AngularJS 模型 AngularJS 控制器

AngularJS Scope(作用域)

Scope(作用域) 是应用在 HTML (视图) 和 JavaScript (控制器)之间的纽带。

Scope 是一个对象，有可用的方法和属性。

Scope 可应用在视图和控制器上。

如何使用 Scope

当你在 AngularJS 创建控制器时，你可以将 **$scope** 对象当作一个参数传递:

AngularJS 实例

控制器中的属性对应了视图上的属性：

```
<div ng-app="myApp" ng-controller="myCtrl">

<h1>{{carname}}</h1>

</div>

<script>
var app = angular.module('myApp', []);

app.controller('myCtrl', function($scope) {
    $scope.carname = "Volvo";
});
</script>
```

尝试一下 »

当在控制器中添加 **$scope** 对象时，视图 (HTML) 可以获取了这些属性。

视图中，你不需要添加 **$scope** 前缀, 只需要添加属性名即可，如： **{{carname}}**。

Scope 概述

AngularJS 应用组成如下：

- View(视图), 即 HTML。
- Model(模型), 当前视图中可用的数据。
- Controller(控制器), 即 JavaScript 函数，可以添加或修改属性。

scope 是模型。

scope 是一个 JavaScript 对象，带有属性和方法，这些属性和方法可以在视图和控制器中使用。

AngularJS 实例

如果你修改了视图，模型和控制器也会相应更新：

```
<div ng-app="myApp" ng-controller="myCtrl">
    <input ng-model="name">
    <h1>{{greeting}}</h1>
    <button ng-click='sayHello()'>点我</button>
</div>

<script>
var app = angular.module('myApp', []);
app.controller('myCtrl', function($scope) {
    $scope.name = "Runoob";
    $scope.sayHello = function() {
        $scope.greeting = 'Hello ' + $scope.name + '!';
    };
});
</script>
```

尝试一下 »

Scope 作用范围

了解你当前使用的 scope 是非常重要的。

在以上两个实例中，只有一个作用域 scope，所以处理起来比较简单，但在大型项目中， HTML DOM 中有多个作用域，这时你就需要知道你使用的 scope 对应的作用域是哪一个。

AngularJS 实例

当我们使用 **ng-repeat** 指令时，每个重复项都访问了当前的重复对象:

```
<div ng-app="myApp" ng-controller="myCtrl">

<ul>
    <li ng-repeat="x in names">{{x}}</li>
</ul>

</div>

<script>
var app = angular.module('myApp', []);

app.controller('myCtrl', function($scope) {
    $scope.names = ["Emil", "Tobias", "Linus"];
});
</script>
```

尝试一下 »

每个 **<li>** 元素可以访问当前的重复对象，这里对应的是一个字符串， 并使用变量 **x** 表示。

根作用域

所有的应用都有一个 **$rootScope**，它可以作用在 **ng-app** 指令包含的所有 HTML 元素中。

$rootScope 可作用于整个应用中。是各个 controller 中 scope 的桥梁。用 rootscope 定义的值，可以在各个 controller 中使用。

AngularJS 实例

创建控制器时，将 $rootScope 作为参数传递，可在应用中使用：

```
<div ng-app="myApp" ng-controller="myCtrl">

<h1>{{lastname}} 家族成员:</h1>
```

```
<ul>
    <li ng-repeat="x in names">{{x}} {{lastname}}</li>
</ul>

</div>

<script>
var app = angular.module('myApp', []);

app.controller('myCtrl', function($scope, $rootScope) {
    $scope.names = ["Emil", "Tobias", "Linus"];
    $rootScope.lastname = "Refsnes";
});
</script>
```

尝试一下 »

