day22-JavaScript基础

学习目标

1. 能够说出五种原始的数据类型
2. 能够使用JS中常用的运算符
3. 能够使用JS中的流程控制语句
4. 能够在JS中定义命名函数和匿名函数
5. 能够使用JS中常用的事件
6. 能够使用window对象常用的方法
7. 能够使用location对象常用的方法和属性
8. 能够使用history对象常用的方法

一,JavaScript基础

1.JS基本概念

1.1什么是JavaScript

JavaScript是运行在浏览器端的脚本语言，它不需要编译,通过浏览器解释就可以执行. 它的解释器被称为JavaScript引擎，为浏览器的一部分，广泛用于客户端的脚本语言

Jan 2018	Jan 2017	Change	Programming Language	Ratings	Change
1	1		Java	14.215%	-3.06%
2	2		C	11.037%	+1.69%
3	3		C++	5.603%	-0.70%
4	5	^	Python	4.678%	+1.21%
5	4	v	C#	3.754%	-0.29%
6	7	^	JavaScript	3.465%	+0.62%
7	6	v	Visual Basic .NET	3.261%	+0.30%
8	16	^^	R	2.549%	+0.76%
9	10	^	PHP	2.532%	-0.03%
10	8	v	Perl	2.419%	-0.33%

1.2JS的作用

- HTML与用户没有交互的功能，网页只能看，不能操作。. JavaScript用来制作web页面交互效果，提升用户体验。

  web前端三层来说：

  结构层　HTML：　　从语义的角度，描述页面结构

  样式层　CSS　：　　从审美的角度，美化页面

行为层　JavaScript：　从交互的角度，提升用户体验

- html当做毛坯房, css当做装修, js当做智能家居

1.3Java和JS比较

特点	Java	JavaScript
面向对象	完全面向对象	基于对象，有些面向对象的特性没有。
运行方式	编译型，生成中间代码，字节码文件	解释型，运行一行就解释一型，不会生成中间文件。
跨平台	安装 JVM 就可以在任何系统上运行	运行在浏览器中，只要系统有浏览器就可以执行。
大小写	区分大小写	区分大小写
数据类型	强类型	弱类型

1.4JS的组成部分

ECMAScript，描述了该语言的语法和基本对象。[2]

文档对象模型（DOM），描述处理网页内容的方法和接口。[2]

浏览器对象模型（BOM），描述与浏览器进行交互的方法和接口。[2]

javascript组成

- ECMAScript核心: js基本语法,数据类型,语句,函数(方法)...
- DOM:定义了一组操作文档(HTML)的方法和接口. 操作HTML
- BOM:定义了一组和浏览器相关的方法和接口. 说白了就是控制浏览器的

2.JS和HTML的整合

如果需在 HTML 页面中插入 JavaScript，请使用 会告诉 JavaScript 在何处开始和结束。 之间的代码行包含了 JavaScript. 常见的方式有两种:

2.1内嵌式

- 通过script标签即可,可以放在任意位置.

```
<script>
        alert("哈哈哈");
</script>
```

2.2外联式

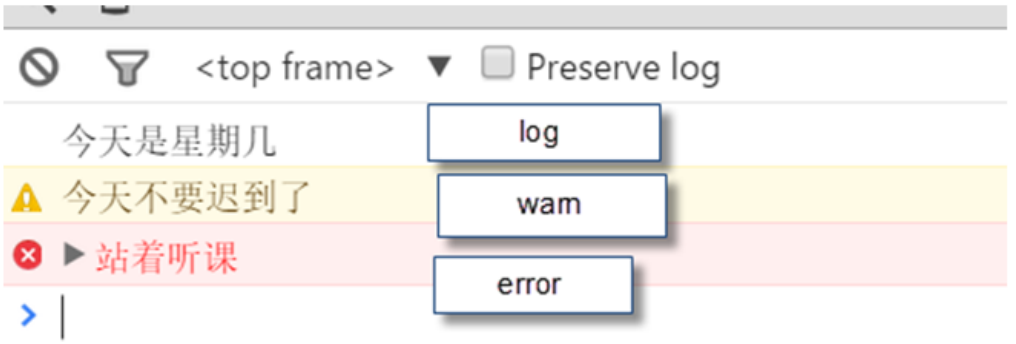
- 定义一个js文件,扩展名是js
- 通过script标签引入

```
<script type="text/javascript" src="../../js/test.js" >
        //js代码不会执行(不要写代码)
</script>

属性:
        src:js文件路径
注意:
        通过外部引入,script标签里面的js代码不会执行
```

