# 回顾

Html, css: 标记语言

Javascript: 编程语言

# Javascript：JS，基于客户端的脚本语言



Day1: 变量

Day2: 循环， 语句

Day3: 函数

Day4: 系统内部的函数

Day5-8: 案例项目

# 变量：用来保存东西（数据）

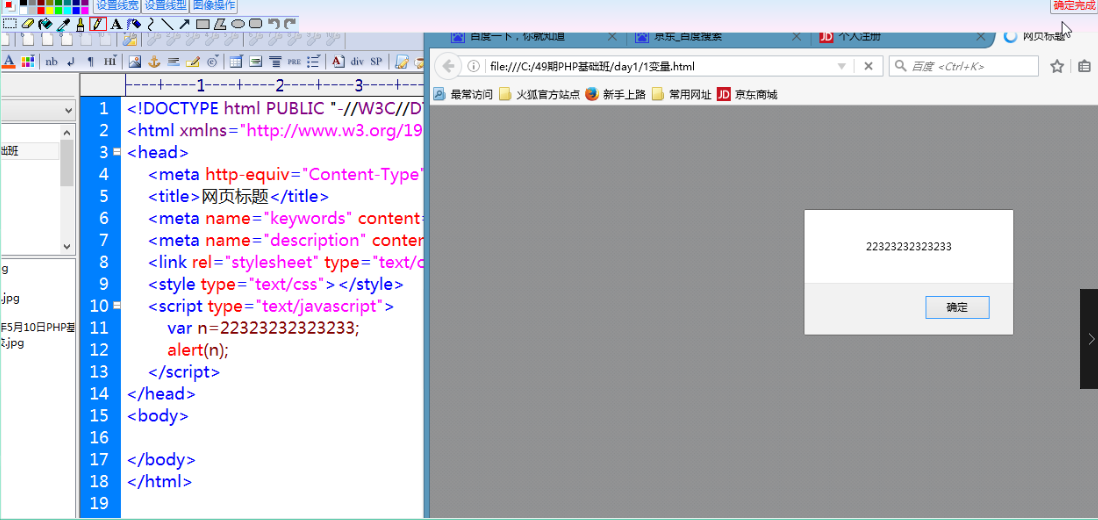
[123456@qq.com](mailto:123456@qq.com)

张三丰

## 语法：var 变量名=值;

Var n=23;

Var m=24;



## Var 系统内部的关键字，用来声明变量

# 变量名

命名规则：

1. 变量名不能以数字开头；
2. 变量名只能包含字母、数字或下划线；
3. 区分大小写；
4. 变量名不能和系统内部关键字相同；
5. 建议：变量名要起得有意义

