# 学习网址

1. http://www.w3school.com.cn/jquery/jquery\_ref\_dom\_element\_methods.asp

# 当前流程前端框架

Vue.js node.js jquery.js angular.js

# Html

## 学习网址

http://www.w3school.com.cn/tags/tag\_html.asp

## 概念

超文本标记语言，[标准通用标记语言](https://baike.baidu.com/item/%E6%A0%87%E5%87%86%E9%80%9A%E7%94%A8%E6%A0%87%E8%AE%B0%E8%AF%AD%E8%A8%80/6805073)下的一个应用。

**“**[超文本](https://baike.baidu.com/item/%E8%B6%85%E6%96%87%E6%9C%AC/2832422)**”**就是指页面内可以包含图片、[链接](https://baike.baidu.com/item/%E9%93%BE%E6%8E%A5/2665501)，甚至音乐、[程序](https://baike.baidu.com/item/%E7%A8%8B%E5%BA%8F/71525)等非文字元素。

超文本标记语言的结构包括**“**头”部分（英语：Head）、和“主体”部分（英语：Body），其中“头”部提供关于网页的信息，“主体”部分提供网页的[具体](https://baike.baidu.com/item/%E5%85%B7%E4%BD%93/4577821)内容。

## 特点

超级文本标记语言文档制作不是很复杂，但功能强大，支持不同数据格式的[文件](https://baike.baidu.com/item/%E6%96%87%E4%BB%B6" \t "_blank)镶入，这也是万维网（[WWW](https://baike.baidu.com/item/WWW)）盛行的原因之一，其主要特点如下：

1. 简易性：超级文本标记语言[版本升级](https://baike.baidu.com/item/%E7%89%88%E6%9C%AC%E5%8D%87%E7%BA%A7)采用[超集](https://baike.baidu.com/item/%E8%B6%85%E9%9B%86)方式，从而更加灵活方便。
2. [可扩展性](https://baike.baidu.com/item/%E5%8F%AF%E6%89%A9%E5%B1%95%E6%80%A7)：超级文本标记语言的广泛应用带来了加强功能，增加[标识符](https://baike.baidu.com/item/%E6%A0%87%E8%AF%86%E7%AC%A6)等要求，超级文本标记语言采取子类元素的方式，为系统扩展带来保证。
3. 平台无关性：虽然[个人计算机](https://baike.baidu.com/item/%E4%B8%AA%E4%BA%BA%E8%AE%A1%E7%AE%97%E6%9C%BA" \t "_blank)大行其道，但使用[MAC](https://baike.baidu.com/item/MAC)等其他机器的大有人在，超级文本标记语言可以使用在广泛的平台上，这也是[万维网](https://baike.baidu.com/item/%E4%B8%87%E7%BB%B4%E7%BD%91" \t "_blank)（[WWW](https://baike.baidu.com/item/WWW)）盛行的另一个原因。
4. 通用性：另外，[HTML](https://baike.baidu.com/item/HTML" \t "_blank)是网络的通用语言,一种简单、通用的全置标记语言。它允许网页制作人建立文本与图片相结合的复杂页面，这些页面可以被网上任何其他人浏览到，无论使用的是什么类型的电脑或浏览器。

## Html和xhtml区别

xmlns 属性在 XHTML 中是必需的，但在 HTML 中不是。不过，即使 XHTML 文档中的 <html> 没有使用此属性，W3C 的验证器也不会报错。这是因为 "xmlns=http://www.w3.org/1999/xhtml" 是一个固定值，即使您没有包含它，此值也会被添加到 <html> 标签中。

## 标签

1. <!DOCTYPE html> 定义文档类型。

# Css

## 概念

层叠样式表(英文全称：Cascading Style Sheets)是一种用来表现[HTML](https://baike.baidu.com/item/HTML" \t "_blank)（[标准通用标记语言](https://baike.baidu.com/item/%E6%A0%87%E5%87%86%E9%80%9A%E7%94%A8%E6%A0%87%E8%AE%B0%E8%AF%AD%E8%A8%80/6805073)的一个应用）或[XML](https://baike.baidu.com/item/XML" \t "_blank)（标准通用标记语言的一个子集）等文件样式的计算机语言。CSS不仅可以静态地修饰网页，还可以配合各种脚本语言动态地对网页各元素进行格式化。

