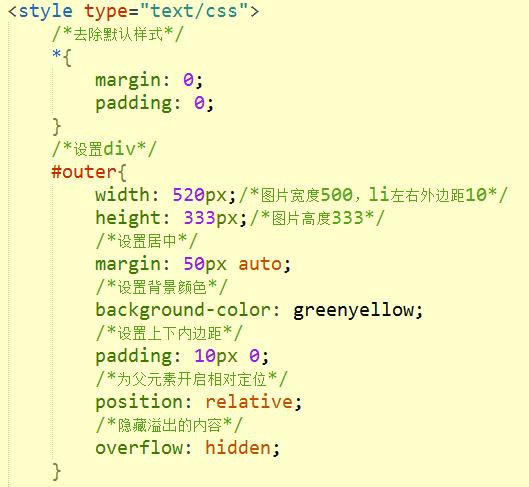
# 轮播图

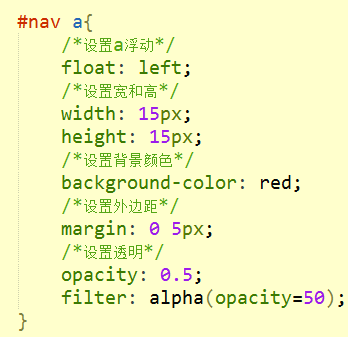
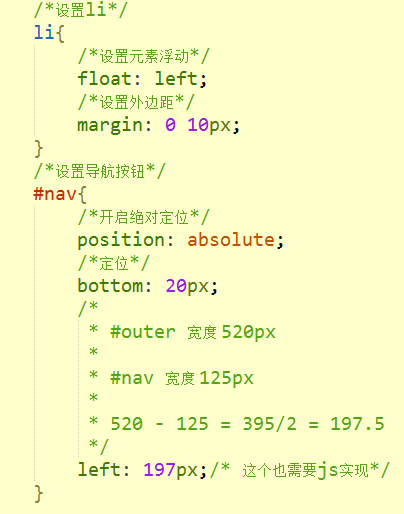
## 结构



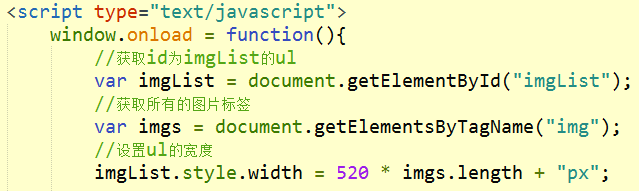
## 样式



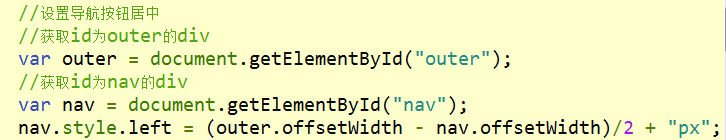


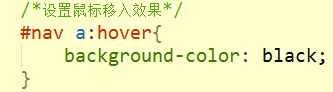


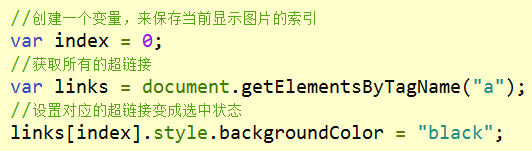
## 点击切换图片



根据图片数量来自动修改ul宽度







\* 点击超链接切换到对应的图片

\* 点击第一个超链接，切换到第一个图片

\* 点击第二个超链接，切换到第二个图片

//为所有的超链接绑定单击响应函数

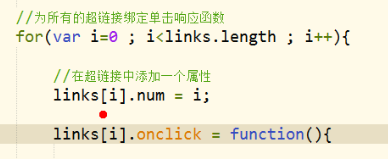
for(var i=0 ; i<links.length ; i++){

links[i].onclick = function(){

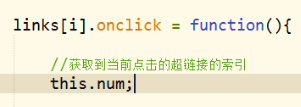
点击超链接切换到对应的图片，需要获取当前点击的超链接的索引，由于for循环先执行，这里i是for循环执行的最后一个值，不能使用i；而this输出的都是javascript:;，也不能判断到底是第几个超链接；可以在单击响应函数前在超链接中添加一个属性

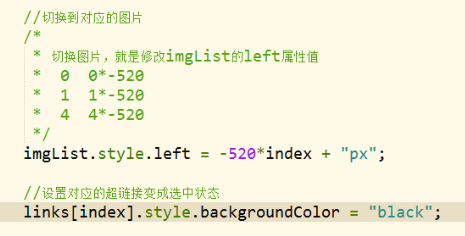
}

}



这样for循环每执行一次，都给各个超链接添加一个num属性，当i=0时，给第一个超链接属性num设置为0，当i=1时，给第二个超链接属性num设置为1，以此类推；这样在单击响应函数里通过this.num就可以获取当前点击的超链接的索引



