昨天

父级

parentNode

兄弟 nextSibling previousSibling

子节点 firstChild lastChild

孩子节点 childNodes children

创建节点 document.createElement(“li”)；

插入节点 appendChild insertBefore（插入的节点，参照节点）；

移出节点 removeChild()

克隆节点 cloneNode(ture&&false);

# 设置节点属性：

1.

获取节点属性

getAttribute(属性) 获得节点属性  
//可以获取属性，没法获取样式。例如href ,type，class，id,name

2. 设置节点属性

SetAttribute(“属性”，“值”)；

比如我们想把box的class由box1改成box2，那么就

setAttribute(‘clas’，‘box2’)

3.删除节点属性

removeAttribute(‘属性’)；

缓动动画

就是将匀速的动画变成由快到慢或者由慢到快的非匀速动画；

缓动动画可以通过缓动公式来达成，数学好的可以自己做都行，这里只是叫一个方法

其实就是等差数列

I= 10,10,10,10,10,10

等差数列

I=10,8,6,4,2,0；

一个盒子，他的初始值为0；要他缓动的走到500px的地方

初始值设为leader ,目标值设为target

Leader=leader+(target-leader)/10;

之前，我们讲过函数自己调用自己，叫做递归调用

今天，学习一个新东西，叫做回调函数，牢记 重要程度：\*\*\*\*\*\*\*\*\*\*

当前函数执行完之后，再执行另一个函数，就叫做回调函数

最简单的无缝轮播图

<script type="text/javascript">

var box=document.getElementById("box");

var ul = document.getElementsByTagName('ul')[0];

var ol = document.getElementsByTagName('ol')[0];

var btn1 = document.getElementsByTagName('button')[0];

var btn2 = document.getElementsByTagName('button')[1];

var timer=null;

var t=null;

var count=0;//当前显示的图片索引

var leader=0,target;

for(var i = 1; i < ul.children.length; i++) {

var li = document.createElement('li');

ol.appendChild(li);

}

var lis = ol.children;

lis[0].setAttribute('class', 'lis');

t=setTimeout(fun,1000);

function fun(){

count++;

if(count>4){

count=1;

leader=0;

ul.style.left='0px';

}

target=count\*-790;

timer=setInterval(animate(),1);

for(var l = 0; l < lis.length; l++) {

lis[l].removeAttribute('class');

}

count>3?lis[0].setAttribute('class', 'lis'):lis[count].setAttribute('class', 'lis');;

}

function animate(){

leader=leader+(target-leader)/10;

ul.style.left=leader+'px';

if(Math.round(leader)<=count\*-790){

clearInterval(timer);

t=setTimeout(fun,1000);

}

}

box.onmouseover=function(){

clearTimeout(t);

}

box.onmouseout=function(){

t=setTimeout(fun,1000);

}

for(var j=0;j<lis.length;j++){

lis[j].index=j;

lis[j].onmouseover=function(){

count=this.index;

t=setTimeout(fun,1);

}

}

</script>

优点，写得快，无缝轮播

缺点，只能自己轮播，无法控制

这种写法，只适用于自动轮播

真正做焦点轮播（就是可以通过小圆点和左右按钮控制的轮播，肯定要用节点做，而不是用一排图片做）；

元素.offsetWidth

获得元素的自身宽度

缓动公式2

Var speed;//步长

Speed=(目标值-本身)/8；

本身=本身+speed；