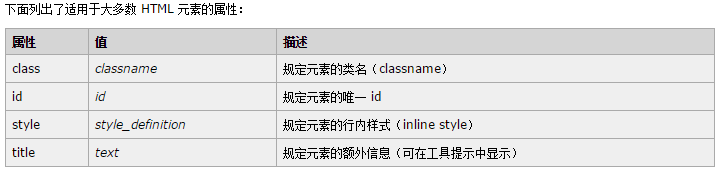
**HTML**

1.属性值应该始终被包括在引号内，最好小写。双引号是最常用的，不过使用单引号也没有问题。

在某些个别的情况下，比如属性值本身就含有双引号，那么您必须使用单引号，例如：

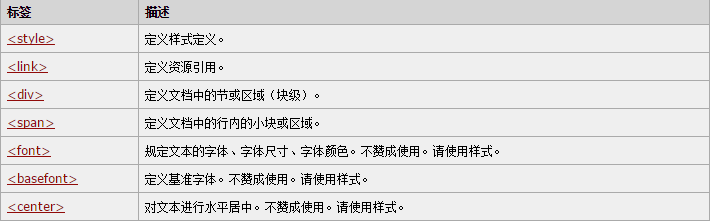
name='Bill "HelloWorld" Gates'

2.

1. pre 标签很适合显示计算机代码，此标签内的内容保持原格式。
2. <a href="/example/html/lastpage.html" style="text-decoration:none">

这是一个无下划线的链接！

</a>

5.

6.设置背景图片：<body background="/i/eg\_background.jpg">

7.

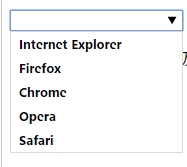


8.<meta> 标签提供关于 HTML 文档的元数据。元数据不会显示在页面上，但是对于机器是可读的。典型的情况是，meta 元素被用于规定页面的描述、关键词、文档的作者、最后修改时间以及其他元数据。

<meta> 标签始终位于 head 元素中。元数据可用于浏览器（如何显示内容或重新加载页面），搜索引擎（关键词），或其他 web 服务。



1. 通过 <datalist> 设置预定义值的 <input> 元素：

<form action="action\_page.php">

<input list="browsers">

<datalist id="browsers">

<option value="Internet Explorer">

<option value="Firefox">

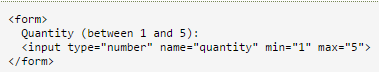
<option value="Chrome">

<option value="Opera">

<option value="Safari">

</datalist> </form>

10.



**JavaScript**

1.<p id="demo">JavaScript 能改变 HTML 元素的内容。</p>

<script>

function myFunction()

{x=document.getElementById("demo"); // 找到元素

x.innerHTML="Hello JavaScript!"; // 改变内容

}</script>

2.function changeImage() //能使一个灯的图片变亮变暗

{element=document.getElementById('myimage')

if (element.src.match("bulbon"))

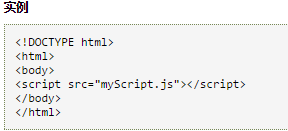
{

element.src="/i/eg\_bulboff.gif";

}else

{ element.src="/i/eg\_bulbon.gif";}}

如需使用外部文件，请在 <script> 标签的 "src" 属性中设置该 .js 文件，外部脚本不能包含 <script> 标签。：



1. JavaScript 是脚本语言。浏览器会在读取代码时，逐行地执行脚本代码。而对于传统编程来说，会在执行前对所有代码进行编译。
2. JavaScript 变量还能保存其他数据类型，比如文本值 (name="Bill Gates")。

在 JavaScript 中，类似 "Bill Gates" 这样一条文本被称为字符串。

JavaScript 变量有很多种类型，但是现在，我们只关注数字和字符串。

当您向变量分配文本值时，应该用双引号或单引号包围这个值。

当您向变量赋的值是数值时，不要使用引号。如果您用引号包围数值，该值会被作为文本来处理。

1. String对象的toUpperCase() 方法来把文本转换为大写

7.验证邮箱的js：html><head>

<script type="text/javascript">

function validate\_email(field,alerttxt)

{with (field)

{

apos=value.indexOf("@")

dotpos=value.lastIndexOf(".")

if (apos<1||dotpos-apos<2)

{alert(alerttxt);return false}

else {return true}

}

}

function validate\_form(thisform)

{

with (thisform)

{

if (validate\_email(email,"Not a valid e-mail address!")==false)

{email.focus();return false}

}}

</script>

</head><body>

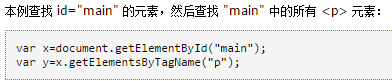
<form action="submitpage.htm"onsubmit="return validate\_form(this);" method="post">

Email: <input type="text" name="email" size="30">

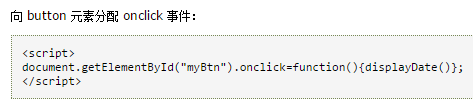
<input type="submit" value="Submit">

</form></body></html>

8.