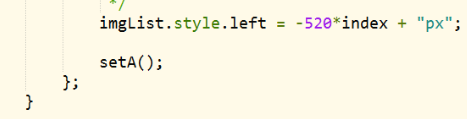
但每点一个超链接都会使背景颜色变为黑色，应该是当前点击的变成黑色，而其他的还是红色，可以定义一个函数，来设置超链接的选中状态



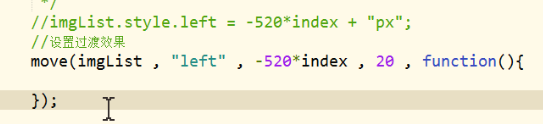
但这个将所有超链接的内联样式都设置成红色，:hover的样式不起作用了，



设置为空串后，相当于把内联样式的背景颜色删了，删了以后则会用原来的样式，这样再调函数就可以了

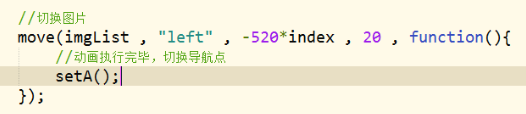


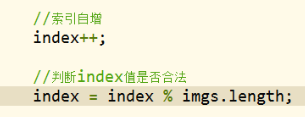
现在是点击哪个图片直接就跳转到哪个，需要一个过渡动画，利用move函数



## 自动切换图片



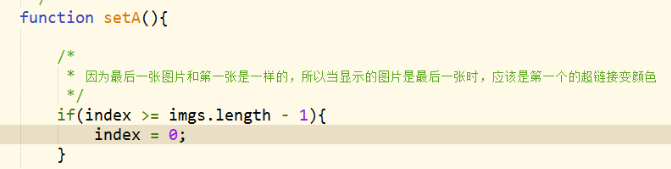




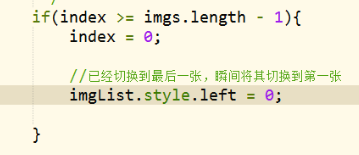
加了判断后，此时从最后一张切换到第一张图片会从最后一张缓慢倒着回退到第一张，而且图片切换时间只有3s，还没完全到第一张，又开始切换图片

