng-clickng-blurng-on-changeparemeter.

```
<input ng-disabled="myCtrl.isDisabled()" ng-model="myCtrl.name">
```

isDisabled. myCtrl

```
vm.isDisabled = function(){  
    if(someCondition)  
        return true;  
    else  
        return false;  
}
```

◦ ◦

```
vm.isDisabled
```

```
if(someCondition)
  vm.isDisabled = true
else
  vm.isDisabled = false
```

◦

```
<input ng-disabled="myCtrl.isDisabled" ng-model="myCtrl.name">
```

◦

## 7. Angular

AngularJS ◦

1. **angular.forEach** for the loops“”◦
2. **DOMangular.element**
3. **angular.copy**
4. ◦ ◦
5. Chrome◦
6. **Batarang** ◦ Chrome

◦

**AngularJS** <https://riptutorial.com/zh-CN/angularjs/topic/3208/angularjs>

# 7: Angular\$ q

## Examples

### \$ q.allpromise

promise.then\$q.all\$q.all◦

### JS

```
$scope.data = []

$q.all([
  $http.get("data.json"),
  $http.get("more-data.json"),
]).then(function(responses) {
  $scope.data = responses.map((resp) => resp.data);
});
```

json\$http.get 2getpromisespromise .thenresponsespromises◦

### HTML

```
<ul>
  <li ng-repeat="d in data">
    <ul>
      <li ng-repeat="item in d">{{item.name}}: {{item.occupation}}</li>
    </ul>
  </li>
</ul>
```

### JSON

```
[{
  "name": "alice",
  "occupation": "manager"
}, {
  "name": "bob",
  "occupation": "developer"
}]
```

### \$ qpromise

\$qAPIpromise◦

\$ q{...}

resolvereject promise◦

### 1

```
function $timeout(fn, delay) {
  return = $q(function(resolve, reject) {
    setTimeout(function() {
      try {
        let r = fn();
        resolve(r);
      }
      catch (e) {
        reject(e);
      }
    }, delay);
  });
}
```

[WindowTimers.setTimeout API](#) [promise](#)。 [AngularJS](#)。 [AngularJS \\$ timeoutAPI](#)。

## 2

```
$scope.divide = function(a, b) {
  return $q(function(resolve, reject) {
    if (b===0) {
      return reject("Cannot devide by 0")
    } else {
      return resolve(a/b);
    }
  });
}
```

[promisifiedpromise](#)。

.then

```
$scope.divide(7, 2).then(function(result) {
  // will return 3.5
}, function(err) {
  // will not run
})

$scope.divide(2, 0).then(function(result) {
  // will not run as the calculation will fail on a divide by 0
}, function(err) {
  // will return the error string.
})
```

## \$ q.defer

[\\$qpromise](#) [\\$q.deferpromise](#)。

[\\$q\\$q.deferpromise](#)。

```
var runAnimation = function(animation, duration) {
  var deferred = $q.defer();
  try {
    ...
```

```

        // run some animation for a given duration
        deferred.resolve("done");
    } catch (err) {
        // in case of error we would want to run the error handler of .then
        deferred.reject(err);
    }
    return deferred.promise;
}

// and then
runAnimation.then(function(status) {}, function(error) {})

```

1. deferred.promise.then

2. .then

## \$q

\$q°

\$q\$scope.ScopeUI°

getMyData **promise**° resolved° rejected **2**°

## Angular

```

function getMyData($timeout, $q) {
    return function() {
        // simulated async function
        var promise = $timeout(function() {
            if(Math.round(Math.random())) {
                return 'data received!'
            } else {
                return $q.reject('oh no an error! try again')
            }
        }, 2000);
        return promise;
    }
}

```

## Promise

```

angular.module('app', [])
.factory('getMyData', getMyData)
.run(function(getData) {
    var promise = getData()
    .then(function(string) {
        console.log(string)
    }, function(error) {
        console.error(error)
    })
    .finally(function() {
        console.log('Finished at:', new Date())
    })
})

```



```
})
```

**promises** \$q ◦ getMyData \$q ◦

```
var defer = $q.defer();
```

`$q.defer()` **deferred**

**promise** `promise` ◦ `$q.deferred()` `resolve()` `reject()` `notify()` ◦

- `resolve(value)` - **promise** ◦
- `reject(reason)` - ◦
- `notify(value)` - **promise** ◦ ◦

**promise** `promise` ◦ `promise` - {**Promise**} - `promise` ◦

**promise** `deferred.promise` ◦

`promise` ◦

-

- `then(successCallback, [errorCallback], [notifyCallback])` - ◦ ◦ ◦
- `catch(errorCallback)` - `catch(errorCallback)` **null** `errorCallback`
- `finally(callback, notifyCallback)` - ◦

◦ ◦

**1**

```
// Creates a promise that when resolved, returns 4.
function getNumbers() {

    var promise = $timeout(function() {
        return 4;
    }, 1000);

    return promise;
}

// Resolve getNumbers() and chain subsequent then() calls to decrement
// initial number from 4 to 0 and then output a string.
getNumbers()
    .then(function(num) {
        // 4
        console.log(num);
        return --num;
    })
    .then(function (num) {
        // 3
        console.log(num);
```

```

    return --num;
  })
  .then(function (num) {
    // 2
    console.log(num);
    return --num;
  })
  .then(function (num) {
    // 1
    console.log(num);
    return --num;
  })
  .then(function (num) {
    // 0
    console.log(num);
    return 'And we are done!';
  })
  .then(function (text) {
    // "And we are done!"
    console.log(text);
  });

```

## \$ q.whenpromise

### promise

```

//OVERLY VERBOSE
var defer;
defer = $q.defer();
defer.resolve(['one', 'two']);
return defer.promise;

```

```

//BETTER
return $q.when(['one', 'two']);

```

## \$ q.when\$ q.resolve

\$ q◦ promisepromise◦

[- AngularJS \\$ qAPI - \\$ q.when](#)

## AngularJS v1.4.1

### ES6resolve

```

//ABSOLUTELY THE SAME AS when
return $q.resolve(['one', 'two'])

```

## \$ q

```

var myDeferred = $q.defer();

```

```
$http(config).then(function(res) {
  myDeferred.resolve(res);
}, function(error) {
  myDeferred.reject(error);
});

return myDeferred.promise;
```

\$ http\$q.defer◦

```
//INSTEAD
return $http(config);
```

\$ http◦

Angular\$ q <https://riptutorial.com/zh-CN/angularjs/topic/4379/angular--q>

# 8: HTTP

AngularJS\$ httpHTTP。 。 。 http。 。

## Examples

Angular\$httpHTTP。 。

### httpInterceptor

#### HTML

```
<!DOCTYPE html>
<html>
<head>
  <title>Angular Interceptor Sample</title>
  <script src="https://code.angularjs.org/1.5.8/angular.min.js"></script>
  <script src="app.js"></script>
  <script src="appController.js"></script>
  <script src="genericInterceptor.js"></script>
</head>
<body ng-app="interceptorApp">
  <div ng-controller="appController as vm">
    <button ng-click="vm.sendRequest()">Send a request</button>
  </div>
</body>
</html>
```

#### “app.js” JavaScript

```
var interceptorApp = angular.module('interceptorApp', []);

interceptorApp.config(function($httpProvider) {
  $httpProvider.interceptors.push('genericInterceptor');
});
```

#### 'appController.js'

```
(function() {
  'use strict';

  function appController($http) {
    var vm = this;

    vm.sendRequest = function() {
      $http.get('http://google.com').then(function(response) {
        console.log(response);
      });
    };
  };
}

angular.module('interceptorApp').controller('appController', ['$http', appController]);
```

```
})();
```

## 'genericInterceptor.js'

```
(function() {  
  "use strict";  
  
  function genericInterceptor($q) {  
    this.responseError = function (response) {  
      return $q.reject(response);  
    };  
  
    this.requestError = function(request){  
      if (canRecover(rejection)) {  
        return responseOrNewPromise  
      }  
      return $q.reject(rejection);  
    };  
  
    this.response = function(response){  
      return response;  
    };  
  
    this.request = function(config){  
      return config;  
    }  
  }  
  
  angular.module('interceptorApp').service('genericInterceptor', genericInterceptor);  
})();
```

## 'genericInterceptor'.

## httpFlash

### htmlhtmlindex.htmldivflashdiv

```
<div class="flashmessage" ng-if="isVisible">  
  {{flashMessage}}  
</div>
```

## angularconfighttpProviderhttpProviderrootScope.

```
var interceptorTest = angular.module('interceptorTest', []);  
  
interceptorTest.config(['$httpProvider',function ($httpProvider) {  
  
  $httpProvider.interceptors.push(["$rootScope",function ($rootScope) {  
    return { //intercept only the response  
      'response': function (response)  
        {  
  
$rootScope.showFeedBack(response.status,response.data.message);  
  
        return response;  
      }  
    }  
  }]);
```

```
        });  
    });  
})
```

`httpProvider.rootScope.run(function($rootScope,`

`$timeout,flash, 3000,`

```
interceptorTest.run(["$rootScope", "$timeout", function($rootScope, $timeout) {  
    $rootScope.showFeedBack = function(status, message) {  
  
        $rootScope.isVisible = true;  
        $rootScope.flashMessage = message;  
        $timeout(function() { $rootScope.isVisible = false }, 3000)  
    }  
}]);
```

**\$rootScope**。

**HTTP** <https://riptutorial.com/zh-CN/angularjs/topic/6484/http>

---

# 9: ng-class

## Examples

### ng-ng

Angular`ng-class`

---

## 1.

```
<span ng-class="MyClass">Sample Text</span>
```

Angular`$ scope`. Angular`$ scope`"MyClass". "MyClass"`<span>`

```
$scope.MyClass = "bold-red deleted error";
```

Angular`MyClass``$ scope`. "bold-red""deleted""error"`<span>`

`<span>` `$scope` HTML`ng-class`

---

---

## 2.

`ng-class`

`true`

```
<style>
  .red { color: red; font-weight: bold; }
  .blue { color: blue; }
  .green { color: green; }
  .highlighted { background-color: yellow; color: black; }
</style>
```

```
<span ng-class="{ red: ShowRed, blue: ShowBlue, green: ShowGreen, highlighted: IsHighlighted }">Sample Text</span>
```

```
<div>Red: <input type="checkbox" ng-model="ShowRed"></div>
<div>Green: <input type="checkbox" ng-model="ShowGreen"></div>
<div>Blue: <input type="checkbox" ng-model="ShowBlue"></div>
<div>Highlight: <input type="checkbox" ng-model="IsHighlighted"></div>
```

---

---

## 3.

## 12.

```
<style>
  .bold { font-weight: bold; }
  .strike { text-decoration: line-through; }
  .orange { color: orange; }
</style>

<p ng-class="[ UserStyle, {orange: warning} ]">Array of Both Expression Types</p>
<input ng-model="UserStyle" placeholder="Type 'bold' and/or 'strike'"><br>
<label><input type="checkbox" ng-model="warning"> warning (apply "orange" class)</label>
```

UserStyle◦ <p>◦ warning◦ <p>◦

**ng-class** <https://riptutorial.com/zh-CN/angularjs/topic/2395/ng-class>



# 10: NG-

ng-viewangular。 {info} ngRouteangular.jsjavascriptangular-route.js。 \$routeProvider“when”。  
templateUrlcontroller。

## Examples

### NG-

ng-view\$route。 Index.html“/”templateURL home.htmlIndex.htmlng-view。

```
angular.module('ngApp', ['ngRoute'])

.config(function($routeProvider) {
  $routeProvider.when("/",
    {
      templateUrl: "home.html",
      controller: "homeCtrl"
    }
  );
});

angular.module('ngApp').controller('homeCtrl', ['$scope', function($scope) {
  $scope.welcome= "Welcome to stackoverflow!";
}]);

//Index.html
<body ng-app="ngApp">
  <div ng-view></div>
</body>

//Home Template URL or home.html
<div><h2>{{welcome}}</h2></div>
```

1.

```
var Registration=angular.module("myApp", ["ngRoute"]);
```

2. “ngRoute”\$routeProvider

```
Registration.config(function($routeProvider) {
});
```

3. “/ add”“/ add”regi.htm

```
Registration.config(function($routeProvider) {
  $routeProvider
  .when("/add", {
    templateUrl : "regi.htm"
  })
});
```

```
});
```

NG- <https://riptutorial.com/zh-CN/angularjs/topic/8833/ng->

# 11: NG-

ngRepeat ◦ ◦ \$index ◦

- `<element ng-repeat="expression"></element>`
- `<div ng-repeat="(key, value) in myObj">...</div>`
- `<div ng-repeat="variable in expression">...</div>`

\$index	<code>0..length-1</code>
\$first	<i>boolean</i> <code>true</code> ◦
\$middle	<i>boolean</i> <code>true</code> ◦
\$last	<i>boolean</i> <code>true</code> ◦
\$even	<i>boolean</i> <code>\$index \$index truefalse</code> ◦
\$odd	<i>boolean</i> <code>\$index \$index truefalse</code> ◦

AngularJSng-repeatng-repeatHTML ◦

## Examples

```
<div ng-repeat="(key, value) in myObj"> ... </div>
```

```
<div ng-repeat="n in [42, 42, 43, 43]">
  {{n}}
</div>
```

ngRepeat [\\$ watchCollection](#) ◦ ngRepeat DOM

- DOM ◦
- DOM ◦
- DOM ◦
  
- track by ◦
- track by ◦
- track by `[ngRepeat:dupes]`

```
$scope.numbers = ['1','1','2','3','4'];

<ul>
  <li ng-repeat="n in numbers track by $index">
    {{n}}
  </li>
</ul>
```

```
</li>  
</ul>
```

## ng-repeat-start + ng-repeat-end

### AngularJS 1.2 ng-repeatng-repeat-startng-repeat-end

```
// table items  
$scope.tableItems = [  
  {  
    row1: 'Item 1: Row 1',  
    row2: 'Item 1: Row 2'  
  },  
  {  
    row1: 'Item 2: Row 1',  
    row2: 'Item 2: Row 2'  
  }  
];  
  
// template  
<table>  
  <th>  
    <td>Items</td>  
  </th>  
  <tr ng-repeat-start="item in tableItems">  
    <td ng-bind="item.row1"></td>  
  </tr>  
  <tr ng-repeat-end>  
    <td ng-bind="item.row2"></td>  
  </tr>  
</table>
```

11

12

21

22

NG- <https://riptutorial.com/zh-CN/angularjs/topic/8118/ng->

---

# 12: NG-

'ngStyle'HTMLCSS。 AngularJSHTMLstyle/angularjsng-style

- `<ANY ng-style="expression"></ANY >`
- `<ANY class="ng-style: expression;"> ... </ANY>`

## Examples

ng

“status”。

```

```

NG- <https://riptutorial.com/zh-CN/angularjs/topic/8773/ng->

# 13: SignalRAngularJs

“AngularJsSignalR”“angularjs”“/”SignalR“<https://www.codeproject.com/Tips/590660/Introduction-to-SignalR>。”

## Examples

### SignalRAngularJs [ChatProject]

#### 1

```
- Application
  - app.js
  - Controllers
    - appController.js
  - Factories
    - SignalR-factory.js
- index.html
- Scripts
  - angular.js
  - jquery.js
  - jquery.signalR.min.js
- Hubs
```

SignalRsignalR-2.2.1

#### 2Startup.csChatHub.cs

“/ Hubs”2[Startup.csChatHub.cs]

#### Startup.cs

```
using Microsoft.Owin;
using Owin;
[assembly: OwinStartup(typeof(SignalR.Hubs.Startup))]

namespace SignalR.Hubs
{
    public class Startup
    {
        public void Configuration(IAppBuilder app)
        {
            app.MapSignalR();
        }
    }
}
```

#### ChatHub.cs

```
using Microsoft.AspNet.SignalR;

namespace SignalR.Hubs
```

```

{
  public class ChatHub : Hub
  {
    public void Send(string name, string message, string time)
    {
      Clients.All.broadcastMessage(name, message, time);
    }
  }
}

```

### 3

*“/ Application”[app.js]*

#### app.js

```
var app = angular.module("app", []);
```

### 4SignalR

*“/ Application / Factories”[SignalR-factory.js]*

#### SignalR-factory.js

```

app.factory("signalR", function () {
  var factory = {};

  factory.url = function (url) {
    $.connection.hub.url = url;
  }

  factory.setHubName = function (hubName) {
    factory.hub = hubName;
  }

  factory.connectToHub = function () {
    return $.connection[factory.hub];
  }

  factory.client = function () {
    var hub = factory.connectToHub();
    return hub.client;
  }

  factory.server = function () {
    var hub = factory.connectToHub();
    return hub.server;
  }

  factory.start = function (fn) {
    return $.connection.hub.start().done(fn);
  }

  return factory;
});

```

#### 5app.js.

```

var app = angular.module("app", []);

app.run(function(signalR) {
    signalR.url("http://localhost:21991/signalr");
});

```

## localhost21991 / signalr | SignalR Hubs Urls

### 6

#### “/ Application / Controllers”[appController.js]

```

app.controller("ctrl", function ($scope, signalR) {
    $scope.messages = [];
    $scope.user = {};

    signalR.setHubName("chatHub");

    signalR.client().broadcastMessage = function (name, message, time) {
        var newChat = { name: name, message: message, time: time };

        $scope.$apply(function() {
            $scope.messages.push(newChat);
        });
    };

    signalR.start(function () {
        $scope.send = function () {
            var dt = new Date();
            var time = dt.getHours() + ":" + dt.getMinutes() + ":" + dt.getSeconds();

            signalR.server().send($scope.user.name, $scope.user.message, time);
        }
    });
});

```

signalR.setHubName“chatHub”| [ChatHub]> ChatHub.cs

HubName Case。

signalR.client |“chatHub”“broadcastMessage”;

### 7index.html

#### index.html

```

<!DOCTYPE html>
<html ng-app="app" ng-controller="ctrl">
<head>
    <meta charset="utf-8" />
    <title>SignalR Simple Chat</title>
</head>
<body>
    <form>
        <input type="text" placeholder="name" ng-model="user.name" />
        <input type="text" placeholder="message" ng-model="user.message" />
        <button ng-click="send()">send</button>
    </form>

```



```
<ul>
  <li ng-repeat="item in messages">
    <b ng-bind="item.name"></b> <small ng-bind="item.time"></small> :
    {{item.message}}
  </li>
</ul>
</form>

<script src="Scripts/angular.min.js"></script>
<script src="Scripts/jquery-1.6.4.min.js"></script>
<script src="Scripts/jquery.signalR-2.2.1.min.js"></script>
<script src="signalr/hubs"></script>
<script src="app.js"></script>
<script src="SignalR-factory.js"></script>
</body>
</html>
```

1

2

SignalRAngularJs <https://riptutorial.com/zh-CN/angularjs/topic/9964/signalrangularjs>

# 14: UI

## ui-router

Angular UI-RouterAngularJS。

SPAURL。 URLSPA。

UI-Router。 UI-Router。

API。 ui-router VS ng-router。 ng-routerngNewRouterng-routerAngular 1.5 + / 2.0ui-router。  
ngNewRouter。

## Examples

### app.js

```
angular.module('myApp', ['ui.router'])
  .controller('controllerOne', function() {
    this.message = 'Hello world from Controller One!';
  })
  .controller('controllerTwo', function() {
    this.message = 'Hello world from Controller Two!';
  })
  .controller('controllerThree', function() {
    this.message = 'Hello world from Controller Three!';
  })
  .config(function($stateProvider, $urlRouterProvider) {
    $stateProvider
      .state('one', {
        url: "/one",
        templateUrl: "view-one.html",
        controller: 'controllerOne',
        controllerAs: 'ctrlOne'
      })
      .state('two', {
        url: "/two",
        templateUrl: "view-two.html",
        controller: 'controllerTwo',
        controllerAs: 'ctrlTwo'
      })
      .state('three', {
        url: "/three",
        templateUrl: "view-three.html",
        controller: 'controllerThree',
        controllerAs: 'ctrlThree'
      });

    $urlRouterProvider.otherwise('/one');
  });
```

### index.html

```

<div ng-app="myApp">
  <nav>
    <!-- links to switch routes -->
    <a ui-sref="one">View One</a>
    <a ui-sref="two">View Two</a>
    <a ui-sref="three">View Three</a>
  </nav>
  <!-- views will be injected here -->
  <div ui-view></div>
  <!-- templates can live in normal html files -->
  <script type="text/ng-template" id="view-one.html">
    <h1>{{ctrlOne.message}}</h1>
  </script>

  <script type="text/ng-template" id="view-two.html">
    <h1>{{ctrlTwo.message}}</h1>
  </script>

  <script type="text/ng-template" id="view-three.html">
    <h1>{{ctrlThree.message}}</h1>
  </script>
</div>

```

## app.js

```

angular.module('myApp', ['ui.router'])
  .controller('controllerOne', function() {
    this.message = 'Hello world from Controller One!';
  })
  .controller('controllerTwo', function() {
    this.message = 'Hello world from Controller Two!';
  })
  .controller('controllerThree', function() {
    this.message = 'Hello world from Controller Three!';
  })
  .controller('controllerFour', function() {
    this.message = 'Hello world from Controller Four!';
  })
  .config(function($stateProvider, $urlRouterProvider) {
    $stateProvider
      .state('one', {
        url: "/one",
        views: {
          "viewA": {
            templateUrl: "view-one.html",
            controller: 'controllerOne',
            controllerAs: 'ctrlOne'
          },
          "viewB": {
            templateUrl: "view-two.html",
            controller: 'controllerTwo',
            controllerAs: 'ctrlTwo'
          }
        }
      })
      .state('two', {
        url: "/two",
        views: {
          "viewA": {
            templateUrl: "view-three.html",

```

```

        controller: 'controllerThree',
        controllerAs: 'ctrlThree'
    },
    "viewB": {
        templateUrl: "view-four.html",
        controller: 'controllerFour',
        controllerAs: 'ctrlFour'
    }
}
});

$urlRouterProvider.otherwise('/one');
});

```

## index.html

```

<div ng-app="myApp">
  <nav>
    <!-- links to switch routes -->
    <a ui-sref="one">Route One</a>
    <a ui-sref="two">Route Two</a>
  </nav>
  <!-- views will be injected here -->
  <div ui-view="viewA"></div>
  <div ui-view="viewB"></div>
  <!-- templates can live in normal html files -->
  <script type="text/ng-template" id="view-one.html">
    <h1>{{ctrlOne.message}}</h1>
  </script>

  <script type="text/ng-template" id="view-two.html">
    <h1>{{ctrlTwo.message}}</h1>
  </script>

  <script type="text/ng-template" id="view-three.html">
    <h1>{{ctrlThree.message}}</h1>
  </script>

  <script type="text/ng-template" id="view-four.html">
    <h1>{{ctrlFour.message}}</h1>
  </script>
</div>

```

## resolve

### app.js

```

angular.module('myApp', ['ui.router'])
  .service('User', ['$http', function User ($http) {
    this.getProfile = function (id) {
      return $http.get(...) // method to load data from API
    };
  }])
  .controller('profileCtrl', ['profile', function profileCtrl (profile) {
    // inject resolved data under the name of the resolve function
    // data will already be returned and processed
    this.profile = profile;
  }])