## 变量名练习

Abc 可以

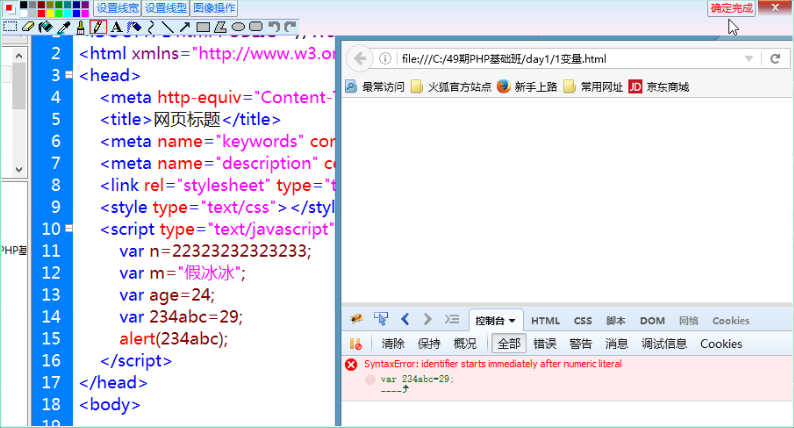
Abcd\_abcd 可以

Ab\_ab\_ab 可以

Abc234 可以

\_\_\_\_\_\_ 可以

234abc 不可以



# 数据类型

数字类型：number, 数字

字符类型：string, 字母，汉字，符号

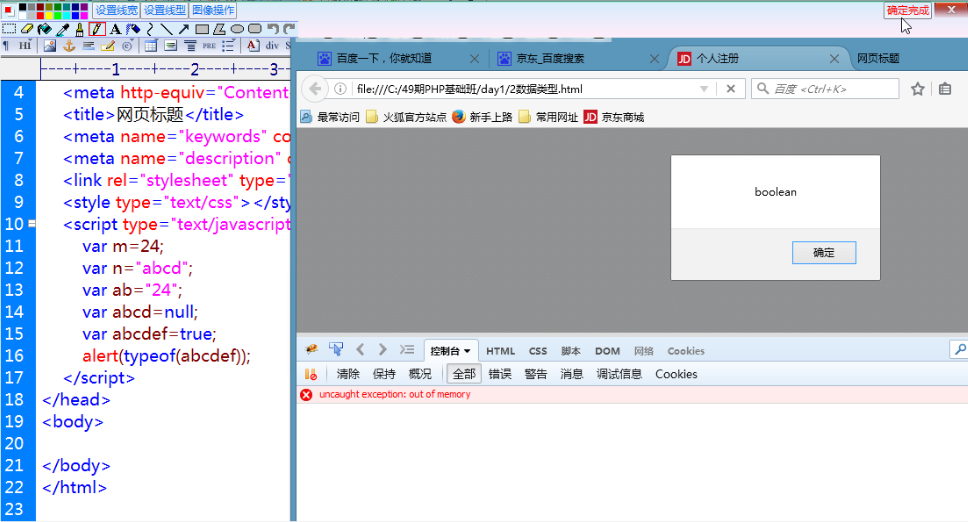
空类型：null, 表示什么都没有

未定义类型：undefined, 表示没有定义

布尔类型：true, false, boolean(bool)

复合数据类型： array 数组，object 对象

## Typeof 查看数据类型



Alert：弹出一个窗口（会停止程序的运行）

# Firebug：调试

注意：不要太相信firebug，认真就输了。

1. 安装firebug
2. 控制台（点击启用）
3. 刷新
4. 找错



# 运算符

1.赋值运算符：符号【=】，表达的意思是把符号右边的数据赋值给左边的变量

2.算数运算符：符号【+，-，\*，/，%，++（递增1），--（递减1）】

a.%取余：取得两个数相除的余数

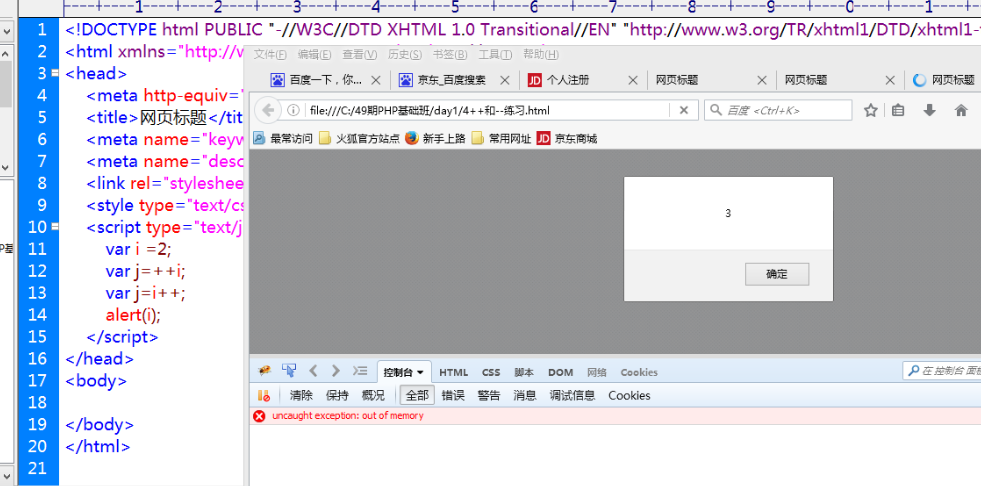
b.++：var i=2; i++; 给变量i加1(++i)

c.--：var i=2; i--; 给变量i减1(--i)

注意：当有赋值运算符参与到++和--中的时候，i++和i--的含义就不同

1) var j=i++; 先把i的值赋值给j，之后i再自加1

2) var j=++i; 先把i自加1，再赋值给j



## 练习++

Var i=6;

Var j=i++;

Var m=j++;

Var n=++i;

Var a=m++;

Var b=m++;

Var ab=++b;

