3. === 全等

判断两个值是否全等，它和相等类似，不同的是它不会做自动的类型转换，如果两个值类型不同直接返回false.

4 !== 不全等

条件运算（三元运算

语法：?:

条件表达式？语句1：语句2. 会返回结果

↑ 描述 ↑ true ↑ false.

$x = a > b ? a : b$; $x = max(a, b)$

可以套娃娃

运算符优先级

1. 先乘除后加减
2. 有的优先级大于或

Memo No.
Date / /

语句

在JS中可以使用{}来为语句分组
同一个{}中的语句要么都执行
要么都不执行
代码块内容，外部是完全可见的

流程控制语句

语句分类：条件判断 if
条件分支 switch.
循环

对象

创建对象

```
var obj = new Object();
obj.name = "XX";  添加属性.

读取 obj.name;
修改 obj.name = "新值";
删 delete obj.name;
```

属性名可以用任意名字.
但是使用特殊的属性名需采用
obj["123"] = 值
来操作.

用[]这种形式去操作属性更加灵活
在[]中可以使用变量

```
obj["123"] = 789;
var n = "123";
log(obj[n]); => 789;
```

```
/*
in运算符
检查一个对象中是否含有指定的属性
"name" in obj;
*/
```

基本和引用数据类型

基本中变量都是在栈内存中
值与值之间是相互独立

对象都是在堆内存中，变量保存的是地址

当比较两个对象时，比较的是地址.

Memo No.
Date / /

对象字面量

使用对象字面量来创建一个对象

```
var obj = {name:"猪八戒"};
```

函数(对象)

创建

```
var fun = new Function();
```

使用函数声明创建函数

```
function fun2(){
    ....
};
```

使用函数表达式来创建一个函数

```
function(){}  //匿名函数，无意义

var fun3 = function(){};
```

参数（可以是函数，对象）

例 求两数和：

```
function sum(a,b){
a+b;
}
```

sum()
调用函数

sum.
函数对象。

sum(1,2);

调用函数时，解析器也不会检查实参的数量
多余实参不会被赋值

sum(1,2,"hello",true,null); → 3.

如果实参数少于形参，则对应实参的形参为 undefined.

返回值

例

```
function sum(a,b,c){
var d=a+b+c;
return d;
}
```

立即执行函数

```
(function() {
    ......
})();
```

函数定义完可立即执行，不等调用，这种函数只会执行一次。

对象（补）

对象属性值可以是函数 → 方法

```
var obj = new Object();
obj.sayName = function() {
    ...
};

var obj2 = {
    ...
    sayName : function() {
        ...
    },
```

枚举对象中的属性

for in 语句

for(var n in obj){ (每次将对象中的一个属性名赋值给变量n)

console.log(n);

}

作用域 {全局：变量作为window对象的属性保存

函数

变量声明提前

var定义声明变量，会在所有代码执行之前被声明，但不会赋值

函数声明法同样也会提到创建

故可在其创建之前调用

函数作用域中也有声明提前特性

扫描全能王 创建

this

解析器在调用函数每次都会向函数内部隐藏
进一个隐含的参数 ——> this
this 指向的是一个对象，这个对象我们称为函数执
行的上下文

工厂方法创建对象

```
function creatPerson(){
    var obj = new Object();
    return obj;
obj.name = "孙武圣";
obj.age = 18;
obj.·····
    return obj;
}
```

构造函数 ——> 一个普通函数 ——> 首字母一般大写
普通函数直接调用，构造函数需使用关键
字new.

1. 立刻创建一个新的对象
2. 将新建对象设为函数中 this
3. 逐行执行函数中的代码
4. 将新建的对象作为返回值返回

instance of
检查一个对象是否是一个类的实例
用法
对象 instance of 构造函数

所有对象都是object的后代

构造函数 → 类
function person(...){
...
...
}
var obj=new person()

扫描全能王 创建