```

```

.config(['$stateProvider', '$urlRouterProvider', function ($stateProvider,
$urlRouterProvider) {
  $stateProvider
    .state('profile', {
      url: "/profile/:userId",
      templateUrl: "profile.html",
      controller: 'profileCtrl',
      controllerAs: 'vm',
      resolve: {
        profile: ['$stateParams', 'User', function ($stateParams, User) {
          // $stateParams will contain any parameter defined in your url
          return User.getProfile($stateParams.userId)
          // .then is only necessary if you need to process returned data
          .then(function (data) {
            return doSomeProcessing(data);
          });
        }]
      }
    });

  $urlRouterProvider.otherwise('/');
});

```

## profile.html

```

<ul>
  <li>Name: {{vm.profile.name}}</li>
  <li>Age: {{vm.profile.age}}</li>
  <li>Sex: {{vm.profile.sex}}</li>
</ul>

```

## [UI-Router Wiki](#) ◦

\$stateProvider ResolveUI ◦ UI ◦ UI ◦

/

## app.js

```

var app = angular.module('myApp', ['ui.router']);

app.config(function ($stateProvider, $urlRouterProvider) {

  $stateProvider

    .state('home', {
      url: '/home',
      templateUrl: 'home.html',
      controller: function ($scope) {
        $scope.text = 'This is the Home'
      }
    })

    .state('home.nested1', {
      url: '/nested1',
      templateUrl: 'nested1.html',
      controller: function ($scope) {

```

```

        $scope.text1 = 'This is the nested view 1'
    }
})

.state('home.nested2',{
    url: '/nested2',
    templateUrl:'nested2.html',
    controller: function($scope){
        $scope.fruits = ['apple','mango','oranges'];
    }
});

$urlRouterProvider.otherwise('/home');

});

```

## index.html

```

<div ui-view></div>
<script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.5.8/angular.min.js"></script>
<script src="angular-ui-router.min.js"></script>
<script src="app.js"></script>

```

## home.html

```

<div>
<h1> {{text}} </h1>
<br>
<a ui-sref="home.nested1">Show nested1</a>
<br>
<a ui-sref="home.nested2">Show nested2</a>
<br>

<div ui-view></div>
</div>

```

## nested1.html

```

<div>
<h1> {{text1}} </h1>
</div>

```

## nested2.html

```

<div>
<ul>
<li ng-repeat="fruit in fruits">{{ fruit }}</li>
</ul>
</div>

```

UI <https://riptutorial.com/zh-CN/angularjs/topic/2545/ui>

# 15:

## Examples

### angularjs

o

```
'use strict';

/**
 * @ngdoc factory
 * @name app.factory:storageService
 * @description This function will communicate with HTML5 sessionStorage via Factory Service.
 */

app.factory('storageService', ['$rootScope', function($rootScope) {

    return {
        get: function(key) {
            return sessionStorage.getItem(key);
        },
        save: function(key, data) {
            sessionStorage.setItem(key, data);
        }
    };
}]);
```

### storageService

```
app.controller('myCtrl', ['storageService', function(storageService) {

    // Save session data to storageService
    storageService.save('key', 'value');

    // Get saved session data from storageService
    var sessionData = storageService.get('key');

}]);
```

<https://riptutorial.com/zh-CN/angularjs/topic/8201/>

# 16: ES6

## Examples

### FileSizeES6

costum

```
let fileSize=function (size,unit,fixedDigit) {
return size.toFixed(fixedDigit) + ' '+unit;
};

let fileSizeFilter=function () {
return function (size) {
if (isNaN(size))
size = 0;

if (size < 1024)
return size + ' octets';

size /= 1024;

if (size < 1024)
return fileSize(size,'Ko',2);

size /= 1024;

if (size < 1024)
return fileSize(size,'Mo',2);

size /= 1024;

if (size < 1024)
return fileSize(size,'Go',2);

size /= 1024;
return fileSize(size,'To',2);
};
};
export default fileSizeFilter;
```

```
import fileSizeFilter from 'path...';
let myMainModule =
angular.module('mainApp', [])
.filter('fileSize', fileSizeFilter);
```

html

```
<div ng-app="mainApp">

<div>
<input type="text" ng-model="size" />
</div>
<div>
```



```
<h3>Output:</h3>
  <p>{{size| Filesize}}</p>
</div>
</div>
```

ES6 <https://riptutorial.com/zh-CN/angularjs/topic/9421/es6>

# 17: ngModelController

## Examples

```
<rating min="0" max="5" nullifier="true" ng-model="data.rating"></rating>
```

CSS;

```
0 1 2 3 4 5 x
```

;"x"null。

```
app.directive('rating', function() {

    function RatingController() {
        this._ngModel = null;
        this.rating = null;
        this.options = null;
        this.min = typeof this.min === 'number' ? this.min : 1;
        this.max = typeof this.max === 'number' ? this.max : 5;
    }

    RatingController.prototype.setNgModel = function(ngModel) {
        this._ngModel = ngModel;

        if( ngModel ) {
            // KEY POINT 1
            ngModel.$render = this._render.bind(this);
        }
    };

    RatingController.prototype._render = function() {
        this.rating = this._ngModel.$viewValue != null ? this._ngModel.$viewValue : -
Number.MAX_VALUE;
    };

    RatingController.prototype._calculateOptions = function() {
        if( this.min == null || this.max == null ) {
            this.options = [];
        }
        else {
            this.options = new Array(this.max - this.min + 1);
            for( var i=0; i < this.options.length; i++ ) {
                this.options[i] = this.min + i;
            }
        }
    };

    RatingController.prototype.setValue = function(val) {
        this.rating = val;
        // KEY POINT 2
        this._ngModel.$setViewValue(val);
    };

    // KEY POINT 3
```

```

Object.defineProperty(RatingController.prototype, 'min', {
  get: function() {
    return this._min;
  },
  set: function(val) {
    this._min = val;
    this._calculateOptions();
  }
});

Object.defineProperty(RatingController.prototype, 'max', {
  get: function() {
    return this._max;
  },
  set: function(val) {
    this._max = val;
    this._calculateOptions();
  }
});

return {
  restrict: 'E',
  scope: {
    // KEY POINT 3
    min: '<?',
    max: '<?',
    nullifier: '<?'
  },
  bindToController: true,
  controllerAs: 'ctrl',
  controller: RatingController,
  require: ['rating', 'ngModel'],
  link: function(scope, elem, attrs, ctrls) {
    ctrls[0].setNgModel(ctrls[1]);
  },
  template:
    '<span ng-repeat="o in ctrl.options" href="#" class="rating-option" ng-
class="{\'rating-option-active\': o <= ctrl.rating}" ng-click="ctrl.setValue(o)">{{ o
}}</span>' +
    '<span ng-if="ctrl.nullifier" ng-click="ctrl.setValue(null)" class="rating-
nullifier">&#10006;</span>'
  };
});

```

1. ngModel.\$render
2. ngModel.\$setViewValue()
3. ;'Angular' = 1.5 - 。 JavaScriptObject.defineProperty() Object.defineProperty() 。

1 ctrl.options 。 ;DOMmin / max/。

2 'Angular <1.5。 Angular' = 1.5\$onInit()minmax 。

<https://jsfiddle.net/h81mgxma/>

;。 。 。

```
<input-person ng-model="data.thePerson"></input-person>
```

```
<input-address ng-model="data.thePerson.address"></input-address>
```

```
function Person(data) {
  data = data || {};
  this.name = data.name;
  this.address = data.address ? new Address(data.address) : null;
}

function Address(data) {
  data = data || {};
  this.street = data.street;
  this.number = data.number;
}
```

```
app.directive('inputAddress', function() {

  InputAddressController.$inject = ['$scope'];
  function InputAddressController($scope) {
    this.$scope = $scope;
    this._ngModel = null;
    this.value = null;
    this._unwatch = angular.noop;
  }

  InputAddressController.prototype.setNgModel = function(ngModel) {
    this._ngModel = ngModel;

    if( ngModel ) {
      // KEY POINT 3
      ngModel.$render = this._render.bind(this);
    }
  };

  InputAddressController.prototype._makeWatch = function() {
    // KEY POINT 1
    this._unwatch = this.$scope.$watchCollection(
      (function() {
        return this.value;
      }).bind(this),
      (function(newval, oldval) {
        if( newval !== oldval ) { // skip the initial trigger
          this._ngModel.$setViewValue(newval !== null ? new Address(newval) : null);
        }
      }).bind(this)
    );
  };

  InputAddressController.prototype._render = function() {
    // KEY POINT 2
    this._unwatch();
    this.value = this._ngModel.$viewValue ? new Address(this._ngModel.$viewValue) : null;
    this._makeWatch();
  };

  return {
    restrict: 'E',
    scope: {},
    bindToController: true,
    controllerAs: 'ctrl',
  };
});
```

```

    controller: InputAddressController,
    require: ['inputAddress', 'ngModel'],
    link: function(scope, elem, attrs, ctrls) {
        ctrls[0].setNgModel(ctrls[1]);
    },
    template:
        '<div>' +
            '<label><span>Street:</span><input type="text" ng-model="ctrl.value.street"
/></label>' +
            '<label><span>Number:</span><input type="text" ng-model="ctrl.value.number"
/></label>' +
            '</div>'
    };
});

```

1. ;◦ \$setViewValue()◦ ◦
2. ◦ ◦ this\_unwatch();this.\_makeWatch();◦ UI◦
3. ngModel.\$render() ngModel.\$setViewValue()◦

◦ <input-address>◦ ◦

```

app.directive('inputPerson', function() {

    InputPersonController.$inject = ['$scope'];
    function InputPersonController($scope) {
        this.$scope = $scope;
        this._ngModel = null;
        this.value = null;
        this._unwatch = angular.noop;
    }

    InputPersonController.prototype.setNgModel = function(ngModel) {
        this._ngModel = ngModel;

        if( ngModel ) {
            ngModel.$render = this._render.bind(this);
        }
    };

    InputPersonController.prototype._makeWatch = function() {
        this._unwatch = this.$scope.$watchCollection(
            (function() {
                return this.value;
            }).bind(this),
            (function(newval, oldval) {
                if( newval !== oldval ) { // skip the initial trigger
                    this._ngModel.$setViewValue(newval !== null ? new Person(newval) : null);
                }
            }).bind(this)
        );
    };

    InputPersonController.prototype._render = function() {
        this._unwatch();
        this.value = this._ngModel.$viewValue ? new Person(this._ngModel.$viewValue) : null;
        this._makeWatch();
    };

    return {

```

```
restrict: 'E',
scope: {},
bindToController: true,
controllerAs: 'ctrl',
controller: InputPersonController,
require: ['inputPerson', 'ngModel'],
link: function(scope, elem, attrs, ctrls) {
    ctrls[0].setNgModel(ctrls[1]);
},
template:
    '<div>' +
        '<label><span>Name:</span><input type="text" ng-model="ctrl.value.name"
/></label>' +
        '<input-address ng-model="ctrl.value.address"></input-address>' +
        '</div>'
    };
});
```

◦ ;JSON◦ angular.copy()◦ ◦

<https://jsfiddle.net/3tzyqfko/2/>

<https://jsfiddle.net/agj4cp0e/https://jsfiddle.net/ugb6Lw8b/>

[ngModelController https://riptutorial.com/zh-CN/angularjs/topic/2438/ngmodelcontroller](https://riptutorial.com/zh-CN/angularjs/topic/2438/ngmodelcontroller)

# 18: ngRoute

ngRoute°

ngRoute <https://docs.angularjs.org/api/ngRoute>

## Examples

controllerAs3°

angular .config

1. \$routeProvider.config
2. .when°
3. .whentemplate templateUrl controller controllerAs

app.js

```
angular.module('myApp', ['ngRoute'])
  .controller('controllerOne', function() {
    this.message = 'Hello world from Controller One!';
  })
  .controller('controllerTwo', function() {
    this.message = 'Hello world from Controller Two!';
  })
  .controller('controllerThree', function() {
    this.message = 'Hello world from Controller Three!';
  })
  .config(function($routeProvider) {
    $routeProvider
      .when('/one', {
        templateUrl: 'view-one.html',
        controller: 'controllerOne',
        controllerAs: 'ctrlOne'
      })
      .when('/two', {
        templateUrl: 'view-two.html',
        controller: 'controllerTwo',
        controllerAs: 'ctrlTwo'
      })
      .when('/three', {
        templateUrl: 'view-three.html',
        controller: 'controllerThree',
        controllerAs: 'ctrlThree'
      })
      // redirect to here if no other routes match
      .otherwise({
        redirectTo: '/one'
      });
  });
```

HTML `<a href="#/helloRoute">My route</a>`

## index.html

```

<div ng-app="myApp">
  <nav>
    <!-- links to switch routes -->
    <a href="#/one">View One</a>
    <a href="#/two">View Two</a>
    <a href="#/three">View Three</a>
  </nav>
  <!-- views will be injected here -->
  <div ng-view></div>
  <!-- templates can live in normal html files -->
  <script type="text/ng-template" id="view-one.html">
    <h1>{{ctrlOne.message}}</h1>
  </script>

  <script type="text/ng-template" id="view-two.html">
    <h1>{{ctrlTwo.message}}</h1>
  </script>

  <script type="text/ng-template" id="view-three.html">
    <h1>{{ctrlThree.message}}</h1>
  </script>
</div>

```

- 1.
2. Controller\$routeParams

## app.js

```

angular.module('myApp', ['ngRoute'])
.controller('controllerOne', function() {
  this.message = 'Hello world from Controller One!';
})
.controller('controllerTwo', function() {
  this.message = 'Hello world from Controller Two!';
})
.controller('controllerThree', ['$routeParams', function($routeParams) {
  var routeParam = $routeParams.paramName

  if ($routeParams.message) {
    // If a param called 'message' exists, we show it's value as the message
    this.message = $routeParams.message;
  } else {
    // If it doesn't exist, we show a default message
    this.message = 'Hello world from Controller Three!';
  }
}])
.config(function($routeProvider) {
  $routeProvider
  .when('/one', {
    templateUrl: 'view-one.html',
    controller: 'controllerOne',
    controllerAs: 'ctrlOne'
  })
  .when('/two', {

```



```

    templateUrl: 'view-two.html',
    controller: 'controllerTwo',
    controllerAs: 'ctrlTwo'
  })
  .when('/three', {
    templateUrl: 'view-three.html',
    controller: 'controllerThree',
    controllerAs: 'ctrlThree'
  })
  .when('/three/:message', { // We will pass a param called 'message' with this route
    templateUrl: 'view-three.html',
    controller: 'controllerThree',
    controllerAs: 'ctrlThree'
  })
  // redirect to here if no other routes match
  .otherwise({
    redirectTo: '/one'
  });
});
});

```

◦

## index.html

```

<div ng-app="myApp">
  <nav>
    <!-- links to switch routes -->
    <a href="#/one">View One</a>
    <a href="#/two">View Two</a>
    <a href="#/three">View Three</a>
    <!-- New link with custom message -->
    <a href="#/three/This-is-a-message">View Three with "This-is-a-message" custom message</a>
  </nav>
  <!-- views will be injected here -->
  <div ng-view></div>
  <!-- templates can live in normal html files -->
  <script type="text/ng-template" id="view-one.html">
    <h1>{{ctrlOne.message}}</h1>
  </script>

  <script type="text/ng-template" id="view-two.html">
    <h1>{{ctrlTwo.message}}</h1>
  </script>

  <script type="text/ng-template" id="view-three.html">
    <h1>{{ctrlThree.message}}</h1>
  </script>
</div>

```

◦

## 1 routes.js requireAuth

```

angular.module('yourApp').config(['$routeProvider', function($routeProvider) {
  $routeProvider
    .when('/home', {
      templateUrl: 'templates/home.html',
      requireAuth: true
    }

```

```
    })
    .when('/login', {
        templateUrl: 'templates/login.html',
    })
    .otherwise({
        redirectTo: '/home'
    });
  });
}])
```

## 2ng-viewangular \$routeProvidernewUrlrequireAuth

```
angular.module('YourApp').controller('YourController', ['$scope', 'session', '$location',
function($scope, session, $location) {

    $scope.$on('$routeChangeStart', function(angularEvent, newUrl) {

        if (newUrl.requireAuth && !session.user) {
            // User isn't authenticated
            $location.path("/login");
        }

    });
}
]);
```

ngRoute <https://riptutorial.com/zh-CN/angularjs/topic/2391/ngroute>

# 19:

## Examples

### /HTML

#### html。

```
<!DOCTYPE html>
<html ng-app="myDemoApp">
  <head>
    <script src="https://code.angularjs.org/1.5.8/angular.min.js"></script>
    <script>

      function HideShowController() {
        var vm = this;
        vm.show=false;
        vm.toggle= function() {
          vm.show=!vm.show;
        }
      }

      angular.module("myDemoApp", [/* module dependencies go here */])
        .controller("hideShowController", [HideShowController]);
    </script>
  </head>
  <body ng-cloak>
    <div ng-controller="hideShowController as vm">
      <a style="cursor: pointer;" ng-show="vm.show" ng-click="vm.toggle()">Show Me!</a>
      <a style="cursor: pointer;" ng-hide="vm.show" ng-click="vm.toggle()">Hide Me!</a>
    </div>
  </body>
</html>
```

1. ng-app="myDemoApp" ngApp angularDOM“myDemoApp”[angular.module](#)。
2. <script src="//angular include">js。
3. HideShowControllertoggle。
4. angular.module(...)。
5. .controller(...) [Angular Controller](#);
6. ng-controller - - 。
7. true<sub>ng-show</sub> HTML。
8. true<sub>ng-hide</sub> HTML。
9. ng-click

<https://riptutorial.com/zh-CN/angularjs/topic/7644/>

## 20:

- ;
  - ;
  - \$ getFn;
  - ;
  - ;
- "Provider"
- - 
  - 
  - 
  -
- 
- ◦

## Examples

Constant

```
angular.module('app', [])
  .constant('endpoint', 'http://some.rest.endpoint') // define
  .config(function(endpoint) {
    // do something with endpoint
    // available in both config- and run phases
  })
  .controller('MainCtrl', function(endpoint) { // inject
    var vm = this;
    vm.endpoint = endpoint; // usage
  });
```

```
<body ng-controller="MainCtrl as vm">
  <div>endpoint = {{ ::vm.endpoint }}</div>
</body>
```

endpoint = <http://some.rest.endpoint>

Value

```
angular.module('app', [])
  .value('endpoint', 'http://some.rest.endpoint') // define
  .run(function(endpoint) {
    // do something with endpoint
    // only available in run phase
  })
```

```
.controller('MainCtrl', function(endpoint) { // inject
  var vm = this;
  vm.endpoint = endpoint; // usage
});
```

```
<body ng-controller="MainCtrl as vm">
  <div>endpoint = {{ ::vm.endpoint }}</div>
</body>
```

endpoint = <http://some.rest.endpoint>

Factory°

Factory° °

Factory°

```
angular.module('app', [])
  .factory('endpointFactory', function() {
    return {
      get: function() {
        return 'http://some.rest.endpoint';
      }
    };
  })
  .controller('MainCtrl', function(endpointFactory) {
    var vm = this;
    vm.endpoint = endpointFactory.get();
  });
```

```
<body ng-controller="MainCtrl as vm">
  <div>endpoint = {{::vm.endpoint }}</div>
</body>
```

endpoint = <http://some.rest.endpoint>

Service°

ServiceValueFactory*new*° °

```
angular.module('app', [])
  .service('endpointService', function() {
    this.get = function() {
      return 'http://some.rest.endpoint';
    };
  })
  .controller('MainCtrl', function(endpointService) {
    var vm = this;
    vm.endpoint = endpointService.get();
  });
```

```
<body ng-controller="MainCtrl as vm">
  <div>endpoint = {{::vm.endpoint }}</div>
</body>
```

---

endpoint = <http://some.rest.endpoint>

Provider◦

Provider\$get◦

APIProviderAPI◦ ◦

```
angular.module('app', [])
  .provider('endpointProvider', function() {
    var uri = 'n/a';

    this.set = function(value) {
      uri = value;
    };

    this.$get = function() {
      return {
        get: function() {
          return uri;
        }
      };
    };
  })
  .config(function(endpointProviderProvider) {
    endpointProviderProvider.set('http://some.rest.endpoint');
  })
  .controller('MainCtrl', function(endpointProvider) {
    var vm = this;
    vm.endpoint = endpointProvider.get();
  });
```

---

```
<body ng-controller="MainCtrl as vm">
  <div>endpoint = {{::vm.endpoint }}</div>
</body>
```

---

endpoint = <http://some.rest.endpoint>

config

endpoint = n / a

<https://riptutorial.com/zh-CN/angularjs/topic/5169/>

# 21:

- myApp.controller('MyController',function(\$ scope){...}; //
- myApp.controller('MyController',['\$ scope',function(\$ scope){...}]; //
- function MyController()  
MyController.\$ inject = ['\$ scope'];  
myApp.controller('MyController',MyController); // \$  
• \$ injector.get "; ///

run◦  
config◦  
◦

## Examples

Angular - Angular Controller\$scope

```
angular.module('myModule', [])  
.controller('myController', ['$scope', function($scope) {  
    $scope.members = ['Alice', 'Bob'];  
    ...  
}])
```

\$scopecontroller ◦ ◦

Angular ◦ ◦

DI - - ◦

“” ◦ ◦ ◦ ◦ Angular ◦

- ◦

◦

◦ \$injector

```
myModule.controller('myController', ['$injector', function($injector) {  
    var myService = $injector.get('myService');  
}]);
```

◦ ◦

## \$ inject

\$inject

```
var MyController = function($scope) {  
    // ...  
}  
MyController.$inject = ['$scope'];  
myModule.controller('MyController', MyController);
```

## vanilla JavaScriptAngularJS

AngularJS injector() vanilla JavaScriptAngularJS ◦ angular.element() jqLite injector() ◦

```
var service;  
var serviceName = 'myService';  
  
var ngAppElement = angular.element(document.querySelector('[ng-app],[data-ng-app]') ||  
document);  
var injector = ngAppElement.injector();  
  
if(injector && injector.has(serviceNameToInject)) {  
    service = injector.get(serviceNameToInject);  
}
```

AngularJS ngAppElement jqLite ◦ angular.element() ng-app data-ng-app DOM document ◦ ngAppElement  
ngAppElement.injector() ◦ injector.has() service injector.get() ◦

<https://riptutorial.com/zh-CN/angularjs/topic/1582/>



# 22:

Angular. . . .

## Examples

### ngStorage

index.html [ngStorage](#).

ngStorage SRC

```
<head>
  <title>Angular JS ngStorage</title>
  <script src =
"http://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/angular.min.js"></script>
  <script src="https://rawgithub.com/gsklee/ngStorage/master/ngStorage.js"></script>
</head>
```

ngStorage2 \$localStorage\$sessionStorage. ngStorageInject.

ng-app="myApp" ngStorage

```
var app = angular.module('myApp', ['ngStorage']);
app.controller('controllerOne', function($localStorage,$sessionStorage) {
  // an object to share
  var sampleObject = {
    name: 'angularjs',
    value: 1
  };
  $localStorage.valueToShare = sampleObject;
  $sessionStorage.valueToShare = sampleObject;
})
.controller('controllerTwo', function($localStorage,$sessionStorage) {
  console.log('localStorage: '+ $localStorage +'sessionStorage: '+$sessionStorage);
})
```

\$localStorage\$sessionStorage.