【I=8, J=7, M=8, N=8, A=6, B=8, Ab=8】

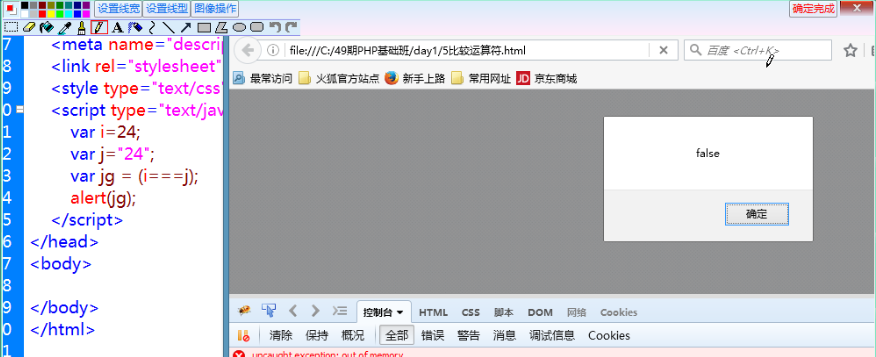
## 比较运算符

符号【> >= < <= == !=(不等于) ===(全等于) !==(不全等于)】

==：判断两个值是否相等（值相等）

===：判断两个值是否相等（值相等，数据类型相同）

## 练习比较运算符：



## 逻辑元素符

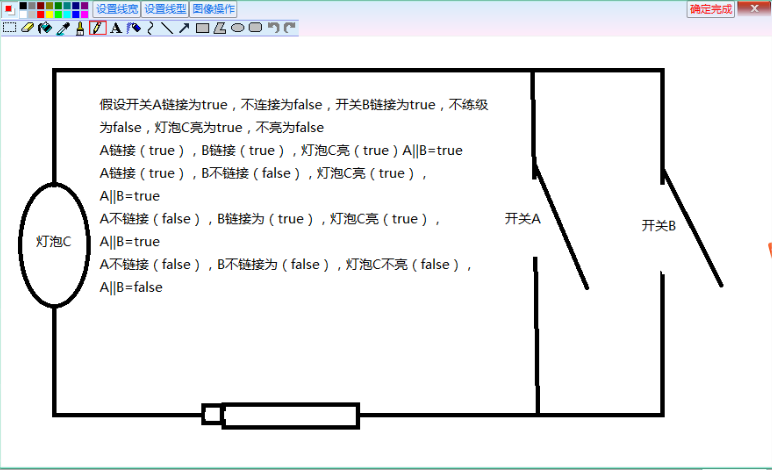
符号【！，&&，||】

！：逻辑非：不是，非

&&：逻辑与：如果两个条件同时满足，为true;如果两个条件中有任意一个不满足，为false.

||：逻辑或：如果两个条件中有任意一个满足，为true;如果两个条件都不满足，为false.





## 复合赋值运算符

符号【+= -= \*= /=】

+=: var i=6;i+=6; 【i=i+6】

-=: var i=6;i-=2; 【i=i-2】

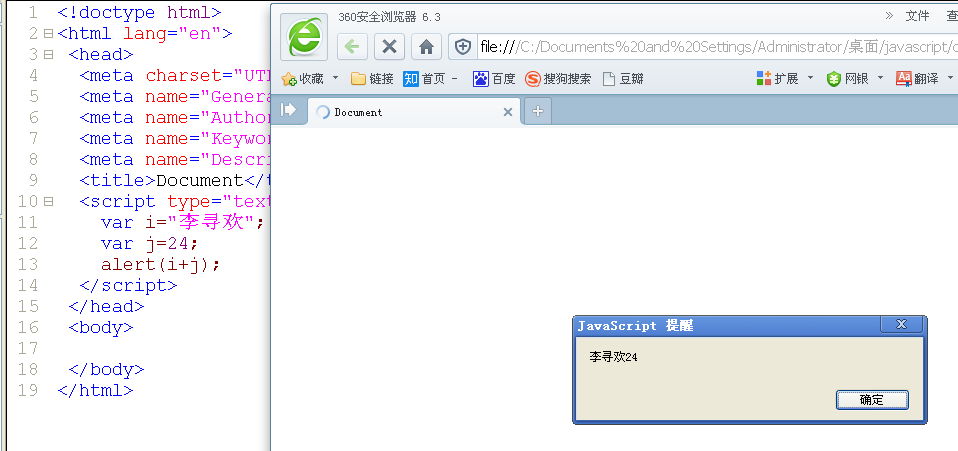
\*=: var i=6;i\*=6; 【i=i\*6】

/=: var i=6;i/=6; 【i=i/6】

## 字符串运算符

符号【+】：连接符号两边的内容

当+两边，任意有一个是字符串就会执行字符串+



# JS的语法规则

1. 区分大小写；
2. 所有的符号都必须是英文状态下的；
3. 每行表达出所要表达的意思即可；
4. 建议：每行后面必须加上英文状态下的分号（不加分号也不会报错，JS语法规则允许）

