ECMAScript:翻译 核心 翻译器

DOM ： document object model HTML DOM 定义了访问和操作 HTML 文档的标准方法。

BOM： 浏览器对象模型（**B**rowser **O**bject **M**odel (BOM)）尚无正式标准。

1. 事件

Onmouseover\onclick\onmouseout

改变引入的css文件源：ol.href=”css.css”

1. 修改文字内容：input.value=”aasssd”;
2. If判断

if(odiv.style.display=="block"){

odiv.style.display="none";

}else{

odiv.style.display="block";

}

5.<a href=”javascript:;”>链接</a>//不会像#那样调到顶部

6. document.getElementById();

函数传参

1. 改变背景颜色

参数就是占位符，当有部分代码确定不下来的时候可以选择使用

function show(color){

var odiv=document.getElementById("div");

odiv.style.background=color;

}

1. 改变div任意样式 使用[ ]

function setStyle(name,value){

var odiv=document.getElementById("div");

odiv.style[name]=value;

}

Onclick=”setStyle(‘width’,’400px’);”

3．style 与classname

style:属性 style.background=’red’;

通过style加的样式都是加的行间样式，style取样式也是行间样式，意思就是style操作的都是行间样式

样式优先级：

通配符\*<标签<class<ID<行间

classname 。。先改了style就改不了classname了，建议选择一种统一方法来修改样式。

**提取行间事件**

匿名函数：没有名字的，直接用的函数

window.onload 整个页面加载完之后执行

行为js、结构html、样式css分离

**获取一组元素**

getElementsByTagName 获得到的是一个数组

**循环**

if(){} 只会执行一次

for(初始化；条件；自增){ 语句 }

while(条件){ 语句 }

**this**

当前发生事件的元素

选项卡中按钮的样式制作，可以先把样式清空，再给当前按钮加样式

**innerHTML**

设置标签里的文字，div.innerHTML=input.value; 利用这个制作小年历

**数组的使用**

定义arr=[1,2,3…]， 使用时： arr[0]

**字符串的链接**

作用：连接两个字符串；问题：注意连接中的优先级（使用括号）

**06**

**变量类型**

变量: typeof 运算符

常见的六个数据类型：number\string\boolean\function\object\undefined

alert(typeof a );//输出a的数据类型；

出现undefined会有两种情况：一是没有定义，二是定义了没有赋值；

**强制类型转换**

字符串转成数字：**parseInt()**; 工作原理是，从左到右扫，遇到非数字的东西就会跳出去

var a=”12dksdjs”;

parseInt(a); //结果为12

var a=”abc”;

parseInt(a); //结果为**NaN**  全称为Not a Number 非数字 ；NaN和NaN是不相等的

isNaN() 判断是否是一个NaN

**parseFloat()** 转成小数

**隐式类型转换（自己偷偷地就转了）**

==、===

减法

var a=5;var b=’5’;

alert(a==b); //输出为true 先转换类型，然后比较

alert(a===b); //输出为false 不转换类型，直接比

var a=’12’;

var b=’5’;

alert(a-b); //7 1.数字相减

alert(a+b); //125 1.字符串连接 2.数字相加

**变量的作用域和闭包**

局部变量和全局变量

闭包： 子函数可以使用父函数的局部变量

**命名规范**

命名规范及必要性

可读性：

规范性

匈牙利命名法

类型前缀：aDiv

首字母大写



07

**运算符**

算数：加减乘除取模

实例：隔行变色、秒转时间

赋值：。。。

关系：< 、 >、>=、<=、==、===、!=、!==

逻辑：&&与并且、|| 或、 ！否

运算优先级：括号

**程序流程控制**

1.if（）{}

2.switch（变量）{

case 值1：

语句1

break；

case 值2：

语句2

break;

…….

default:（可省）

语句n

}

1. 三目运算符 三元运算符

？：

if（条件）{

语句1

}

else{

语句2

}

相当于

条件？语句1：语句2

**break与continue**

break //整个循环中断

Continue//本次循环中断

**什么是真什么是假**

真的：true 非零数字 非空字符串 非空对象

假的：false 0 空字符串 null undefined

**Json**

什么是json:是用来存数据的

var json={a:12,b:5,a:'abc'}

alert(json.b);

Json和数组

//alert(json.a);

//alert(json['a']); //.可用[ ]代替但是里面是的字符串

alert(arr[0]);

Json没有length

循环json 使用 for ( var i in json ){ alert( json[i]); }

**08 深入javascript**

**函数返回值**

什么是函数返回值

函数的执行结果

也可以没有return

一个函数应该只返回一种数据类型

**函数传参**

可变参（不定参）：arguments

1.arguments是一个数组有长度，可利用for循环来实现求和

2.css函数

3.取非行间样式

oDiv.currentStyle.width; //获取非行间样式，IE

getComputedStyle(oDiv,null).width //第二个参数可以随便写，火狐浏览器

解决兼容问题：使用if判断，**封装成一个函数，**用的时候调用即可

只能取单一样式，没有办法取符合样式

复合样式：background、border…

单一样式：width、height ….