HTML5localStoragesessionStorage. HTML5 localStorage.

```
var myObj = {
  firstname: "Nic",
  lastname: "Raboy",
  website: "https://www.google.com"
}
//if you wanted to save into localStorage, serialize it
window.localStorage.set("saved", JSON.stringify(myObj));

//unserialize to get object
var myObj = JSON.parse(window.localStorage.get("saved"));
```

servicesetget controllers°

```
app.service('setGetData', function() {
  var data = '';
  getData: function() { return data; },
  setData: function(requestData) { data = requestData; }
});
```

```
app.controller('myCtrl1', ['setGetData',function(setGetData) {

  // To set the data from the one controller
  var data = 'Hello World !!';
  setGetData.setData(data);

}]);

app.controller('myCtrl2', ['setGetData',function(setGetData) {

  // To get the data from the another controller
  var res = setGetData.getData();
  console.log(res); // Hello World !!

}]);
```

myCtrl1setting myCtrl2getting° °

<https://riptutorial.com/zh-CN/angularjs/topic/1923/>

## 23:

### Examples

-

type="text" type="number"Angular◦

```
var app = angular.module('app', []);

app.controller('ctrl', function($scope) {
  $scope.textInput = {
    value: '5'
  };
  $scope.numberInput = {
    value: 5
  };
});
```

```
<div ng-app="app" ng-controller="ctrl">
  <input type="text" ng-model="textInput.value">
  {{ textInput.value + 5 }}
  <input type="number" ng-model="numberInput.value">
  {{ numberInput.value + 5 }}
</div>
```

- +55 \* ◦
- + 10 \* ◦

\* - ◦

### ngRepeat

ng-repeatAngular◦

ng-

```
<ul>
  <li ng-repeat="item in itemCollection">
    {{item.Name}}
  </li>
</ul>
```

**item =**

**itemCollection =**

## ng-

```
<ul>
  <li ng-repeat="(key, value) in myObject">
    {{key}} : {{value}}
  </li>
</ul>
```

**key =**

**value =**

**myObject =**

## ng-repeat

```
<input type="text" ng-model="searchText">
<ul>
  <li ng-repeat="string in stringArray | filter:searchText">
    {{string}}
  </li>
</ul>
```

**searchText =**

**stringArray =**['string', 'array']

as aliasName

```
<input type="text" ng-model="searchText">
<ul>
  <li ng-repeat="string in stringArray | filter:searchText as filteredStrings">
    {{string}}
  </li>
</ul>
<p>There are {{filteredStrings.length}} matching results</p>
```

## ng-repeat-startng-repeat-end

DOMng-repeat-startng-repeat-end°

```
<ul>
  <li ng-repeat-start="item in [{a: 1, b: 2}, {a: 3, b:4}]">
    {{item.a}}
  </li>
  <li ng-repeat-end>
    {{item.b}}
  </li>
</ul>
```

- 1
- 2
- 3
- 4

ng-repeat-end ng-repeat-start

ng-repeat

\$index	\$ index === 0 true; \$first
\$first	true
\$last	true
\$middle	\$first \$last true
\$even	true \$index % 2 === 0
\$odd	true \$index % 2 === 1

ngRepeat

track by track by \$index

```
<div ng-repeat="item in itemCollection track by item.id">
<div ng-repeat="item in itemCollection track by $index">
```

## ngRepeat

ngRepeat repeat

ngRepeat

```
scope val: {{val}}<br/>
ctrlAs val: {{ctrl.val}}
<ul>
  <li ng-repeat="item in itemCollection">
    <a href="#" ng-click="$parent.val=item.value; ctrl.val=item.value;">
      {{item.label}} {{item.value}}
    </a>
  </li>
</ul>

$scope.val = 0;
this.val = 0;

$scope.itemCollection = [
  id: 0,
  value: 4.99,
  label: 'Football'
],
```

```

{
  id: 1,
  value: 6.99,
  label: 'Baseball'
},
{
  id: 2,
  value: 9.99,
  label: 'Basketball'
}
];

```

ng-clickval = item.value val ◦ \$parentcontrollerAsng-controller="mainController as ctrl" ◦

## ng-repeat

ng-repeat◦

```

<div ng-repeat="values in test">
  <div ng-repeat="i in values">
    [{{parent.$index}},{{i}}] {{i}}
  </div>
</div>

var app = angular.module("myApp", []);
app.controller("ctrl", function($scope) {
  $scope.test = [
    ['a', 'b', 'c'],
    ['d', 'e', 'f']
  ];
});

```

ng-repeatng-repeat\$parent.\$index ◦

## ngShowngHide

ng-showtruefalseHTML◦ ◦ ◦

ng-hide◦ falsyHTML◦ ◦

## JSBin

```

var app = angular.module('app', []);

angular.module('app')
  .controller('ExampleController', ExampleController);

function ExampleController() {

  var vm = this;

  //Binding the username to HTML element
  vm.username = '';

  //A taken username
  vm.taken_username = 'StackOverflow!';
}

```

```
}
```

```
<section ng-controller="ExampleController as main">

  <p>Enter Password</p>
  <input ng-model="main.username" type="text">

  <hr>

  <!-- Will always show as long as StackOverflow is not typed in -->
  <!-- The expression is always true when it is not StackOverflow -->
  <div style="color:green;" ng-show="main.username != main.taken_username">
    Your username is free to use!
  </div>

  <!-- Will only show when StackOverflow is typed in -->
  <!-- The expression value becomes falsy -->
  <div style="color:red;" ng-hide="main.username != main.taken_username">
    Your username is taken!
  </div>

  <p>Enter 'StackOverflow' in username field to show ngHide directive.</p>

</section>
```

## ngOptions

ngOptions **html** ◦ ngOptions ngOptions <select><option>◦

ng-options **select**

```
<select ng-model="selectedFruitNgOptions"
  ng-options="curFruit as curFruit.label for curFruit in fruit">
</select>
```

ng-repeat select ng-repeat for Each ◦ ng-options select ◦

ng-repeat

```
<select ng-model="selectedFruit">
  <option ng-repeat="curFruit in fruit" value="{{curFruit}}">
    {{curFruit.label}}
  </option>
</select>
```

◦

```
$scope.fruit = [
  { label: "Apples", value: 4, id: 2 },
  { label: "Oranges", value: 2, id: 1 },
  { label: "Limes", value: 4, id: 4 },
  { label: "Lemons", value: 5, id: 3 }
];
```

```
<!-- label for value in array -->
<select ng-options="f.label for f in fruit" ng-model="selectedFruit"></select>
```

```
<option value="{ label: 'Apples', value: 4, id: 2 }"> Apples </option>
```

f.label<option>◦

---

```
<!-- select as label for value in array -->
<select ng-options="f.value as f.label for f in fruit" ng-model="selectedFruit"></select>
```

```
<option value="4"> Apples </option>
```

f.value 4◦

---

```
<!-- label group by group for value in array -->
<select ng-options="f.label group by f.value for f in fruit" ng-
model="selectedFruit"></select>
```

```
<option value="{ label: 'Apples', value: 4, id: 2 }"> Apples </option>
```

value◦ value

---

```
<!-- label disable when disable for value in array -->
<select ng-options="f.label disable when f.value == 4 for f in fruit" ng-
model="selectedFruit"></select>
```

```
<option disabled="" value="{ label: 'Apples', value: 4, id: 2 }"> Apples </option>
```

disable when f.value==4“Apples”“Limes”◦ value=4

---

```
<!-- label group by group for value in array track by trackexpr -->
<select ng-options="f.value as f.label group by f.value for f in fruit track by f.id" ng-
model="selectedFruit"></select>
```

```
<option value="4"> Apples </option>
```

trackByAngularid◦

---

```
<!-- label for value in array | orderBy:orderexpr track by trackexpr -->
<select ng-options="f.label for f in fruit | orderBy:'id' track by f.id" ng-
model="selectedFruit"></select>
```

```
<option disabled="" value="{ label: 'Apples', value: 4, id: 2 }"> Apples </option>
```



orderByAngularJS“Oranges”1stid = 1。

---

ng-options <select>ng-model 。

## ngModel

ng-model {{myAge}} 。

```
<input type="text" ng-model="myName">
<p>{{myName}}</p>
```

。

ng-model\$scope.myName controllerAs

```
<div ng-controller="myCtrl as mc">
  <input type="text" ng-model="mc.myName">
  <p>{{mc.myName}}</p>
</div>
```

ng-controllerng-model \$scopeng-modelthis.myName。

## ngClass

CSS。 Angular。 Angular。

/。 true。

```
<style>
  .active { background-color: green; color: white; }
  .inactive { background-color: gray; color: white; }
  .adminUser { font-weight: bold; color: yellow; }
  .regularUser { color: white; }
</style>

<span ng-class="{
  active: user.active,
  inactive: !user.active,
  adminUser: user.level === 1,
  regularUser: user.level === 2
}">John Smith</span>
```

Angular\$scope.useractivelevel。 Angular<span> 。

## ngIf

ng-ifng-showDOM。 Angular 1.1.5ng-If。 1.1.5ng-if。 Angularng-ifng-ifAngular。

ng-show ng-if 。

---

# JavaScript

```
angular.module('MyApp', []);

angular.module('MyApp').controller('myController', ['$scope', '$window', function
myController($scope, $window) {
    $scope.currentUser= $window.localStorage.getItem('userName');
}]);
```

```
<div ng-controller="myController">
  <div ng-if="currentUser">
    Hello, {{currentUser}}
  </div>
  <div ng-if="!currentUser">
    <a href="/login">Log In</a>
    <a href="/register">Register</a>
  </div>
</div>
```

## DOM<sub>currentUser</sub>

```
<div ng-controller="myController">
  <div ng-if="currentUser">
    Hello, {{currentUser}}
  </div>
  <!-- ng-if: !currentUser -->
</div>
```

## DOM<sub>currentUser</sub>

```
<div ng-controller="myController">
  <!-- ng-if: currentUser -->
  <div ng-if="!currentUser">
    <a href="/login">Log In</a>
    <a href="/register">Register</a>
  </div>
</div>
```

## ngIftruefalse。

```
<div ng-if="myFunction()">
  <span>Span text</span>
</div>
```

## true。

```
$scope.myFunction = function() {
  var result = false;
  // Code to determine the boolean value of result
  return result;
};
```

Angular。

## ngMouseenterngMouseleave

DOM ng-mouseenterng-mouseleaveCSS。

ng-mouseenterDOM

### HTML

```
<div ng-mouseenter="applyStyle = true" ng-class="{ 'active': applyStyle }">
```

div applyStyletrue ng-class.active CSS ng-class 。

ng-mouseleaveDOM

### HTML

```
<div ng-mouseenter="applyStyle = true" ng-mouseleave="applyStyle = false" ng-
class="{ 'active': applyStyle }">
```

div .active。

## ngDisabled

。

ng-disabledtruthyfalsy。

ng-disabledinputdisabled。

### HTML

```
<input type="text" ng-model="vm.name">
<button ng-disabled="vm.name.length===0" ng-click="vm.submitMe">Submit</button>
```

input0vm.name.length===0trueng-clickclick

## ngDbclick

DOM ng-dblclick。

## HTML

```
<input type="number" ng-model="num = num + 1" ng-init="num=0">
<button ng-dblclick="num++">Double click me</button>
```

input◦

ng-app **AngularJS**◦

ng-init◦

ng-bind({})◦

ng-bind-template◦

ng-non-bindable ◦

ng-bind-html **HTMLHTML**◦

ng-change◦

ng-checked◦

ng-class **classcss**◦

ng-cloak **AngularJS**◦

ng-click◦

ng-controller◦

ng-disabled **disabled**

ng-form

ng-href **AngularJS**href◦

ng-include **HTML**◦

ng-if **DOM**

ng-switch◦

ng-model **modelinputselecttextarea**◦

ng-readonly **readonly**◦

ng-repeat◦

ng-selected◦

ng-show/ng-hide/◦

ng-srcAngularJSsrc◦

ng-submitonsubmit◦

ng-value◦

ng-requiredonsubmit◦

ng-styleHTMLCSS◦

ng-patternngModel◦

ng-maxlengthmaxlengthngModel◦

ng-minlength minlengthminlengthngModel◦

ng-classevenngRepeat◦

ng-classoddngRepeat◦

ng-cut◦

ng-copy◦

ng-paste◦

ng-options◦

ng-list◦

ng-openopenngOpen◦

## ngClick

ng-clickclickDOM◦

ng-clickDOM◦

◦

events\$event

## HTML

```
<input ng-click="onClick($event)">Click me</input>
```

```
.controller("ctrl", function($scope) {  
  $scope.onClick = function(evt) {
```

```
        console.debug("Hello click event: %o ",evt);
    }
})
```

## HTML

```
<button ng-click="count = count + 1" ng-init="count=0">
  Increment
</button>
<span>
  count: {{count}}
</span>
```

## HTML

```
<button ng-click="count()" ng-init="count=0">
  Increment
</button>
<span>
  count: {{count}}
</span>
```

```
...
$scope.count = function(){
  $scope.count = $scope.count + 1;
}
...
```

onClick “Hello click event”。

## ngRequired

ng-requiredrequiredinputrequire。

input。HTML。

## HTML

```
<input type="checkbox" ng-model="someBooleanValue">
<input type="text" ng-model="username" ng-required="someBooleanValue">
```

## NG-

ng-model-optionsng-modelng-model。

。

```
<input type="text" ng-model="myValue" ng-model-options="{ 'debounce': 500 }">
```

myValue500input myValue500。

### 1. updateOn

```
ng-model-options="{ updateOn: 'blur'}" // will update on blur
```

### 2. debounce

```
ng-model-options="{debounce': 500}" // will update the model after 1/2 second
```

### 3. allowInvalid undefined ◦

### 4. getterSetter ng-modelgetter / setter◦ ◦

```
<input type="text" ng-model="myFunc" ng-model-options="{getterSetter': true}">
$scope.myFunc = function() {return "value";}
```

### 5. timezone datetime ◦

## ngCloak

ngCloakAngular html◦ -

### HTML

```
<div ng-cloak>
  <h1>Hello {{ name }}</h1>
</div>
```

ngCloakbodyngCloak◦

ngCloak◦

## ngInclude

ng-include◦ ◦

```
<div ng-include
  src="'/gridview'"
  ng-controller='gridController as gc'>
</div>
```

/gridviewWebURL◦

src -attributeAngular◦ ◦ URL◦ ◦

/gridview htmlgridController

```
<div class="row">
  <button type="button" class="btn btn-default" ng-click="gc.doSomething()"></button>
```

```
</div>
```

## ngSrc

`src` `{{hash}}` `Angular` `o` `{{hash}}` `URL` `Angular` `{{hash}}` `o` `ng-src` `image` `src`

```
<div ng-init="pic = 'pic_angular.jpg'">
  <h1>Angular</h1>
  
</div>
```

## ngPattern

`ng-pattern` `o`

`<input>` `ng-model` `IP` `o`

```
<input type="text" ng-model="ipAddr" ng-pattern="ipRegex" name="ip" required>
```

```
$scope.ipRegex = /\b(?:?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.{3}(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\b/;
```

## ngValue

`ngRepeat` `ng-repeat` `ngValue`

```
<script>
  angular.module('valueExample', [])
    .controller('ExampleController', ['$scope', function($scope) {
      $scope.names = ['pizza', 'unicorns', 'robots'];
      $scope.my = { favorite: 'unicorns' };
    }]);
</script>
<form ng-controller="ExampleController">
  <h2>Which is your favorite?</h2>
  <label ng-repeat="name in names" for="{{name}}">
    {{name}}
    <input type="radio"
      ng-model="my.favorite"
      ng-value="name"
      id="{{name}}"
      name="favorite">
  </label>
  <div>You chose {{my.favorite}}</div>
</form>
```

[plnkr](#)

## ngCopy

`ngCopy` `o`



```
<p ng-copy="blockCopy($event)">This paragraph cannot be copied</p>
```

```
$scope.blockCopy = function(event) {  
  event.preventDefault();  
  console.log("Copying won't work");  
}
```

## ngPaste

ngPaste

```
<input ng-paste="paste=true" ng-init="paste=false" placeholder='paste here'>  
pasted: {{paste}}
```

## ngHref

hrefngHrefhref。 ngHrefhrefhtmlhref。

AngularJSngHref。

### 1

```
<div ng-init="linkValue = 'http://stackoverflow.com'">  
  <p>Go to <a ng-href="{{linkValue}}">{{linkValue}}</a>!</p>  
</div>
```

2hrefhref。

```
<input ng-model="value" />  
<a id="link" ng-href="{{value}}">link</a>
```

### 3

```
<script>  
angular.module('angularDoc', [])  
.controller('myController', function($scope) {  
  // Set some scope value.  
  // Here we set bootstrap version.  
  $scope.bootstrap_version = '3.3.7';  
  
  // Set the default layout value  
  $scope.layout = 'normal';  
});  
</script>  
<!-- Insert it into Angular Code -->  
<link rel="stylesheet" ng-href="//maxcdn.bootstrapcdn.com/bootstrap/{{ bootstrap_version  
}}/css/bootstrap.min.css">  
<link rel="stylesheet" ng-href="layout-{{ layout }}.css">
```

## ngList

```
ng-list°
```

```
ng-list", " °
```

```
ng-listng-list="; " °
```

```
°
```

```
ng-listng-trim true° ng-trim false° ng-trim="false"ng-list°
```

```
angular.module('test', [])  
  .controller('ngListExample', ['$scope', function($scope) {  
    $scope.list = ['angular', 'is', 'cool!'];  
  }]);
```

```
; ° °
```

```
<body ng-app="test" ng-controller="ngListExample">  
  <input ng-model="list" ng-list="; " ng-trim="false">  
</body>
```

```
angular; is; cool!
```

<https://riptutorial.com/zh-CN/angularjs/topic/706/>

# 24:

## Examples

### angular.equals

angular.equals2 angular.equalstrue◦

angular.equalsvalue1value2

1. ===
2. angular.equals
3. NaN
4. ◦

◦

```
angular.equals(1, 1) // true
angular.equals(1, 2) // false
angular.equals({}, {}) // true, note that {}==={} is false
angular.equals({a: 1}, {a: 1}) // true
angular.equals({a: 1}, {a: 2}) // false
angular.equals(NaN, NaN) // true
```

### angular.isString

string angular.isStringtrue

angular.isString

```
angular.isString("hello") // true
angular.isString([1, 2]) // false
angular.isString(42) // false
```

```
typeof someValue === "string"
```

### angular.isArray

Array angular.isArraytrue◦

angular.isArray

```
angular.isArray([]) // true
angular.isArray([2, 3]) // true
angular.isArray({}) // false
angular.isArray(17) // false
```

```
Array.isArray(someValue)
```

## angular.merge

angular.merge。

### angular.merge

```
angular.merge({}, {}) // {}  
angular.merge({name: "king roland"}, {password: "12345"})  
// {name: "king roland", password: "12345"}  
angular.merge({a: 1}, [4, 5, 6]) // {0: 4, 1: 5, 2: 6, a: 1}  
angular.merge({a: 1}, {b: {c: {d: 2}}}) // {"a":1,"b":{"c":{"d":2}}}
```

## angular.isDefinedangular.isUndefined

angular.isDefinedangular.isDefined

### angular.isDefinedsomeValue

```
value !== undefined; // will evaluate to true is value is defined
```

```
angular.isDefined(42) // true  
angular.isDefined([1, 2]) // true  
angular.isDefined(undefined) // false  
angular.isDefined(null) // true
```

angular.isUndefinedangular.isDefined

### angular.isUndefinedsomeValue

```
value === undefined; // will evaluate to true is value is undefined
```

```
!angular.isDefined(value)
```

```
angular.isUndefined(42) // false  
angular.isUndefined(undefined) // true
```

## angular.isDate

Date angular.isDatetrue。

### angular.isDate

```
angular.isDate("lone star") // false  
angular.isDate(new Date()) // true
```

## angular.isNumber

## Number angular.isNumber true+ Infinity-InfinityNaN

### angular.isNumber

```
"23" == 23 // true
```

```
angular.isNumber("23") // false  
angular.isNumber(23) // true  
angular.isNumber(NaN) // true  
angular.isNumber(Infinity) // true
```

```
"23" == 23 // true
```

## angular.isFunction

angular.isFunction true

### angular.isFunctionFN

```
var onClick = function(e) {return e};  
angular.isFunction(onClick); // true  
  
var someArray = ["pizza", "the", "hut"];  
angular.isFunction(someArray ); // false
```

## angular.toJson

angular.toJson JSON

JSON.stringify \$\$ \$\$

```
angular.toJson(object)
```

JSON

.toString

```
angular.toJson({name: "barf", occupation: "mog", $$somebizzareproperty: 42})  
// {"name":"barf","occupation":"mog"}  
angular.toJson(42)  
// "42"  
angular.toJson([1, "2", 3, "4"])  
// "[1,\"2\",3,\"4"]"  
var fn = function(value) {return value}  
angular.toJson(fn)  
// undefined, functions have no representation in JSON
```

## angular.fromJson

angular.fromJson JSONObjectArray

## angular.fromJson

◦

```
angular.fromJson("{\"yogurt\": \"strawberries\"}")
// Object {yogurt: "strawberries"}
angular.fromJson('{jam: "raspberries"}')
// will throw an exception as the string is not a valid JSON
angular.fromJson(this)
// Window {external: Object, chrome: Object, _gaq: Y, angular: Object, ng339: 3...}
angular.fromJson([1, 2])
// [1, 2]
typeof angular.fromJson(new Date())
// "object"
```

## angular.noop

angular.noopangular.noop ◦

### angular.noop

angular.noop◦

```
$scope.onSomeChange = function(model, callback) {
  updateTheModel(model);
  if (angular.isFunction(callback)) {
    callback();
  } else {
    throw new Error("error: callback is not a function!");
  }
};

$scope.onSomeChange(42, function() {console.log("hello callback")});
// will update the model and print 'hello callback'
$scope.onSomeChange(42, angular.noop);
// will update the model
```

```
angular.noop() // undefined
angular.isFunction(angular.noop) // true
```

## angular.isObject

angular.isObject**false**null typeof nullobject ◦

### angular.isObject

◦

```
angular.isObject({name: "skroob", job: "president"})
// true
angular.isObject(null)
// false
angular.isObject([null])
```

```
// true
angular.isObject(new Date())
// true
angular.isObject(undefined)
// false
```

## angular.isElement

DOMjQueryElementangular.isElementtrue◦

angular.isElementELEM

◦

```
angular.isElement(document.querySelector("body"))
// true
angular.isElement(document.querySelector("#some_id"))
// false if "some_id" is not using as an id inside the selected DOM
angular.isElement("<div></div>")
// false
```

