## 2018.6.15

jsp:

JSP页面主要由指令标签，注释，嵌入Java代码，JSP动作标签，HTML标记语言等五个元素组成。

include指令（用于文件包含，该指令可以在一个JSP页面中包含另外一个文件中的内容，并且只能支持静态包含，意思是被包含文件的所有内容都被原样包含到时该JSP页面中，被包含的文件可以是一段Java代码，HTML代码，或者是另一个JSP页面。）Eg:<%@include file=”other.jsp” %>

page指令（）<%@ page attr1=”value1” atrr2=”value2”.......%>

taglib指令（ taglib指令用于加载用户自定义标签，使用此指令后可以直接在JSP页面使用自定义标签。类似于JSTL标签，就必须采用此指令. ） Eg:<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

在JSP页面嵌入Java代码，必须满足语法:<% %>要将Java代码片段写在这两个标签这中，可以编写单行或者多行的Java代码. Eg:<% Date now=new Date();%>

声明用于定义JSP页面的全局变量，它们可以被整个JSP页面访问，例如：定义全局变量<%! Long startTime=System.nanoTime();%>

JSP表达式:<%=表达式 %>可以直接把Java代码的表达式结果输出到JSP页面中。类似于out对象的输出功能。Eg:<%=Math.PI()%>;

HTML注释：<!--注释文本-->

JSP注释:<%--注释文本-->

JSP动态注释:<!--<%=new Date()%>-->

动态标签

包含标签<jsp:include>

转发标签<jsp:forward>

参数标签<jsp:param>

操作JavaBean的动作标签 <jsp:useBean>标签 -----<jsp:setProperty>标签----<jsp:getProperty>标签

<c:if> 标签必须要有test属性，当test中的表达式结果为true时，则会执行本体内容；如果为false，则不会执行。

<c:if test="${aaa}"></c:if>

<c:forEach>标签的作用就是迭代输出标签内部的内容。它既可以进行固定次数的迭代输出，也可以依据集合中对象的个数来决定迭代的次数。

<c:forEach var="每个变量名字" items="要迭代的list" varStatus="每个对象的状态" begin="循环从哪儿开始" end="循环到哪儿结束" step="循环的步长">

循环要输出的东西

</c:forEach>

jsp中〈%= %〉和＄｛｝两种输出,前者就是输出一个java表达式或者语句的值，后者是EL表达式。后者能输出的，前者一定也能输出，但是反过来就不一定了。

\"是转译符，因为在双引号是出现双引号，会出现组合混乱，所以转译，表示此双引号非彼双引号。

for...in 语句用于对数组或者对象的属性进行循环操作。

|  |
| --- |
| **for** (变量 **in** 对象)  {      在此执行代码  } |

[for循环](https://www.baidu.com/s?wd=for%E5%BE%AA%E7%8E%AF&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank)是对数组的元素进行循环，而不能引用于非数组对象。

|  |  |
| --- | --- |
| 1  2  3 | **for**(int 变量初始值;条件;递增或递减){      在此执行代码  } |

$.each(data, function(index, value){

}

遍历处理data，可以是数组、DOM、json等，取决于直接给定或者ajax返回的类型

[width()](http://www.runoob.com/jquery/css-width.html) - 设置或返回元素的宽度

[height()](http://www.runoob.com/jquery/css-height.html) - 设置或返回元素的高度

[innerWidth()](http://www.runoob.com/jquery/html-innerwidth.html) - 返回元素的宽度（包含 padding）

[innerHeight()](http://www.runoob.com/jquery/html-innerheight.html) - 返回元素的高度（包含 padding）

[outerWidth()](http://www.runoob.com/jquery/html-outerwidth.html) - 返回元素的宽度（包含 padding 和 border）

outerHeight() 方法返回第一个匹配元素的外部高度(包含 padding 和 border)