# JavaScript

## 概念

JavaScript一种直译式脚本语言，是一种动态类型、弱类型、基于原型的语言，内置支持类型。它的解释器被称为JavaScript引擎，为浏览器的一部分，广泛用于客户端的脚本语言，最早是在[HTML](https://baike.baidu.com/item/HTML)（标准通用标记语言下的一个应用）网页上使用，用来给HTML网页增加动态功能。

# Jsp

## 概念

JSP与Servlet一样，是在服务器端执行的。通常返回给客户端的就是一个HTML文本，因此客户端只要有浏览器就能浏览。

JSP全名为Java Server Pages，中文名叫java[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8)页面，其根本是一个简化的[Servlet](https://baike.baidu.com/item/Servlet)设计，它 [1]  是由[Sun Microsystems](https://baike.baidu.com/item/Sun%20Microsystems)公司倡导、许多公司参与一起建立的一种[动态网页](https://baike.baidu.com/item/%E5%8A%A8%E6%80%81%E7%BD%91%E9%A1%B5)技术标准。JJSP技术有点类似ASP技术，它是在传统的[网页](https://baike.baidu.com/item/%E7%BD%91%E9%A1%B5" \t "_blank)HTML（[标准通用标记语言](https://baike.baidu.com/item/%E6%A0%87%E5%87%86%E9%80%9A%E7%94%A8%E6%A0%87%E8%AE%B0%E8%AF%AD%E8%A8%80)的子集）文件(\*.htm,\*.[html](https://baike.baidu.com/item/html))中插入Java[程序段](https://baike.baidu.com/item/%E7%A8%8B%E5%BA%8F%E6%AE%B5)(Scriptlet)和JSP标记(tag)，从而形成JSP文件，后缀名为(\*.jsp)。 用JSP开发的Web应用是[跨平台](https://baike.baidu.com/item/%E8%B7%A8%E5%B9%B3%E5%8F%B0/8558902)的，既能在Linux下运行，也能在其他操作系统上运行。

# Jquery.js

## 学习网址

<https://www.runoob.com/jquery/jquery-tutorial.html>

<http://www.w3school.com.cn/jquery/jquery_selectors.asp>

## 概念

jQuery就是javascript的一个库，把我们常用的一些功能进行了封装，方便我们来调用，提高我们的开发效率。极大地简化了 JavaScript 编程。

Javascipt跟jQuery的区别：

Javascript是一门编程语言，我们用它来编写客户端浏览器脚本。

jQuery是javascript的一个库，包含多个可重用的函数，用来辅助我们简化javascript开发

jQuery能做的javascipt都能做到，而javascript能做的事情，jQuery不一定能做到。

## 选择器

最基础的三大选择器：Id选择器（#id)、class类选择器（.class）、标签选择器(p/div)。

其他选择器：

1. 组合选择器（.class,#id）
2. 层次选择器

$("#id>.classname ") //子元素选择器

$("#id .classname ") //后代元素选择器

$("#id + .classname ") //紧邻下一个元素选择器

$("#id ~ .classname ") //兄弟元素选择器

1. 表单选择器

$(":input") //匹配所有 input, textarea, select 和 button 元素

$(":text") //所有的单行文本框，$(":text") 等价于$("[type=text]")，推荐使用$("input:text")效率更高，下同

$(":password") //所有密码框

$(":radio") //所有单选按钮

$(":checkbox") //所有复选框

$(":submit") //所有提交按钮

$(":reset") //所有重置按钮

$(":button") //所有button按钮

$(":file") //所有文件域

1. 过滤选择器

$("li:first") //第一个li

$("li:last") //最后一个li

$("li:even") //挑选下标为偶数的li

$("li:odd") //挑选下标为奇数的li

$("li:eq(4)") //下标等于 4 的li(第五个 li 元素)