## angular.copy

angular.copy◦

angular.copy

```
let obj = {name: "vespa", occupation: "princess"};
let cpy = angular.copy(obj);
cpy.name = "yogurt"
// obj = {name: "vespa", occupation: "princess"}
// cpy = {name: "yogurt", occupation: "princess"}
```

```
var w = [a, [b, [c, [d]]]];
var q = angular.copy(w);
// q = [a, [b, [c, [d]]]]
```

angular.equals(w, q)true.equals◦ w === qfalse◦

## angular.identity

angular.identity◦

angular.identity

◦

```
angular.identity(42) // 42
```

```
var mutate = function(fn, num) {
```

```
    return angular.isFunction(fn) ? fn(num) : angular.identity(num)
  }

mutate(function(value) {return value-7}, 42) // 35
mutate(null, 42) // 42
mutate("mount. rushmore", 42) // 42
```

## angular.forEach

angular.forEach /<sup>o</sup> /<sup>o</sup>

Array.prototype.forEachJSnullundefined<sup>o</sup>

**angular.forEach**objectfunctionvaluekey{// function};

```
angular.forEach({"a": 12, "b": 34}, (value, key) => console.log("key: " + key + ", value: " +
value))
// key: a, value: 12
// key: b, value: 34
angular.forEach([2, 4, 6, 8, 10], (value, key) => console.log(key))
// will print the array indices: 1, 2, 3, 4, 5
angular.forEach([2, 4, 6, 8, 10], (value, key) => console.log(value))
// will print the array values: 2, 4, 6, 7, 10
angular.forEach(undefined, (value, key) => console.log("key: " + key + ", value: " + value))
// undefined
```

<https://riptutorial.com/zh-CN/angularjs/topic/3032/>



# 25: -

## Examples

Angular<sub>XHR</sub> ◦ ◦

◦ ◦ GruntGulp ◦

\$templateCache ◦ angular ◦

html2jshtml2js ◦ ◦

```
module.exports = function (grunt) {
  //set up the location of your views here
  var viewLocation = ['app/views/**/*.html'];

  grunt.initConfig({
    pkg: require('./package.json'),
    //section that sets up the settings for concatenation of the html files into one
file
    html2js: {
      options: {
        base: '',
        module: 'app.templates', //new module name
        singleModule: true,
        useStrict: true,
        htmlmin: {
          collapseBooleanAttributes: true,
          collapseWhitespace: true
        }
      },
      main: {
        src: viewLocation,
        dest: 'build/app.templates.js'
      }
    },
    //this section is watching for changes in view files, and if there was a change,
it will regenerate the production file. This task can be handy during development.
    watch: {
      views:{
        files: viewLocation,
        tasks: ['buildHTML']
      },
    }
  });

  //to automatically generate one view file
  grunt.loadNpmTasks('grunt-html2js');

  //to watch for changes and if the file has been changed, regenerate the file
  grunt.loadNpmTasks('grunt-contrib-watch');

  //just a task with friendly name to reference in watch
  grunt.registerTask('buildHTML', ['html2js']);
};
```

## 2index.html

```
<script src="build/app.templates.js"></script>
```

```
angular.module('app', ['app.templates'])
```

ui-router

```
.state('home', {
  url: '/home',
  views: {
    "@": {
      controller: 'homeController',
      //this will be picked up from $templateCache
      templateUrl: 'app/views/home.html'
    },
  },
})
})
```

## JS。 JS。

。 Angularminificaiton。 。

minificaiton \$scope myService。

```
.controller('myController', function($scope, myService){
})
```

## Angular。

```
.controller('myController', ['$scope', 'myService', function($scope, myService){
}])
```

- 。
- ng-annotate
- uglify。

module.exports = functiongrunt{//= ['app / scripts / \* .js'];

```
grunt.initConfig({
  pkg: require('./package.json'),
  //add necessary annotations for safe minification
  ngAnnotate: {
    angular: {
      src: ['staging/concatenated.js'],
      dest: 'staging/anotated.js'
    }
  },
  //combines all the files into one file
  concat: {
    js: {
```

```

        src: scriptLocation,
        dest: 'staging/concatenated.js'
    }
},
//final uglifying
uglify: {
    options: {
        report: 'min',
        mangle: false,
        sourceMap:true
    },
    my_target: {
        files: {
            'build/app.min.js': ['staging/anotated.js']
        }
    }
},
},
    //this section is watching for changes in JS files, and if there was a change, it will
    regenerate the production file. You can choose not to do it, but I like to keep concatenated
    version up to date
    watch: {
        scripts: {
            files: scriptLocation,
            tasks: ['buildJS']
        }
    }
});

//module to make files less readable
grunt.loadNpmTasks('grunt-contrib-uglify');

//mdule to concatenate files together
grunt.loadNpmTasks('grunt-contrib-concat');

//module to make angularJS files ready for minification
grunt.loadNpmTasks('grunt-ng-annotate');

//to watch for changes and if the file has been changed, regenerate the file
grunt.loadNpmTasks('grunt-contrib-watch');

//task that sequentially executes all steps to prepare JS file for production
//concatinate all JS files
//annotate JS file (prepare for minification
//uglify file
grunt.registerTask('buildJS', ['concat:js', 'ngAnnotate', 'uglify']);
};

```

- <https://riptutorial.com/zh-CN/angularjs/topic/4434/--->

# 26:

## Examples

7

### 1 ng-repeat

`ng-repeat` `ng-repeat` ◦

```
<div ng-repeat="user in userCollection">
  <div ng-repeat="details in user">
    {{details}}
  </div>
</div>
```

◦ `ng-repeat` track by `$index id` ◦ `ng-repeat` track by `Angular $index object id` ◦

```
<div ng-repeat="user in userCollection track by $index">
  {{user.data}}
</div>
```

◦

---

## 2

Angular ◦ ◦

```
<!-- Default data binding has a performance cost -->
<div>{{ my.data }}</div>
```

AngularJS > = 1.3

```
<!-- Bind once is much faster -->
<div>{{ ::my.data }}</div>

<div ng-bind="::my.data"></div>

<!-- Use single binding notation in ng-repeat where only list display is needed -->
<div ng-repeat="user in ::userCollection">
  {{::user.data}}
</div>
```

“bind once” Angular ◦ AngularDOM ◦

{{}} ◦

ng-bind° ng-bind°

\$digest°

---

## 3

AngularJS° ° °

```
<div ng-controller="bigCalculations as calc">
  <p>{{calc.calculateMe()}}</p>
  <p>{{calc.data | heavyFilter}}</p>
</div>
```

```
<div ng-controller="bigCalculations as calc">
  <p>{{calc.preCalculatedValue}}</p>
  <p>{{calc.data | lightFilter}}</p>
</div>
```

```
app.controller('bigCalculations', function(valueService) {
  // bad, because this is called in every digest loop
  this.calculateMe = function() {
    var t = 0;
    for(i = 0; i < 1000; i++) {
      t += i;
    }
    return t;
  }
  // good, because this is executed just once and logic is separated in service to keep
  the controller light
  this.preCalculatedValue = valueService.valueCalculation(); // returns 499500
});
```

---

## 4

° UI° °

Angular°

\$watch() - \$watch()

\$watchCollection() - \$watch\$watch

\$watch(..., true) - ""watchCollection

- {{::variable}}°

° [@WordsJared](#)

```
(function() {
```

```

var root = angular.element(document.getElementsByTagName('body')),
    watchers = [],
    f = function(element) {
    angular.forEach(['$scope', '$isolateScope'], function(scopeProperty) {
        if(element.data() && element.data().hasOwnProperty(scopeProperty)) {
            angular.forEach(element.data()[scopeProperty].$$watchers, function(watcher) {
                watchers.push(watcher);
            });
        }
    });

    angular.forEach(element.children(), function(childElement) {
        f(angular.element(childElement));
    });
};

f(root);

// Remove duplicate watchers
var watchersWithoutDuplicates = [];
angular.forEach(watchers, function(item) {
    if(watchersWithoutDuplicates.indexOf(item) < 0) {
        watchersWithoutDuplicates.push(item);
    }
});
console.log(watchersWithoutDuplicates.length);
})();

```

## 5ng-if / ng-show

- [ng-if](#) [DOM](#) [ng-show](#) ◦ ng-if ◦
- 
- ng-if
- /ng-show/ng-hide

```

<div ng-repeat="user in userCollection">
  <p ng-if="user.hasTreeLegs">I am special<!-- some complicated DOM --></p>
  <p ng-show="user.hasSubscribed">I am awesome<!-- switch this setting on and off --></p>
</div>

```

- ng-if **test**

## 6

◦ ◦

```

angular.module('exampleApp', []).config(['$compileProvider', function ($compileProvider) {
    $compileProvider.debugInfoEnabled(false);
}]);

```

# 7

- ◦
- Angular\$ inject ◦ ◦

PRO TIP \$ inject ◦ annotateif\$ inject = fn ◦ \$ inject ◦ \$ inject

```
var app = angular.module('DemoApp', []);

var DemoController = function (s, h) {
  h.get('https://api.github.com/users/angular/repos').success(function (repos) {
    s.repos = repos;
  });
}
// $inject property annotation
DemoController['$inject'] = ['$scope', '$http'];

app.controller('DemoController', DemoController);
```

2ng-appng-strict-diDI ◦

```
<html ng-app="DemoApp" ng-strict-di>
```

```
angular.bootstrap(document, ['DemoApp'], {
  strictDi: true
});
```

Angular ◦ Angular ◦

◦  
{{my.data}}

::◦ my.data ◦ ◦ Angular ◦

```
{{::my.data}}
<span ng-bind="::my.data"></span>
<span ng-if="::my.data"></span>
<span ng-repeat="item in ::my.data">{{item}}</span>
<span ng-class="::{'my-class': my.data }"></div>
```

my.data ◦ ◦

AngularJS ◦ ◦

```
<div ng-controller="bigCalculations as calc">
  <p>{{calc.calculateMe()}}</p>
  <p>{{calc.data | heavyFilter}}</p>
</div>
```

```
<div ng-controller="bigCalculations as calc">
  <p>{{calc.preCalculatedValue}}</p>
  <p>{{calc.data | lightFilter}}</p>
</div>
```

```
.controller("bigCalculations", function(valueService) {
  // bad, because this is called in every digest loop
  this.calculateMe = function() {
    var t = 0;
    for(i = 0; i < 1000; i++) {
      t = t + i;
    }
    return t;
  }
  //good, because it is executed just once and logic is separated in service to keep the
  controller light
  this.preCalculatedValue = valueService.caluclateSumm(); // returns 499500
});
```

◦

\$watch() \$watchCollection◦

**Watcher◦ Angular - \$watch() \$watchCollection◦**

```
(function() {
  angular.module("app", []).controller("ctrl", function($scope) {
    $scope.value = 10;
    $scope.$watch(
      function() { return $scope.value; },
      function() { console.log("value changed"); }
    );
  }
})();
```

◦ **UI◦**

**Angular◦**

\$watch() -

\$watchCollection() - **\$ watch**

\$watch(..., true) - **""watchCollection**

- **{{::variable}}**

◦ **@Words Like Jared -**

```
(function() {
  var root = angular.element(document.getElementsByTagName("body")),
      watchers = [];
```



```

var f = function(element) {

    angular.forEach(["$scope", "$isolateScope"], function(scopeProperty) {
        if(element.data() && element.data().hasOwnProperty(scopeProperty)) {
            angular.forEach(element.data()[scopeProperty].$$watchers, function(watcher) {
                watchers.push(watcher);
            });
        }
    });

    angular.forEach(element.children(), function(childElement) {
        f(angular.element(childElement));
    });

};

f(root);

// Remove duplicate watchers
var watchersWithoutDuplicates = [];
angular.forEach(watchers, function(item) {
    if(watchersWithoutDuplicates.indexOf(item) < 0) {
        watchersWithoutDuplicates.push(item);
    }
});

console.log(watchersWithoutDuplicates.length);

})();

```

[ng-stats](#) [Angular](#) [showAngularStats](#)

```

showAngularStats({
    "position": "topleft",
    "digestTimeThreshold": 16,
    "autoload": true,
    "logDigest": true,
    "logWatches": true
});

```

◦



## ng-if vs ng-show

◦ [ng-if](#) [DOM](#) ◦ [ng-if](#) ◦ [ng-show](#)

**NG-**

[ngIf](#) [DOM](#) ◦ [ngIf](#) [false](#) [DOM](#) [DOM](#) ◦

ngShowngShowHTML。 ng-hide CSS。

```
<div ng-repeat="user in userCollection">
  <p ng-if="user.hasTreeLegs">I am special
    <!-- some complicated DOM -->
  </p>
  <p ng-show="user.hasSubscribed">I am awesome
    <!-- switch this setting on and off -->
  </p>
</div>
```

95ng-if ;DOMng-show 。

ng-if

ng-ifng-showng-hide。 \$parent.property 。

```
<div ng-controller="ExampleController">
  <form name="userForm">
    Name:
    <input type="text" name="userName"
      ng-model="user.name"
      ng-model-options="{ debounce: 1000 }" />
    <button ng-click="userForm.userName.$rollbackViewValue();
user.name=''>Clear</button><br />
  </form>
  <pre>user.name = </pre>
</div>
```

10001。 ng-model\$digest。

Angular。 。 ng-model。

```
//always deregister these
$scope.$on(...);
$scope.$parent.$on(...);
```

angular

```
//no need to deregister this
$scope.$on(...);
```

\$rootScope.\$on 。

```
angular.module('app').controller('badExampleController', badExample);
badExample.$inject = ['$scope', '$rootScope'];
```

```
function badExample($scope, $rootScope) {
    $rootScope.$on('post:created', function postCreated(event, data) {});
}
```

```
angular.module('app').controller('goodExampleController', goodExample);
goodExample.$inject = ['$scope', '$rootScope'];

function goodExample($scope, $rootScope) {
    var deregister = $rootScope.$on('post:created', function postCreated(event, data) {});

    $scope.$on('$destroy', function destroyScope() {
        deregister();
    });
}
```

<https://riptutorial.com/zh-CN/angularjs/topic/1921/>

---

# 27:

## Examples

VS

- 'this'.

- 

- 

- 

- 

- ◦

JavaScript.

ES6. ◦

- 

\$ get. ◦

-

```
mod.provider("myProvider", fun
```

```
  this.$get = function() {
```

```
    return new function()
```

```
      this.getValue = fu
```

```
        return "My Va
```

```
      };
```

```
    };
```

```
  };
```

```
});
```

<https://riptutorial.com/zh-CN/angularjs/topic/7099/>

## 28:

AngularJS。 [Jasmine Jasmine](#)。 [ngMock](#)。

## Examples

```
angular.module('myModule', []).filter('multiplier', function() {
  return function(number, multiplier) {
    if (!angular.isNumber(number)) {
      throw new Error(number + " is not a number!");
    }
    if (!multiplier) {
      multiplier = 2;
    }
    return number * multiplier;
  }
});
```

```
describe('multiplierFilter', function() {
  var filter;

  beforeEach(function() {
    module('myModule');
    inject(function(multiplierFilter) {
      filter = multiplierFilter;
    });
  });

  it('multiply by 2 by default', function() {
    expect(filter(2)).toBe(4);
    expect(filter(3)).toBe(6);
  });

  it('allow to specify custom multiplier', function() {
    expect(filter(2, 4)).toBe(8);
  });

  it('throws error on invalid input', function() {
    expect(function() {
      filter(null);
    }).toThrow();
  });
});
```

inject+ *Filter*。 [Filter](#)。

## 1.5+

```
angular.module('myModule', []).component('myComponent', {
  bindings: {
    myValue: '<'
  },
  controller: function(MyService) {
    this.service = MyService;
  }
});
```

```

    this.componentMethod = function() {
        return 2;
    };
}
});

```

```

describe('myComponent', function() {
    var component;

    var MyServiceFake = jasmine.createSpyObj(['serviceMethod']);

    beforeEach(function() {
        module('myModule');
        inject(function($componentController) {
            // 1st - component name, 2nd - controller injections, 3rd - bindings
            component = $componentController('myComponent', {
                MyService: MyServiceFake
            }, {
                myValue: 3
            });
        });
    });

    /** Here you test the injector. Useless. */

    it('injects the binding', function() {
        expect(component.myValue).toBe(3);
    });

    it('has some cool behavior', function() {
        expect(component.componentMethod()).toBe(2);
    });
});

```

```

angular.module('myModule', [])
    .controller('myController', function($scope) {
        $scope.num = 2;
        $scope.doSomething = function() {
            $scope.num += 2;
        }
    });

```

```

describe('myController', function() {
    var $scope;
    beforeEach(function() {
        module('myModule');
        inject(function($controller, $rootScope) {
            $scope = $rootScope.$new();
            $controller('myController', {
                '$scope': $scope
            })
        });
    });
    it('should increment `num` by 2', function() {
        expect($scope.num).toEqual(2);
        $scope.doSomething();
        expect($scope.num).toEqual(4);
    });
});

```

```
});
```

```
angular.module('myModule', [])  
  .service('myService', function() {  
    this.doSomething = function(someNumber) {  
      return someNumber + 2;  
    }  
  });
```

```
describe('myService', function() {  
  var myService;  
  beforeEach(function() {  
    module('myModule');  
    inject(function(_myService_) {  
      myService = _myService_;  
    });  
  });  
  it('should increment `num` by 2', function() {  
    var result = myService.doSomething(4);  
    expect(result).toEqual(6);  
  });  
});
```

```
angular.module('myModule', [])  
  .directive('myDirective', function() {  
    return {  
      template: '<div>{{greeting}} {{name}}!</div>',  
      scope: {  
        name: '=',  
        greeting: '@'  
      }  
    };  
  });
```

```
describe('myDirective', function() {  
  var element, scope;  
  beforeEach(function() {  
    module('myModule');  
    inject(function($compile, $rootScope) {  
      scope = $rootScope.$new();  
      element = angular.element("<my-directive name='name' greeting='Hello'></my-directive>");  
      $compile(element)(scope);  
      /* PLEASE NEVER USE scope.$digest(). scope.$apply use a protection to avoid to run a  
      digest loop when there is already one, so, use scope.$apply() instead. */  
      scope.$apply();  
    })  
  });  
  
  it('has the text attribute injected', function() {  
    expect(element.html()).toContain('Hello');  
  });  
  
  it('should have proper message after scope change', function() {  
    scope.name = 'John';  
    scope.$apply();  
    expect(element.html()).toContain("John");  
    scope.name = 'Alice';  
  });  
});
```



```
expect(element.html()).toContain("John");  
scope.$apply();  
expect(element.html()).toContain("Alice");  
});  
});
```

<https://riptutorial.com/zh-CN/angularjs/topic/1689/>

# 29:

## Examples

[node.js](#) npm

GitHub @ <https://github.com/mikkoviitala/angular-grunt-run-local>

Web

[grunt](#) [javascript](#) [npm](#) [bower](#)

- **root 3**
  - package.json
  - bower.json
  - gruntfile.js



---

### package.json

**grunt matchdep grunt-express** [gruntWeb](#) **grunt-open** [grunturl](#) /

“”

```
{
  "name": "app",
  "version": "1.0.0",
  "dependencies": {},
  "devDependencies": {
    "grunt": "~0.4.1",
    "matchdep": "~0.1.2",
    "grunt-express": "~1.0.0-beta2",
    "grunt-open": "~0.2.1"
  },
  "scripts": {
    "postinstall": "bower install"
  }
}
```

### bower.json

Bower

```
{
```

```
"name": "app",
"version": "1.0.0",
"dependencies": {
  "angular": "~1.3.x"
},
"devDependencies": {}
}
```

## gruntfile.js

gruntfile.js [http:// localhost9000 /](http://localhost9000/)

```
'use strict';

// see http://rhumaric.com/2013/07/renewing-the-grunt-livereload-magic/

module.exports = function(grunt) {
  require('matchdep').filterDev('grunt-*').forEach(grunt.loadNpmTasks);

  grunt.initConfig({
    express: {
      all: {
        options: {
          port: 9000,
          hostname: 'localhost',
          bases: [__dirname]
        }
      }
    },
    open: {
      all: {
        path: 'http://localhost:<%= express.all.options.port%>'
      }
    }
  });

  grunt.registerTask('app', [
    'express',
    'open',
    'express-keepalive'
  ]);
};
```

◦ /◦

```
npm install -g grunt-cli bower
npm install
```











```
grunt app
```

◦ [GitHub](#) ◦

◦ index.html app.jsapp.css ◦ [Git](#) ◦

## **AngularJS application**

Hello Stack Overflow Documentation (beta)

-  .bowerrc
-  .gitignore
-  LICENSE
-  README.MD
-  app.css
-  app.js
-  bower.json
-  gruntfile.js
-  index.html
-  package.json

<https://riptutorial.com/zh-CN/angularjs/topic/6077/>

# 30:

## Examples

◦ ◦ addZ ◦ "Z"◦

### example.js

```
angular.module('main', [])
  .filter('addZ', function() {
    return function(value) {
      return value + "Z";
    }
  })
  .controller('MyController', ['$scope', function($scope) {
    $scope.sample = "hello";
  }])
```

### example.html

{ variable | filter} ◦ sampleaddZ◦

```
<div ng-controller="MyController">
  <span>{{sample | addZ}}</span>
</div>
```

helloZ

\$filter

```
angular
  .module('filters', [])
  .filter('percentage', function($filter) {
    return function (input) {
      return $filter('number')(input * 100) + ' %';
    };
  });
```

◦

```
angular
  .module('app', [])
  .controller('MyController', function($scope) {
    $scope.example = 0.098152;
  })
  .filter('percentage', function($filter) {
    return function (input, decimals) {
      return $filter('number')(input * 100, decimals) + ' %';
    };
  });
```

```
};  
});
```

percentage

```
<span ng-controller="MyController">{{ example | percentage: 2 }}</span>  
=> "9.81 %"
```

...

```
<span ng-controller="MyController">{{ example | percentage }}</span>  
=> "9.8152 %"
```

<https://riptutorial.com/zh-CN/angularjs/topic/2552/>

# 31: AngularJS TypeScript

- [\\$ scopeng.IScope - typescript](#).

