一、JAVA的环境搭建

1. 什么是JRE、JDK

JRE（Java Runtime Enviroment ---Java运行环境）：包括Java虚拟机和Java程序所需的核心类库等，如果想要运行一个开发好的Java程序，计算机中只需要安装JRE即可。

JDK（Java Development Kit---Java开发工具包）：JDK是提供给Java开发人员使用的，包含了Java的开发工具，也包括JRE。

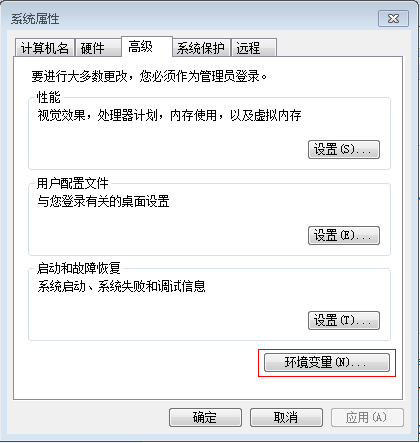
一句话总结：使用JDK开发完成的Java程序，交给JRE去运行。

1. 下载安装JDK

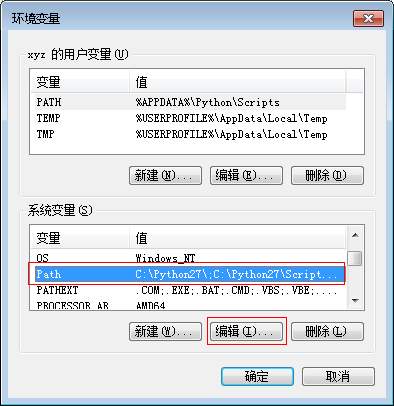
到官网<http://www.oracle.com/>去下载JDK，JDK中已包含JRE，不用再单独下载JRE了。

1. 配置环境变量

我的电脑-右键属性-高级系统设置-高级-环境变量



系统变量-找到Path变量-编辑

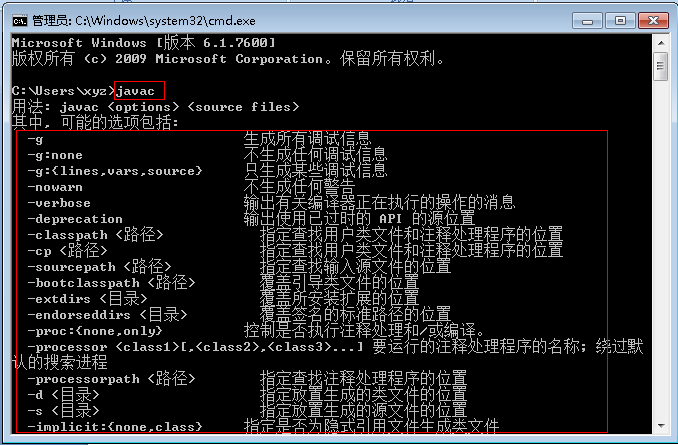


变量值：将鼠标移到最前面，把JDK安装文件下的bin目录路径填写在此处，后面要加个“;”号



1. 验证是否成功

打开命令提示，任意路径下运行javac命令，不报错就说明已经安装成功。



1. 运算符
2. 算术运算符

+ - \* / %（取余，模运算） +（连接符）

++(自增，在原有数据基础上+1，再赋给原有数据)

--（自减，在原有数据基础上-1，再赋给原有数据）

1. 赋值运算符

=(将右边的数据赋值给左边) +=（将左右二边的和赋值给左边） -=（将左右二边的差赋值给左边）

\*=（将左右二边的乘积……） /=（将左右二边的商……） %=（将左右二边的模……）

1. 比较运算符

运算完的结果是true或者false

|  |  |
| --- | --- |
| 运算符 | 运算 范例 结果 |
| == | 相等于 8==10 false |
| != | 不等于 8!=10 true |
| < | 小于 8<10 true |
| > | 大于 8>10 false |
| <= | 小于或等于 8<=10 true |
| >= | 大于或等于 8>=10 false |

1. 逻辑运算符

用于连接二个boolean类型的表达式。

&：与，运算规律：运算的二边只要有一个false，结果为false,二边都为true，结果才为true。

|：或，运算规律：运算的二边只要有一个true，结果为true,二边都为false，结果为false。

^：异或，二边结果相反为true，二边结果相同为false。

!：非运算，取相反的结果。

1. 三元运算符

格式：（条件表达式）?表达式1：表达式2;

如果条件为true，运算后的结果是表达式1，如果条件为false，运算后的结果是表达式2。

1. 判断结构

If语句

1. if(条件表达式)

{

执行语句;

}

示例：

Int x=1;

If(x<2)

{

System.out.printIn(“yes”);

}

1. if(条件表达式)

{

执行语句;

}

else

{

执行语句;

}

示例：

int x=1;

If(x>1)

{

System.out.printIn(“yes”);

}

else

{

System.out.printIn(“no”);

}

1. if(条件表达式)

{

执行语句;

}

else if(条件表达式)

{

执行语句;

}

……

else

{

执行语句;

}

示例：

int x=1;

If(x>1)

{

System.out.printIn(“a”);

}

else if

{

System.out.printIn(“b”);

}

else

{

System.out.printIn(“c”);

}

1. 选择结构

switch语句

格式：

switch(表达式)

{

case 取值1:

执行语句;

break;

case 取值2:

执行语句;

break;

……

default:

执行语句;

break;