**数组基础**

定义

var arr=[1,2,3];

var arr=new Array(1,2,3);

两种方法没有任何差别，第一种使用多，因为代码短。

数组的属性

length：既可以获取，又可以设置。

例子：快速清空数组

使用原则：数组中应该只存一种类型的数据

**数组的方法**

push //尾部添加数组元素

pop //尾部删除数组元素

shift //头部删除

unshift //头部添加

**splice**(开始，长度，元素。。) 替换

splice（删除起点，删除长度） 用来删除元素

splice（起点，0，元素） 插入

concat a.concat(b); 数组的链接:

join 分隔符 a.join(‘---’);

sort a.sort();排序从小到大 只认识字符数字也会按字符串处理

sort有一个比较函数，可以解决这一问题

a.sort(function(){})

arr.sort(function(n1,n2){

if(n1<n2){

return -1;

}else if(n1>n2){

return 1;

}else{

return 0;

}

})

if判断也可使用**return n1-n2**；代替

09 **定时器的使用**

定时器的作用

开启定时器

setInterval（函数，时间）； 间隔型

setTimeout(函数，时间) 延时型，只执行一次

停止定时器

clearInterval（）

clearTimeout（定时器名称） 可以指定停止哪个定时器

**数码时钟**

效果思路

获取系统时间

date对象

getHours、getMinutes、getSeconds

显示系统时间

字符串连接

空位补零

设置图片的路径

charAt方法 //取字符串的某一位

不兼容IE7：不能获取str[i]

使用charAt解决兼容问题

**Date**

alert(oDate.getFullYear());

alert(oDate.getMonth()+1);

alert(oDate.getDate());

alert(oDate.getDay()); //0代表周日

延时提示框

setTimeout();

**无缝滚动**

物体运动基础

让div动起来 offsetleft//左边距

offsettop//上边距

写两个一样的ul是控制left值，判断offsetLeft值

**11 DOM基础**

**DOM基础**

什么是DOM document

浏览器支持

IE 10%支持 IE9之前

chrome 60%支持，不是那么的好也不是特别差

FF 最标准的浏览器 99%支持 第一次启动特别慢

**DOM节点**

子节点 **childNodes** 用来获取子节点 ，获取的子节点包括文本节点和元素节点

可使用**nodeType**来判断获取的是文本节点还是元素节点

**children** 获取的只有元素节点，比上面的方法简单，没有兼容问题

父节点

parentNode 获取父节点

例子：点击链接，隐藏整个li

offsetParent 获取元素用来定位的父级

例子：获取元素在页面上的实际位置

**DOM节点**

首尾子节点

有兼容问题

firstchild 兼容IE低版本浏览器 可以利用if 判断解决兼容问题

firstElementChild 兼容所有浏览器

lastChild

lastElementChild

兄弟节点

有兼容问题

nextSibling、nextElementSibling

previousSibling、previousElementSibling

**操作元素属性**

第一种：oTxt.value="adsdsdadad";

第二种： oTxt['value']="dddd";

第三种：DOM方式： oTxt.**setAttribute**('value','dsdsadsdasd');

一般是用不到的，特殊情况会用到

获取：getAttribute（属性名称）；

设置：setAttribute（属性名称，属性值）；

删除：removeAttribute（属性名称）；

DOM元素灵活查找

用**className**选择元素

**DOM操作应用**

创建DOM元素

需要两个步骤：一，创建一个元素createElement(“li”)；二，添加给父级appendChild(“li”)