AngularJS 模型 AngularJS 控制器

笔记列表

$rootscope设置的变量在所有controller里面都是可以直接用{{$root.变量名}}来显示的，当然也可以赋值给$scope.

```
<div ng-app="myApp">
    <div ng-controller="myCtrl">
        <h1>姓氏为 {{lastname}} 家族成员:</h1>
        <ul>
            <li ng-repeat="x in names">{{x}} {{lastname}}</li>
        </ul>
    </div>
    <div ng-controller="myCtrl_1">
        <div>scope中的值是{{lastname_1}}</div>
        <div>rootscope中的值是{{$root.lastname}}</div>
    </div>
</div>
```

尝试一下 »

小强 5个月前 (07-14)

$rootScope 全局对象的属性可在所有子作用域中访问，子作用域互相无法访问对方的私有变量，这一点与js的函数作用域完全一致。

```
var app = angular.module('myApp',[]);
// 两个控制器
app.controller('myCtrl', ['$scope', '$rootScope',myCtrl]);
app.controller('myCtrl2', ['$scope','$rootScope', myCtrl2]);

function myCtrl($scope,$rootScope){
  $scope.myf = 1;
  $rootScope.allf = '全局1';
}

function myCtrl2($scope,$rootScope){
  $scope.mys = 2;
  $rootScope.alls = '全局2';
}
```

尝试一下 »

大大大 3个月前 (09-06)

1、不同的控制器域内可以有同名的变量。但除非必要，否则不建议这么做，自己都会乱掉。

2、后面控制器中定义的rootScope变量，在它前面的控制器中访问不到。

```
function myCtrl($scope,$rootScope){

  $scope.myf = 1;
  $rootScope.allf = '全局1';
  alert($rootScope.alls);  // 访问不到，未定义
}
```

```
function myCtrl2($scope,$rootScope){
  $scope.mys = 2;
  $rootScope.alls = '全局2';
  alert($rootScope.allf);  // 可以访问
  alert($rootScope.alls);  // 可以访问
}
```

xalple 1个月前 (11-07)

实例

```
<script src="https://cdn.bootcss.com/angular.js/1.6.6/angular.min.js"></script>
<body ng-app="myApp" >
    <div ng-controller="myCtrl">
        在控制器中使用scope：{{ name }}<br/>
        在控制器中使用rootScope：{{ nameall }}<br/>
        在控制器中使用scope按钮事件：
        <input type="button" value="scope点击事件" ng-click="sayHello()"/>
    </div>
    不在控制器中使用scope：{{ name }}<br/>
    不在控制器中使用rootScope：{{ nameall }}<br/>
    不在控制器中使用scope按钮事件：
    <input type="button" value="scope点击事件" ng-click="sayHello()"/>
</body>
```

尝试一下 »

zxclcy 1个月前 (11-17)

```
<div ng-controller="myContrl">
  <h1>{{lastname}}家族成员：</h1>
  <ul>
    <li ng-repeat="x in names track by $index">
      {{x}}.{{lastname}}
      <button ng-click="delPerson($index)">删除</button>
    </li>
```

```
  </ul>
  <p>添加成员：
    <input type="text" ng-model="name" placeholder="请输入需要添加的成员名字"></p>
  <button ng-click="addPerson(name)">确认添加</button>
</div>
```

尝试一下 »

上面是代码，直接从webstrom里面复制过来的，运行正常；

我在使用 ng-repeat 循环的基础上利用我会的有限的知识扩展了一下功能，加了两个方法，一个是 addPerson(),添加新的成员，一个是 delPerson(),删除现有成员；都是利用 $scope 实现的，angular会自动监控，一旦存放循环对象的数组 names 里面的数组对象出现变动，都会实时更新并反映到页面的显示上。所以我的添加和删除功能也是根据这个特性实现的。我最主要想要反映的是 ng-repeat 的训话对象是不能出现重复项的，所以如果有重复的就会报错，应该是 key value的问题吧，不是很了解内部运行机制；经过查询发现 在 循环后面加上 track by $index 就会解决问题，也就可以有重复对象了

breaty 1个月前 (11-18)

点我分享笔记

笔记需要是本篇文章的内容扩展！

注册邀请码获取方式

反馈/建议