angular

监听滚动事件，滚动元素必须有高度，且可以滚动

指令里面不放表达式

# angular.element()

## 引入JQ

angular.element('#test').html()

## 不引入JQ

1、angular.element(document.querySelector(''));

2、angular.element(document).find('xx').addClass('yy')

3、var par = document.getElementsByTagName('p')[0];

angular.element(par).addClass('blue');

4、angular.element('<div id="test">1111</div>').html()

angular.bootstrap(document.body,[‘模块名’]🡪在body插入模块

# 控制器 controller

<html ng-app="demo">

<head>

<meta charset="UTF-8">

<title></title>

</head>

<body>

<div ng-controller="oaoafly">

</div>

<div ng-controller="winds">

</div>

</body>

<script>

angular.module("demo",[])

.controller("oaoafly",[function(){

//code 控制器的逻辑

}])

.controller("winds",[function(){

}]);

</script>

</html>

# 服务

$scope ,$document，$rootScope,$window,$event，$cookiestroe(需要注入ngCookie模块)

$event =>JQ 的 事件e参数



## $sce

$sce.trustAsHtml(str)要经过此方法过滤的html结构才能用到html里

## $http

$http({

method:"get",

url:"Oaoafly.php"

params:{}

})

.then(function(data){

//成功的回调

},function(err){

//失败的回调

})

## $filter

$filter("过滤器名字")("处理的数据","过滤器的参数")

## $location

url = http://qiaole.sinaapp.com?#name=cccccc

$location.absUrl();

// http://qiaole.sinaapp.com?#name=cccccc

$location.host();

// qiaole.sinaapp.com

$location.port();

// 80

$location.protocol();

// http

$location.url();

// ?#name=cccccc

// 获取url参数

$location.search().name;

// or

$location.search()['name'];

## 延时器/定时器

$timeout(function{},1000)

$interval(function(){},1000)

# 自定义服务

app.service(‘name’,function(){return obj})

app.service('Math', function() {

this.add = function(a, b) {

return a + b;

};

this.multipy = function(a, b) {

return a \* b

};

this.num=1;

this.str="wscats.php";

})

也可以跟factory的方法一样

app.factory(‘name’,function(){return obj})

app.factory('Math', function() {

var obj = {

add: function(a, b) {

return a + b;

},

multipy: function(a, b) {

return a \* b

},

num:1,

str:"oaoafly.js"

}

return obj;

})

app.value(‘name’,’value’)