插入到标签的末尾

插入元素

insertBefore(节点，原有节点 ) 在已有元素前插入

例子：倒叙插入li

删除DOM 元素

removeChild（节点） 从父级上移除子节点

**文档碎片**

文档碎片可以提高DOM操作性能（理论上） 事实上，几乎看不出来差距

var oFrag=document.createDocumentFragment(); //创建一个文档碎片

**DOM操作高级应用**

**表格应用**

获取：tBodies、tHead、tFoot、rows获取行、cells获取单元格

隔行变色：鼠标移入高亮（之前用过if判断i%2）

添加、删除一行

**搜索**

**可以忽略大小写**：把所有的转化成小写字母比较 .toLowerCase()

**模糊搜索**：search的使用 search(结果中包含的字符串)

sTab.toLowerCase().search(sTxt.toLowerCase())!=-1

=-1说明没找到

**多关键字：split 切割字符串 split（“ ”）以空格为切割点**

把输入的字符串切割成一个关键字数组，循环数组，把有关键词的内容选出来

**高亮显示，筛选**

高亮显示：背景颜色发生改变（上面的搜索例子都是用高亮显示的效果）

筛选：将结果设置display：block; 非结果里的设置成display：none；

**排序**

移动li 使用比较函数sort（function（n1,n2）{ return n1-n2;}）

**表单应用**

**表单基础知识**

什么是表单：向服务器提交数据，比如用户注册

action :提交到哪里

表单事件

onsubmit 提交时发生

onreset 重置时发生（清楚数据）

表单内容验证

阻止用户输入非法字符 阻止事件

输入时、失去焦点时验证 onkeyup onblur

提交时检查 onsubmit

后台数据检查

15. **js运动基础**

**运动基础**

让div动起来

**运动框架及应用**

运动框架 （意思大概是运动流程？？？）

在运动开始时，关闭已有的定时器 clearInterval( );

把运动和停止隔开（if/else）

**运动框架实例**

例子1“分享到”侧边栏 （侧边栏的收缩运动）

例子2 淡入淡出的图片（透明度）

**缓冲运动**

逐渐变慢，最后停止（速度和距离成反比） var speed=(300-oDiv.offsetLeft)/10;

速度不能是一个小数，计算机会自动舍去小数点后面的数，比如291.5px，会被计算机认为是291px;

Math.ceil() //向上取整 Math.floor() //向下取整

距离越远速度越小

速度由距离决定

速度=（目标值-当前值）/缩放系数

例子：缓冲菜单

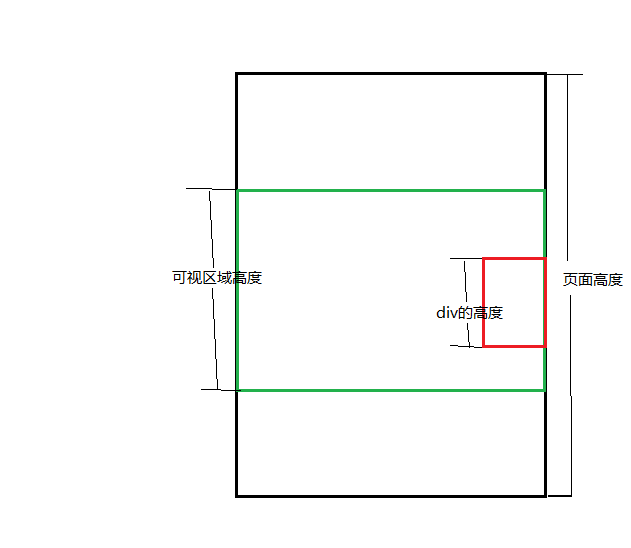
bug：**速度取整**

**跟随页面滚动的缓冲侧边栏**

潜在问题：目标值不是整数时(会出现抖得情况，取整就好了)

对联悬浮框，会出现抖动的现象，因除以2之后会有小数点，所以了把算出来的目标值取整（采用parseInt()）

startMove(parseInt((document.documentElement.clientHeight-oDiv.offsetHeight)/2)+scrollTop);



匀速运动的停止条件

匀速运动：距离足够近时停止（Math.abs(iTarget-oDiv.offsetLeft)<=7）

Math.abs() 取绝对值函数

缓冲运动：两点重合

**17 js运动应用**

多物体运动框架

多个物体同时运动

例子：多个物体同时变宽

单个定时器，存在问题

每个div 一个定时器

多个div同时运动时，可给每个div加定时器，如果使用同一个定时器会造成当前div变化，之前div没有返回默认状态，

aDiv[i].timer=null; //给每一个div加一个定时器,

clearInterval(obj.timer); //清楚定时器 各用各的定时器，互不影响

多个物体运动的程序中，不出现公用的属性，否则会出现自相矛盾的情况，例子：多个图片的淡入淡出

任意值的运动框架

offset属性的bug

有边框值的div变窄时：会出现变宽的现象，原因是style.width操作的是行间样式offsetwidth值会加上border值再赋值给width就是不断的增加了