3.体验JS常用的小功能

- alert(): 弹出警示框
- console.log(): 向控制台打印日志

控制台输出	
console.log()	控制台输出正常语句
console.warn()	控制台警示框
console.error()	控制台错误提示



- document.writte(); 文档打印. 向页面输出内容. 这个语句 主要就是给用户看的

4.JS基本语法

4.1变量

- JavaScript 是一种弱类型语言，javascript的变量类型由它的值来决定。 定义变量需要用关键字 'var'

```
var 变量名 = 值;


注意:
        1.var可以省略不写,建议保留
        2.最后一个分号可以省略,建议保留
        3.同时定义多个变量可以用","隔开，公用一个‘var’关键字. var c = 45,d='qwe',f='68';
```

4.2数据类型

1.五种原始数据类型

关键字	说明
number	数值型：整数、浮点
boolean	布尔类型，包含：true/false
字符串	包含字符和字符串，既可以使用双引号又可以使用单引号
object	对象类型，如：Date
undefined	未定义类型，一个变量没有赋值之前的状态

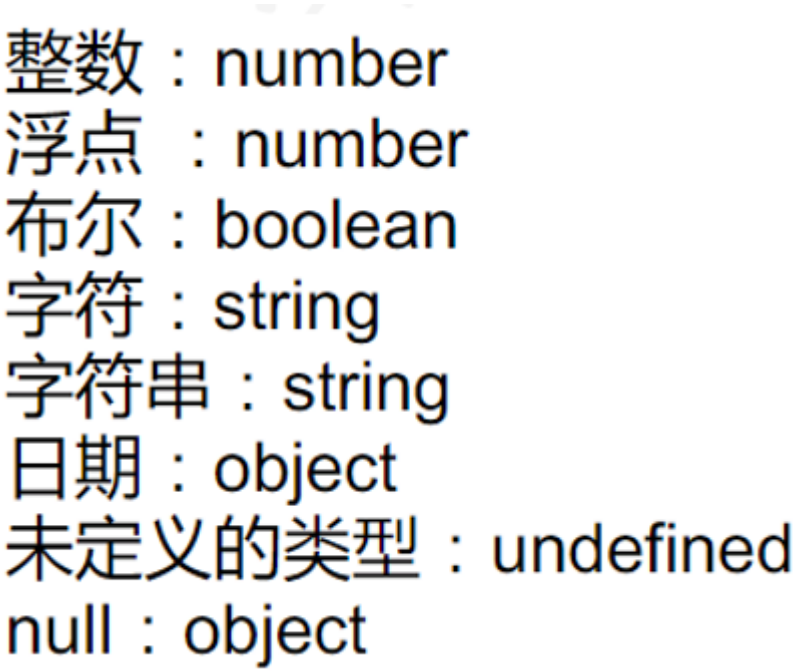
2.typeof操作符

- 作用： 用来判断变量是什么类型
- 写法：typeof(变量名) 或 typeof 变量名
- null与undefined的区别：

  null: 对象类型，已经知道了数据类型，但对象为空。
  undefined：未定义的类型，并不知道是什么数据类型。

3.小练习

- 定义不同的变量,输出类型,



- 代码

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
</head>
<body>
<script type="text/javascript">
    var i = 5;   //整数
    var f = 3.14;  //浮点
    var b = true;  //布尔
    var c = 'a';  //字符串
    var str = "abc";   //字符串
    var d = new Date();  //日期
    var u;   //未定义类型
    var n = null; //空
    document.write("整数: " + typeof(i) + "<br/>");
    document.write("浮点 : " + typeof(f) + "<br/>");
    document.write("布尔: " + typeof(b) + "<br/>");
    document.write("字符: " + typeof(c) + "<br/>");
    document.write("字符串: " + typeof(str) + "<br/>");
    document.write("日期: " + typeof(d) + "<br/>");
    document.write("未定义的类型: " + typeof(u) + "<br/>");
    document.write("null: " + typeof(n) + "<br/>");
</script>
</body>
</html>
```

4.2字符串转换成数字类型

- 全局函数(方法)，就是可以在JS中任何的地方直接使用的函数，不用导入对象。不属于任何一个对象

转换函数	作用
parseInt()	将一个字符串转成整数，如果一个字符串包含非数字字符，那么 parseInt 函数会从首字母开始取数字字符，一旦发现非数字字符，马上停止获取内容。如果转换失败，则返回 NaN=Not a Number，不是一个数。
parseFloat()	将一个字符串转成小数，转换原理同上。
isNaN()	转换前判断被转换的字符串是否是一个数字，非数字返回 true　isNaN = is not a number

4.3运算符

- 关系运算符:> >= < <=
- number类型和字符串做-,*,/的时候,字符串自动的进行类型转换,前提字符串里面的数值要满足number类型