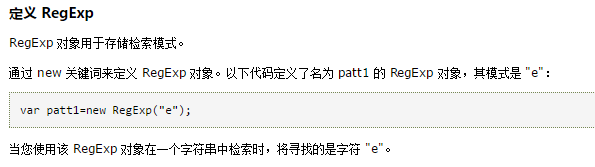


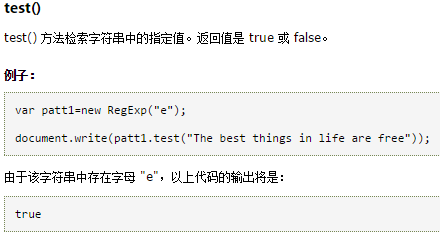
9.

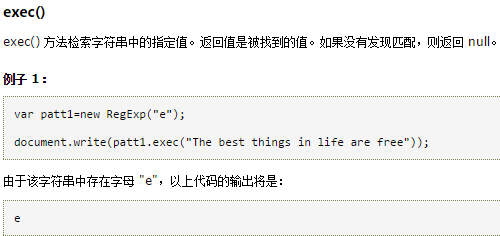


1. RegExp 是正则表达式的缩写。

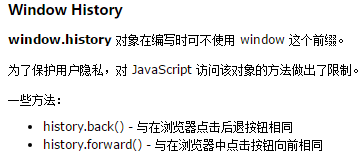
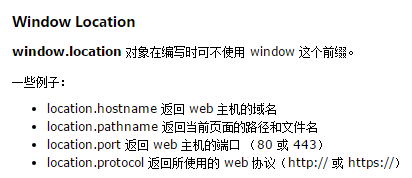
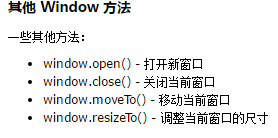
当您检索某个文本时，可以使用一种模式来描述要检索的内容。RegExp 就是这种模式。

简单的模式可以是一个单独的字符。更复杂的模式包括了更多的字符，并可用于解析、格式检查、替换等等。您可以规定字符串中的检索位置，以及要检索的字符类型，等等





11.



12.

AngularJS通过ng-directives 扩展了 HTML。

ng-app指令定义一个AngularJS应用程序。

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。

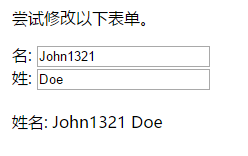
当网页加载完毕，AngularJS 自动开启。

ng-app 指令告诉 AngularJS，<div> 元素是 AngularJS 应用程序 的"所有者"。

ng-model 指令把输入域的值绑定到应用程序变量 name。

ng-bind 指令把应用程序变量 name 绑定到某个段落的 innerHTML。

13.<script src="https://cdn.static.runoob.com/libs/angular.js/1.4.6/angular.min.js"></script>

</head>

<body>

<p>尝试修改以下表单。</p>

<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>

14.AngularJS 表达式 与 JavaScript 表达式

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。

15.ng-app

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。

稍后您将学习到 ng-app 如何通过一个值（比如 ng-app="myModule"）连接到代码模块。

ng-init

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。

稍后您将学习更多有关控制器和模块的知识。

ng-model

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

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

16.

17.



18.

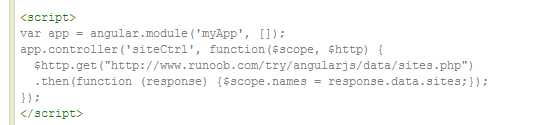


19.AngularJS $http

AngularJS $http 是一个用于读取web服务器上数据的服务。

$http.get(url) 是用于读取服务器数据的函数。



简写：

20.



21.1、载入了实现路由的 js 文件：angular-route.js。

2、包含了 ngRoute 模块作为主应用模块的依赖模块。

angular.module('routingDemoApp',['ngRoute'])

3、使用 ngView 指令。

<div ng-view></div>

该 div 内的 HTML 内容会根据路由的变化而变化。

4、配置 $routeProvider，AngularJS $routeProvider 用来定义路由规则。

module.config(['$routeProvider', function($routeProvider){

$routeProvider

.when('/',{template:'这是首页页面'})

.when('/computers',{template:'这是电脑分类页面'})

.when('/printers',{template:'这是打印机页面'})

.otherwise({redirectTo:'/'});

}]);

AngularJS 模块的 config 函数用于配置路由规则。通过使用 configAPI，我们请求把$routeProvider注入到我们的配置函数并且使用$routeProvider.whenAPI来定义我们的路由规则。

$routeProvider 为我们提供了 when(path,object) & otherwise(object) 函数按顺序定义所有路由，函数包含两个参数:

第一个参数是 URL 或者 URL 正则规则。第二个参数是路由配置对象。