$("li:gt(2)") //下标大于 2 的li

$("li:lt(2)") //下标小于 2 的li

$("li:not(#runoob)") //挑选除 id="runoob" 以外的所有li

$("li:hidden") //匹配所有不可见元素，或type为hidden的元素

$("li:visible") //匹配所有可见元素

状态过滤选择器

$("input:enabled") // 匹配可用的 input

$("input:disabled") // 匹配不可用的 input

$("input:checked") // 匹配选中的 input

$("option:selected") // 匹配选中的 option

属性过滤选择器（li[name=XX] ul li[name$=’name’] $表示以什么结尾）

$("div[id]") //所有含有 id 属性的 div 元素

$("div[id='123']") // id属性值为123的div 元素

$("div[id!='123']") // id属性值不等于123的div 元素

$("div[id^='qq']") // id属性值以qq开头的div 元素

$("div[id$='zz']") // id属性值以zz结尾的div 元素

$("div[id\*='bb']") // id属性值包含bb的div 元素

$("input[id][name$='man']") //多属性选过滤，同时满足两个属性的条件的元素

内容过滤

$("div:contains('Runob')") // 包含 Runob文本的元素

$("td:empty") //不包含子元素或者文本的空元素

$("div:has(selector)") //含有选择器所匹配的元素

$("td:parent") //含有子元素或者文本的元素

## 元素内容设置

text() - 设置或返回所选元素的文本内容

html() - 设置或返回所选元素的内容（包括 HTML 标记）

val() - 设置或返回表单字段的值

attr() – 设置或返回所选元素的某属性值

css() – 设置或返回所选元素的样式值，多个属性的设置：$("p").css({"background-color":"yellow","font-size":"200%"});

备注：若函数为一个参数，表示获取，若为两个参数表示赋值。例如：$(p).attr(name)-获取name属性值 , $(p).attr(name,”123”)-设置name的属性值为123。

## Jquery元素追加

1. append() - 在被选元素的结尾插入内容，即使是div块级标签，使用该函数追加时，不会换行仍然在其尾部直接追加。
2. prepend() - 在被选元素的开头插入内容，显示的位置与选中元素的标签级别无关，紧邻目标元素进行追加不会换行。
3. after() - 在被选元素之后插入内容，若被选中元素是块级标签则换行追加，若为行级标签将在尾部追加。
4. before() - 在被选元素之前插入内容，显示的位置与选中元素的标签级别有关。

## jquery元素删除

remove() - 删除被选元素（及其子元素）

empty() - 从被选元素中删除子元素

## jquery操作css

addClass() - 向被选元素添加一个或多个类

removeClass() - 从被选元素删除一个或多个类

toggleClass() - 对被选元素进行添加/删除类的切换操作，若设置的元素所有当前样式则删除，若没有则添加。

css() - 设置或返回样式属性

## jquery遍历

### 向上遍历

这些 jQuery 方法很有用，它们用于向上遍历 DOM 树：

1. parent()：返回被选元素的直接父元素。该方法只会向上一级对 DOM 树进行遍历。
2. parents()：parents() 方法返回被选元素的所有祖先元素，它一路向上直到文档的根元素 (<html>)。
3. parentsUntil()：parentsUntil() 方法返回介于两个给定元素之间的所有祖先元素。例子：

<https://www.runoob.com/try/try.php?filename=tryjquery_parentsuntil>

### 向下遍历

下面是两个用于向下遍历 DOM 树的 jQuery 方法：

1. children()：返回被选元素的所有直接子元素。该方法只会向下一级对 DOM 树进行遍历。
2. find()：返回被选元素的后代元素，一路向下直到最后一个后代。

### 同级遍历

有许多有用的方法让我们在 DOM 树进行水平遍历：

1. siblings()：返回被选元素的所有同胞元素
2. next()：被选元素的下一个同胞元素
3. nextAll()：返回被选元素的所有跟随的同胞元素
4. nextUntil()：返回介于两个给定参数之间的所有跟随的同胞元素
5. prev()： 被选元素的上一个同胞元素
6. prevAll()：返回被选元素的所有上面的同胞元素
7. prevUntil()：返回介于两个给定参数之间的所有上面的同胞元素

## load

load() 方法从服务器加载数据，并把返回的数据放入被选元素中。$(selector).load(URL,data,callback);

例如：

$("#div1").load("demo\_test.txt");