```
var i = 3;
var j = "6";
alert(j-i);//结果是3, "6" ==> 6
alert(j*i);//结果是18,
alert(j/i);//结果是2,
```

- 除法,保留小数

```
var i = 2;
var j = 5;
alert(j/i);
```

- `==` 比较数值, `===` 比较数值和类型

```
var i = 2;
var j = "2";
alert(i==j); // ==比较的仅仅是数值, true
alert(i===j); // ===比较的是数值和类型.false
```

4.4语句

- for循环

```
  //99乘法表
  <script>
    for(var i = 1; i<=9 ; i++){
        for(var j =1; j <= i;j++){
                document.write(j+""+i+"="+ji);
                //空格
                document.write(" ");
        }
        //换行
    document.write("<br />");
   }
 </script>
```

- if... else

```
var a = 6;
if(a==1)
{
    alert('语文');
}
else if(a==2)
{
    alert('数学');
}
else
{
    alert('不补习');
}
```

- switch

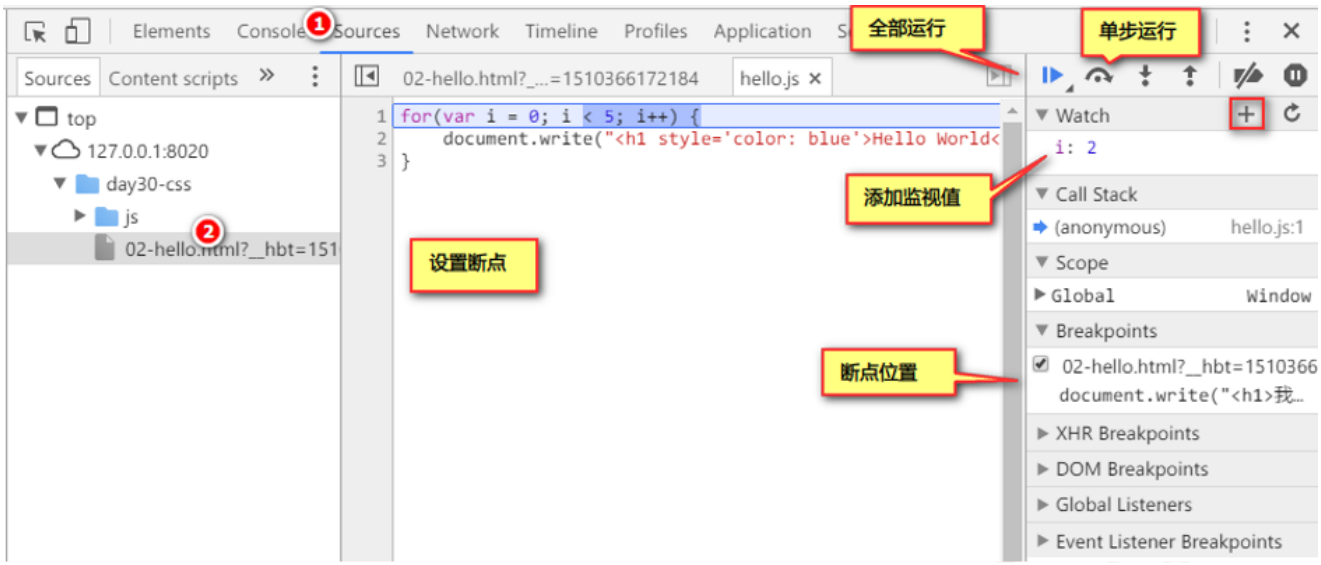
```
<script>
        var str = "java";

        switch (str){
                case "java":
                        alert("java");
                        break;
                case "C++":
                        alert("C++");
                        break;