用currentStyle代替offset

有bug不能修改opacity值

Math.round(3.4) -- 3 四舍五入值 不会出现0.300000000015这种的

例子：仿Flash图片展示-1

效果思路

两边的按钮:淡入淡出

1. **js运动初级**

链式运动框架

完美运动框架

运动框架总结

**链式运动框架**

回调函数

运动停止时，执行函数 startMove(oDiv,”width”,300,function(){前面运动停止后要执行的函数})

**完美运动框架**

多个值同时变化(可以让任意的属性同时变化)

getStyle同时设置多个属性值

参数传递

json的使用 json={a:1,b:2}

for in 遍历

**运动框架总结**

运动框架总结过程

startMove(iTarget) 运动框架

startMove(obj,iTarget) 多物体

startMove(obj,attr,iTarget) 任意值

startMove(obj,attr,iTarget,fn) 链式运动

startMove(obj,json) 多值运动

startMove(obj,json,fn) 完美运动框架

**20 JS事件**

**event对象和事件冒泡**

什么是event事件对象

用来获取事件的详细信息：鼠标事件、键盘按键

event.clientX 获取横坐标 //IE

event.clientY 获取纵坐标 //IE

ev.clientX 获取横坐标 //FF

document的本质：document.childNodes[0].tagName //获取第一个子节点 是！

**获取event对象（兼容性写法**）

var oEvent=ev||event;

oEvent.clientX;

**事件流**

事件冒泡 ：从里到外依次执行，子级事件会传递给父级的父级直到html

取消冒泡：oEvent.cancelBubble=true;

例子：仿select控件

DOM事件流

**鼠标事件**

可视区位置：clientX、clientY (当页面出现滚动条时，位置是可视区内的位置)

例子：跟随鼠标的div onmousemove

消除滚动条带来的影响：加上可视区域页面顶部的距离

var scrollTop=document.documentElement.scrollTop||document.body.scrollTop;

获取鼠标在页面的而绝对位置

封装函数

例子2：一串跟随鼠标的div

**键盘事件**

keyCode

获取用户按下按下键盘的哪个键

例子：键盘控制div移动

Onkeydown

Onkeyup

document.onkeydown=function(ev){

var oEvent=ev||event;

alert(oEvent.keyCode); //打出按的哪个键

};

其他属性

ctrlKey、shiftKey、altKey

例子：提交留言

回车提交

Ctrl+回车提交

**JS事件中级**

默认行为：浏览器自带的行为（如右键菜单）

document.oncontextmenu 右键菜单事件

**阻止默认行为**

普通写法：在菜单事件中写入 return false；

例子1：屏蔽右键菜单

弹出自定义右键菜单

例子2.只能输入的数字框

keydown keyup

**拖拽**

鼠标和左上角的距离不变

鼠标按下去-----弹起来

三个事件 onmousedown、onouesemove、onmouseup

靠谱拖拽

原有拖拽的问题

直接给document加事件

防止拖出页面

l>document.documentElement.clientWidth 获取可视区宽度

**JS事件高级应用**

**事件绑定**

window.onload=function(){} 只能有一个，多了的话最后个起作用

可以使用事件绑定的方法解决这个问题

attachEvent(事件名， 函数名) **添加事件** 兼容IE

addEventListener(事件名，函数，false/true) 兼容FF

解决兼容问题，可以通过if判断

detachEvent(事件名，函数); **解除绑定**

removeEventListener(事件名，函数，false/true)

