课程名称

第 1 天课堂笔记（本课程共 8 天）

前端与移动开发学院

http://web.itcast.cn

目录

一、复习

昨天学习到了程序的两种控制流向语句，原来的程序，只能一条一条的执行，不能有别的分支。

“流程控制语句”：if、for。

1.1 if

选择语句，给程序添加了多种执行路线。

```
if(){
    语句 1
}else if(){
    语句 2
}else if(){
    语句 3
}else{
    语句 4
}
```

有且仅有一条出路。注意跳楼现象。

所以我们发现，计算机的两个基本能力：1）计算能力 2）流程控制能力

1.2 for

循环语句，顾名思义，就是将结构类似的语句重复执行。

```
for(var i = 0 ; i <= 100 ; i++){
    console.log(i);
}
```

for 语句能够简化程序的书写，不用大量的 ctrl+C、ctrl+V 了；

for 语句充分体现计算机的“奴隶性”。

比如题目：寻找 1~1000 之内，所有能被 5 整除、或者能被 6 整除的数字

```
for(var i = 1 ; i <= 1000 ; i++){
    if(i % 5 == 0 || i % 6 == 0){
        console.log(i);
    }
}
```

上面这个算法，我们有一个术语**“穷举法”**。

穷：　完整。欲穷千里目，更上一层楼。穷尽。

举：　列举，推举。

穷举法：就是一个一个试。我们现在要寻找 1~1000 之内，所有能被 5 整除、或者能被 6 整除的数字。

我们的思路，就是将 1、2、3、4、……998、999、1000 依次去试验。看看这个数字，能不能被 5 或者 6 整除。

这时候你说，老师，那计算机太辛苦了。你看，13 这个数字，为什么还要试呢？打眼一看就不能被 5、

6 整除。

计算机就是一个奴隶。它没有思维，它就是一个不吃饭，有着极强计算力的东西。

二、几道算法练习

2.1 题目 1：报 7 游戏的安全数

大家从小到大，都玩儿过的一个庸俗的游戏：

游戏玩儿法就是，大家轮流报数，如果报到能被 7 整除的数字，或者尾数是 7 的数字，都算踩地雷了。就应该罚唱歌。

请在控制台输出 1~60 之间的所有“安全数”。

比如：

1、2、3、4、5、6、8、9、10、11、12、13、15、16、18、19、20、22、23、24、25、26、29、30……

答案见案例

2.2 题目 2：水仙花数

水仙花数是一种特殊的三位数，它的特点就是，每个数位的立方和，等于它本身。

比如 153 就是水仙花数。因为：

$$1^3+5^3+3^3 = 153$$

100~999 之内，只有 4 个水仙花数，请找出来。

特别经典的算法，是每个学习编程的人，都要会做。

答案见案例，我们只列出数值答案：　153、370、371、407

2.3 题目 3：求 1~100 的和

求和的题目，涉及到了新的一种算法思想，叫做“累加器”。

1+2+3+4+5+6……

正确的：

```
var sum = 0;  //累加器
//遍历 1~100，将所有的数字扔到累加器里面
for(var i = 1 ; i <= 100 ; i++){
    sum = sum + i;
}
console.log(sum);
```

初学者常见的错误：

错误 1：不声明 sum，不行的，因为所有的变量都要声明：

```
//遍历 1~100，将所有的数字扔到累加器里面
for(var i = 1 ; i <= 100 ; i++){
    sum = sum + i;
}
console.log(sum);
```

错误 2： sum 不能在 for 里直接声明：

```
//遍历 1~100，将所有的数字扔到累加器里面
for(var i = 1 ; i <= 100 ; i++){
    var sum = sum + i;
}
console.log(sum);
```

错误 3： 每次 for 循环都要 var 一次，是不正确的：

```
//遍历 1~100，将所有的数字扔到累加器里面
for(var i = 1 ; i <= 100 ; i++){
    var sum = 0;  //累加器
    sum = sum + i;
}
console.log(sum);
```