                case "Android":
                        alert("Android");
                        break;  }
</script>
```

5.在浏览器中的调试

几乎所有的浏览器都支持JS代码的调式，IE、Chrome、FireFox中调试的快捷键：F12

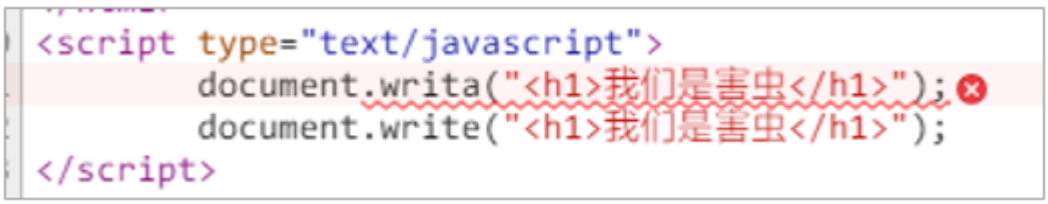
5.1设置断点

- 注：设置断点以后要重新刷新页面才会在断点停下来



5.2出现错误

- 如果出现错误, 有些浏览器会出现提示



案例一: 使用JS完成简单的数据校验

一,案例需求

- 点击注册按钮.判断用户名是否为空,如果为空,给用户一个提示,不让用户提交

二,技术分析

1.JS函数【重点】

- 函数就是重复执行的代码。

1,1 有函数名的函数

```
语法:
function 函数名(参数列表){
  函数体
}

注意:
        1.不管有没有返回值,函数格式是一样的 function 函数名(参数列表){函数体}
        2.如果有参数,参数不需要加var关键字(不需要加类型)
        3. JS中函数是没有重载的,后面的会把前面给覆盖掉
```

1.2 匿名函数

```
//匿名函数(通常和事件绑定一起用)
var sum = function(a,b) {
  return a+b;
}

console.log(sum(10,20));
```

2.获取元素(标签)方法

可以使用内置对象document上的getElementById方法来获取页面上设置了id属性的元素，获取到的是一个html对象，然后将它赋值给一个变量，比如：

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>

</head>
<body>

    <input id="inputId" type="text" value="哈哈"/>

</body>
<script>
    //获得input输入框标签(对象)
    var inputEle =  document.getElementById("inputId");
    // 获得value
    var  str = inputEle.value;
    console.log("str="+str);

</script>

</html>
```

3.点击事件

- 方式1:通过标签的事件属性绑定

```
<input  type="button" onclick="函数名(参数列表)" />
```

- 方式2:派发事件(注意:页面加载成功之后才可以派发)

```
元素对象.onclick=function(){};  元素对象可以通过:document.getElementById("id值")来获得;
```

三,思路分析

- 给表单设置一个 提交事件

```
<form  onsubmit="return checkData()">
</form>
```

- 创建checkData()函数响应这个事件

```
function checkData(){
  //1. 获得用户输入的用户名
  //2. 判断用户名是否为null
  //3. 如果为null,  给用户一个警告, 阻止表单提交

}
```

四,代码实现

- HTMl

```
<form action="http://www.baidu.com" method="get" id="myform" onsubmit="return checkData()">
    <table class="main" border="0" cellspacing="0" cellpadding="0">
```

- JS代码

```
<script type="text/javascript">
           //创建checkData()函数响应这个事件
           function checkData () {
               //1. 获得用户输入的用户名
               var username =  document.getElementById("usernameId").value;
               //2. 判断用户名是否为null,如果为null,  ,
               if(username == null || username == ""){
                   //给用户一个警告
                   alert("用户名不能为空....");
                   //阻止表单提交
                   return false;
               }

               return true;

           }
       </script>
```

> 作业: 判断一个两次输入的密码是否一致, 不一致 阻止提交; 一致的提交.

五,JS事件总结【重点】

属性	当以下情况发生时，出现此事件	FF	N	IE
onabort	图像加载被中断	1	3	4
onblur	元素失去焦点	1	2	3
onchange	用户改变域的内容	1	2	3
onclick	鼠标点击某个对象	1	2	3
ondblclick	鼠标双击某个对象	1	4	4
onerror	当加载文档或图像时发生某个错误	1	3	4
onfocus	元素获得焦点	1	2	3
onkeydown	某个键盘的键被按下	1	4	3
onkeypress	某个键盘的键被按下或按住	1	4	3
onkeyup	某个键盘的键被松开	1	4	3
onload	某个页面或图像被完成加载	1	2	3
onmousedown	某个鼠标按键被按下	1	4	4
onmousemove	鼠标被移动	1	6	3
onmouseout	鼠标从某元素移开	1	4	4
onmouseover	鼠标被移到某元素之上	1	2	3
onmouseup	某个鼠标按键被松开	1	4	4
onreset	重置按钮被点击	1	3	4
onresize	窗口或框架被调整尺寸	1	4	4
onselect	文本被选定	1	2	3
onsubmit	提交按钮被点击	1	2	3
onunload	用户退出页面	1	2	3