**高级拖拽**

复习拖拽原理

距离不变

三个事件：onmousedown onmousemove onmouseup

限制范围

对位置进行判断

例子1：不能脱出窗口的div

例子2：不能拖出指定对象的div

磁性吸附 ：就是判断一定距离内直接设置为一定值

**事件捕获**

oBtn. setCapture() 事件捕获作用就是把所有事件捕获后都添加到一个按钮上去 //只兼容IE

可利用事件捕获**解决拖拽时文字被选中的问题**：因为加上事件捕获后文字图片的获取不到页面的事件了都添加给了被拖拽的div

odiv.releaseCapture() 解除事件捕获

**自定义滚动条**

拖拽

只有横向拖拽

限制范围----范围的大小

计算比例---当前值/最大值

控制其他对象

例子1：控制物体的大小

例子2：控制物体的透明度

例子3：控制文字滚动

**AJAX基础**

什么是服务器

网页浏览器分析

如何配置自己的服务器（AMP）：利用wamp配置一个自己的服务器

文件放在根目录下的www文件下

访问时 localhost：//aaaa.html或是127.0.0.1 也是本地地址

什么是ajax

无刷新获取数据

读取服务器数据

使用ajax

基础：请求并显示静态text文件

字符集编码：将文本文件的字符集编码选择为utf-8 统一一个就可以

缓存、阻止缓存

动态数据：请求JS(json)数据

AJAX中级

创建ajax对象

链接到服务器

发送请求

接收返回值

//1创建Ajax对象

var oAjax=new XMLHttpRequest(); //只是不兼容IE6

var oAjax=new ActiceXObject(“Microsoft.XMLHTTP”); 兼容IE6

在js里 用没有定义的变量-----浏览器会报错

用没有定义的属性------undefined

//2链接服务器

oAjax.open(‘方法’ ，文件名，异步传输);

oAjax.open('get','a.text',true)

同步：一件一件的来

异步：多件事可以一起处理

ajax处理方式为异步传输 所以open（）的参数是true

//3.发送请求

oAjax.send();

//4.接收返回值 （数据接收会触发一个事件）

oAjax.onreadytatechange=function(){ };

//oAjax.readyState //浏览器和服务器进行到哪一步了

oAjax.readyState==0 （未初始化）还没有调用open方法

oAjax.readyState==1 （载入） 已经调用send 方法

oAjax.readyState==2 （载入完成）send（）方法已经完成，已收到全部响应内容

oAjax.readyState==3 （解析） 正在解析响应内容

oAjax.readyState==4 （完成） 响应内容解析已完成，可以在客户端调用了

oAjax.responseText //获取服务器获取来的文本

Ajax数据

什么叫数据类型---英语、中文

XML（浪费资源，很老很老）、json

字符集

所有字符集相同

**26 . JS面向对象基础**

**什么是对象**

对象：黑盒子，不了解内部的结构，知道表面的各种操作（按钮）

类似电视：用户知道怎么用，但是可以不知道原理

**不了解原理，会使用功能**

**Date（） getFullYear()**

面向对象时一种通用思想，并非只有编程中能用，任何事情都可以用

**JS中的面向对象**

面向对象编程的（OOP）的特点

抽象：抓住核心问题，把与问题相关的问题抽出来

封装：不考虑内部实现

继承：从已有对象上，继承出一些方法和属性，子类，又有一些自己的特性

多重继承：从多个父类继承

多态

对象的组成

方法------函数：过程、动态的

属性------属性：状态、静态的

this

当前发生事件的对象（之前）

当前的方法属于谁，this就是谁

不能在系统对象中随意的添加属性或是方法，否责会覆盖已有的方法

object

没有功能，是一个空白的东西，所以在编写面向对象的程序是用的是new Object;

构造函数：在工能上解释为构造一个对象

prototype 原型 类似class

**面向对象实例**

面向对象的选项卡

面向过程的选项卡转换为面向对象的选项卡

原则：不能有嵌套函数，但可以有全局变量

过程：

onload----构造函数

全局变量---属性

函数----方法

改错

this 事件 闭包 传参

**JS面向对象高级**

Json方式的面向对象

把方法包在一个Json里

用json 实现：简单，但是不适合多个对象

单体

命名空间：可以让很多相同名字的函数同时存在

var json={};

json.conmmon={};

json.fx={};

json.site={};

json.conmmon.getUser=function(){

alert('a');

};

json.fx.getUser=function(){

alert('b');

};

json.conmmon.getUser();

**继承：继承函数/方法**

**a.call**(this)

function A(){

this.abc=12;

}

A.prototype.show=function(){

alert(this.abc);

};

//继承A 用call

function B(){

//this ---是-new了一个B()放到A里

A.call(this);

}

B.prototype=A.prototype; //继承A的方法

var obj=new B();

//alert(obj.abc);

obj.show();