2.4 求阶乘

所谓的阶乘，比如 6 的阶乘，就是 1*2*3*4*5*6 = 720。

现在，计算 13 的阶乘。

```
var result = 1;        //累乘器

for(var i = 1 ; i <= 13 ; i++){
        result = result * i;
}
console.log(result);
```

2.5 用户输入一个数，输出因数的个数。

昨天晚上的作业 3，就是用户输入一个数，输出所偶遇能够整除它的数字。

比如，

用户输入 48，此时输出 1、2、3、4、6、8、12、16、24、48 。　共 **10** 个数字。

用户输入 21，此时输出 1、3、7、21.共 4 个数字。

今天的这道题目，和昨天的这个题目非常像，不过不输出完成序列，只输出个数。

也就是说，用户输入 48，弹出 **10**。

用户输入 21，弹出 **4**

用户输入 11，弹出 **2**

提示：此题用到累加器。

2.6 判断质数

质数：就是只能被 1 和自己整除。

翻译过来：它的因数个数是 2。

比如：2、3、5、7、11、13、17、19、23、29、31、37……

用户输入一个数字，弹出这个数字是否是质数。

提示，用 2.5 的思路。

答案见案例。

三、函数

3.1 初步认识函数

```
    <script type="text/javascript">
        console.log("你好");
        sayHello();          //调用函数

        //定义函数：
        function sayHello(){
            console.log("欢迎");
            console.log("welcome");
        }
    </script>
</body>
```

```
你好
欢迎
welcome
```

函数，是一种封装。就是将一些语句，封装到函数里面。通过调用的形式，执行这些语句。

函数的使用，是两个步骤，**第一步，函数的定义**：

语法：

```
function  函数名字(){

}
```

function 就是英语“函数”、“功能”的意思。顾名思义，将一些功能封装到函数里面。

function 是一个关键字，和 var、typeof 一样，都是关键字，后面要加空格。

函数名字的命名规定，和变量的命名规定一样。只能是字母、数字、下划线、美元符号，不能以数字开头。

后面有一对儿空的小括号，里面是放参数用的，下午介绍。大括号里面，是这个函数的语句。

常见错误：

不能小括号包裹大括号：

```
function sayHello({

})
```

不能忘了小括号对儿：

```
function sayHello{

}
```

第二步，函数的调用。

函数如果不调用，等于白写。

调用一个函数，太简单了，就是这个函数的名字后面加小括号对儿。

语法：

```
函数名字();
```

定义函数，可以在调用的后面：这是 JS 的语法特性，函数声明头的提升。知道就行了。

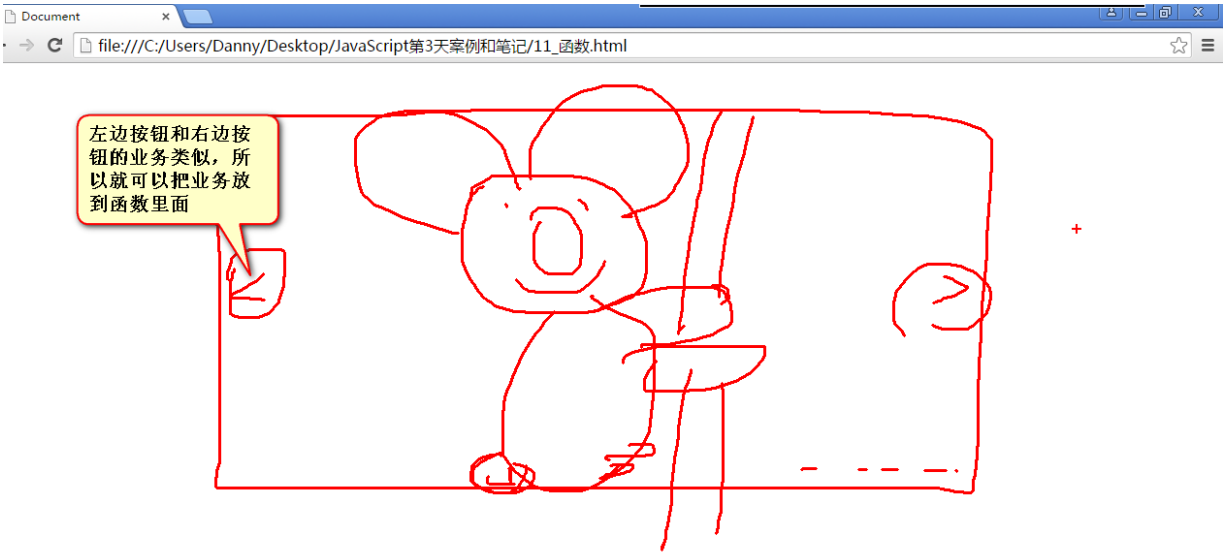
```
        console.log("你好");
        sayHello();          //调用函数

        //定义函数：
        function sayHello(){
            console.log("欢迎");
            console.log("welcome");
        }
```