# 组建/指令

| **指令** | **作用** |
| --- | --- |
| ng-app | 定义应用程序的根元素。 |
| ng-bind | 绑定 HTML 元素到应用程序数据 |
| ng-bind-html | 绑定 HTML 元素的 innerHTML 到应用程序数据，并移除 HTML 字符串中危险字符 |
| ng-bind-template | 规定要使用模板替换的文本内容 |
| ng-blur | 规定blur事件的行为 |
| ng-change | 规定在内容改变时要执行的表达式 |
| ng-checked | 规定元素是否被选中 |
| ng-class | 指定 HTML 元素使用的 CSS类 后面跟一个对象  {red:’true/false’} |
| ng-class-even | 类似 ng-class，但只在偶数行起作用 |
| ng-class-odd | 类似 ng-class，但只在奇数行起作用 |
| ng-click | 定义元素被点击时的行为 |
| ng-cloak | 在应用正要加载时防止其闪烁 |
| ng-controller | 定义应用的控制器对象 |
| ng-copy | 规定拷贝事件的行为 |
| ng-csp | 修改内容的安全策略 |
| ng-cut | 规定剪切事件的行为 |
| ng-dblclick | 规定双击事件的行为 |
| ng-disabled | 规定一个元素是否被禁用 |
| ng-focus | 规定聚焦事件的行为 |
| ng-form | 指定 HTML表单继承控制器表单 |
| ng-hide | 隐藏或显示 HTML元素 |
| ng-href | 为 the元素指定链接 |
| ng-if | 如果条件为 false移除 HTML元素 |
| ng-include | 在应用中包含 HTML文件 |
| ng-init | 定义应用的初始化值 |
| ng-jq | 定义应用必须使用到的库，如：jQuery |
| ng-keydown | 规定按下按键事件的行为 |
| ng-keypress | 规定按下按键事件的行为 |
| ng-keyup | 规定松开按键事件的行为 |
| ng-list | 将文本转换为列表 (数组) |
| ng-model | 绑定 HTML 控制器的值到应用数据 |
| ng-mousedown | 规定按下鼠标按键时的行为 |
| ng-model-options | 规定如何更新模型 |
| ng-mouseenter | 规定鼠标指针穿过元素时的行为 |
| ng-mouseleave | 规定鼠标指针离开元素时的行为 |
| ng-mousemove | 规定鼠标指针在指定的元素中移动时的行为 |
| ng-mouseover | 规定鼠标指针位于元素上方时的行为 |
| ng-mouseup | 规定当在元素上松开鼠标按钮时的行为 |
| ng-non-bindable | 规定元素或子元素不能绑定数据 |
| ng-open | 指定元素的open属性 |
| ng-options | 在 <select>列表中指定 <options> |
| ng-paste | 规定粘贴事件的行为 |
| ng-pluralize | 根据本地化规则显示信息 |
| ng-readonly | 指定元素的readonly属性 |
| ng-repeat | 定义集合中每项数据的模板 (ng-repeat=”ele in arr”)如果有重复的内容，需要在后面加track by $index |
| ng-selected | 指定元素的selected属性 |
| ng-show | 显示或隐藏HTML元素 |
| ng-src | 指定 元素的src属性 |
| ng-srcset | 指定 元素的srcset属性 |
| ng-style | 指定元素的 style属性 |
| ng-submit | 规定 onsubmit事件发生时执行的表达式 |
| ng-switch | 规定显示或隐藏子元素的条件 |
| ng-transclude | 规定填充的目标位置 |
| ng-value | 规定 input元素的值 |

# 自定义指令

app.directive(‘name’,function(){

return {

restrict:’ECMA’(可忽略)

template:’html结构’

templateUrl:’路径’

controller:[‘name’,function($scope){}]

controller : ['$scope',function($scope){

//共享数据存放在这里

$scope.name = "Wscats";

//使用this共享给其他指令 传递数据被transclude插槽中的子组件

this.name = "Wscats";