例子：https://www.runoob.com/jquery/jquery-ajax-load.html

## get post

两种在客户端和服务器端进行请求-响应的常用方法是：GET 和 POST。

1. *GET* - 从指定的资源请求数据：GET 基本上用于从服务器获得（取回）数据。注释：GET 方法可能返回缓存数据。
2. *POST* - 向指定的资源提交要处理的数据：POST 也可用于从服务器获取数据。不过，POST 方法不会缓存数据，并且常用于连同请求一起发送数据。
3. 区别：

https://www.runoob.com/tags/html-httpmethods.html

## Jquery校验

https://www.runoob.com/jquery/jquery-plugin-validate.html

## Click、on、onClick

1. Onclick给当前dom元素绑定一个鼠标点击事件，即本身为一个触发点击事项
2. Click是jquery封装事件函数，该函数主要用于触发onclick事件。若某一对象绑定了click函数并绑定了onclick事件，则在点击时onclick事件会先于click函数执行，即onclick事件中绑定的函数执行完毕以后该对象绑定的click函数才会触发。
3. On为jquery事件绑定函数，若页面的dom对象加载时存在即可调用使用触发click也可用on绑定触发函数。但在dom对象加载完毕后给追加的dom元素设置点击事件时，只能通过on函数进行单击函数的绑定，直接绑定click函数是不会触发执行的。

# AngularJS

## 学习网址

https://www.runoob.com/angularjs/angularjs-tutorial.html

## 概念

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**](https://www.runoob.com/angularjs/angularjs-directives.html) 扩展了 HTML，且通过 [**表达式**](https://www.runoob.com/angularjs/angularjs-expressions.html) 绑定数据到 HTML。

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。

菜鸟教程建议：建议把脚本放在 <body> 元素的底部。这会提高网页加载速度，因为 HTML 加载不受制于脚本加载。

## 使用

AngularJS的使用有两个关键概念：指令、表达式。Angular整个应用程序的定义和控制由：module和controller进行操作。

AngularJS 应用组成如下：

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

### 指令

#### 定义

AngularJS 指令是以 **ng** 作为前缀的 HTML 属性。

当网页加载完毕，AngularJS 自动开启。

HTML5 允许扩展的（自制的）属性，以 data- 开头。  
AngularJS 属性以 ng- 开头，但是您可以使用 data-ng- 来让网页对 HTML5 有效。例如：

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

#### 使用

1. **ng-app** 指令告诉 AngularJS，<div> 元素是 AngularJS **应用程序** 的"所有者"。
2. **ng-model** 指令把输入域的值绑定到应用程序变量 **name**。使用该指令以后，可通过对应的属性查看当前值的合法性，例如：

<form ng-app="" name="myForm" ng-init="myText = 'test@runoob.com'">  
    Email:  
    <input type="email" name="myAddress" ng-model="myText" required></p>  
    <h1>状态</h1>  
    {{myForm.myAddress.$valid}} (如果输入的值是合法的则为 true)  
    {{myForm.myAddress.$dirty}} (如果值改变则为 true)  
    {{myForm.myAddress.$touched}} (如果通过触屏点击则为 true)  
</form>

Ng-model基于html提供了css，例如：

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
* ng-invalid:未通过验证的表单
* ng-valid:布尔型属性，它指示表单是否通过验证。如果表单当前通过验证，他将为true
* ng-dirty:布尔值属性，表示用户是否修改了表单。如果为 ture，表示有修改过；false 表示修没有修改过
* ng-touched:布尔值属性，表示用户是否和控件进行过交互
* ng-pristine:布尔值属性，表示用户是否修改了表单。如果为ture，表示没有修改过；false表示修改过
* ng-valid: 验证通过
* ng-invalid: 验证失败
* ng-valid-[key]: 由$setValidity添加的所有验证通过的值
* ng-invalid-[key]: 由$setValidity添加的所有验证失败的值
* ng-pristine: 控件为初始状态
* ng-dirty: 控件输入值已变更
* ng-touched: 控件已失去焦点
* ng-untouched: 控件未失去焦点
* ng-pending: 任何为满足$asyncValidators的情况

