**AngularJS**

# AngularJS简介

AngularJS 是一个 JavaScript 框架。它可通过 <script> 标签添加到 HTML 页面。

AngularJS 通过 指令 扩展了 HTML，且通过 表达式 绑定数据到 HTML。

### angularJS可以干什么

AngularJS 使得开发现代的单一页面应用程序（SPAs：Single Page Applications）变得更加容易。

* AngularJS 把应用程序数据绑定到 HTML 元素。
* AngularJS 可以克隆和重复 HTML 元素。
* AngularJS 可以隐藏和显示 HTML 元素。
* AngularJS 可以在 HTML 元素"背后"添加代码。
* AngularJS 支持输入验证。

### 如何扩展HTML？

AngularJS 通过 ng-directives 扩展了 HTML。

* ng-app 指令定义一个 AngularJS 应用程序。
* ng-init 指令初始化 AngularJS 应用程序变量。
* ng-model 指令把元素值（比如输入域的值）绑定到应用程序。
* ng-bind 指令把应用程序数据绑定到 HTML 视图。

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

名: <input type="text" ng-model="firstName"><br>

姓: <input type="text" ng-model="lastName"><br>

<br>

姓名: {{firstName + " " + lastName}}

</div>

<script>

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

app.controller('myCtrl', function($scope) {

$scope.firstName= "John";

$scope.lastName= "Doe";

});

</script>

# angularJS语法

### angularJS组成

* 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。

### 表达式

AngularJS 表达式写在双大括号内：{{ expression }}

和JS表达式的异同

* 类似于 JavaScript 表达式，AngularJS 表达式可以包含字母，操作符，变量。
* 与 JavaScript 表达式不同，AngularJS 表达式可以写在 HTML 中。
* 与 JavaScript 表达式不同，AngularJS 表达式不支持条件判断，循环及异常。
* 与 JavaScript 表达式不同，AngularJS 表达式支持过滤器。

### 指令

AngularJS 通过被称为 指令 的新属性来扩展 HTML。

AngularJS 通过内置的指令来为应用添加功能。

AngularJS 允许你自定义指令。

#### 指令

AngularJS 指令是扩展的 HTML 属性，带有前缀 ng-。

* ng-app 指令初始化一个 AngularJS 应用程序。

ng-app 指令定义了 AngularJS 应用程序的 根元素。

ng-app 指令在网页加载完毕时会自动引导（自动初始化）应用程序。

* ng-init 指令初始化应用程序数据。

ng-init 指令为 AngularJS 应用程序定义了 初始值。

通常情况下，不使用 ng-init。您将使用一个控制器或模块来代替它。

* ng-model 指令把元素值（比如输入域的值）绑定到应用程序。

ng-model 指令 绑定 HTML 元素 到应用程序数据。

ng-model 指令也可以：

* 为应用程序数据提供类型验证（number、email、required）。
* 为应用程序数据提供状态（invalid、dirty、touched、error）。
* 为 HTML 元素提供 CSS 类。
* 绑定 HTML 元素到 HTML 表单。
* ng-repeat 指令克隆html元素

ng-repeat 指令对于集合中（数组中）的每个项会 克隆一次 HTML 元素。

<div ng-app="" ng-init="firstName='John'">

<p>在输入框中尝试输入：</p>

<p>姓名：<input type="text" ng-model="firstName"></p>

<p>你输入的为： {{ firstName }}</p>

</div>

#### 自定义指令

除了 AngularJS 内置的指令外，我们还可以创建自定义指令。

你可以使用 .directive 函数来添加自定义的指令。

要调用自定义指令，HTML 元素上需要添加自定义指令名。

使用驼峰法来命名一个指令， runoobDirective, 但在使用它时需要以 - 分割, runoob-directive

<body ng-app="myApp">

<runoob-directive></runoob-directive>

<script>

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

app.directive("runoobDirective", function() {

return {

template : "<h1>自定义指令!</h1>"

};

});

</script>

</body>

**调用自定义指令的方式：**

* 元素名
* 属性
* 类名
* 注释



**限制调用方式：**

//通过添加 restrict 属性,并设置值为 "A", 来设置指令只能通过属性的方式来调用:

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

app.directive("runoobDirective", function() {

return {

restrict : "A",

template : "<h1>自定义指令!</h1>"

};

});

restrict 值可以是以下几种:

* E 作为元素名使用
* A 作为属性使用
* C 作为类名使用
* M 作为注释使用

restrict 默认值为 EA, 即可以通过元素名和属性名来调用指令。

### 实现效果

#### 实现数据绑定（单向绑定/双向绑定）

<div ng-app="" ng-init="quantity=1;price=5">

<h2>价格计算器</h2>

数量： <input type="number" ng-model="quantity">

价格： <input type="number" ng-model="price">

<p><b>总价：</b> {{ quantity \* price }}</p>

</div>

#### 实现重复html元素

<div ng-app="" ng-init="names=['Jani','Hege','Kai']">

<p>使用 ng-repeat 来循环数组</p>

<ul>

<li ng-repeat="x in names">

{{ x }}

</li>

</ul>

</div>

#### 验证用户输入

<form ng-app="" name="myForm">

Email:

<input type="email" name="myAddress" ng-model="text">

<span ng-show="myForm.myAddress.$error.email">不是一个合法的邮箱地址</span>

</form>

<form ng-app="" name="myForm" ng-init="myText = 'test@runoob.com'">

Email:

<input type="email" name="myAddress" ng-model="myText" required>

<p>编辑邮箱地址，查看状态的改变。</p>

<h1>状态</h1>

<p>Valid: {{myForm.myAddress.$valid}} (如果输入的值是合法的则为 true)。</p>

<p>Dirty: {{myForm.myAddress.$dirty}} (如果值改变则为 true)。</p>

<p>Touched: {{myForm.myAddress.$touched}} (如果通过触屏点击则为 true)。</p>

</form>

#### 更改css样式

<style>

input.ng-invalid {

background-color: lightblue;

}

</style>

<body>

<form ng-app="" name="myForm">

输入你的名字:

<input name="myAddress" ng-model="text" required>

</form>

ng-model 指令根据表单域的状态添加/移除以下类：

* ng-empty
* ng-not-empty
* ng-touched
* ng-untouched
* ng-valid
* ng-invalid
* ng-dirty
* ng-pending
* ng-pristine

### 作用域scope

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。

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:

<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>

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}}。

**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。

<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>

### 控制器

AngularJS 应用程序由 ng-app 定义。应用程序在 <div> 内运行。

ng-controller="myCtrl" 属性是一个 AngularJS 指令。用于定义一个控制器。

myCtrl 函数是一个 JavaScript 函数。

AngularJS 使用$scope 对象来调用控制器。

在 AngularJS 中， $scope 是一个应用对象(属于应用变量和函数)。

控制器的 $scope （相当于作用域、控制范围）用来保存AngularJS Model(模型)的对象。

控制器在作用域中创建了两个属性 (firstName 和 lastName)。

ng-model 指令绑定输入域到控制器的属性（firstName 和 lastName）。

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

名: <input type="text" ng-model="firstName"><br>

姓: <input type="text" ng-model="lastName"><br>

<br>

姓名: {{firstName + " " + lastName}}

</div>

<script>

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

app.controller('myCtrl', function($scope) {

$scope.firstName = "John";

$scope.lastName = "Doe";

});

</script>

### 过滤器



过滤器可以通过一个管道字符（|）和一个过滤器添加到表达式中。.

uppercase 过滤器将字符串格式化为大写：

<div ng-app="myApp" ng-controller="personCtrl">

<p>姓名为 {{ lastName | uppercase }}</p>

</div>

自定义过滤器

app.filter('myFormat',['hexafy', function(hexafy) {

return function(x) {

return hexafy.myFunc(x);

};

}]);

<ul>

<li ng-repeat="x in counts">{{x | myFormat}}</li>

</ul>

### 服务

在 AngularJS 中，服务是一个函数或对象，可在你的 AngularJS 应用中使用。

AngularJS 内建了30 多个服务。

#### 内置服务

有个 $location 服务，它可以返回当前页面的 URL 地址。

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

app.controller('customersCtrl', function($scope, $location) {

$scope.myUrl = $location.absUrl();

});

$http 是 AngularJS 应用中最常用的服务。 服务向服务器发送请求，应用响应服务器传送过来的数据。

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

app.controller('myCtrl', function($scope, $http) {

$http.get("welcome.htm").then(function (response) {

$scope.myWelcome = response.data;

});

});

AngularJS $timeout 服务对应了 JS window.setTimeout 函数。

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

app.controller('myCtrl', function($scope, $timeout) {

$scope.myHeader = "Hello World!";

$timeout(function () {

$scope.myHeader = "How are you today?";

}, 2000);

});

AngularJS $interval 服务对应了 JS window.setInterval 函数。

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

app.controller('myCtrl', function($scope, $interval) {

$scope.theTime = new Date().toLocaleTimeString();

$interval(function () {

$scope.theTime = new Date().toLocaleTimeString();

}, 1000);

});

#### 自定义服务

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

this.myFunc = function (x) {

return x.toString(16);

}

});

要使用自定义服务，需要在定义控制器的时候独立添加，设置依赖关系:

//使用自定义的的服务 hexafy 将一个数字转换为16进制数:

app.controller('myCtrl', function($scope, hexafy) {

$scope.hex = hexafy.myFunc(255);

});

### 服务中的http服务

$http 是 AngularJS 中的一个核心服务，用于读取远程服务器的数据。

// 简单的 GET 请求，可以改为 POST

$http({

method: 'GET',

url: '/someUrl'

}).then(function successCallback(response) {

// 请求成功执行代码

}, function errorCallback(response) {

// 请求失败执行代码

});

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

app.controller('siteCtrl', function($scope, $http) {

$http({

method: 'GET',

url: 'https://www.runoob.com/try/angularjs/data/sites.php'

}).then(function successCallback(response) {

$scope.names = response.data.sites;

}, function errorCallback(response) {

// 请求失败执行代码

});

});