}],

transclude:true/false 在html标签里面插入ng-transclude

link:function(scope,ele,attr){ }

attr =>该组建父节点属性值

ele =>jqLite

| **方法** | **用处** |
| --- | --- |
| addClass() | 为每个匹配的元素添加指定的样式类名 |
| after() | 在匹配元素集合中的每个元素后面插入参数所指定的内容，作为其兄弟节点 |
| append() | 在每个匹配元素里面的末尾处插入参数内容 |
| attr() | 获取匹配的元素集合中的第一个元素的属性的值 |
| bind() | 为一个元素绑定一个事件处理程序 |
| children() | 获得匹配元素集合中每个元素的子元素，选择器选择性筛选 |
| clone() | 创建一个匹配的元素集合的深度拷贝副本 |
| contents() | 获得匹配元素集合中每个元素的子元素，包括文字和注释节点 |
| css() | 获取匹配元素集合中的第一个元素的样式属性的值 |
| data() | 在匹配元素上存储任意相关数据 |
| detach() | 从DOM中去掉所有匹配的元素 |
| empty() | 从DOM中移除集合中匹配元素的所有子节点 |
| eq() | 减少匹配元素的集合为指定的索引的哪一个元素 |
| find() | 通过一个选择器，jQuery对象，或元素过滤，得到当前匹配的元素集合中每个元素的后代 |
| hasClass() | 确定任何一个匹配元素是否有被分配给定的（样式）类 |
| html() | 获取集合中第一个匹配元素的HTML内容 |
| next() | 取得匹配的元素集合中每一个元素紧邻的后面同辈元素的元素集合。如果提供一个选择器，那么只有紧跟着的兄弟元素满足选择器时，才会返回此元素 |
| on() | 在选定的元素上绑定一个或多个事件处理函数 |
| off() | 移除一个事件处理函数 |
| one() | 为元素的事件添加处理函数。处理函数在每个元素上每种事件类型最多执行一次 |
| parent() | 取得匹配元素集合中，每个元素的父元素，可以提供一个可选的选择器 |
| prepend() | 将参数内容插入到每个匹配元素的前面（元素内部） |
| prop() | 获取匹配的元素集中第一个元素的属性（property）值 |
| ready() | 当DOM准备就绪时，指定一个函数来执行 |
| remove() | 将匹配元素集合从DOM中删除。（同时移除元素上的事件及 jQuery 数据。） |
| removeAttr() | 为匹配的元素集合中的每个元素中移除一个属性（attribute） |
| removeClass() | 移除集合中每个匹配元素上一个，多个或全部样式 |
| removeData() | 在元素上移除绑定的数据 |
| replaceWith() | 用提供的内容替换集合中所有匹配的元素并且返回被删除元素的集合 |
| text() | 得到匹配元素集合中每个元素的合并文本，包括他们的后代 |
| toggleClass() | 在匹配的元素集合中的每个元素上添加或删除一个或多个样式类,取决于这个样式类是否存在或值切换属性。即：如果存在（不存在）就删除（添加）一个类 |
| triggerHandler() | 为一个事件执行附加到元素的所有处理程序 |
| unbind() | 从元素上删除一个以前附加事件处理程序 |
| val() | 获取匹配的元素集合中第一个元素的当前值 |
| wrap() | 在每个匹配的元素外层包上一个html元素 |

}

})

$scope.$appy();手动执行脏值检测

# 控制器之间通信数据

## 父->子

$scope.$broadcast(‘name’,data)

## 子->父

$scope.$emit(‘name’,data)

## 接收

$scope.$on(‘name’,function(event,data){(data就是接收的数据)})

## 兄弟传播

借助$rootScope, $rootScope.$on(‘name’,function(event,data){(data就是接收的数据)})

# 本地存储

localStorage.setItem(‘name’,value)

localStorage.getItem(‘name’,value)

# 表达式

<p>{{name}}</p>

<p id="{{name}}">{{name}}</p>

<style>

.blue{

color: {{color}};

}

</style>

# 过滤器

## currency

可以在后面接上其他参数来更改货币符号

Model

$scope.num = 100000;

View

<p>{{num|currency : '¥'}}</p>

## date

Model

$scope.date = 1477966081000;//时间戳(精确到毫秒)

View

<p>{{date|date:'yyyy/MM/dd hh:mm:ss EEEE'}}</p>

## orderBy和filter和limitTo

1. orderBy接受一个布尔值，实现排序
2. filter可以接受字符串，即需要筛选的键值名，也可以接受一个对象，实现筛选
3. limitTo接受两个数字，第一个数字是长度，第二个是索引值，可以用在数组也可以用在字符串截取上

//Model

$scope.filter = "name";

$scope.orderBy = true;

$scope.names = [{

name: 'Wscats',

id: 1

}, {

name: 'Oaoafly',

id: 3

}, {

name: 'Windiest',

id: 2

}];

//View

<ul>

<li ng-repeat="obj in arr | orderBy:sort:'true'">{{obj.name}} {{obj.skill}} {{obj.age}}</li>

<li ng-repeat="obj in arr | filter:{name:sort}">{{obj.name}} {{obj.skill}} {{obj.age}}</li>

<li ng-repeat="obj in arr | limitTo:1:page">{{obj.name}} {{obj.skill}} {{obj.age}}</li>