1. **ng-bind** 指令把应用程序变量 name 绑定到某个段落的 innerHTML。
2. **ng-show**属性返回 true 的情况下显示。例如：

<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>

1. **ng-init**指令初始化 AngularJS 应用程序变量。多个参数可用分号隔开，例如：ng-init="quantity=1;cost=5"。
2. **ng-repeat** 指令会重复一个 HTML 元素。例如。使用ng-init初始化一个数组，使用ng-repeat进行循环显示。
3. 初始化数组：Ng-init=“points=[1,5,6,8]”
4. 循环遍历数组，注意需要在指定html元素中设置ng-repeat指令。例如：<ul><li ng-repeat=”x in points”>{{x}}</li></ul>
5. 循环指令显示结果：

<li>1</li>

<li>2</li>

<li>6</li>

<li>8</li>

1. 添加自定义指令：
2. 使用 **.directive** 函数来添加自定义的指令。
3. 要调用自定义指令，HTML 元素上需要添加自定义指令名。
4. 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**:
5. 例如：

<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>

1. 指令的调用
2. 元素名：<runoob-directive></runoob-directive>
3. 属性：<div runoob-directive></div>
4. 类名：<div class="runoob-directive"></div>
5. 注释：<!-- directive: runoob-directive -->
6. 指令的限制调用，通过添加 **restrict** 属性,限制调用方式。

**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

### 表达式

#### 定义

AngularJS 表达式写在双大括号内：**{{ expression }}**。

AngularJS 表达式把数据绑定到 HTML，这与 **ng-bind** 指令有异曲同工之妙。

AngularJS 将在表达式书写的位置"输出"数据。

**AngularJS 表达式** 很像 **JavaScript 表达式**：它们可以包含文字、运算符和变量。

实例 {{ 5 + 5 }} 或 {{ firstName + " " + lastName }}

#### 使用

Angular的表达式中可进行加减乘除的操作。

1. 数字 例如：{{ 2 \* 5}}—输出10
2. 字符串 例如：{{ 2 +‘’+ 5}}—输出25
3. 对象 例如：初始化指令

ng-init=”person={name:‘lhb’,age:25}”

表达式 {{person.name}}输出结果：lhb

1. 数组 例如：初始化指令

ng-init=”points=[1,4,6,2]”

表达式{{points[0]}}输出结果：1

### Module

#### 定义

AngularJS **模块（Module）** 定义了 AngularJS 应用。

例如：

<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>

#### 使用

### Controller

#### 定义

AngularJS **控制器（Controller）** 用于控制 AngularJS 应用。

**ng-app**指令指明了应用, **ng-controller** 指明了控制器。

例如：

<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>

#### 使用

### Scope

#### 定义

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。

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。

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。

#### 使用

<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>

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

$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。

# VueJS /vjuː/

## 学习网址

https://www.runoob.com/vue2/vue-tutorial.html

## 概念

Vue.js（读音 /vjuː/, 类似于 view） 是一套构建用户界面的渐进式框架。

Vue 只关注视图层， 采用自底向上增量开发的设计。

Vue 的目标是通过尽可能简单的 API 实现响应的数据绑定和组合的视图组件。

Vue 学习起来非常简单，本教程基于 Vue 2.1.8 版本测试。

# ReactJS [riˈækt]

## 学习网址

http://www.runoob.com/react/react-tutorial.html

## 概念

React 是一个用于构建用户界面的 JAVASCRIPT 库。

React主要用于构建UI，很多人认为 React 是 MVC 中的 V（视图）。

React 起源于 Facebook 的内部项目，用来架设 Instagram 的网站，并于 2013 年 5 月开源。

React 拥有较高的性能，代码逻辑非常简单，越来越多的人已开始关注和使用它。

# NodeJS

## 学习网址

https://www.runoob.com/nodejs/nodejs-tutorial.html

## 概念

简单的说 Node.js 就是运行在服务端的 JavaScript。

Node.js 是一个基于Chrome JavaScript 运行时建立的一个平台。

Node.js是一个事件驱动I/O服务端JavaScript环境，基于Google的V8引擎，V8引擎执行Javascript的速度非常快，性能非常好。