## Examples

### AngularJS

Controllerng-controllerDOMAngularControllerController。 \$ scopeController。

### typescript

```
module App.Controllers {
  class Address {
    line1: string;
    line2: string;
    city: string;
    state: string;
  }
  export class SampleController {
    firstName: string;
    lastName: string;
    age: number;
    address: Address;
    setUpWatches($scope: ng.IScope): void {
      $scope.$watch(() => this.firstName, (n, o) => {
        //n is string and so is o
      });
    };
    constructor($scope: ng.IScope) {
      this.setUpWatches($scope);
    }
  }
}
```

### Javascript

```
var App;
(function (App) {
  var Controllers;
  (function (Controllers) {
    var Address = (function () {
      function Address() {
      }
      return Address;
    })();
    var SampleController = (function () {
      function SampleController($scope) {
        this.setUpWatches($scope);
      }
      SampleController.prototype.setUpWatches = function ($scope) {
        var _this = this;
        $scope.$watch(function () { return _this.firstName; }, function (n, o) {
          //n is string and so is o
        });
      };
    })();
  });
})(App);
```

```

        });
    };
    ;
    return SampleController;
}());
Controllers.SampleController = SampleController;
})(Controllers = App.Controllers || (App.Controllers = {}));
})(App || (App = {}));
//# sourceMappingURL=ExampleController.js.map

```

js

```

app
  .module('app')
  .controller('exampleController', App.Controller.SampleController)

```

## ControllerControllerAs

**Controller** controller as° \$scope °

controller as someName\$scope° \$ scope°

```

// we are using $scope object.
app.controller('MyCtrl', function ($scope) {
  $scope.name = 'John';
});

<div ng-controller="MyCtrl">
  {{name}}
</div>

```

**controller as**

```

// we are using the "this" Object instead of "$scope"
app.controller('MyCtrl', function() {
  this.name = 'John';
});

<div ng-controller="MyCtrl as info">
  {{info.name}}
</div>

```

## JavaScript“”

```

var jsClass = function () {
  this.name = 'John';
}
var jsObj = new jsClass();

```

jsObjjsClass °

angular° °



/

\$scope. angular. ExampleController.

```
function n(n){this.setUpWatches(n)}
```

\$scope

\$inject string[] DI.

```
module App.Controllers {
  class Address {
    line1: string;
    line2: string;
    city: string;
    state: string;
  }
  export class SampleController {
    firstName: string;
    lastName: string;
    age: number;
    address: Address;
    setUpWatches($scope: ng.IScope): void {
      $scope.$watch(() => this.firstName, (n, o) => {
        //n is string and so is o
      });
    };
    static $inject : string[] = ['$scope'];
    constructor($scope: ng.IScope) {
      this.setUpWatches($scope);
    }
  }
}
```

## ControllerAs

ControllerJavaScript. function context **this**.

1

```
this.constFunction = function() { ... }
```

controller object\$scope . .

```
<a href="#123" ng-click="constFunction()"></a> // It will not work
```

2

```
$scope.scopeFunction = function() { ... }
```

\$scope objectcontroller object . \$scope.

```
<a href="#123" ng-click="scopeFunction()"></a> // It will work
```

## ControllerAs

- **ControllerAs** oneCtrl.name ◦ oneCtrl.name anotherCtrl.name name \$scope.name **HTML** `{{ name }}` ◦
- \$scope intermediary object ◦ this.\* ◦

```
<div ng-controller="FirstCtrl">
  {{ name }}
  <div ng-controller="SecondCtrl">
    {{ name }}
    <div ng-controller="ThirdCtrl">
      {{ name }}
    </div>
  </div>
</div>
```

`{{ name }}` ◦

```
<div ng-controller="FirstCtrl as first">
  {{ first.name }}
  <div ng-controller="SecondCtrl as second">
    {{ second.name }}
    <div ng-controller="ThirdCtrl as third">
      {{ third.name }}
    </div>
  </div>
</div>
```

- \$scope **\$ scope** \$watch \$digest \$emit \$http ◦
- /\$scope \$scope ◦

**AngularJS TypeScript** <https://riptutorial.com/zh-CN/angularjs/topic/3477/angularjstypescript>

## 32: ES6

### Examples

#### ES6angularJS

```
class exampleContoller{

  constructor (service1, service2, ...serviceN) {
    let ctrl=this;
    ctrl.service1=service1;
    ctrl.service2=service2;
    .
    .
    .
    ctrl.service1=service1;
    ctrl.controllerName = 'Example Controller';
    ctrl.method1 (controllerName)

  }

  method1 (param) {
    let ctrl=this;
    ctrl.service1.serviceFunction();
    .
    .
    ctrl.scopeName=param;
  }
  .
  .
  .
  methodN (param) {
    let ctrl=this;
    ctrl.service1.serviceFunction();
    .
    .
  }

}

exampleContoller.$inject = ['service1', 'service2', ..., 'serviceN'];
export default exampleContoller;
```

ES6 <https://riptutorial.com/zh-CN/angularjs/topic/9419/es6>

# 33:

o

```
.controller('MyController', function($scope, Profile, EVENT))
```

- \$scope
- Profile
- EVENT

## Examples

```
angular
  .module('MyApp', [])
  .constant('VERSION', 1.0);
```

```
angular
  .module('MyApp')
  .controller('FooterController', function(VERSION) {
    this.version = VERSION;
  });
```

```
<footer ng-controller="FooterController as Footer">{{ Footer.version }}</footer>
```

/.....

- o o o .....  
.....

```
angular
  .module('MyApp')
  .constant('EVENTS', {
    LOGIN_VALIDATE_FORM: 'login::click-validate',
    LOGIN_FORGOT_PASSWORD: 'login::click-forgot',
    LOGIN_ERROR: 'login::notify-error',
    ...
  });
```

```
angular
  .module('MyApp')
  .controller('LoginController', function($scope, EVENT) {
    $scope.$on(EVENT.LOGIN_VALIDATE_FORM, function() {
      ...
    });
  });
```

...

- o

```
angular
  .module('MyApp')
  .constant('CONFIG', {
    BASE_URL: {
      APP: 'http://localhost:3000',
      API: 'http://localhost:3001'
    },
    STORAGE: 'S3',
    ...
  });
```

• ◦ ...◦

```
angular
  .module('MyApp')
  .constant('HARDCODED', {
    KEY: 'KEY',
    RELATION: 'has_many',
    VAT: 19.6
  });
```

.....

```
$scope.settings = {
  username: Profile.username,
  relation: 'has_many',
  vat: 19.6
}
```

```
$scope.settings = {
  username: Profile.username,
  relation: HARDCODED.RELATION,
  vat: HARDCODED.VAT
}
```

<https://riptutorial.com/zh-CN/angularjs/topic/3967/>

# 34:

1.

```
angular.module('lazy', [  
  'alreadyLoadedDependency1',  
  'alreadyLoadedDependency2',  
  ...  
{  
  files: [  
    'path/to/lazily/loaded/dependency1.js',  
    'path/to/lazily/loaded/dependency2.js', //<--- requires lazily loaded dependency1  
    'path/to/lazily/loaded/dependency.css'  
  ],  
  serie: true //Sequential load instead of parallel  
}  
]);
```

## Examples

ocLazyLoad.js

```
//Make sure you put the correct dependency! it is spelled different than the service!  
angular.module('app', [  
  'oc.lazyLoad',  
  'ui-router'  
]);
```

\$ocLazyLoad

```
.controller('someCtrl', function($ocLazyLoad) {  
  $ocLazyLoad.load('path/to/file.js').then(...);  
});
```

o

```
$ocLazyLoad.load([  
  'bower_components/bootstrap/dist/js/bootstrap.js',  
  'bower_components/bootstrap/dist/css/bootstrap.css',  
  'partials/template1.html'  
]);
```

## UI

```
.state('profile', {  
  url: '/profile',  
  controller: 'profileCtrl as vm'  
  resolve: {  
    module: function($ocLazyLoad) {  
      return $ocLazyLoad.load([  
        'path/to/profile/module.js',
```

```
    'path/to/profile/style.css'
  });
}
}
});
```

## ngRoute

```
.when('/profile', {
  controller: 'profileCtrl as vm'
  resolve: {
    module: function($ocLazyLoad) {
      return $ocLazyLoad.load([
        'path/to/profile/module.js',
        'path/to/profile/style.css'
      ]);
    }
  }
});
```

module.js

```
//lazy_module.js
angular.module('lazy', [
  'alreadyLoadedDependency1',
  'alreadyLoadedDependency2',
  ...
  [
    'path/to/lazily/loaded/dependency.js',
    'path/to/lazily/loaded/dependency.css'
  ]
]);
```

```
<div oc-lazy-load="['path/to/lazy/loaded/directive.js',
'path/to/lazy/loaded/directive.html']">

<!-- myDirective available here -->
<my-directive></my-directive>

</div>
```

<https://riptutorial.com/zh-CN/angularjs/topic/6400/>

# 35:

## Examples

- 
- ◦

### 1. Chrome

- [javascriptcsscpu](#)◦
- [JS ProfilerJS](#)◦ ...

### 2. FireBug Firefox

### 3. Dynatrace IE

### 4. Batarang Chrome

Chrome◦ ◦ ◦

### 5. Watcher Chrome

AngularUI◦

### 6. @Words Jared

```
(function() {
  var root = angular.element(document.getElementsByTagName('body')),
      watchers = [],
      f = function(element) {
        angular.forEach(['$scope', '$isolateScope'], function(scopeProperty) {
          if(element.data() && element.data().hasOwnProperty(scopeProperty)) {
            angular.forEach(element.data()[scopeProperty].$$watchers, function(watcher) {
              watchers.push(watcher);
            });
          }
        });
      };

  angular.forEach(element.children(), function(childElement) {
    f(angular.element(childElement));
  });
});

f(root);

// Remove duplicate watchers
var watchersWithoutDuplicates = [];
angular.forEach(watchers, function(item) {
  if(watchersWithoutDuplicates.indexOf(item) < 0) {
    watchersWithoutDuplicates.push(item);
  }
});
```



```
});  
console.log(watchersWithoutDuplicates.length);  
})();
```

7. /。

[https //www.webpagetest.org/](https://www.webpagetest.org/)

IEChrome。 。

。 。 。 。

:)

<https://riptutorial.com/zh-CN/angularjs/topic/7033/>

## 36:

cssng-hide。 ng-show / hide。

## Examples

```
angular.module('core').factory('print_service', ['$rootScope', '$compile', '$http',
'$timeout', '$q',
function($rootScope, $compile, $http, $timeout, $q) {

    var printHtml = function (html) {
        var deferred = $q.defer();
        var hiddenFrame = $('<iframe style="display:
none"></iframe>').appendTo('body')[0];

        hiddenFrame.contentWindow.printAndRemove = function() {
            hiddenFrame.contentWindow.print();
            $(hiddenFrame).remove();
            deferred.resolve();
        };

        var htmlContent =    "<!doctype html>" +
                            "<html>" +
                                '<head><link rel="stylesheet" type="text/css"
href="/style/css/print.css"/></head>' +
                                '<body onload="printAndRemove();">' +
                                    html +
                                '</body>' +
                            "</html>";

        var doc = hiddenFrame.contentWindow.document.open("text/html", "replace");
        doc.write(htmlContent);
        doc.close();
        return deferred.promise;
    };

    var openNewWindow = function (html) {
        var newWindow = window.open("debugPrint.html");
        newWindow.addEventListener('load', function() {
            $(newWindow.document.body).html(html);
        }, false);
    };

    var print = function (templateUrl, data) {

        $rootScope.isBeingPrinted = true;

        $http.get(templateUrl).success(function(template) {
            var printScope = $rootScope.$new()
            angular.extend(printScope, data);
            var element = $compile($('<div>' + template + '</div>'))(printScope);
            var waitForRenderAndPrint = function() {
                if(printScope.$$phase || $http.pendingRequests.length) {
                    $timeout(waitForRenderAndPrint, 1000);
                } else {
                    // Replace printHtml with openNewWindow for debugging
                    printHtml(element.html());
                }
            };
        });
    };
});
```

```

        printScope.$destroy();
    }
};
waitForRenderAndPrint();
});
};

var printFromScope = function (templateUrl, scope, afterPrint) {
    $rootScope.isBeingPrinted = true;
    $http.get(templateUrl).then(function(response) {
        var template = response.data;
        var printScope = scope;
        var element = $compile($('

' + template + '</div>'))(printScope);
        var waitForRenderAndPrint = function() {
            if (printScope.$$phase || $http.pendingRequests.length) {
                $timeout(waitForRenderAndPrint);
            } else {
                // Replace printHtml with openNewWindow for debugging
                printHtml(element.html()).then(function() {
                    $rootScope.isBeingPrinted = false;
                    if (afterPrint) {
                        afterPrint();
                    }
                });
            }
        };
        waitForRenderAndPrint();
    });
};

return {
    print : print,
    printFromScope : printFromScope
}
}
});


```

```

var template_url = '/views/print.client.view.html';
print_service.printFromScope(template_url, $scope, function() {
    // Print Completed
});

```

<https://riptutorial.com/zh-CN/angularjs/topic/6750/>

## 37:

- `<htmlElement ng-controller =“controllerName”> ... </ htmlElement>`
- `<script> app.controller'controllerName'controllerFunction; </ SCRIPT>`

## Examples

Angular。HTML。

Angular

```
<!DOCTYPE html>

<html lang="en" ng-app='MyFirstApp'>
  <head>
    <title>My First App</title>

    <!-- angular source -->
    <script src="https://code.angularjs.org/1.5.3/angular.min.js"></script>

    <!-- Your custom controller code -->
    <script src="js/controllers.js"></script>
  </head>
  <body>
    <div ng-controller="MyController as mc">
      <h1>{{ mc.title }}</h1>
      <p>{{ mc.description }}</p>
      <button ng-click="mc.clicked()">
        Click Me!
      </button>
    </div>
  </body>
</html>
```

```
<html ng-app='MyFirstApp'>
```

ng-appJavaScript。

```
<script src="js/controllers.js"></script>
```

JavaScript/。

```
<div ng-controller="MyController as mc">
```

ng-controllerDOM。

... as mc MyController 。

```
<h1>{{ mc.title }}</h1>
```

```
{{ ... }}Angular◦ <h1>mc.title◦
```

Angularmc.title◦

◦ Angular{{ 1 + 2 }}{{ "Hello " + "World" }}◦

```
<button ng-click="mc.clicked()">
```

ng-clickAngularclickMyControllerclicked()◦

---

MyController◦ js/controller.js◦

JavascriptAngular◦

```
var app = angular.module("MyFirstApp", []);
```

ng-appHTML◦

app◦

```
app.controller('MyController', function(){
  var ctrl = this;

  ctrl.title = "My First Angular App";
  ctrl.description = "This is my first Angular app!";

  ctrl.clicked = function(){
    alert("MyController.clicked()");
  };
});
```

this◦

◦

```
angular
  .module('app')
  .controller('SampleController', SampleController)

SampleController.$inject = ['$log', '$scope'];
function SampleController($log, $scope){
  $log.debug('*****SampleController*****');

  /* Your code below */
}
```

.\$inject◦ ◦

## Minification

◦

◦

```
var app = angular.module('app');
app.controller('sampleController', ['$scope', '$http', function(a, b){
  //logic here
}]);
```

◦ \$scope\$http a\$scope b\$http ◦ ◦

\$inject◦

```
var app = angular.module('app');
app.controller('sampleController', sampleController);
sampleController.$inject = ['$scope', '$http'];
function sampleController(a, b) {
  //logic here
}
```

◦

◦

\$scope\$http◦ ◦

```
// Intentional Bug: injected dependencies are reversed which will cause a problem
app.controller('sampleController', ['$scope', '$http',function($http, $scope) {
  $http.get('sample.json');
}]);
```

## Angular JSControllerAs

Angular \$scopeControllerView◦ ◦ Angular\$ scopeController As◦

Controller As -

### controllerAs

```
<div ng-controller="CustomerController as customer">
  {{ customer.name }}
</div>
```

### controllerAs

```
function CustomerController() {
  this.name = {};
  this.sendMessage = function() { };
}
```

### controllervm

```
function CustomerController() {
  /*jshint validthis: true */
  var vm = this;
  vm.name = {};
  vm.sendMessage = function() { };
}
```

controllerAs\$scope ◦ **View**\$scope ◦ controllerAs ◦ -

- \$scope ◦ this.\* ◦ **JavaScript** ◦
- controllerAs\$parent ◦
- **""customer.name""** ◦
- **Views**\$parent ◦
- controllerAs ◦ vm **ViewModel** ◦ this ◦ ◦

controllerAscontrollerAs

```
<div ng-controller="Controller as vm">...</div>
```

vm\$scope.vm ◦

## Minification-Safe

controller ◦

module.controller ◦

;

```
var app = angular.module('myApp');
```

```
app.controller('ctrlInject',
  [
    /* Injected Parameters */
    '$Injectable1',
    '$Injectable2',
    /* Controller Function */
    function($injectable1Instance, $injectable2Instance) {
      /* Controller Content */
    }
  ]
);
```

◦

```
var
a=angular.module('myApp');a.controller('ctrlInject', ['$Injectable1', '$Injectable2', function(b,c) { /*
Controller Content */}]);
```

```
$Injectable1Instance bapp a $Injectable2Instance c $Injectable1Instance
```

```
$scope ◦ $scope $scope ◦
```

```
.controller('parentController', function ($scope) {
    $scope.parentVariable = "I'm the parent";
});

.controller('childController', function ($scope) {
    $scope.childVariable = "I'm the child";

    $scope.childFunction = function () {
        $scope.parentVariable = "I'm overriding you";
    };
});
```

◦

```
<body ng-controller="parentController">
  What controller am I? {{parentVariable}}
  <div ng-controller="childController">
    What controller am I? {{childVariable}}
    <button ng-click="childFunction()"> Click me to override! </button>
  </div>
</body>
```

◦ ngController - ◦

<https://riptutorial.com/zh-CN/angularjs/topic/601/>



# 38: SelfThis

AngularJS。

## Examples

“ng-controller。

```
<div ng-controller="MainCtrl as main">
</div>
```

◦ ***greeting***

```
<div ng-controller="MainCtrl as main">
  {{ main.greeting }}
</div>
```

greeting\$ scope

```
angular
.module('ngNjOrg')
.controller('ForgotPasswordController',function ($log) {
  var self = this;

  self.greeting = "Hello World";
})
```

HTML。 ***self***。

```
angular
.module('ngNjOrg')
.controller('ForgotPasswordController',function ($log) {
  var self = this;

  self.greeting = "Hello World";

  function itsLate () {
    this.greeting = "Goodnight";
  }
})
```

***itsLate***。 JavaScriptitsLate“this”“this”。

```
angular
.module('ngNjOrg')
.controller('ForgotPasswordController',function ($log) {
  var self = this;

  self.greeting = "Hello World";
```

```
function itsLate () {  
    self.greeting = "Goodnight";  
}  
  
})
```

“” - 。

**SelfThis** <https://riptutorial.com/zh-CN/angularjs/topic/8867/selfthis>

# 39:

Angular。 Scope

## Examples

```
<p ng-bind="message"></p>
```

“”。

```
$scope.message = "Hello World";
```

HTML。 “Hello World”innerHTML。 Angular。 Digest CycleWatchersDOM。

Scope。 ◦ **\$\$ WatchersArray** ◦ \$ digestScopeWatchersArray。

AngularWatchersArray

1. {{}} - ng-model。
2. \$ scope。 \$ watch'expression / function' - JavaScript。

### \$ watch

1. ◦
2. ◦ DOM。
3. ◦ ◦ \$ watch

```
Scope.prototype.$watch = function(watchFn, listenerFn) {  
  var watcher = {  
    watchFn: watchFn,  
    listenerFn: listenerFn || function() { },  
    last: initWatchVal // initWatchVal is typically undefined  
  };  
  this.$$watchers.push(watcher); // pushing the Watcher Object to Watchers  
};
```

AngularDigest Cycle。 \$ digest\$ scope。 \$ digest。 ng-click\$ scope。 AngularJS\$ digest\$ digest。  
ng-click/ng-model\$ timeout\$ digest。 \$ digest。

```
Scope.prototype.$digest = function() {  
  var dirty;  
  do {  
    dirty = this.$$digestOnce();  
  } while (dirty);  
}  
Scope.prototype.$$digestOnce = function() {  
  var self = this;
```

```

var newValue, oldValue, dirty;
_.forEach(this.$$watchers, function(watcher) {
    newValue = watcher.watchFn(self);
    oldValue = watcher.last;    // It just remembers the last value for dirty checking
    if (newValue !== oldValue) { //Dirty checking of References
// For Deep checking the object , code of Value
// based checking of Object should be implemented here
        watcher.last = newValue;
        watcher.listenerFn(newValue,
            (oldValue === initWatchVal ? newValue : oldValue),
            self);
        dirty = true;
    }
});
return dirty;
};

```

JavaScriptsetTimeoutAngular。 \$ apply\$ digest。 DOM\$ apply。 \$ applyAngular。 \$ apply\$ digest  
。 \$ apply。

```

Scope.prototype.$apply = function(expr) {
    try {
        return this.$eval(expr); //Evaluating code in the context of Scope
    } finally {
        this.$digest();
    }
};

```

<https://riptutorial.com/zh-CN/angularjs/topic/2342/>

# 40:

## Examples

```
angular.module("app")
  .service("counterService", function(){

    var service = {
      number: 0
    };

    return service;
  });
```

```
angular.module("app")

  // Custom services are injected just like Angular's built-in services
  .controller("step1Controller", ['counterService', '$scope', function(counterService,
$scope) {
    counterService.number++;
    // bind to object (by reference), not to value, for automatic sync
    $scope.counter = counterService;
  })
```

```
// editable
<input ng-model="counter.number" />
```

```
// read-only
<span ng-bind="counter.number"></span>
```

◦ ◦

---

## Angularjs

### \$ injector◦

- ◦

## angular.factory

```
.factory('dataService', function() {
  var dataObject = {};
  var service = {
    // define the getter method
    get data() {
      return dataObject;
    },
    // define the setter method
    set data(value) {
      dataObject = value || {};
    }
  };
});
```

```

    }
};
// return the "service" object to expose the getter/setter
return service;
})

```

```

.controller('controllerOne', function(dataService) {
    // create a local reference to the dataService
    this.dataService = dataService;
    // create an object to store
    var someObject = {
        name: 'SomeObject',
        value: 1
    };
    // store the object
    this.dataService.data = someObject;
})

.controller('controllerTwo', function(dataService) {
    // create a local reference to the dataService
    this.dataService = dataService;
    // this will automatically update with any changes to the shared data object
    this.objectFromControllerOne = this.dataService.data;
})

```

## \$sce -

\$sce “Strict Contextual Escaping”。

HTML<sub>\$sce</sub>。

\$sce `。

```

.filter('sanitizer', ['$sce', [function($sce) {
    return function(content) {
        return $sce.trustAsResourceUrl(content);
    };
}]);

```

```

<div ng-repeat="item in items">

    // Sanitize external sources
    <iframe ng-src="{{item.youtube_url | sanitizer}}">

    // Sanitize and render HTML
    <div ng-bind-html="{{item.raw_html_content | sanitizer}}"></div>

</div>

```

”

```

angular.module("app")
    .service("counterService", ["fooService", "barService", function(anotherService,
barService){

```

```

var service = {
  number: 0,
  foo: function () {
    return fooService.bazMethod(); // Use of 'fooService'
  },
  bar: function () {
    return barService.bazMethod(); // Use of 'barService'
  }
};

return service;
});

```

## angular.module API

```

angular.module('myApp.services', []).factory('githubService', function() {
  var serviceInstance = {};
  // Our first service
  return serviceInstance;
});

```

```

// Creating the factory through using the
// bracket notation
angular.module('myApp.services', [])
.factory('githubService', [function($http) {
}]);

```

◦

```

angular.module('myApp.services', [])
  .factory('githubService', function($http) {
    var githubUrl = 'https://api.github.com';
    var runUserRequest = function(username, path) {
      // Return the promise from the $http service
      // that calls the Github API using JSONP
      return $http({
        method: 'JSONP',
        url: githubUrl + '/users/' +
            username + '/' +
            path + '?callback=JSON_CALLBACK'
      });
    }
  })
// Return the service object with a single function
// events
return {
  events: function(username) {
    return runUserRequest(username, 'events');
  }
};

```

## 1

constructornew **javascript** AngularJsnew ◦

1. new

\$http

```
function StudentDetailsService($http) {
  this.getStudentDetails = function getStudentDetails() {
    return $http.get('/details');
  };
}

angular.module('myapp').service('StudentDetailsService', StudentDetailsService);
```

```
function StudentController(StudentDetailsService) {
  StudentDetailsService.getStudentDetails().then(function (response) {
    // handle response
  });
}

angular.module('app').controller('StudentController', StudentController);
```

.service() ◦ **API**getStudentDetails() ◦ **API**.service()◦

## 2

.factory().services() .factory() "" .service()◦ .service()

.factory()◦

- 1.
- 2.