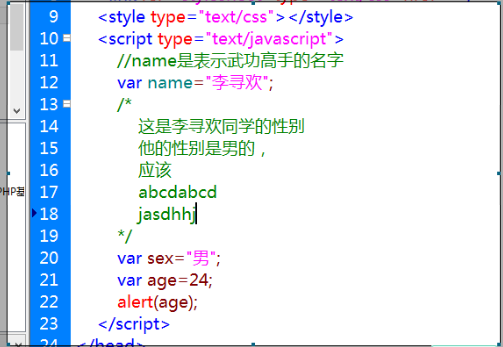
# JS的注释

注释就是给某段代码加上标注或者说明，记录

注释的内容不会被代码执行

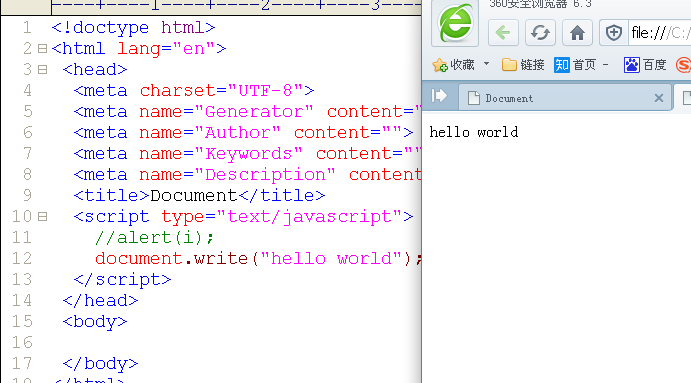
两种：

1. 单行注释//：符号//后面的内容都是注释内容；
2. 多行注释/\*\*/:符号/\*内容\*/中间的内容都是注释内容



Alert:弹出警告窗口

# Document.write()输出到页面上



# 回顾

变量：var 变量名=值；

Var:声明一个变量

变量名命名规则

数据类型

1. 数字类型：number
2. 字符类型：string
3. 布尔类型：boolean/bool
4. 空类型：null 表示空的，什么都没有
5. 未定义类型：undifined 表示没有定义
6. 复合数据类型：复杂数据 array/object

运算符

1. 赋值运算符
2. 算术运算符
3. 比较运算符
4. 逻辑运算符
5. 复合赋值运算符
6. 字符串运算符

# 位运算符

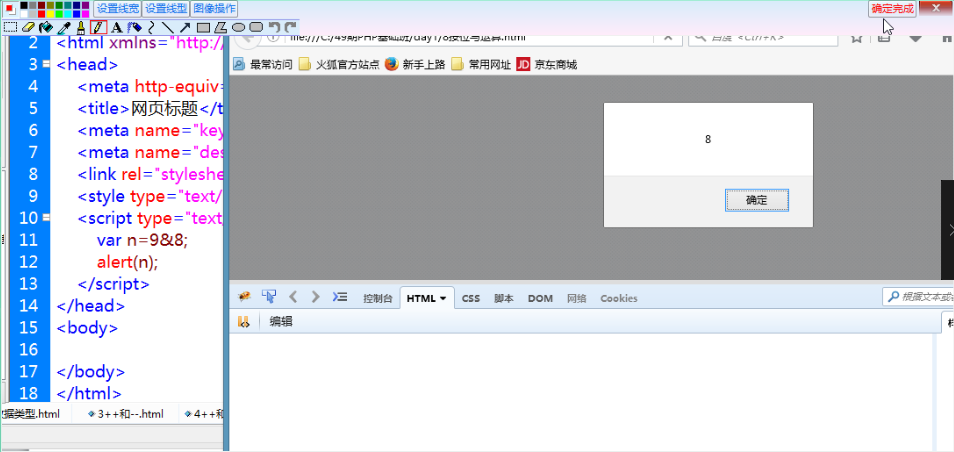
二进制：

十进制：

&（按位与）：获取两个数的二进制位，比较相同位数上的数字，如果都为1，则为1；否则为0

Var n=7&8;