**提示：**如需包含 margin，请使用 outerHeight(true)。

1）.css('height')和.height()之间的区别是后者返回一个没有单位的数值（例如，400），前者是返回带有完整单位的字符串（例如，400px）。

2）.height()总是返回内容宽度,不管CSSbox-sizing属性值。.height('value')设置的容器宽度是根据CSSbox-sizing属性来定的, 将这个属性值改成border-box，将造成这个函数改变这个容器的outerHeight，而不是原来的内容高度。jQuery.extend(object);为扩展jQuery类本身.为类添加新的方法。   
jQuery.fn.extend(object);给jQuery对象添加方法。

window.location.href 是本页面跳转

window.top.location.href是最外层的页面跳转

top.location.reload();   刷新整页    
self.location.reload();  刷新本页

Javascript刷新页面的几种方法：   
1 history.go(0)   
2 location.reload()   
3 location=location.href(兼容所有的）  
4 location.assign(location)   
5 document.execCommand('Refresh')  刷新当前文档  
6 window.navigate(location)   
7 location.replace(location)   
8 document.URL=location.href

location.reload(true)时，相当于强制刷新  
location.reload(false)时，刷新，但可以引用缓存的数据。

history.go(0)是直接读取缓存数据，不会从服务器端获取数据  
location.reload()是会从服务端获取数据的

replace() 方法不会在 History 对象中生成一个新的记录。当使用该方法时，新的 URL 将覆盖 History 对象中的当前记录。

window.location.assign(url) ： 加载 URL 指定的新的 HTML 文档。 就相当于一个链接，跳转到指定的url，当前页面会转为新页面内容，可以点击后退返回上一个页面。

window.navigate("http://ww.baidu.com/") 这个方法是只针对IE的，不适用于火狐等其他浏览器，

我尝试用jQuery控制HTML5视频，两个视频分别在两个tab中，我希望点中tab后，该tab里的视频可以立即播放，而另外tab里的视频能够停止。  
我的代码是这样的：

|  |  |
| --- | --- |
| 1  2  3 | $('#playMovie1').click(function(){  $('#movie1').play();  }); |

但发现这样不行，而用以下的js是可以的：

|  |  |
| --- | --- |
| 1 | document.getElementById('movie1').play(); |

**解决方法：**  
play并不是jQuery的函数，而是DOM元素的函数，所以我们需要通过DOM来调用play，代码如下：

|  |  |
| --- | --- |
| 1 | $('#videoId').get(0).play(); |

**最简单的方法实现Play和Pause:**

|  |  |
| --- | --- |
| 1  2 | $('video').trigger('play');  $('video').trigger('pause'); |

-----------------------------------------------------------------------

$nextTick 是在下次 DOM 更新循环结束之后执行延迟回调，在修改数据之后使用 $nextTick，则可以在回调中获取更新后的 DOM，

# 关于SetInterval(code,time)的第一个参数的几种用法详解

setInterval("fun()",1000); //全局作用域下正常执行

其中这种加引号的调用就可以理解为 可执行代码  就行eval 一样去执行第一个参数，就是对fun方法的调用 理所当然的弹出 1  一秒钟间隔，一直执行。

在全局作用域下  setInterval("fun()",1000); 可以正常执行，但放到window.onload函数中却不能执行

而setInterval现在把以引号包括的可执行代码进行处理。就像eval一样给予执行。其在执行中 fun() 执行环境发生了变化，不是在window.onload方法下，而是在全局环境中也就是window.大家应该知道JavaScript存在作用域链，由内向外依次查找。内部可以访问其上层的函数和变量，而外部却不能访问内部的函数和变量。JavaScript有一个预编译处理，首先对函数和变量进行预编译。也就是说其函数和变量作用域是在其声明的时候确定的，而不是在执行的时候确定。当setInterval把"fun()"执行环境换为全局的后，对fun的调用是无效的。因为全局不能访问局部的函数和变量。window.onload相对于window来说就是局部的。

eval() 中的代码执行于eval() 语句所处的作用域内,setTimeout的作用域为顶级作用域

setInterval(fun(),1000); //调用函数正常，setInterval调用出错

fun() 是对函数的直接调用，也就是说当setInterval还没有开始函数func就执行了。如果这个函数没有返回值或者返回值不是可执行的函数或者其他的代码的话，就以上代码而言只是弹出1，之后就遍报错了.

setInterval(fun,1000);  //正确方法

setInterval(匿名函数,1000);  //正确方法