.factory()

## Revealing

```
function StudentDetailsService($http) {
  function getStudentDetails() {
    return $http.get('/details');
  }
  return {
    getStudentDetails: getStudentDetails
  };
}

angular.module('myapp').factory('StudentDetailsService', StudentDetailsService);
```

```
function StudentController(StudentDetailsService) {
  StudentDetailsService.getStudentDetails().then(function (response) {
    // handle response
  });
}

angular.module('app').controller('StudentController', StudentController);
```

◦

```
function closureFunction(name) {
  function innerClosureFunction(age) { // innerClosureFunction() is the inner function, a
    closure
  }
}
```



```
    // Here you can manipulate 'age' AND 'name' variables both
  };
};
```

“name ◦

.factory()

```
function StudentDetailsService($http) {
  function closureFunction(name) {
    function innerClosureFunction(age) {
      // Here you can manipulate 'age' AND 'name' variables
    };
  };
};

angular.module('myapp').factory('StudentDetailsService', StudentDetailsService);
```

```
function StudentController(StudentDetailsService) {
  var myClosure = StudentDetailsService('Student Name'); // This now HAS the
  innerClosureFunction()
  var callMyClosure = myClosure(24); // This calls the innerClosureFunction()
};

angular.module('app').controller('StudentController', StudentController);
```

/

.service()new◦ .factory()new

```
function StudentDetailsService($http) {
  function Student() {
    this.age = function () {
      return 'This is my age';
    };
  }
  Student.prototype.address = function () {
    return 'This is my address';
  };
  return Student;
};

angular.module('myapp').factory('StudentDetailsService', StudentDetailsService);
```

```
function StudentController(StudentDetailsService) {
  var newStudent = new StudentDetailsService();

  //Now the instance has been created. Its properties can be accessed.

  newStudent.age();
  newStudent.address();

};

angular.module('app').controller('StudentController', StudentController);
```



# 41:

## Examples

- Angular◦

```
angular
  .module('app', []);
```

- Array [] app Empty Array[] ◦

```
angular.module('app', [
  'app.auth',
  'app.dashboard'
]);
```

```
angular
  .module('app');
```

- ◦

- Angular◦

- AngularJs◦

- 

```
var app = angular.module('myApp', []);
// Empty array is list of modules myApp is depends on.
// if there are any required dependancies,
// then you can add in module, Like ['ngAnimate']

app.controller('myController', function() {

  // write your business logic here
});
```

1. - ◦

```
angular.module('myModule', []).
  config(function(injectables) {
    // here you can only inject providers in to config blocks.
  });
```

2. - ◦

```
angular.module('myModule', []).
  run(function(injectables) {
```

```
// here you can only inject instances in to config blocks.  
});
```

<https://riptutorial.com/zh-CN/angularjs/topic/844/>

## 42:

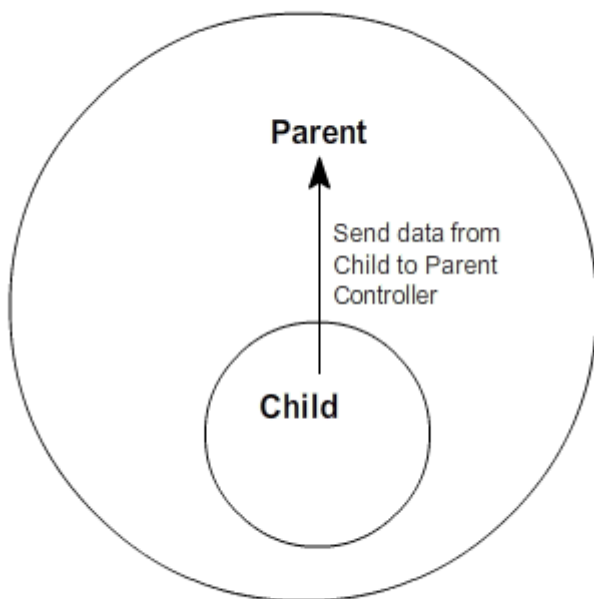
```
Object {name“eventName”targetScopeScopedefaultPreventedfalsecurrentScope  
ChildScope}
```

ARGS

## Examples

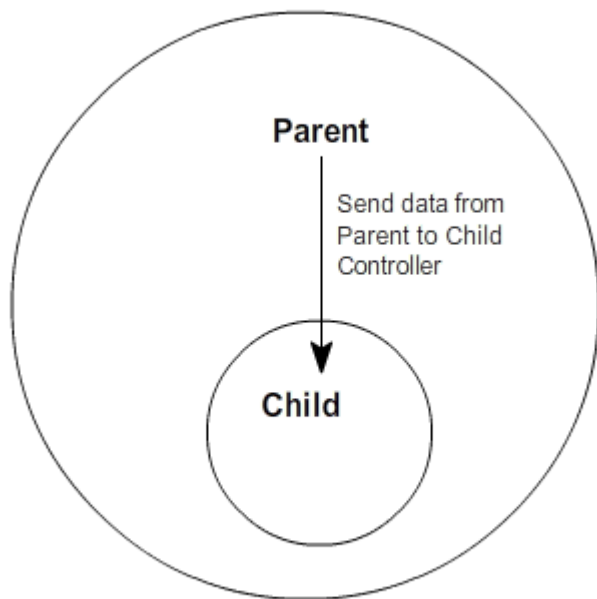
`$scope.$emit` `$scope.$emit`

```
$scope.$emit($scope.$emit)
```



`$scope.$broadcast` `$scope.$on`

```
$scope.$broadcast($scope.$on)
```



```
// firing an event upwards
$scope.$emit('myCustomEvent', 'Data to send');

// firing an event downwards
$scope.$broadcast('myCustomEvent', {
  someProp: 'some value'
});

// listen for the event in the relevant $scope
$scope.$on('myCustomEvent', function (event, data) {
  console.log(data); // 'Data from the event'
});
```

```
$rootScope.$scope
```

## AngularJS

o o

```
// firing an event upwards
$rootScope.$emit('myEvent', 'Data to send');

// listening an event
var listenerEventHandler = $rootScope.$on('myEvent', function() {
  //handle code
});

$scope.$on('$destroy', function() {
  listenerEventHandler();
});
```

2.

\$emit\$broadcast ◦ ◦

---

1. ◦ \$scope

---

2. ◦ \$rootScope

ProductListController CartController ◦ CartController /◦ \$rootScope◦

---

\$scope.\$emit

```
<html>
  <head>
    <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.4/angular.js"></script>
    <script>
      var app = angular.module('app', []);

      app.controller("FirstController", function ($scope) {
        $scope.$on('eventName', function (event, args) {
          $scope.message = args.message;
        });
      });

      app.controller("SecondController", function ($scope) {
        $scope.handleClick = function (msg) {
          $scope.$emit('eventName', {message: msg});
        };
      });

    </script>
  </head>
  <body ng-app="app">
    <div ng-controller="FirstController" style="border:2px ;padding:5px;">
      <h1>Parent Controller</h1>
      <p>Emit Message : {{message}}</p>
      <br />
      <div ng-controller="SecondController" style="border:2px;padding:5px;">
        <h1>Child Controller</h1>
        <input ng-model="msg">
        <button ng-click="handleClick(msg);">Emit</button>
      </div>
    </div>
  </body>
</html>
```

\$scope.\$broadcast

```
<html>
  <head>
    <title>Broadcasting</title>
    <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.4/angular.js"></script>
    <script>
      var app = angular.module('app', []);
```

```

    app.controller("FirstController", function ($scope) {
        $scope.handleClick = function (msg) {
            $scope.$broadcast('eventName', {message: msg});
        };
    });

    app.controller("SecondController", function ($scope) {
        $scope.$on('eventName', function (event, args) {
            $scope.message = args.message;
        });
    });

</script>
</head>
<body ng-app="app">
    <div ng-controller="FirstController" style="border:2px solid ; padding:5px;">
        <h1>Parent Controller</h1>
        <input ng-model="msg">
        <button ng-click="handleClick(msg);">Broadcast</button>
        <br /><br />
        <div ng-controller="SecondController" style="border:2px solid ;padding:5px;">
            <h1>Child Controller</h1>
            <p>Broadcast Message : {{message}}</p>
        </div>
    </div>
</body>
</html>

```

## \$ destroy\$rootScope。 \$

### \$rootScope。 \$on。 。

```

angular.module('app').controller('badExampleController', badExample);

badExample.$inject = ['$scope', '$rootScope'];
function badExample($scope, $rootScope) {

    $rootScope.$on('post:created', function postCreated(event, data) {});

}

```

```

angular.module('app').controller('goodExampleController', goodExample);

goodExample.$inject = ['$scope', '$rootScope'];
function goodExample($scope, $rootScope) {

    var deregister = $rootScope.$on('post:created', function postCreated(event, data) {});

    $scope.$on('$destroy', function destroyScope() {
        deregister();
    });

}

```

<https://riptutorial.com/zh-CN/angularjs/topic/1922/>



# 43:

- \$ scope. \$ watchwatchExpressioncallback[deep compare]
- \$. \$
- \$. \$[]

## Examples

Angular. [DOMjs](#).

Angular“ *digest loop*” - DOM.

ng-modelkeyup [eventListener](#)

```
<input ng-model="variable" />
```

keyup ◦

```
<span>{{variable}}</span>
```

::

### 1. html

- ng-modelkeyup
- spanexpression

### 2.

- keyup
- 
- 

## \$ digest\$ watch

- **\$ digest** DOM =>
- **\$ watch**=> DOM

```
<input id="input"/>
<span id="span"></span>
```

```
var $watches = [];
function $digest() {
    $watches.forEach(function($w) {
        var val = $w.val();
        if($w.prevVal !== val) {
            $w.callback(val, $w.prevVal);
            $w.prevVal = val;
        }
    });
}
function $watch(val, callback) {
```

```
$watches.push({val:val, callback:callback, prevVal: val() })
}
```

## DOMangular

```
var realVar;
//this is usually done by ng-model directive
input1.addEventListener('keyup',function(e){
    realVar=e.target.value;
    $digest()
}, true);

//this is usually done with {{expressions}} or ng-bind directive
$watch(function(){return realVar},function(val){
    span1.innerHTML = val;
});
```

<https://jsfiddle.net/azofxd4j/>

## \$ scope

html。

-

```
<span ng-repeat="number in [1,2,3,4,5]">{{number}}</span>
```

ng-repeat<sup>55</sup>number

---

angular。。

DOM \$digest\$watch。

DOM

```
<div>
  <input ng-model="person.name" />
  <span ng-repeat="number in [1,2,3,4,5]">{{number}} {{person.name}}</span>
</div>
```

ng-repeat。 - angular。

\$digest\$digest\$digest。

- - div\$digest <sup>5</sup> \$digest\$digest - 。

---

```
function $scope(){
    this.$children = [];
    this.$watches = [];
}
```

```
$scope.prototype.$digest = function(){
  this.$watches.forEach(function($w){
    var val = $w.val();
    if($w.prevVal !== val){
      $w.callback(val, $w.prevVal);
      $w.prevVal = val;
    }
  });
  this.$children.forEach(function(c){
    c.$digest();
  });
}

$scope.prototype.$watch = function(val, callback){
  this.$watches.push({val:val, callback:callback, prevVal: val() })
}
```

<https://riptutorial.com/zh-CN/angularjs/topic/3156/>

# 44:

=	◦ ◦
<	◦
@	◦
	◦
-	-
<b>LifeCycle Hooks</b>	angular.version >= 1.5.3
<b>\$OnInit</b>	◦ ◦
<b>\$ onChangeschangesObj</b>	◦ changesObj{ currentValue, previousValue, isFirstChange() }◦
<b>\$onDestroy</b>	◦ ◦
<b>\$ postLink</b>	◦ Angular 2ngAfterViewInitngAfterContentInit◦
<b>\$ doCheck</b>	◦ ◦ ;\$ onChanges◦

Component◦ Angular 1.5AngularJS◦

<https://docs.angularjs.org/guide/component>

## Examples

### LifeCycle

- Angular 2◦ WebHTML◦

```
angular.module('myApp', [])  
  .component('helloWorld', {  
    template: '<span>Hello World!</span>'  
  });
```

```
<div ng-app="myApp">  
  <hello-world> </hello-world>  
</div>
```

```
angular.module("myApp", [])
  .component("helloWorld", {
    template: '<span>Hello {{$ctrl.name}}!</span>',
    bindings: { name: '@' }
  });
```

```
<div ng-app="myApp">
  <hello-world name="'John'" > </hello-world>
</div>
```

◦

```
angular.module("myApp", [])
  .component("helloWorld", {
    template: "Hello {{$ctrl.name}}, I'm {{$ctrl.myName}}!",
    bindings: { name: '@' },
    controller: function() {
      this.myName = 'Alain';
    }
  });
```

```
<div ng-app="myApp">
  <hello-world name="John"> </hello-world>
</div>
```

## CodePen

### Angular\$onInit◦

```
angular.module("myApp", [])
  .component("helloWorld", {
    template: "Hello {{$ctrl.name}}, I'm {{$ctrl.myName}}!",
    bindings: { name: '@' },
    controller: function() {
      this.$onInit = function() {
        this.myName = "Mac" + this.name;
      }
    }
  });
```

“Hello JohnMacJohn”。

\$ctrlcontrollerAsAngular◦

## “require”

◦

## require

### \$onInit

```
angular.module("myApp", [])
  .component("helloWorld", {
    template: "Hello {{$ctrl.name}}, I'm {{$ctrl.myName}}!",
    bindings: { name: '@' },
    require: {
      parent: '^parentComponent'
    },
    controller: function () {
      // here this.parent might not be initiated yet

      this.$onInit = function() {
        // after $onInit, use this.parent to access required controller
        this.parent.foo();
      }
    }
  });
```

◦

## JS

angularJSHTML<html><ANYTHING>◦◦◦ angular◦◦

```
angular.module("myApp", []).component("customer", {})
```

◦◦

```
angular.module("myApp", []).component("customer", {
  templateUrl : "customer.html", // your view here
  controller: customerController, //your controller here
  controllerAs: "cust"           //alternate name for your controller
})
```

“myApp”customer◦ html

```
<customer></customer>
```

◦

◦

<https://riptutorial.com/zh-CN/angularjs/topic/892/>

# 45:

## AngularJS Directives

	◦ false true { @ =< } ◦
falsy	◦ ◦
	◦ ◦
{ @ }	DOM ◦ ◦
{ = }	◦
{ < }	DOM ◦ ◦ ◦
{ }	◦
	DOM ◦ tElement tAttr ◦ ◦ post-link pre-link post-link ◦
/	link ◦ scope iElement DOM iAttrs DOM controller transclude Fn ◦ DOM DOM ◦ ◦
	◦ ◦ ◦
	◦ ◦
restrict	DOM ◦ demoDirective E - <demo-directive></demo-directive> A - <div demo-directive></div> C - <div class="demo-directive"></div> M - <!-- directive: demo-directive --> ◦ restrict - restrict: "AC" <b>Attribute OR Class</b> ◦ "EA" ◦
'demoDirective'	demoDirective ◦ ◦
'demoDirective'	demoDirective null fn ◦
'^ demoDirective'	demoDirective ◦ ◦
'^^ demoDirective'	demoDirective ◦ ◦
'^ demoDirective'	demoDirective null fn ◦
'^^	demoDirective null fn ◦

demoDirective'
----------------

## Examples

angularjs。 angularjshtmlhtmlhtml。

### directive.js

```
// Create the App module if you haven't created it yet
var demoApp= angular.module("demoApp", []);

// If you already have the app module created, comment the above line and create a reference
of the app module
var demoApp = angular.module("demoApp");

// Create a directive using the below syntax
// Directives are used to extend the capabilities of html element
// You can either create it as an Element/Attribute/class
// We are creating a directive named demoDirective. Notice it is in CamelCase when we are
defining the directive just like ngModel
// This directive will be activated as soon as any this element is encountered in html

demoApp.directive('demoDirective', function () {

    // This returns a directive definition object
    // A directive definition object is a simple JavaScript object used for configuring the
directive's behaviour,template..etc
    return {
        // restrict: 'AE', signifies that directive is Element/Attribute directive,
        // "E" is for element, "A" is for attribute, "C" is for class, and "M" is for comment.
        // Attributes are going to be the main ones as far as adding behaviors that get used the
most.
        // If you don't specify the restrict property it will default to "A"
        restrict : 'AE',

        // The values of scope property decides how the actual scope is created and used inside a
directive. These values can be either "false", "true" or "{}". This creates an isolate scope
for the directive.
        // '@' binding is for passing strings. These strings support {{{}} expressions for
interpolated values.
        // '=' binding is for two-way model binding. The model in parent scope is linked to the
model in the directive's isolated scope.
        // '&' binding is for passing a method into your directive's scope so that it can be
called within your directive.
        // The method is pre-bound to the directive's parent scope, and supports arguments.
        scope: {
            name: "@", // Always use small casing here even if it's a mix of 2-3 words
        },

        // template replaces the complete element with its text.
        template: "<div>Hello {{{name}}}</div>",

        // compile is called during application initialization. AngularJS calls it once when html
page is loaded.
        compile: function(element, attributes) {
            element.css("border", "1px solid #cccccc");
        }
    };
});
```



```

// linkFunction is linked with each element with scope to get the element specific data.
var linkFunction = function($scope, element, attributes) {
    element.html("Name: <b>"+$scope.name + "</b>");
    element.css("background-color", "#ff00ff");
};
return linkFunction;
}
};
});

```

## App

```

<html>

  <head>
    <title>Angular JS Directives</title>
  </head>
  <body>
    <script src =
"http://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/angular.min.js"></script>
    <script src="directive.js"></script>
    <div ng-app = "demoApp">
      <!-- Notice we are using Spinal Casing here -->
      <demo-directive name="World"></demo-directive>

    </div>
  </body>
</html>

```

```

demoApp.directive('demoDirective', function () {
  var directiveDefinitionObject = {
    multiElement:
    priority:
    terminal:
    scope: {},
    bindToController: {},
    controller:
    controllerAs:
    require:
    restrict:
    templateNamespace:
    template:
    templateUrl:
    transclude:
    compile:
    link: function(){}
  };
  return directiveDefinitionObject;
});

```

1. **multiElement** - trueDOM
2. **priority** - DOM◦ ◦
3. **terminal** - true
4. **scope** -
5. **bind to controller** -

6. `controller` -
7. `require` -
8. `controllerAs` - `controllerAs`◦
9. `restrict` - `ElementAttributeClassComment`
10. `templateNamespace` - `htmlsvgmath`◦ `html`
11. `template` - `htmltranscludetrue`
12. `templateUrl` - `url`
13. `transclude` - ◦ ◦
14. `compile` - `DOM`
15. `link` - `compile`◦ `linkDOMDOM`◦ ◦

## directive.js

```
angular.module('myApp', [])
  .directive('superman', function() {
    return {
      // restricts how the directive can be used
      restrict: 'E',
      templateUrl: 'superman-template.html',
      controller: function() {
        this.message = "I'm superman!"
      },
      controllerAs: 'supermanCtrl',
      // Executed after Angular's initialization. Use commonly
      // for adding event handlers and DOM manipulation
      link: function(scope, element, attributes) {
        element.on('click', function() {
          alert('I am superman!')
        });
      }
    }
  });
```

## template.html

```
<h2>{{supermanCtrl.message}}</h2>
```

## index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Document</title>
  <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.5.0/angular.js"></script>
  <script src="superman-directive.js"></script>
</head>
<body>
<div ng-app="myApp">
  <superman></superman>
</div>
</body>
</html>
```

AngularJS AngularJS HTML。 HTML。 DOM HTML。 ng-model ng-hide ng-if。

。 Custom。 /angularjs/angular.js。 /。 。

app reusableComponents reusableModuleApp.js

## reusableModuleApp.js

```
(function(){  
  
    var reusableModuleApp = angular.module('reusableModuleApp', ['ngSanitize']);  
  
    //Remember whatever dependencies you have in here should be injected in the app module where  
    it is intended to be used or it's scripts should be included in your main app  
        //We will be injecting ng-sanitize  
  
    reusableModuleApp.directive('toolbar', toolbar)  
  
    toolbar.$inject=['$sce'];  
  
    function toolbar($sce){  
  
        return{  
            restrict : 'AE',  
            //Defining below isolate scope actually provides window for the directive to take data  
            from app that will be using this.  
            scope : {  
                value1: '=',  
                value2: '=',  
            },  
  
            }  
  
            template : '<ul> <li><a ng-click="Add()" href="">{{value1}}</a></li> <li><a ng-  
            click="Edit()" href="#">{{value2}}</a></li> </ul> ',  
            link : function(scope, element, attrs){  
  
                //Handle's Add function  
                scope.Add = function(){  
  
                };  
  
                //Handle's Edit function  
                scope.Edit = function(){  
  
                };  
  
            }  
        }  
    };  
  
});
```

## mainApp.js

```
(function(){  
    var mainApp = angular.module('mainApp', ['reusableModuleApp']); //Inject reusableModuleApp  
    in your application where you want to use toolbar component
```

```

mainApp.controller('mainAppController', function($scope){
    $scope.value1 = "Add";
    $scope.value2 = "Edit";

});

});

```

## index.html

```

<!doctype html>
<html ng-app="mainApp">
<head>
  <title> Demo Making a reusable component
</head>
<body ng-controller="mainAppController">

  <!-- We are providing data to toolbar directive using mainApp'controller -->
  <toolbar value1="value1" value2="value2"></toolbar>

  <!-- We need to add the dependent js files on both apps here -->
  <script src="js/angular.js"></script>
  <script src="js/angular-sanitize.js"></script>

  <!-- your mainApp.js should be added afterwards --->
  <script src="mainApp.js"></script>

  <!-- Add your reusable component js files here -->
  <script src="resuableComponents/reusableModuleApp.js"></script>

</body>
</html>

```

- angularJs ◦ reusableModuleApp.js reusableModuleAppDDO ◦

- 

scope ◦

- scope

- 

- Angular

```

var ProgressBar = function() {
  return {
    scope: { // This is how we define an isolated scope
      current: '=', // Create a REQUIRED bidirectional binding by using the 'current'
attribute
      full: '=?maxValue' // Create an OPTIONAL (Note the '?'): bidirectional binding using
'max-value' attribute to the `full` property in our directive isolated scope
    }
    template: '<div class="progress-back">' +
      ' <div class="progress-bar"' +
      ' ng-style="{width: getProgress()}">' +

```

```

        ' </div>' +
        '</div>',
    link: function(scope, el, attrs) {
        if (scope.full === undefined) {
            scope.full = 100;
        }
        scope.getProgress = function() {
            return (scope.current / scope.size * 100) + '%';
        }
    }
}
}
}