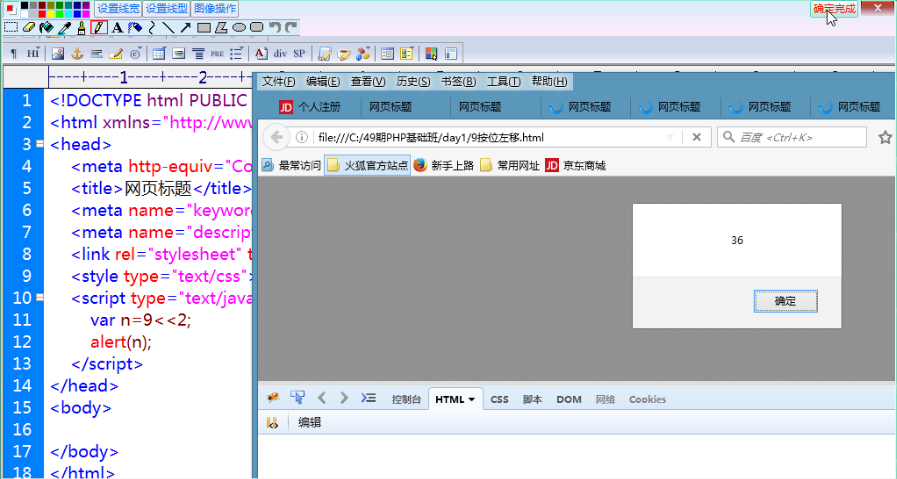
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 7的二进制 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 8的二进制 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 计算 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |



<<（按位左移）:把某个数字的二进制位数左移n位，后面不足的补0，前面挤出去的不管。

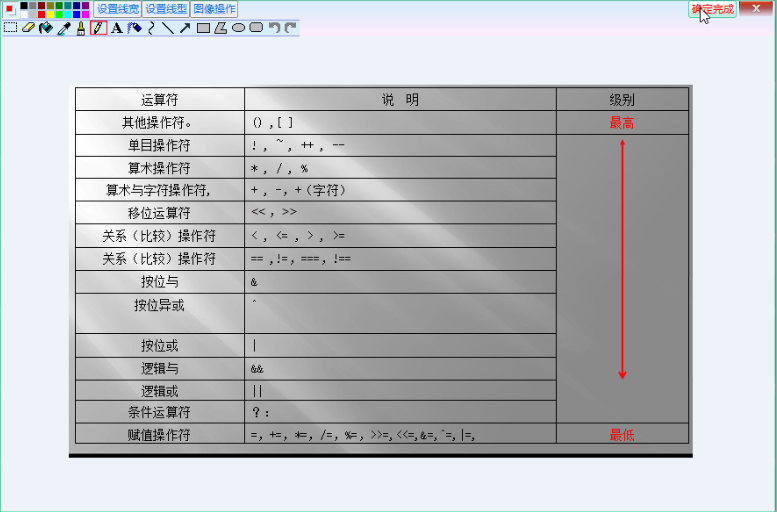
Var n=9<<2;

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 9的二进制 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 左移两位 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |



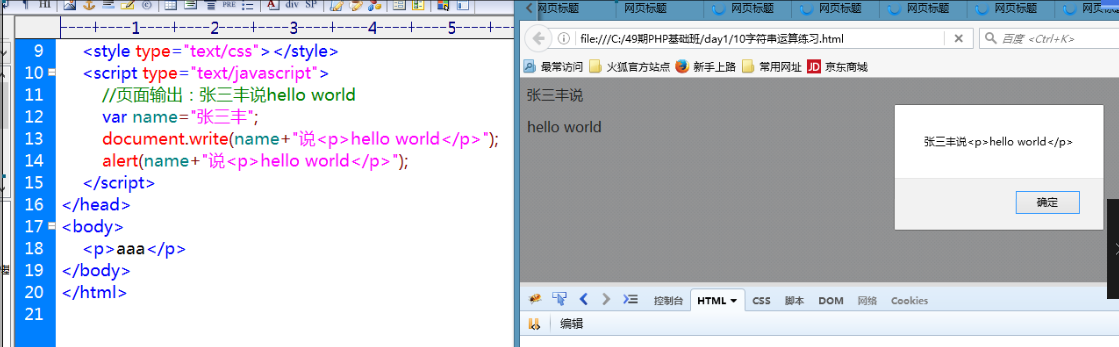
# 运算符的优先级

1. 小括号最优先
2. 逻辑与大于逻辑或
3. 赋值运算符最后进行
4. 当有多个运算符参与运算的时候，必须意识到运算符是有优先级的



## 练习1

1. 在页面上输出内容【用JS输出】



1. 在页面上输出h标签【用JS输出】



## 练习2