可以在最后一张图片后再加一个第一张图片

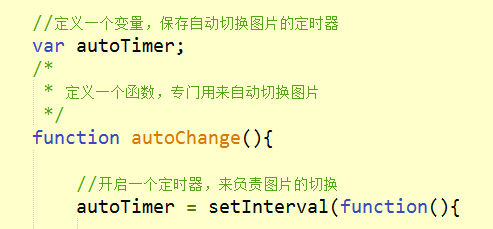


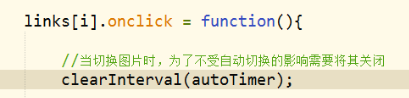


但现在还是缓慢回退到第一张，此时当切换到最后一张时，可以瞬间切换到第一张，由于第一张和最后一张相同，肉眼看不出来

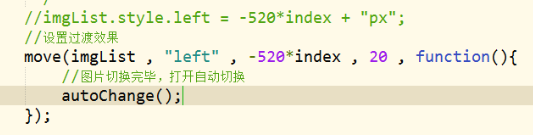


此时自动切换和手动点击切换执行效果互相影响，希望点击时自动切换停止

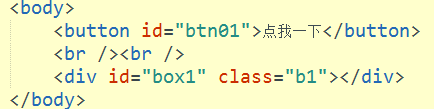


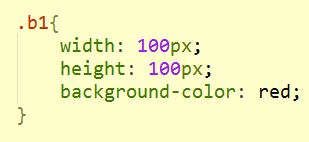


但点击完成，图片切换完毕后，图片不会再自动切换，需要在图片切换完成后，再打开自动切换



# 类的操作

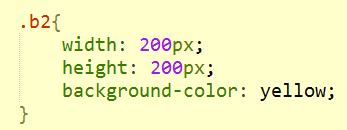


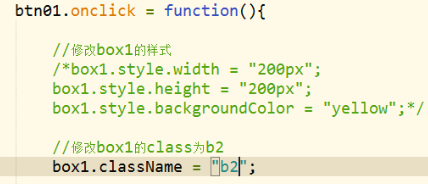




此时每修改一个样式，浏览器都要重新渲染一次，希望修改多个只渲染一次，而且当前每修改一个样式就需要一行代码，希望一行代码修改多个样式

此时box1的样式是通过class为b1设置的，可不可以修改class来修改样式





我们可以通过修改元素的class来间接的修改元素的样式，这样我们可以同时修改多个样式，并且提升浏览器渲染页面的性能

修改class也可以使行为和表现分离（js没有修改CSS），方便后期维护，第一次的修改是通过js去影响CSS，没有分离

但此时b2会覆盖原b1，希望只加入b2，而不覆盖原来的



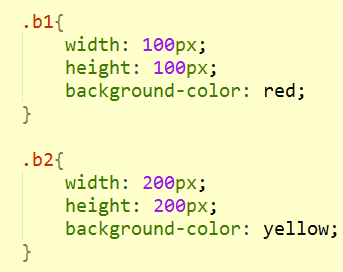


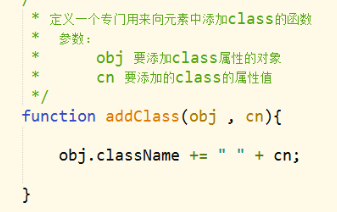


前面需要加一个空格

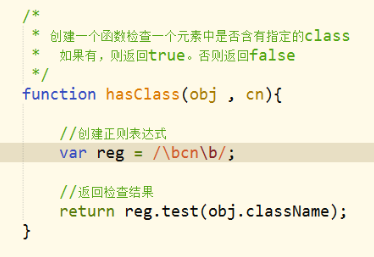


由于b2在b1下面，b2样式会覆盖b1样式

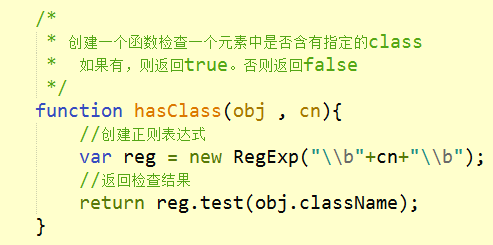


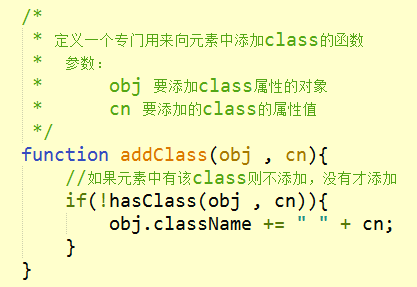


此时每点击一下都会添加一个相同的class属性值

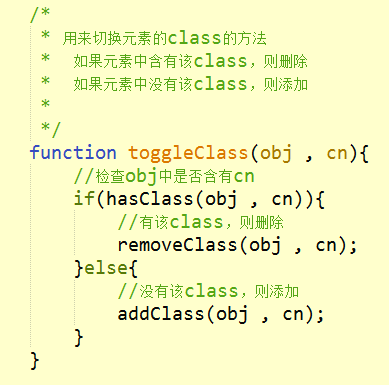


但正则表达式中的是检查有没有cn这种class属性值，这种写正则的方式一写就写死了，需要一种更加灵活的方式，









# 二级菜单

# JSON

JavaScript Object Notation - JS对象表示法

\* - JSON是表示数据的一种格式

\* - JSON实际就是一个字符串，但是JSON这个字符串的格式比较特殊

\* - JSON的语法和JS对象的语法基本一致，但是注意JSON中的属性名必须加双引号

\* - JSON可以被任意的语言任意的平台所识别，并且可以转换为任意语言的对象

\* 在开发中我们经常使用JSON来进行数据的交换，或者是编写配置文件

\* - JSON语法规范：

\* - JSON对象：'{ }'

\* - JSON数组：'[]'

\* - JSON中可以使用的数据类型

\* 1.字符串

\* 2.数值

\* 3.布尔值

\* 4.null

\* 5.对象，普通对象，函数不行

\* 6.数组

在JS中为我们提供了一个工具类，就叫JSON，通过该类可以帮我们将一个JS对象转换为一个JSON，也可以将一个JSON转换为一个JS对象

JS对象 ----> JSON

\* JSON.stringify()

\* - 该方法可以将一个JS对象转换为JSON

\* - 需要一个JS对象作为参数，可以将该对象转换为JSON字符串返回

JSON ----> JS对象

\* JSON.parse()

\* - 可以将一个JSON字符串转换为JS对象

JSON这个工具类在IE7及以下的浏览器中不支持，如果需要兼容这些老版本的浏览器，则可以引入一个外部的JS文件，来处理该问题