</ul>

## uppercase和lowercase

//Model

$scope.name = 'wSsCat';

//View

<p>{{name|uppercase}}</p>

<p>{{name|lowercase}}</p>

## json

json过滤器可以把一个js对象格式化为json字符串，没有参数,作用就和我们熟悉的JSON.stringify()一样，一般用于调试

<p>{{obj|json}}</p>

## number

number过滤器可以为一个数字加上千位分割，像这样，123,456,789，同时接收一个参数，可以指定小float类型保留几位小数

Model

$scope.num = 100000;

View

<p>{{num|number:2}}</p>

# 自定义过滤器

app.filter(‘name’,function(){return 一个函数})

# 路由

## ngRoute（需注入ngRoute模块）

<div ng-view></div>

var app = angular.module(‘ngAapp’,[ngRoute])

//配置

app.config(function($routeProvider){

$routeProvider.when(‘/路由名’,{

template:’模板’,

templateUrl:’路径’,

controller:’控制器名’

}).when(‘路由名’,{

//同上

}).otherwise({

redirectTo:’路由名’,//表示默认

})

})

## ui.router（需注入ui.router模块） $urlRouterProvider.otherwise(‘/路由名’)

嵌套路由

<div ui-view></div>

var app = angular.module(‘ngApp’,[ui.router]);

//配置

app.config(function($stateProvider){

$stateProvider.state(‘关联名’, {

[url:’路由名](url:'路由名)’,

template:’模板’,

templateUrl:’路径’

}).state(‘关联名’, {

[url:’路由名](url:'路由名)’,

template:’模板’,

templateUrl:’路径’

})

})

如果需要嵌套路由，点完一个另外一个才出现，则变成

.state(‘father’, {

[url:’路由名](url:'路由名)’,

template:’模板’,

templateUrl:’路径’

}).state(‘father.child’,{

//同上

})

})

## 路由传参

http://路由#!/name?id=2&skill=ps

🡺 http://路由#!/name/2/ps

前端a标签写法<a href=”/路由名/:id/:skill”></a>

ui.router 获取参数服务$state

ngRoute 获取参数服务$routeParams

$location.path()获取路由

跳转详情页

# 表单验证

<!DOCTYPE html>  
<html>  
<script src="http://apps.bdimg.com/libs/angular.js/1.4.6/angular.min.js"></script>  
<body>  
  
<h2>Validation Example</h2>  
  
<form  ng-app="myApp"  ng-controller="validateCtrl"  
name="myForm" novalidate>  
  
<p>用户名:<br>  
  <input type="text" name="user" ng-model="user" required>  
  <span style="color:red" ng-show="myForm.user.$dirty && myForm.user.$invalid">  
  <span ng-show="myForm.user.$error.required">用户名是必须的。</span>  
  </span>  
</p>  
  
<p>邮箱:<br>  
  <input type="email" name="email" ng-model="email" required>  
  <span style="color:red" ng-show="myForm.email.$dirty && myForm.email.$invalid">  
  <span ng-show="myForm.email.$error.required">邮箱是必须的。</span>  
  <span ng-show="myForm.email.$error.email">非法的邮箱。</span>  
  </span>  
</p>  
  
<p>  
  <input type="submit"  
  ng-disabled="myForm.user.$dirty && myForm.user.$invalid ||  
  myForm.email.$dirty && myForm.email.$invalid">  
</p>  
  
</form>  
  
<script>  
var app = angular.module('myApp', []);  
app.controller('validateCtrl', function($scope) {  
    $scope.user = 'John Doe';  
    $scope.email = 'john.doe@gmail.com';  
});  
</script>  
  
</body>  
</html>

|  |  |
| --- | --- |
| **属性** | **描述** |
| $dirty | 表单有填写记录 |
| $valid | 字段内容合法的 |
| $invalid | 字段内容是非法的 |
| $pristine | 表单没有填写记录 |