ProgressBar.$inject = [];
angular.module('app').directive('progressBar', ProgressBar);

```

```

angular.module('app').controller('myCtrl', function($scope) {
    $scope.currentProgressValue = 39;
    $scope.maxProgressBarValue = 50;
});

```

```

<div ng-controller="myCtrl">
    <progress-bar current="currentProgressValue"></progress-bar>
    <progress-bar current="currentProgressValue" max-value="maxProgressBarValue"></progress-
bar>
</div>

```

## ◦ “Box”

### userBox.js

```

angular.module('simpleDirective', []).directive('userBox', function() {
    return {
        scope: {
            username: '=username',
            reputation: '=reputation'
        },
        templateUrl: '/path/to/app/directives/user-box.html'
    };
});

```

### Controller.js

```

var myApp = angular.module('myApp', ['simpleDirective']);

myApp.controller('Controller', function($scope) {

    $scope.user = "John Doe";
    $scope.rep = 1250;

    $scope.user2 = "Andrew";
    $scope.rep2 = 2850;

});

```

## myPage.js

```
<html lang="en" ng-app="myApp">
  <head>
    <script src="/path/to/app/angular.min.js"></script>
    <script src="/path/to/app/js/controllers/Controller.js"></script>
    <script src="/path/to/app/js/directives/userBox.js"></script>
  </head>

  <body>

    <div ng-controller="Controller">
      <user-box username="user" reputation="rep"></user-box>
      <user-box username="user2" reputation="rep2"></user-box>
    </div>

  </body>
</html>
```

## box.html

```
<div>{{username}}</div>
<div>{{reputation}} reputation</div>
```

```
John Doe
1250 reputation
Andrew
2850 reputation
```

◦ ◦

## \$ inject◦

.config◦ myDirective◦ myDirective◦ Directive◦ []◦

## templateUrl

```
angular.module('myApp').config(function($provide) {
  $provide.decorator('myDirectiveDirective', function($delegate) {
    var directive = $delegate[0]; // this is the actual delegated, your directive
    directive.templateUrl = 'newTemplate.html'; // you change the directive template
    return $delegate;
  })
});
```

## onClick◦

```
angular.module('myApp').config(function ($provide) {
  $provide.decorator('myDirectiveTwoDirective', function ($delegate) {
    var directive = $delegate[0];
    var link = directive.link; // this is directive link phase
    directive.compile = function () { // change the compile of that directive
      return function (scope, element, attrs) {
        link.apply(this, arguments); // apply this at the link phase
      }
    }
  })
});
```

```

        element.on('click', function(){ // when add an onclick that log hello when
the directive is clicked.
            console.log('hello!');
        });
    };
};
return $delegate;
});
});

```

◦

## Angular js◦

### AdirBdir\$ scopeBdirAdir◦

```

angular.module('myApp', []).directive('Adir', function () {
    return {
        restrict: 'AE',
        controller: ['$scope', function ($scope) {
            $scope.logFn = function (val) {
                console.log(val);
            }
        }
    ]
    }
})

```

### require'^ Adir'^◦

```

.directive('Bdir', function () {
    return {
        restrict: 'AE',
        require: '^Adir', // Bdir require Adir
        link: function (scope, elem, attr, Parent) {
            // Parent is Adir but can be an array of required directives.
            elem.on('click', function ($event) {
                Parent.logFn("Hello!"); // will log "Hello! at parent dir scope
                scope.$apply(); // apply to parent scope.
            });
        }
    }
});

```

```

<div a-dir><span b-dir></span></div>
<a-dir><b-dir></b-dir> </a-dir>

```

## HTML◦

<https://riptutorial.com/zh-CN/angularjs/topic/965/>

# 46:

## Examples

Angular。

jQuery。 Angular。

**ng-modelnovalidate。**

```
<form name="form" novalidate>
  <label name="email"> Your email </label>
  <input type="email" name="email" ng-model="email" />
</form>
```

Angular**ng-model。** 。

```
<input type="number" name="postalcode" ng-model="zipcode" />
```

**ng-submit。** 。

```
<form name="signup_form" ng-submit="submitFunc()" novalidate>
  <label name="email"> Your email </label>
  <input type="email" name="email" ng-model="email" />
  <button type="submit">Signup</button>
</form>
```

Angular。

Angular**ng-pristine ng-dirty ng-validng-invalid。** *css/pristine / dirty。*

\$touched	
\$untouched	
\$pristine	
\$dirty	
\$valid	
\$invalid	

truefalse。

。

```
<form name="myForm" novalidate>
```



```
<input name="myName" ng-model="myName" required>
  <span ng-show="myForm.myName.$touched && myForm.myName.$invalid">This name is
invalid</span>
</form>
```

ng-show°

## CSS

### AngularCSS

ng-touched	
ng-untouched	
ng-pristine	
ng-dirty	
ng-valid	
ng-invalid	

```
input.ng-invalid {
  background-color: crimson;
}
input.ng-valid {
  background-color: green;
}
```

## ngMessages

ngMessages°

ngMessagesAngular ng-class° repetitive°

ngMessages°

## Html

```
<form name="ngMessagesDemo">
  <input name="firstname" type="text" ng-model="firstname" required>
  <div ng-messages="ngMessagesDemo.firstname.$error">
    <div ng-message="required">Firstname is required.</div>
  </div>
</form>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.3.16/angular.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.3.16/angular-
```

```
messages.min.js"></script>
```

## JS

```
var app = angular.module('app', ['ngMessages']);

app.controller('mainCtrl', function ($scope) {
  $scope.firstname = "Rohit";
});
```

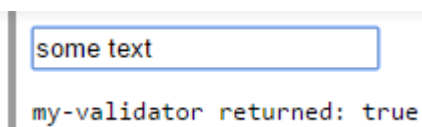
### ◦ Angular ngModelController\$validators

```
angular.module('app', [])
  .directive('myValidator', function() {
    return {
      // element must have ng-model attribute
      // or $validators does not work
      require: 'ngModel',
      link: function(scope, elm, attrs, ctrl) {
        ctrl.$validators.myValidator = function(modelValue, viewValue) {
          // validate viewValue with your custom logic
          var valid = (viewValue && viewValue.length > 0) || false;
          return valid;
        };
      }
    };
  });
```

### ngModel◦

```
<form name="form">
  <input type="text"
    ng-model="model"
    name="model"
    my-validator>
  <pre ng-bind="'my-validator returned: ' + form.model.$valid"></pre>
</form>
```

### my-validator◦ ng-model◦ ui◦



some text

my-validator returned: true

### ◦ HTML5◦ Angular ng-form◦

```
<form name="myForm" noValidate>
  <!-- nested form can be referenced via 'myForm.myNestedForm' -->
  <ng-form name="myNestedForm" noValidate>
    <input name="myInput1" ng-minlength="1" ng-model="input1" required />
    <input name="myInput2" ng-minlength="1" ng-model="input2" required />
  </ng-form>

  <!-- show errors for the nested subform here -->
```

```

<div ng-messages="myForm.myNestedForm.$error">
  <!-- note that this will show if either input does not meet the minimum -->
  <div ng-message="minlength">Length is not at least 1</div>
</div>
</form>

<!-- status of the form -->
<p>Has any field on my form been edited? {{myForm.$dirty}}</p>
<p>Is my nested form valid? {{myForm.myNestedForm.$valid}}</p>
<p>Is myInput1 valid? {{myForm.myNestedForm.myInput1.$valid}}</p>

```

◦ myInput1\$dirty \$dirty ◦ myNestedFormmyForm\$dirty ◦

**\$ http**◦

**users**◦

inputng-model\$asyncValidators◦

◦ ◦

```

ngModel.$asyncValidators.usernameValidate = function (name) {
  if (name) {
    return AuthenticationService.checkIfNameExists(name); // returns a promise
  } else {
    return $q.reject("This username is already taken!"); // rejected promise
  }
};

```

ng-model**promise**◦

<https://riptutorial.com/zh-CN/angularjs/topic/3979/>

# 47:

- ;

**Decorator** ◦ Decorator◦ ◦

---

```
config$provide$provide.decorator◦
```

```
    $delegate◦ $delegate◦ ◦
```

---

```
◦ /◦
```

```
◦
```

## Examples

```
null◦
```

```
angular.module('app', [])
  .config(function($provide) {
    $provide.decorator('myService', function($delegate) {
      $delegate.getDate = function() { // override with actual date object
        return new Date();
      };
      return $delegate;
    });
  })
  .service('myService', function() {
    this.getDate = function() {
      return null; // w/o decoration we'll be returning null
    };
  })
  .controller('myController', function(myService) {
    var vm = this;
    vm.date = myService.getDate();
  });
```

```
<body ng-controller="myController as vm">
  <div ng-bind="vm.date | date:'fullDate'"></div>
</body>
```

---

**Saturday, August 6, 2016**

◦ **\$ delegate0decoratornameDirective◦**

```
myDate $delegate[0]myDateDirective◦
```

◦ ◦ ◦

```
<body>
  <my-date></my-date>
</body>
```

```
angular.module('app', [])
  .config(function($provide) {
    $provide.decorator('myDateDirective', function($delegate, $interval) {
      var directive = $delegate[0]; // access directive

      directive.compile = function() { // modify compile fn
        return function(scope) {
          directive.link.apply(this, arguments);
          $interval(function() {
            scope.date = new Date(); // update date every second
          }, 1000);
        };
      };

      return $delegate;
    });
  })
  .directive('myDate', function() {
    return {
      restrict: 'E',
      template: '<span>Current time is {{ date | date:\'MM:ss\' }}</span>',
      link: function(scope) {
        scope.date = new Date(); // get current date
      }
    };
  });
```

**Current time is 08:33**

nameFilter ◦ filter ◦ repeat ◦ decorator ◦ repeatFilter ◦ n ◦ angular ◦

```
<body>
  <div ng-bind="'i can haz cheeseburger ' | repeat:2"></div>
</body>

angular.module('app', [])
  .config(function($provide) {
    $provide.decorator('repeatFilter', function($delegate) {
      return function reverse(input, count) {
        // reverse repeated string
        return ($delegate(input, count)).split('').reverse().join('');
      };
    });
  })
  .filter('repeat', function() {
    return function(input, count) {
      // repeat string n times
      return (input || '').repeat(count || 1);
    };
  });
```

i can haz cheeseburger i can haz cheeseburger

regrubeseehc zah nac i regrubeseehc zah nac i

<https://riptutorial.com/zh-CN/angularjs/topic/5255/>

---

# 48: MVC

AngularJS MVC JavaScript HTML。 HTML JavaScript。 AngularJS。

## Examples

---

### mvc

```
var indexController = myApp.controller("indexController", function ($scope) {  
    // Application logic goes here  
});
```

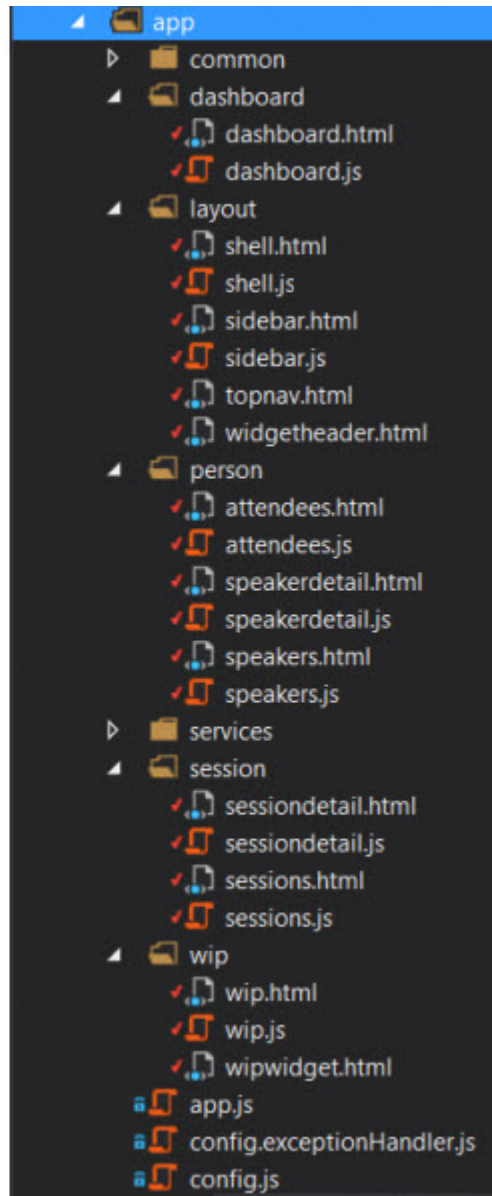
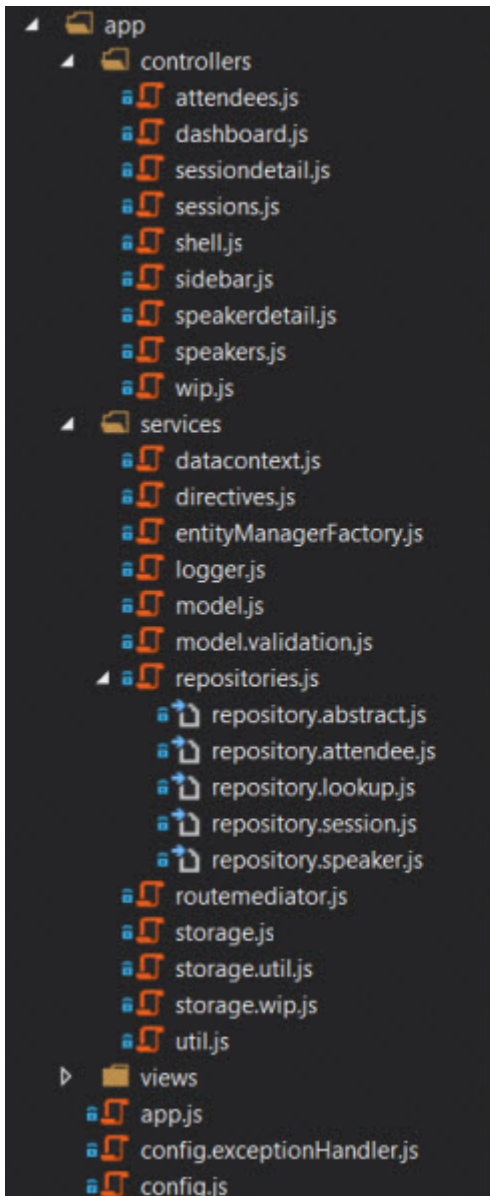
```
var indexController = myApp.controller("indexController", function ($scope) {  
    // controller logic goes here  
    $scope.message = "Hello Hacking World"  
});
```

MVC <https://riptutorial.com/zh-CN/angularjs/topic/8667/mvc>

# 49: -

## Examples

Angular - ""。 。 Sort By Type Sort By Feature 。 。



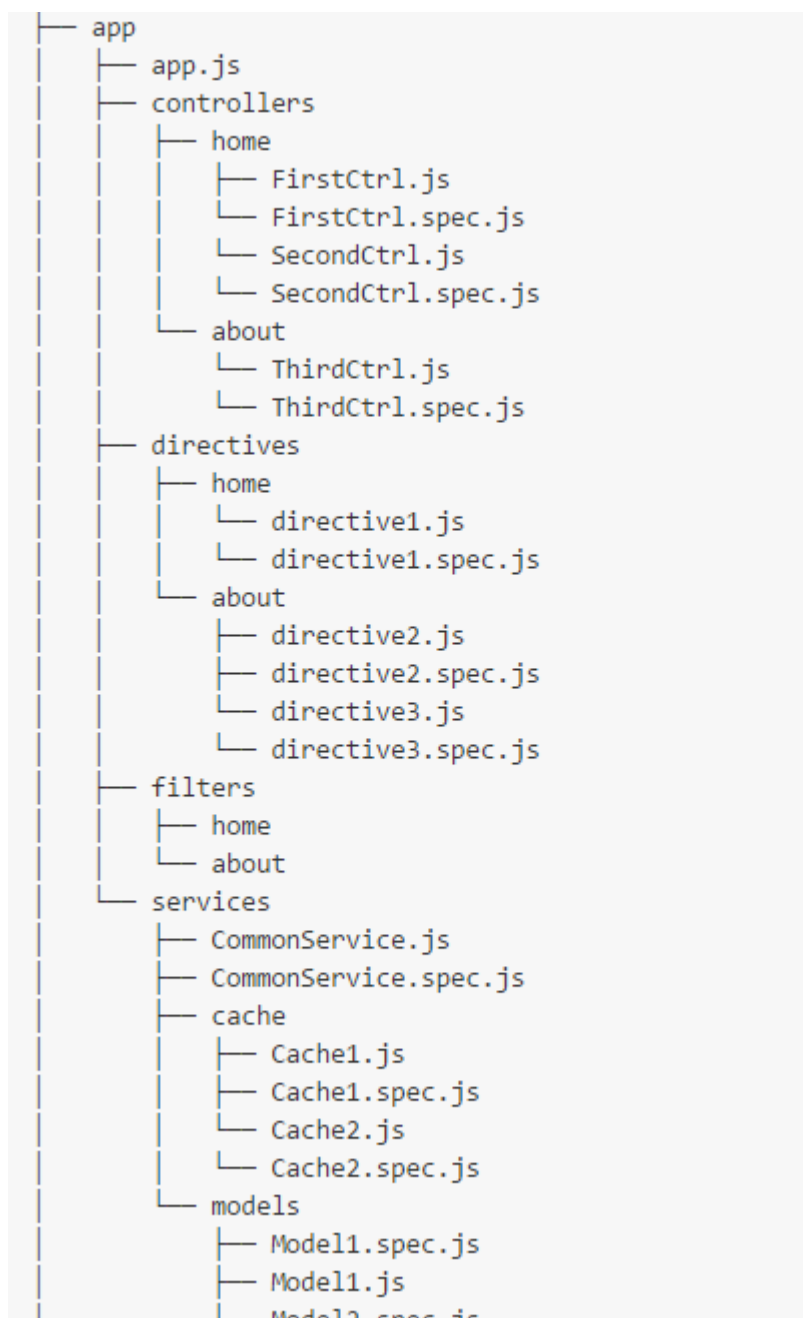
- - Angular。
- - 。 。

admin。

- - 。
-



- ""。



- <https://riptutorial.com/zh-CN/angularjs/topic/6148/--->

# 50:

## Examples

```
<div ng-app="demoApp" ng-controller="mainController as ctrl">
  {{$id}}
  <ul>
    <li ng-repeat="item in ctrl.items">
      {{$id}}<br/>
      {{item.text}}
    </li>
  </ul>
  {{$id}}
  <pre>
    {{ctrl.items | json : 2}}
  </pre>
</div>
```

```
angular.module('demoApp', [])
.controller('mainController', MainController);
```

```
function MainController() {
  var vm = this;
  vm.items = [{
    id: 0,
    text: 'first'
  },
  {
    id: 1,
    text: 'second'
  },
  {
    id: 2,
    text: 'third'
  }
  ]};
}
```

◦ `$scope.$id` \$ scope◦

`ul-tag` \$ id = 2 `ng-repeat`◦

◦ `json` ◦ \n ◦

## ng-inspect chrome

[ng-inspect](#) Chrome AngularJS◦

`ng-inspect`◦

```

Elements Console Sources Network Performance Memory Application Security
▶ <nav class="wm-studio-header navbar navbar-default" headerlabel="my-
projects">...</nav>
  <!-- ngInclude: -->
▶ <ng-include src="//dh2dw20653ig1.cloudfront.net/studio/8.4.1.1.2786/
editor/templates/studioactions.html" data-ng-hide="hidenews" class="ng-
scope">...</ng-include>
▼ <div class="wm-studio-content ng-scope">
  ▼ <div class="wm-projects" data-ng-class="{ 'full-width':
(!showUpdatesPanel && !showProjectInfoPanel)}">
    <!-- ngIf: RELEASE_MESSAGE -->
    ▶ <ul class="nav nav-pills wm-projects-navigation">...</ul>
    ▼ <section role="main" class="projects-list-container">
      ...
      ▶ <div ng-class="[size, spinnerclass]" init-widget title apply-styles
no-animate class="app-spinner ng-isolate-scope ng-hide" name=
"projects-list-spinner" show="false">...</div> == $θ
      ▶ <ul class="projects-list">...</ul>
    </section>
    </div>
    ▶ <aside class="wm-updates" role="complementary">...</aside>
      ::after
    </div>
    <!-- ngInclude:
"//dh2dw20653ig1.cloudfront.net/studio/8.4.1.1.2786/editor/templates/foot
er.html" -->
    ▶ <div data-ng-include="//dh2dw20653ig1.cloudfront.net/studio/
8.4.1.1.2786/editor/templates/footer.html" class="ng-scope">...</div>
      ::after
    </div>
    <script src="//dh2dw20653ig1.cloudfront.net/studio/8.4.1.1.2786/editor/
generated/scripts/projects-listing-page-libs.min.js"></script>
    <script src="//dh2dw20653ig1.cloudfront.net/studio/8.4.1.1.2786/editor/
generated/scripts/projects-listing-page.min.js"></script>
    ▶ <div id="toast-container" ng-class="[config.position, config.animation]"
toaster-options="{ 'limit': 1, 'time-out': 2000, 'extended-time-out': 0,
'position-class': 'toast-bottom-right' 'close-button': true}" class="ng-

```

scope/isolateScope ◦

```

$s      -- scope of the selected node
$sis    -- isolateScope of the selected node
$el     -- jQuery element reference of the selected node (requires jQuery)
$events -- events present on the selected node (requires jQuery)

```

Elements Console Sources Network Performance Memory Application Security

