JS是由EMACjavascript(编译器)、DOM(全名document object model )、BOM(全名 browser object moded)组成

JS中含有部分隐性转换 比如 == 和!==和 + 符号，你在使用这三个符号的时候浏览器可能会默认将符号两边的值转化为一样的类型，尽管你并没有要求这么做

类型命名法



JS中的真假判断



JSON

个人理解:json是一种可以定义下标的数组，它没有长度，向里面添加什么，他就有什么，没添加就什么也没有，特别想python里面的字典

用法为var 变量 = {下标1:内容，下标2:内容}

PS：json可以用for in来循环里面的内容

JS虽然可以使用style来修改CSS样式，但是对于样式表或外部样式表里的样式它其实是没法直接获取的，这个时候需要我们换一种方法来获取它，getComputedStyle是将计算机计算完的数值输出出来(因为一个页面可能有很多种因素控制一个元素的属性，计算机需要经过优先级计算)

TIP：JS中的很多兼容性问题都可以用if判断来解决

数组的length可以直接设置，设置为几就留几个内元素

Offsetwidth获取的是在页面当中元素的宽度，它是包括border以及padding在内的，并不是单纯的获取元素的width，以及类似的offset操作，也是这样。

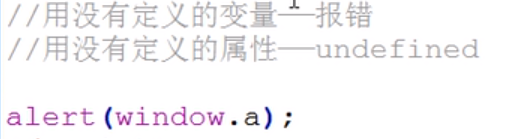
Document包括整个页面，他是html外面的层，也是一个页面最外面的层，js中如果有控制整个页面的操作可以通过他来实现

Var a = false || true 这种写法多用来处理兼容问题，它优先挑选后面两个选项它们中间正确的一项进行赋值

Return false 在事件中可以起到屏蔽浏览器bug的奇效

函数绑定事件或添加函数再或者赋值函数，函数不需要加后面的()，直接写名字就可以

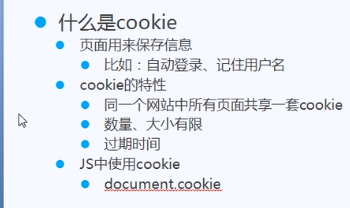
在设置一个元素的宽高时一定要加’px’



全局变量他其实是可以作window上的一个属性存在的，因为全局变量是创建在window上的，这样就会出现上面的现象 顺便说一下，这个全局alert也可以作为window的属性存在

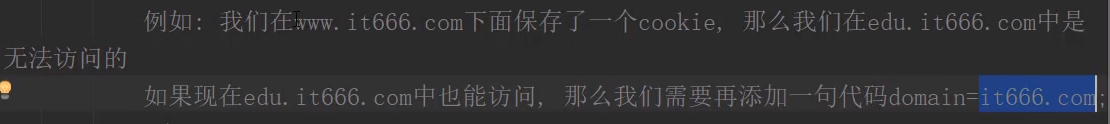
Cookie中的 = 为添加



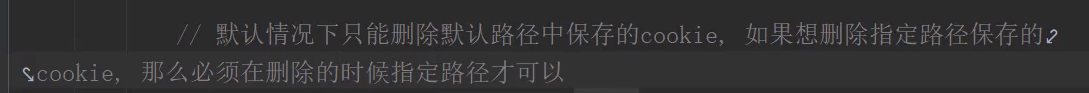


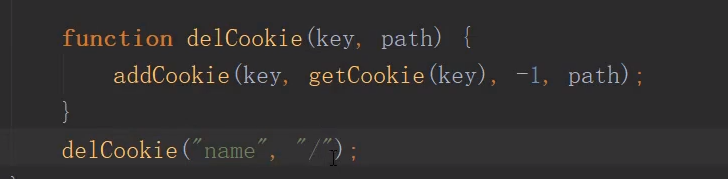
在同一浏览器同一路径(或下一级路径)定的前提下，可以访问其他文件存储的cookie

在添加cookie的时候加上一句path=/； ，就可以将cookie添加到根目录



(不确定)





**hash**

可以跨浏览器保留页面

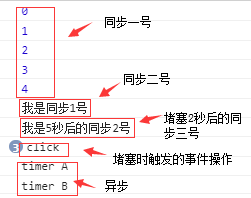
添加方法：window.location.hash = ……;

同方法获取

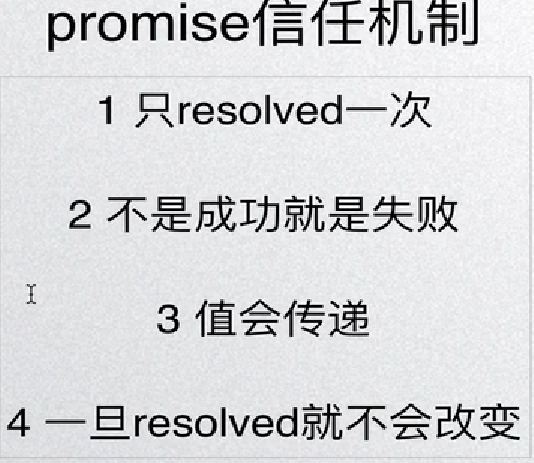
新词：平稳退化

尽可能的保证你的页面在脱离了JS还能够正常访问

JS会先执行完全部的同步操作，才会去执行异步操作，如果同步操作执行中间有事件操作，也会将其添加到异步操作之前



回调函数两大问题：

1. 控制反转、失去对程序的操控
2. 回调执行方式与人类思考方式存在差异
3. 

没声明和没定义是两种概念，但是typeof都会显示undefined，因为JS没有“未声明”这一类型

·typeof null 是object ，这是一个遗留下来的，并且不打算修复的BUG(PS：是无法修法)

NaN(Not a Number)在类型转换过程中，遇到不可转换的(比如“a”/2)数值，会出现NaN来代替(PS：用typeof来检验’a’/2最终会得到number类型)

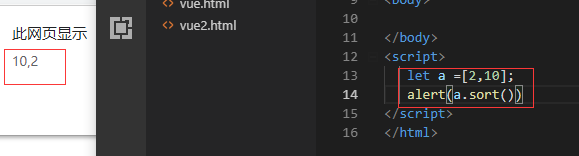
Window上的IsNaN方法在判断数值是否为NaN的时候，会先将判断的数值进行类型转换，只要无法转化为整型的数值会判定为NaN(比如判断一个字符串，因为无法转化为数字，所以结果为true)(PS：ES6中在Number类型上也添加了isNaN方法，不会出现上述的问题)

一个整型/0会返回infinite，/-0会返回-infinite

通过new String(bool、number)方式创建的值类型为object

通过new Array(只传一个值)会创建一个长度为这个值得数组，但是在循环中，这个数组的长度为0

Javascript中的sort()排序，在无callback参数的情况下，会默认将内容转换为string然后对比它们的ASCLL数码 ？？！！！

就比如这种情况

10排在2前面的原因是因为‘1’的ascll码比‘2’小