}

示例：

**int** week=1;

**switch**(week)

{

**case** 1:

System.*out*.println(week+"对应的是星期一");

**break**;

**case** 2:

System.*out*.println(week+"对应的是星期二");

**break**;

//……

}

1. 循环结构

1、while语句格式：

while(条件表达式)

{

执行语句;

}

示例：获取1到10的和

**public** **static** **void** main(String[] args)

{

**int** x=1;

**int** sum=0;

**while**(x<10)

{

sum=sum+x;

x++;

}

System.*out*.println("sum="+sum);

}

2、do while

语句格式：

do

{

执行语句;

}while(条件表达式);

do while特点：条件无论是否满足，循环体至少执行一次。

3、for

语句格式：

for(初始化表达式;循环条件表达式;循环后的操作表达式)

{

执行语句;

}

示例：累加

**public** **static** **void** main(String[] args)

{

**int** sum=0;

**for**(**int** i=0;i<10;i++)

{

sum=sum+x;

}

System.*out*.println("sum="+sum);

}

for的另一种格式:

Java语言的最新版本――J2SE 1.5中，引入了另一种形式的for循环。借助这种形式的for循环，现在可以用一种更简单地方式来进行遍历的工作。

Java的第二种for循环基本是这样的格式：

for (循环变量类型 循环变量名称 : 要被遍历的对象) 循环体

借助这种语法，遍历一个数组的操作就可以采取这样的写法：

清单3：遍历数组的简单方式

/\* 建立一个数组 \*/

int[] integers = {1， 2， 3， 4};

/\* 开始遍历 \*/

for (int i : integers) {

System.out.println(i);/\* 依次输出“1”、“2”、“3”、“4” \*/

}

1. 其他流程控制语句
2. break语句：跳出所在的当前循环，应用范围是选择结构和循环结构。

**public** **static** **void** main(String[] args)

{

**for**(**int** i=0;i<3;i++)

{

System.*out*.println("i="+i);

**break**;

}

}

1. continue语句：继续，结束本次循环，继续下次循环，应用于循环结构。

**public** **static** **void** main(String[] args)

{

**for**(**int** i=0;i<11;i++)

{

**if**(x%2==0)

**continue**;

System.*out*.println("i="+i);

}

}

1. 函数

1、函数的定义

函数就是定义在类中的具有特定功能的一段独立小程序，也称为方法。

格式：

修饰符 返回值类型 函数名（参数类型 形式参数1，参数类型 形式参数2，……）

{

执行语句;

return 返回值;

}

返回值类型：函数运行后的结果的数据类型；

参数类型：形式参数的数据类型；

形式参数：是一个变量，用于存储调用函数时传递给函数的实际参数；

实际参数：传递给形式参数的具体数值；

return：用于结束函数；

返回值：该函数运算后的结果，该结果会返回给调用者。

示例：

**public** **static int** add(**int** a,**int** b)

{

**return** a+b;

}

2、函数的细节——void

特殊情况下，函数没有具体的返回值，return的后面直接用分号结束。返回值类型没有具体的数据类型，用void来表示。

示例：

**public** **static** **void** myPrint()

{

System.*out*.println("hello java");

**return**;

}

3、函数的重载

重载的概念：在同一个类中，允许存在一个以上同名函数，只要它们的参数个数或者参数类型不同即可。

重载的特点：与返回值类型无关，只看参数列表。

重载的好处：方便于阅读，优化了程序设计。

示例：

//返回二个整数的和

**int** add(**int** x,**int** y){**return** x+y;}

//返回三个整数的和

**int** add(**int** x,**int** y,**int** z){**return** x+y+z;}

//返回二个小数的和

**double** add(**double** x,**double** y){**return** x+y;}

1. 数组
2. 概念：同一种类型数据的集合。可以自动给数组中的元素从0开始编号，方便操作这些元素。
3. 格式1：

元素类型 [ ] 数组名=new 元素类型[元素个数或数组长度];

示例：**int**[] arr=**new** **int**[5];

格式2：

元素类型 [ ] 数组名=new 元素类型[]{元素1，元素2，……};

示例：**int**[] arr1=**new** **int**[]{1,2,3,4};

**int**[] arr2={1,2,3,4};

//数组练习：冒泡排序

**public** **static** **void** main(String[] args)

{

**int**[] arr={34,58,4,101,3,87};

*bubbleSort*(arr);

}

**public** **static** **void** bubbleSort(**int**[] arr)

{

**for**(**int** x=0;x<arr.length-1;x++)

{

**for**(**int** y=0;y<arr.length-1-x;y++)

{

**if**(arr[y]>arr[y+1])

{

**int** temp=arr[y];

arr[y]=arr[y+1];

arr[y+1]=temp;

}

}

}

}

1. 二维数组

格式1：int[][] arr=new int[3][2];

二维数组中有3个一维数组，每个一维数组中有2个元素。

格式2：int[][] arr=new int[3][];

二维数组中有3个一维数组，每个一维数组都是默认初始化值null。

arr 二维数组

arr[0] 二维数组中的角标为0的一维数组。

arr[0][0] 二维数组中的角标0的一维数组中角标为0的元素。

arr.length 二维数组的长度，其实就是一维数组的个数。

arr[0].length 二维数组中角标为0的一维数组的长度。

示例：

**int**[][] arr={{3,1,24},{5,8,2,9},{4,1}};

**for**(**int** x=0;x<arr.length;x++)

{

**for**(**int** y=0;y<arr[x].length;y++)

{

System.*out*.println(arr[x][y]);

}

}

1. 面向对象