1.需要掌握的事件

- onclick: 点击
- onsubmit: 表单提交
- onfocus: 获得焦点; onblur: 失去焦点

```
<body>
    <!--给输入框设置获得和失去焦点事件
        this:就是当前对象, 就是谁获得焦点 this就是谁
    -->
    <input id="inputId" type="text" onfocus="_onfocus(this)" onblur="_onblur()" value="哈哈哈"/>

</body>

<script>

    //获得焦点的时候, 获得输入框的value "哈哈哈"
    function _onfocus (obj) {
        //方式一: 先获得input, 再调用value
        //var inputEle =  document.getElementById("inputId");
        //var data  =  inputEle.value;

        //方式二: 传参, obj就是上面传过来的this, 就是输入框对象

        console.log("获得了焦点..."+obj.value);

    }

    function _onblur () {
        console.log("失去了焦点...");
    }

</script>
```

- onload: 等页面加载完成

```
    <script>
        function getData () {
            //获得input输入框标签(对象)
            var inputEle =  document.getElementById("inputId");
            // 获得value
            var  str = inputEle.value;
            console.log("str="+str);
        }

    </script>

</head>
<!--等页面加载完成-->
<body onload="getData()">
    <input id="inputId" type="text" value="哈哈"/>
</body>
```

- onchange 内容改变

```
<select id="selectId" onchange="changePlayer(this)">
    <option value="Jordon">乔丹</option>
    <option value="Iverson">艾弗森</option>
    <option value="Kobe">科比</option>
</select>
</body>

<script>
    function changePlayer (obj) {
        //获得选中的内容
        alert("内容改变了..."+obj.value);
    }
</script>
```

2.需要知道的事件

- 和鼠标相关的事件

```
<body>

    <div class="box"  id="divId" onmousemove="_onmousemove(this)"
        onmousedown="_onmousedown(this)" onmouseout="_onmouseout(this)"
    ></div>

</body>

<script>
    /*function  changeColor(obj,color) {
        obj.style.backgroundColor = color;
    }*/

    //鼠标移动
    function _onmousemove (obj) {
        //div的背景色改成红色(CSS)
        obj.style.backgroundColor = "red";

    }

    //鼠标按下
    function _onmousedown (obj) {
        //div的背景色改成绿色(CSS)
        obj.style.backgroundColor = "green";

    }

    //鼠标离开
    function _onmouseout (obj) {
        //div的背景色改成黄色(CSS)
        obj.style.backgroundColor = "yellow";

    }

</script>
```

- 和键盘相关的事件

```
<body>

    <input type="text" onkeydown="_onkeydown()" onkeyup="_onkeyup()"/>

</body>

<script>
    //键盘按下
    function  _onkeydown() {
        console.log("键盘按下...");

    }

    //键盘抬起
    function  _onkeyup() {
        console.log("键盘抬起...");
    }

</script>
```

案例二:使用JS完成图片轮播效果

一,需求分析

- 实现每过3秒中切换一张图片的效果，一共3张图片，当显示到最后1张的时候，再次显示第1张。

二,技术分析

1.定时任务

- setInterval(code,time) 按照指定的周期（以毫秒计）来调用函数或计算表达式

  参数说明: code即执行的代码;

  方式一: 函数名　　setInterval(show,3000);

  方式二:函数字符串　　setInterval("show()",3000);

  time:时间,单位毫秒

- 示例代码

```
<script>
    //每隔1s向控制台打印hello...

    //1. 创建定时任务
    setInterval("sayHello()",1000);

    //2. 创建打印hello...的函数
    function  sayHello() {
        console.log("hello...");
    }


</script>
```