函数的功能、好处：

1） 将会被大量重复的语句写在函数里面，这样以后需要这些语句的时候，直接调用函数，不用重写那些语句。

2） 简化编程，让编程变的模块化。

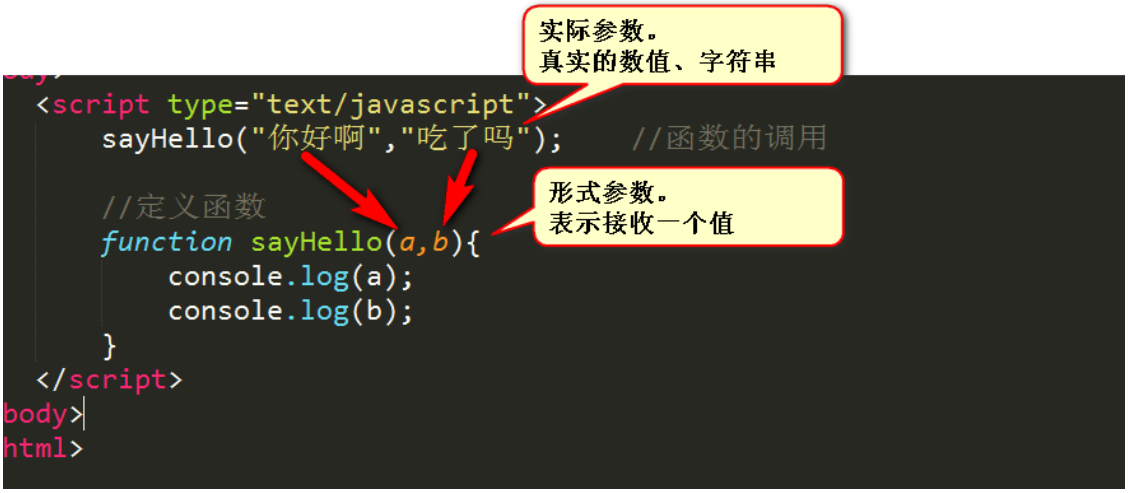


```
body>
  <script type="text/javascript">
      console.log("你好");
      sayHello();      //调用函数
      console.log("么么哒");

      //定义函数:
      function sayHello(){
          console.log("欢迎");
          console.log("welcome");
      }
  </script>
/body>
```

```
你好
欢迎
welcome
么么哒
```

3.2 函数的参数



实际参数和形式参数的个数，要相同。

```
        qiuhe(3,4);
        qiuhe("3",4);
        qiuhe("我爱你","中国");

        function qiuhe(a,b){
            console.log(a + b);
        }
```

```
7
34
我爱你中国
```

3.3 函数的返回值

```
    <script type="text/javascript">
        console.log(qiuhe(3,4));

        function qiuhe(a,b){
            return a + b;
        }
    </script>
```

return 就是英语“返回”的意思，那么就表示此时这个“函数调用的表达式”（红色部分），值就是这个 a+b。

- 函数里面可以没有 return，如果有，只能有一个。不能有多个 return；
- 函数里面，return 后面不允许书写程序了，也就是说写在后面的程序无效；