## 在HTML中使用JavaScript

1. <script>标签

首先强调一点：我认为下载js文件之后就会立即解析，可以按照下载就是解析去做；

1. 具备属性asycn：

<script *asycn* = "asycn" *src* = "c.tet"></script>

<script *asycn* *src* = "a.js"></script>

只适用外部文件！为script应用这种属性后，script引入的脚本你就会立即下载，并且不影响其他浏览器的操作，也就是说，增加这个属性之后，页面会和这个js文件异步执行。但是不一定谁先执行的。而且如果多个script都设置了异步属性，则不一定谁会先执行，后面的script可能在前一个script之前执行。

另外，script的src可以引入外域的任何js文件，同时浏览器也不会检查js文件的扩展名，即使引入的是一个根本不存在的后缀文件，或者没有后缀的文件，都能成功引入并得到解析。但是最好不这样，这样的话就必须得保证服务器能返回一个正确的MIME类型。

c.tet文件的代码：

"use strict";

var con = document.getElementById("a").innerHTML;

alert(con);

HTML文件代码：

<body>

<p *id* = "a">晚上好JavaScript</p>

</body>

1. 具备属性defer:

<script defer = "defer" *src* = "c.tet"></script>

<script defer *src* = "a.tet"></script>

defer属性则是延迟执行，延迟到全部页面解析完毕之后执行，多个defer的script按顺序执行。缺点是必须等页面解析完了才能执行js，因此如果页面量太大用户可能会等很久才能解析Js。内容少的页面肯定是没问题的。

a.js:

"use strict";

var con = document.getElementById("a").innerHTML;

alert(con);

<script defer = "defer" *src* = "a.js"></script>

</head>

<body>

<p *id* = "a">晚上好JavaScript</p>

</body>



1. **defer、window.onload、$(window).load()、$(document).ready() 的区别**

区别肯定是有的,

onload事件在html文档中所有的节点都下载完成后执行，包括js,css,图片的资源完全下载后才执行。  
 如果js 设置了defer之后,js的解析执行在浏览器生成了html文档后执行，不包括图片的资源下载。

$(window).load()很明显和window.onload是一回事,都是得等html文档的一切，包括js，css图片什么的资源下载完才能执行。

$(document).ready()则会等DOM结构绘制完成执行，而无需等到图片或其他媒体下载完毕。

**问题来了，$(document).ready()与defer谁更优先？有待考证。。。**

1. 在XHTML中的用法

function compare(){

if(a < b){ // 在XHTML中这个地方会出现错误，< 会作为一个标签的开始，<后面不能接空格

*console*.log("a小于b");

} else if(a > b){

*console*.log("a大于b");

}

}

因此，如果想用<，就是用 &lt; 代替。

var a = 1,b = 2;

function compare(){

if(a &lt; b){ // 在XHTML中这个地方会出现错误，< 会作为一个标签的开始，<后面不能接空格

*console*.log("a小于b");

} else if(a > b){

*console*.log("a大于b");

}

}

compare();

但是如果希望在XHTML中像在HTML中使用js。那么就使用CData片段来包裹整个script标签中的所有内容。

<script>

var a = 1,b = 2;

<![CDATA[

function compare(){

if(a &lt; b){

*console*.log("a小于b");

} else if(a > b){

*console*.log("a大于b");

}

}

compare();

]]>

</script>

但是有些浏览器不支持XHTML，因而也不支持CData片段。为了在所有现代浏览器中使用，应该这样操作（就是把Cdata所在行注释掉）：

<script>

// <![CDATA[

var a = 1,b = 2;

function compare(){

if(a &lt; b){

*console*.log("a小于b");

} else if(a > b){

*console*.log("a大于b");

}

}

compare();

// ]]>

</script>

如果使用外部文件，就不会出现之前所说的所有问题，XHTML和HTML是一样的。

在XHTML文档中引入js文件所使用script标签的方式和html中有所不同，和link倒是很相似，是一个单标签。

<script *src* = "a.js" */*>

注意：我不知道是我声明XHTML文档的方式不对还是怎么，将DOCTYPE修改为XHTML的严格模式。浏览器似乎依旧按照HTML进行解析，也就是说，之前的 < 问题根本不会发生，写CDATA片段还会报错。标签名改为大写也不会出现问题，在控制台内还会显示小写的标签名。

我的声明方式是<!DOCTYPE *html*

*PUBLIC* "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

其他的同HTML。

我使用的是谷歌浏览器和360浏览器。

1. <noscript>元素

如果浏览器不支持或者没启用javascript，那么浏览器就会显示在noscript元素中的内容。其中可以包含除script标签外的所有body内存在的标签。

noscript元素应该写在body标签内。

<body>

<noscript>

你的浏览器不支持或者没有启用javascript.

</noscript>

<p *id* = "a">晚上好JavaScript</p>

</body>



关于如何关闭谷歌浏览器的javascript。详见百度教程：

https://jingyan.baidu.com/article/e75057f218bbd4ebc91a8928.html