2.使用JS操作图片

- 其实就是改变src的值

```
<body>
    <img id="imgId" src="../img/banner_1.jpg" width="600px" height="200px"/><br/>
    <input type="button" value="上一张" onclick="preImg()"/>
    <input type="button" value="下一张" onclick="nextImg()"/>

</body>

<script>

    var i = 1;

    //上一张
    function  preImg() {
        i--;
        if (i == 0){
            i = 3;
        }

        //1. 获得img标签对象
        var imgEle =  document.getElementById("imgId");
        //2. 改变src值
        imgEle.src = "../img/banner_"+i+".jpg";

    }

    //下一张
    function nextImg() {
        i++;
        if (i == 4){
            i = 1;
        }

        //1. 获得img标签对象
        var imgEle =  document.getElementById("imgId");
        //2. 改变src值
        imgEle.src = "../img/banner_"+i+".jpg";
```

```
  }
```

```
### 三,思路分析

+ 创建一个定时任务
```

setInterval("changeImg()",3000);

```
+ 创建changeImg()函数 响应这个事件,  在这个函数里面切换图片
```

function changeImg(){
  //1. 获得轮播图img标签
  //2. 改变img的src的值(需要做临界点的判断)

}

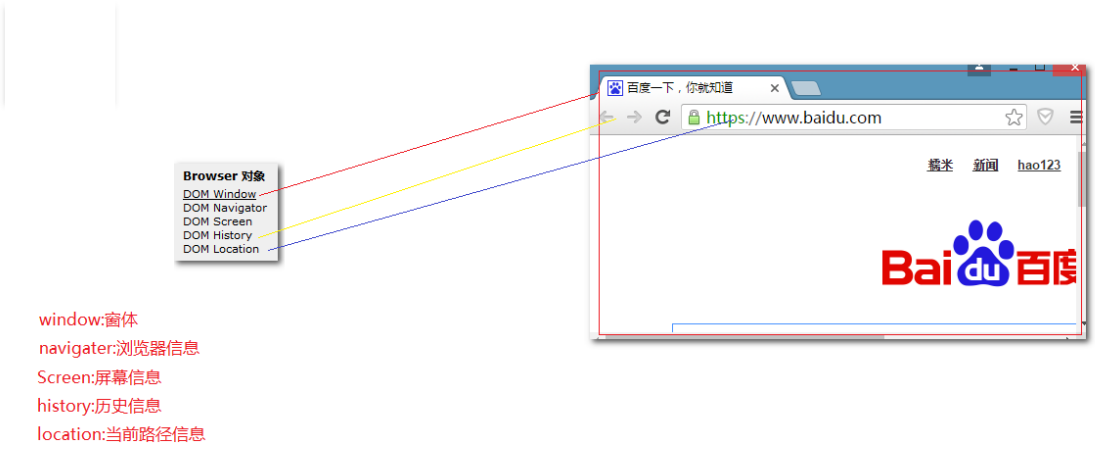
四,代码实现

```
<script>
    setInterval("changeImg()",3000);


    var i = 1;

    //切换图片
    function changeImg() {


        i++;

        if (i == 4){
            i = 1;
        }

        //1. 获得轮播图img标签
        var imgEle =  document.getElementById("imgId");

        //2. 改变img的src的值(需要做临界点的判断)
        imgEle.src = "../img/banner_"+i+".jpg";
    }

</script>
```

五,BOM总结

1.概述

Browser Object Model,为了便于对浏览器的操作，JavaScript封装了对浏览器中各个对象，使得开发者可以方便的操作浏览器中的各个对象。



2.BOM里面的五个对象

2.1window: 窗体对象

方法	作用
alert()	显示带有一段消息和一个确认按钮的警告框
confirm()	显示带有一段消息以及确认按钮和取消按钮的对话框
setInterval()	按照指定的周期（以毫秒计）来调用函数或计算表达式
setTimeout()	在指定的毫秒数后调用函数或计算表达式
clearInterval()	取消由 setInterval() 设置的 Interval()。
clearTimeout()	取消由 setTimeout() 方法设置的 timeout。

2.2,navigator:浏览器对象(了解)

属性	作用
appName	返回浏览器的名称
appVersion	返回浏览器的平台和版本信息

2.3,screen:屏幕对象(了解)

方法	作用
width	返回显示器屏幕的宽度
height	返回显示屏幕的高度

2.4,history:历史对象

属性	作用	作用
back()	加载 history 列表中的前一个 URL	
forword()	加载 history 列表中的下一个 URL	
go()	加载 history 列表中的某个具体页面	

2.5,location:当前路径信息

属性	作用
host	设置或返回主机名和当前 URL 的端口号
href	设置或返回完整的 URL
port	设置或返回当前 URL 的端口号

location.href; 获得路径

location.href = "http://www.baidu.com"; 设置路径,跳转到百度页面