**30． BOM应用**

用来操作浏览器

**BOM基础**

打开、关闭窗口

onclick='window.open("https://www.baidu.com/");'

document.write(); 先清空再写入，会把原来的全部清除掉

window.close(‘index.html’); //只能关闭 自己打开的页面

window.navigator.userAgent //显示浏览器版本信息

window.location //当前网页的地址，还可以赋值，

尺寸及坐标

窗口尺寸、工作区尺寸

可视区尺寸

对于IE chrome ff opera safari

window.innerHeight;

window.innerWidth;

对于IE8/7/6/5

document.documentElement.clientWidth //和内容无关，只和窗口大小有关

document.documentElement.clientHeight //可视区的高度

或者document.body.clientHeight;

滚动距离

document.body.scroll //兼容谷歌浏览器

document.doucmentElement.scrollTop //兼容IE和火狐

处理兼容，用一个||即可

window.onscroll //当页面滚动的时候发生

window.onresize //当窗口大小改变时发生

**常用的方法和事件**

系统对话框

警告框：alert() 没有返回值

选择框：confirm（‘提问的内容’） 返回boolean

输入框：prompt('请输入你的姓名','默认文字'); 返回输入的字符串

window对象常用事件

onload

onscroll

onresize 窗口改变的时候发生

**COOKIE基础与应用**

什么是cookie

页面用来保存信息

比如：自动登录、记住用户名

cookie的特性

同一网站中所有页面公用一个cookie

数量、大小有限

过期时间

js中使用cookie

document.cookie

设置cookie 设置周期

使用cookie split各种切割存入的字符串，来获取值

删除cookie 直接让cookie过期 -1；

**js中的正则表达式**

**正则表达式基础**

复习字符串的操作

search 查找 str.search(‘a’) 返回的是搜索的字符的位置

substring(2,3) 截取字符串，第一个参数是开始位置，第二个参数是结束位置，结束位置不包含在结果里面

charAt(); 返回字符串的某一位

split() ; 字符串切割 返回一个数组

正则表达式

alert(str.match(/\d+/g)); 找出所有的数字

什么叫正则：就是规则、模式，计算机可以理解的规则

强大的字符串匹配工具

是一种人类很难读懂的文字

RegExp对象

JS风格—new RegExp(); 新建一个对象

Perl风格-----/ a/

var re=new RegExp('a','i'); //最简单的正则表达式,就是代表a , i是忽略大小写

**search**

字符串搜索

返回出现的位置

忽略大小写---i ignore

判断浏览器的类型

**match**

获取匹配的项目，把所有匹配的东西返回出来

量词：+ ----个数，类似于若干，多少都可以

量词变化：\d、\d\d回头\d+

全局匹配：g---global 找到全部的结果

例子：找出所有数字

/\d/g -----找出单个数字

/\d\d/g -----找出两个数字挨着的

/\d+/g ----挨着的数字无论是几个都出来了

**replace**

替换所有匹配

返回替换后的结果

例子:敏感词过滤

元字符 **方括号**

[abc]pc 或的关系----apc bpc cpc

[a-z] a到z

[0-9] 0到9 数字

[^a-z] 排除，除了a到z

[^a-z0-9] 除了a-z和0到9

过滤HTML标签

<.+> 任意字符 +代表不限制个数

< [^<>]+> 除了尖括号 之外的所有字符

常用的转义字符

\d 数字 [0-9]

\D 相当于 [^0-9]

\w 英文 数字 下划线 [a-z0-9]

\s 空白字符

**量词**

简单的量词+

个数的意思

**{n}** 正好出现n次

**{n,m}** 至少n次最多m次

？ 相当于{0,1}可有可无的意思

\* { 0.} 不推荐使用, \*代表0次或多次的

电话号码的规则 ：\d{8} 只有八个数字， [1-9]\d{7}完整的写法，开头不能为0

qq号： [1-9]\d{4,10} 开头不是0 最少5位最多11位

可有可无的区号：（0\d{2,3}-）?[1-9]\d{7}

例子

邮箱校验

一串英文、数字、下划线 @一串英文、数字 . 一串英文

w+ @ [a-z0-9]+ \. [a-z]+

re.test() 方法 结果是一部分符合要求就会返回true

^ 不在方括号里 代表行首

$ 代表行尾

var re=/^\w+@[a-z0-9]+\.[a-z]+$/i;

**乘法表**

for(var i=1;i<10;i++){

for(var j=1;j<=i;j++){

document.write(j+"\*"+i+"="+ j\*i+" ");

}

document.write("<br/>")

}

两个变量，怎么样不利用第三个变量，交换值

a=1;b=2;

a=a+b;

b=a-b;

a=a-b;