正课:

1. RegExp

2. Function

1. RegExp:

1. 验证字符串格式: var bool=reg.test(str)

2. 即查找每个关键词的内容，又查找每个关键词的位置

var arr=reg.exec(str);

在str中查找下一个关键词的内容和位置

返回值: 和match不加g的时候是一样的！

如果找到: arr: [ 0: 关键词的内容, index: 位置]

如果没找到返回null

问题: 每次只能找一个关键词的内容和位置，无法找所有。

解决: 正则表达式后+g

并且用循环反复执行exec()。每次执行exec()，exec()都非常聪明的自动找下一个。

但是，因为exec()返回的数组中，一次只能放一个敏感词的内容和位置。所以，第二次找到的只能覆盖第一次找到的。

2. Function

什么是:

用途: 保存一段可重用的代码段的程序结构，再起一个名字。

本质: 内存中保存一段代码端的存储空间——对象

为什么: 代码重用！

何时: 只要一段代码，可能被反复使用时，都要定义在一个函数内，再起一个名字。今后，用函数名等于用代码段。

如何:

1. 创建函数: 3种:

1. 以声明方式创建:

function 函数名(形参列表){

函数体

return 返回值

}

形参:

什么是: 专门接收从函数外部传入函数内数据的变量

为什么: 有些函数执行时，需要动态获得必须的数据，才能正常执行。

何时: 只要一个函数，必须某些数据才能正常执行时

返回值:

什么是: 一个函数的执行结果

为什么: 外部调用者可能需要获得函数的执行结果

何时: 只要外部调用者，需要获得函数的执行结果时。

问题: 会被声明提前(hoist)

什么是: 在程序开始执行前！程序会先将所有var声明的变量和function声明的函数，提前到当前作用域的顶部集中创建。而赋值(=)留在原地。

所以声明提前是js中广受诟病的缺陷。打乱了程序正常的执行顺序。

2. 赋值方式创建:

var函数名=function (形参列表){

函数体

return 返回值

}

说明: 赋值方式创建的函数，和声明方式创建的函数在使用时，是完全一样的！

只不过，在程序开始执行前，赋值方式可避免函数被声明提前。保持了程序原有的执行顺序。

揭示: js中其实函数也是一个普通的对象而已。函数名仅仅是一个普通的变量。函数名变量通过对象地址引用着函数对象。每次调用函数名时，都是通过地址，找到函数对象，再执行其中的内容。

2. 调用函数：

var 变量=函数名(实参值列表)

调用函数名，等于调用函数中的函数体

实参值以赋值方式传递给形参变量

如果函数有返回值，则用变量接住。

强调: 如果一个函数，只是定义，而没有调用，其内部的代码是不执行的！即使写错！也不会发现，也不报错！！