```

scope">...</ng-include>
  <div class="wm-studio-content ng-scope">
    <div class="wm-projects" data-ng-class="{ 'full-width':
      (!showUpdatesPanel && !showProjectInfoPanel)}">
      <!-- ngIf: RELEASE_MESSAGE -->
      <ul class="nav nav-pills wm-projects-navigation">...</ul>
      <section role="main" class="projects-list-container">
        <div ng-class="[size, spinnerclass]" init-widiget title apply-styles
          no-animate class="app-spinner ng-isolate-scope ng-hide" name=
            "projects-list-spinner" show="false">...</div> == $0
        <ul class="projects-list">...</ul>
      </section>
    </div>
  <aside class="wm-updates" role="complementary">...</aside>
html #ng-app div div div section div.app-spinner.ng-isolate-scope.ng-hide

```

Changes Console What's New Request blocking Search Coverage Quick source

top Filter Info

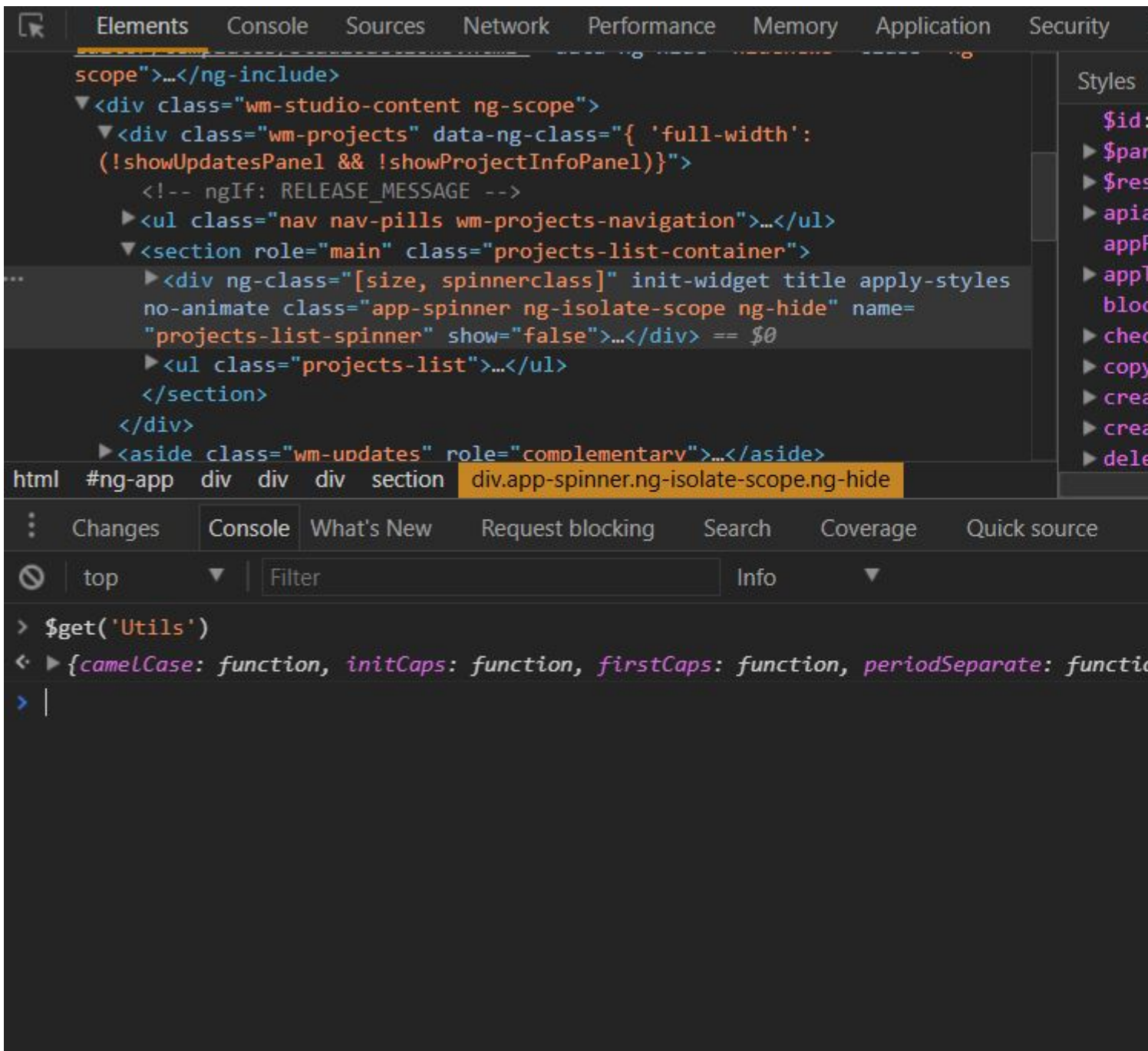
```

> $el
< ▶ [div.app-spinner.ng-isolate-scope.ng-hide, context: div.app-spinner.ng-isolate-scope.
> $events
< undefined
> $s
< ▶ o {$$childTail: o, $$childHead: o, $$nextSibling: null, $$watchers: Array(10), $$list
> $is
< ▶ o {$id: 11, $$childTail: b, $$childHead: b, $$prevSibling: o, $$nextSibling: b...}
> |

```

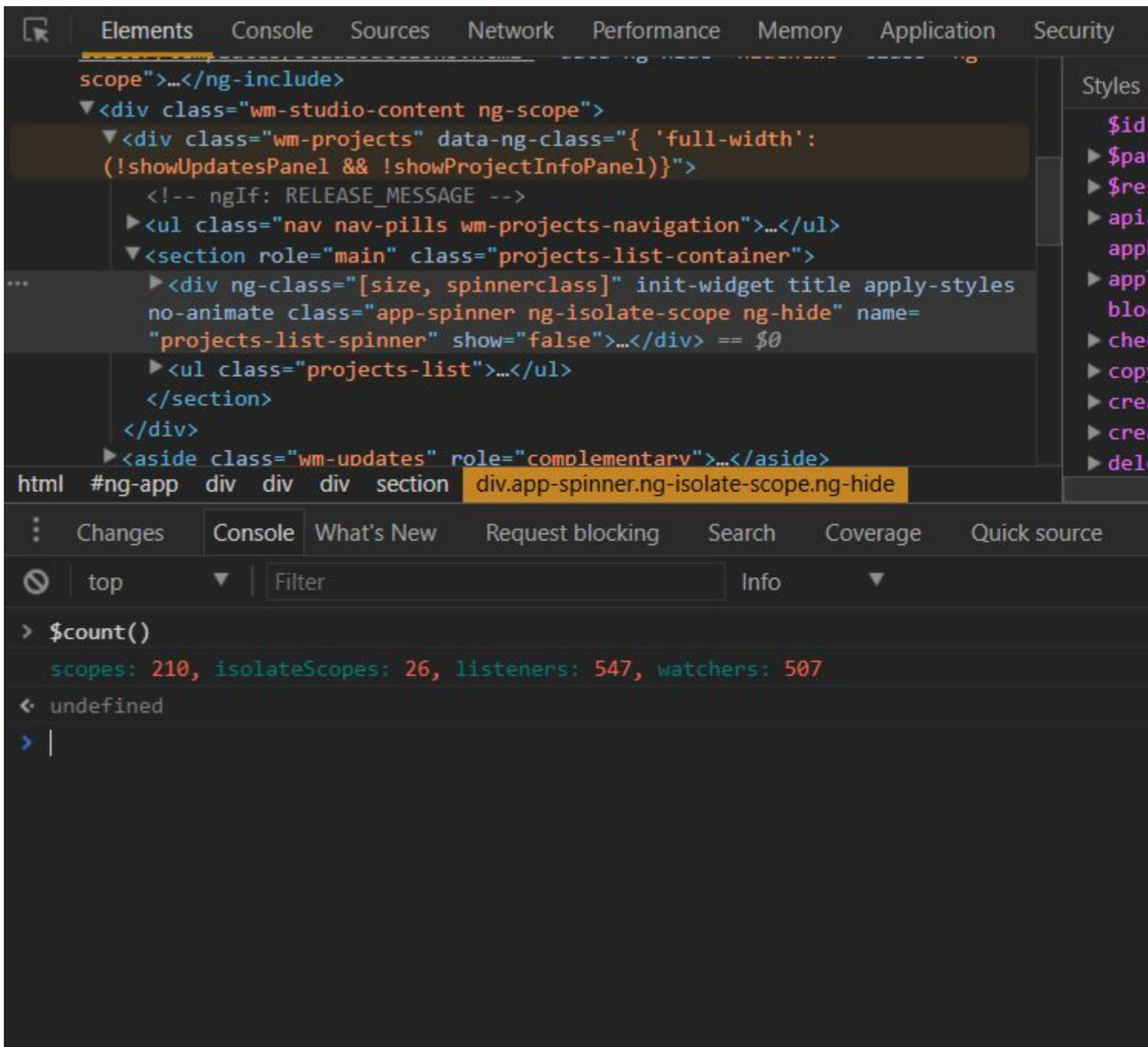
/o

\$get ()/o



no.ofisolateScope。

\$count () isolateScope。



`debugInfo`.

`ng-inspect`

◦

```

angular.element(myDomElement).scope();
e.g.
angular.element(document.getElementById('yourElementId')).scope() //accessing by ID

```

-

```

angular.element('[ng-controller=ctrl]').scope()

```

DOMj“elements”\$ 0。

```
angular.element($0).scope();
```

<https://riptutorial.com/zh-CN/angularjs/topic/4761/>

# 51: Angular 2+

AngularJS TypeScript Angular。

AngularJS。 “Angular。

## Examples

### AngularJS

Angular。 AngularJS Angular。

1.5+ AngularJS。 AngularJS Angular 2+。

[AngularJS](#)。

```
ng-controller ng-controller component
```

---

。 。

UserListController UserListComponent。

### HTML

```
<div ng-controller="UserListController as listctrl">
  <ul>
    <li ng-repeat="user in myUserList">
      {{ user }}
    </li>
  </ul>
</div>
```

### JavaScript

```
app.controller("UserListController", function($scope, SomeService) {

  $scope.myUserList = ['Shin', 'Helias', 'Kalhac'];

  this.someFunction = function() {
    // ...
  }

  // ...
})
```

### HTML



```
<user-list></user-list>
```

## JavaScript

```
app.component("UserList", {
  templateUrl: 'user-list.html',
  controller: UserListController
});

function UserListController(SomeService) {

  this.myUserList = ['Shin', 'Helias', 'Kalhac'];

  this.someFunction = function() {
    // ...
  }

  // ...
}
```

```
$scope.this.myUserList$scope.myUserList ;
```

user-list.component.html

```
<ul>
  <li ng-repeat="user in $ctrl.myUserList">
    {{ user }}
  </li>
</ul>
```

myUserList \$ctrl.myUserList**HTML**\$scope.myUserList ◦

\$ctrl◦

---

ng-controller

```
$stateProvider
  .state('users', {
    url: '/users',
    templateUrl: 'user-list.html',
    controller: 'UserListController'
  })
// ..
```

```
$stateProvider
  .state('users', {
    url: '/',
    template: '<user-list></user-list>'
  })
// ..
```

---

- - 
  - `$onInit () $onChanges () ...`

◦

◦



AngularJSAngular◦

Angular 2+◦

**WebpackES6**

[WebpackES6 TypeScript](#) ◦ ◦

◦

[Angular 2+](https://riptutorial.com/zh-CN/angularjs/topic/9942/angular-2plus) <https://riptutorial.com/zh-CN/angularjs/topic/9942/angular-2plus>

# 52:

## Examples

ng-repeat◦ app.filter()app.filter()◦ ◦

.toUpperCase() javascript

```
var app = angular.module("MyApp", []);

// just like making a controller, you must give the
// filter a unique name, in this case "toUppercase"
app.filter('toUppercase', function(){
  // all the filter does is return a function,
  // which acts as the "filtering" function
  return function(rawString){
    // The filter function takes in the value,
    // which we modify in some way, then return
    // back.
    return rawString.toUpperCase();
  };
});
```

◦

“toUppercase”; app.filter(...)◦ ◦ ◦

◦ ◦

◦ javascriptAngular◦

## Javascript

angular \$filter◦

```
app.controller("MyController", function($scope, $filter){
  this.rawString = "Foo";
  this.capsString = $filter("toUppercase")(this.rawString);
});
```

## HTML

angular |◦ MyControllerrawString◦

```
<div ng-controller="MyController as ctrl">
  <span>Capital rawString: {{ ctrl.rawString | toUppercase }}</span>
</div>
```

## Angular<sup>™</sup>“toUppercase”。

◦ ◦

```
function removeNulls() {
  return function(list) {
    for (var i = list.length - 1; i >= 0; i--) {
      if (typeof list[i] === 'undefined' ||
          list[i] === null) {
        list.splice(i, 1);
      }
    }
    return list;
  };
}
```

## HTML

```
{{listOfItems | removeNulls}}
```

```
listOfItems = removeNullsFilter(listOfItems);
```

◦ ◦

```
function convertToBooleanValue() {
  return function(input) {
    if (typeof input !== 'undefined' &&
        input !== null &&
        (input === true || input === 1 || input === '1' || input
         .toString().toLowerCase() === 'true')) {
      return true;
    }
    return false;
  };
}
```

## HTML

```
{{isAvailable | convertToBooleanValue}}
```

```
var available = convertToBooleanValueFilter(isAvailable);
```

◦

```
(function() {
  "use strict";
  angular
    .module('app', [])
    .controller('mainCtrl', mainCtrl);

  function mainCtrl() {
    var vm = this;
```

```

vm.classifications = ["Saloons", "Sedans", "Commercial vehicle", "Sport car"];
vm.cars = [
  {
    "name": "car1",
    "classifications": [
      {
        "name": "Saloons"
      },
      {
        "name": "Sedans"
      }
    ]
  },
  {
    "name": "car2",
    "classifications": [
      {
        "name": "Saloons"
      },
      {
        "name": "Commercial vehicle"
      }
    ]
  },
  {
    "name": "car3",
    "classifications": [
      {
        "name": "Sport car"
      },
      {
        "name": "Sedans"
      }
    ]
  }
];
}
})();

```

```

<body ng-app="app" ng-controller="mainCtrl as main">
  Filter car by classification:
  <select ng-model="classificationName"
    ng-options="classification for classification in main.classifications"></select>
  <br>
  <ul>
    <li ng-repeat="car in main.cars |
      filter: { classifications: { name: classificationName } } track by $index"
      ng-bind-template="{{car.name}} - {{car.classifications | json}}">
    </li>
  </ul>
</body>

```

[DEMO](#) ◦

\$filter [Angular](#) ◦

```
angular.module("app")
```

```

.service("users", usersService)
.controller("UsersController", UsersController);

function usersService () {
  this.getAll = function () {
    return [{
      id: 1,
      username: "john"
    }, {
      id: 2,
      username: "will"
    }, {
      id: 3,
      username: "jack"
    }
  ];
};
}

function UsersController ($filter, users) {
  var orderByFilter = $filter("orderBy");

  this.users = orderByFilter(users.getAll(), "username");
  // Now the users are ordered by their usernames: jack, john, will

  this.users = orderByFilter(users.getAll(), "username", true);
  // Now the users are ordered by their usernames, in reverse order: will, john, jack
}

```

## ng-repeat

ng-repeat◦ ng-repeat◦ as [variablename]◦

```

<ul>
  <li ng-repeat="item in vm.listItems | filter:vm.myFilter as filtered">
    {{item.name}}
  </li>
</ul>
<span>Showing {{filtered.length}} of {{vm.listItems.length}}</span>

```

<https://riptutorial.com/zh-CN/angularjs/topic/1401/>

S. No		Contributors
1	AngularJS	<a href="#">Abhishek Pandey</a> , <a href="#">After Class</a> , <a href="#">Andrés Encarnación</a> , <a href="#">AnonymousNotReally</a> , <a href="#">badzilla</a> , <a href="#">Charlie H</a> , <a href="#">Chirag Bhatia - chirag64</a> , <a href="#">Community</a> , <a href="#">daniellmb</a> , <a href="#">David G.</a> , <a href="#">Devid Farinelli</a> , <a href="#">Eugene</a> , <a href="#">fracz</a> , <a href="#">Franck Dernoncourt</a> , <a href="#">Gabriel Pires</a> , <a href="#">Gourav Garg</a> , <a href="#">H. Pauwelyn</a> , <a href="#">Igor Raush</a> , <a href="#">jengeb</a> , <a href="#">Jeroen</a> , <a href="#">John F.</a> , <a href="#">Léo Martin</a> , <a href="#">Lotus91</a> , <a href="#">LucyMarieJ</a> , <a href="#">M. Junaid Salaat</a> , <a href="#">Maaz.Musa</a> , <a href="#">Matt</a> , <a href="#">Mikko Viitala</a> , <a href="#">Mistalis</a> , <a href="#">Nemanja Trifunovic</a> , <a href="#">Nhan</a> , <a href="#">Nico</a> , <a href="#">pathe.kiran</a> , <a href="#">Patrick</a> , <a href="#">Pushpendra</a> , <a href="#">Richard Hamilton</a> , <a href="#">Stepan Suvorov</a> , <a href="#">Stephen Leppik</a> , <a href="#">Sunil Lama</a> , <a href="#">superluminary</a> , <a href="#">Syed Priom</a> , <a href="#">timbo</a> , <a href="#">Ven</a> , <a href="#">vincentvanjoe</a> , <a href="#">Yasin Patel</a> , <a href="#">Ze Rubeus</a> , <a href="#">Артем Комаров</a>
2	\$ http	<a href="#">CENT1PEDE</a> , <a href="#">jaredsk</a> , <a href="#">Liron Ilayev</a>
3	Angular \$	<a href="#">Abhishek Maurya</a> , <a href="#">elliott-j</a> , <a href="#">Eugene</a> , <a href="#">jaredsk</a> , <a href="#">Liron Ilayev</a> , <a href="#">MoLow</a> , <a href="#">Prateek Gupta</a> , <a href="#">RamenChef</a> , <a href="#">ryansstack</a> , <a href="#">Tony</a>
4	angularjs	<a href="#">Paresh Maghodiya</a>
5	AngularJS`=@`	<a href="#">Alon Eitan</a> , <a href="#">Lucas L</a> , <a href="#">Makarov Sergey</a> , <a href="#">Nico</a> , <a href="#">zucker</a>
6	AngularJS	<a href="#">Alon Eitan</a> , <a href="#">Cosmin Ababei</a> , <a href="#">doctorsherlock</a> , <a href="#">Faruk Yazıcı</a> , <a href="#">ngLover</a> , <a href="#">Phil</a>
7	Angular\$ q	<a href="#">Alon Eitan</a> , <a href="#">caiocpricci2</a> , <a href="#">ganqqwerty</a> , <a href="#">georgeawg</a> , <a href="#">John F.</a> , <a href="#">Muli Yulzary</a> , <a href="#">Praveen Poonia</a> , <a href="#">Richard Hamilton</a> , <a href="#">Rohit Jindal</a> , <a href="#">svarog</a>
8	HTTP	<a href="#">G Akshay</a> , <a href="#">Istvan Reiter</a> , <a href="#">MeanMan</a> , <a href="#">Mistalis</a> , <a href="#">mnoronha</a>
9	ng-class	<a href="#">Dr. Cool</a>
10	NG-	<a href="#">Aayushi Jain</a> , <a href="#">Manikandan Velayutham</a> , <a href="#">Umesh Shende</a>
11	NG-	<a href="#">Divya Jain</a> , <a href="#">Jim</a> , <a href="#">Sender</a> , <a href="#">zucker</a>
12	NG-	<a href="#">Divya Jain</a> , <a href="#">Jim</a>
13	SignalRAngularJs	<a href="#">Maher</a>
14	UI	<a href="#">George Kagan</a> , <a href="#">H.T</a> , <a href="#">Michael P. Bazos</a> , <a href="#">Ryan Hamley</a> , <a href="#">sgarcia.dev</a>
15		<a href="#">Rohit Jindal</a>

16	ES6	Bouraoui KACEM
17	ngModelController	Nikos Paraskevopoulos
18	ngRoute	Alon Eitan, Alvaro Vazquez, camabeh, DotBot, sgarcia.dev, svarog
19		Gourav Garg
20		Mikko Viitala
21		Andrea, badzilla, Gavishiddappa Gadagi, George Kagan, MoLow, Omri Aharon
22		elliott-j, Grundy, Lex, Mikko Viitala, Mistalis, Nix, prit4fun, Rohit Jindal, sgarcia.dev, Sunil Lama
23		Adam Harrison, Alon Eitan, Aron, AWolf, Ayan, Bon Macalindong, CENT1PEDE, Devid Farinelli, DillonChanis, Divya Jain, Dr. Cool, Eric Siebeneich, George Kagan, Grinn, gustavohenke, IncrediApp, kelvinelove, Krupesh Kotecha, Liron Ilayev, m.e.conroy, Maciej Gurban, Mansouri, Mikko Viitala, Mistalis, Mitul, MoLow, Naga2Raja, ngLover, Nishant123, Piet, redunderthebed, Richard Hamilton, svarog, tanmay, theblindprophet, timbo, Tomislav Stankovic, vincentvanjoe, Vishal Singh
24		MoLow, Pranav C Balan, svarog
25	-	JanisP
26		Ajeet Lakhani, Alon Eitan, Andrew Piliser, Anfelipe, Anirudha, Ashwin Ramaswami, atul mishra, Braiam, bwoebi, chris, Dania, Daniel Molin, daniellmb, Deepak Bansal, Divya Jain, DotBot, Dr. Cool, Durgpal Singh, fracz, Gabriel Pires, George Kagan, Grundy, JanisP, Jared Hooper, jhampton, John Slegers, jusopi, M22an, Matthew Green, Mistalis, Mudassir Ali, Nhan, Psaniko, Richard Hamilton, RyanDawkins, sgarcia.dev, theblindprophet, user3632710, vincentvanjoe, Yasin Patel, Ze Rubeus
27		Deepak Bansal
28		daniellmb, elliott-j, fracz, Gabriel Pires, Nico, ronapelbaum
29		Mikko Viitala
30		doodhwala, Pat, Sylvain
31	AngularJSTypeScript	Parv Sharma, Rohit Jindal



32	ES6	Bouraoui KACEM
33		Sylvain
34		Muli Yulzary
35		Deepak Bansal
36		ziaulain
37		Adam Harrison, Aeolingamenfel, Alon Eitan, badzilla, Bon Macalindong, Braiam, chatuur, DerekMT12, Dr. Cool, Florian, George Kagan, Grundy, Jared Hooper, Liron Ilayev, M. Junaid Salaat, Mark Cidade, Matthew Green, Mike, Nad Flores, Praveen Poonia, RamenChef, Sébastien Deprez, sgarcia.dev, thegreenpizza, timbo, Und3rTow, WMios
38	SelfThis	It-Z, Jim
39		Lucas L, Sasank Sunkavalli, theblindprophet
40		Abdellah Alaoui, Alvaro Vazquez, AnonDCX, DotBot, elliot-j, Flash, Gavishiddappa Gadagi, Hubert Grzeskowiak, Lex, Nishant123
41		Alon Eitan, Ankit, badzilla, Bon Macalindong, Matthew Green, Nad Flores, ojus kulkarni, sgarcia.dev, thegreenpizza
42		CodeWarrior, Nguyen Tran, Rohit Jindal, RyanDawkins, sgarcia.dev, shaN, Shashank Vivek
43		Alon Eitan, chris, MoLow, prit4fun
44		Alon Eitan, Artem K., badzilla, BarakD, Hubert Grzeskowiak, John F., Juri, M. Junaid Salaat, Mansouri, Pankaj Parkar, Ravi Singh, sgarcia.dev, Syed Priom, Yogesh Mangaj
45		Alon Eitan, br3w5, casraf, Cody Stott, Daniel, Everettss, Filipe Amaral, Gaara, Gavishiddappa Gadagi, Jinw, jkris, mnoronha, Pushendra, Rahul Bhooteshwar, Sajal, sgarcia.dev, Stephan, theblindprophet, TheChetan, Yuri Blanc
46		Alon Eitan, fantarama, garyx, Mikko Viitala, Richard Hamilton, Rohit Jindal, shane, svarog, timbo
47		Mikko Viitala
48	MVC	Ashok choudhary, Gavishiddappa Gadagi, Jim
49	-	jitender, Liron Ilayev

50		<a href="#">Aman</a> , <a href="#">AWolf</a> , <a href="#">Vinay K</a>
51	Angular 2+	<a href="#">ShinDarth</a>
52		<a href="#">Aeolingamenfel</a> , <a href="#">developer033</a> , <a href="#">Ed Hinchliffe</a> , <a href="#">fracz</a> , <a href="#">gustavohenke</a> , <a href="#">Matthew Green</a> , <a href="#">Nico</a>