第一个知识点Java跨平台;

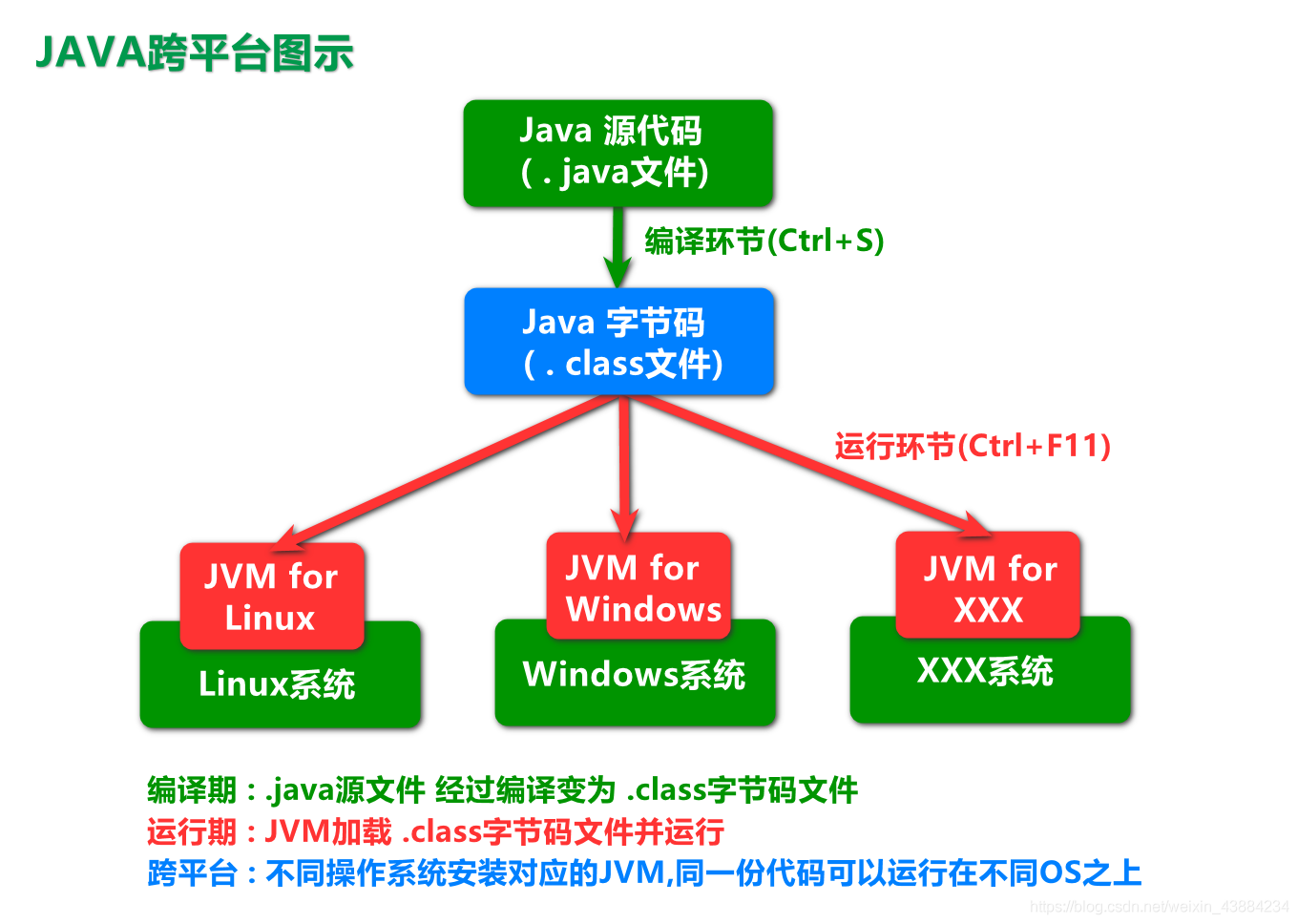
--java语言特性

简单性 安全性 面向对象 高性能 编译性 解释性 分布式处理 健壮性 开源 跨平台

--什么是跨平台性

通过Java语言编写的应用程序在不同的系统平台上都可以运行,那原理是什么？

我们所写的JAVA程序是".java"为后缀的源文件,但这些文件计算机无法直接执行，需要先进行编译环节,通过编译变为以".class"为后缀的字节码文件,这个字节码文件交由JVM(JAVA虚拟机)来运行.



**JVM：Java Virtual Machine** java虚拟机，是java程序运行的基础，它负责将Java字节码转换成机器码并执行。 加载.class并运行.class javap –all test.class

JRE： JRE (Java Runtime Environment)java运行环境 除了包含JVM以外还包含了运行java程序所必须的环境JRE = JVM+java系统类库(小零件)

JDK：java开发工具包除了包含JRE以外还包含了开发java程序所必须的命令工具JDK = JRE+编译、运行等命令工具

说明：运行java程序的最小环境为JRE开发java程序的最小环境为JDK



第二个知识点变量;

--变量



--什么是变量

在JAVA中，我们需要记录一些数据

但这些数据的值是不固定的，总在变，我们可以把这些数据理解为变量。

我们通过三个元素来描述变量：变量类型 变量名以及变量值。

定义变量的两种格式：

格式一：声明变量时并且赋值：int age = 18;

格式二：先声明变量：int tel ; 然后给变量赋值：tel = 119;

--命名规范

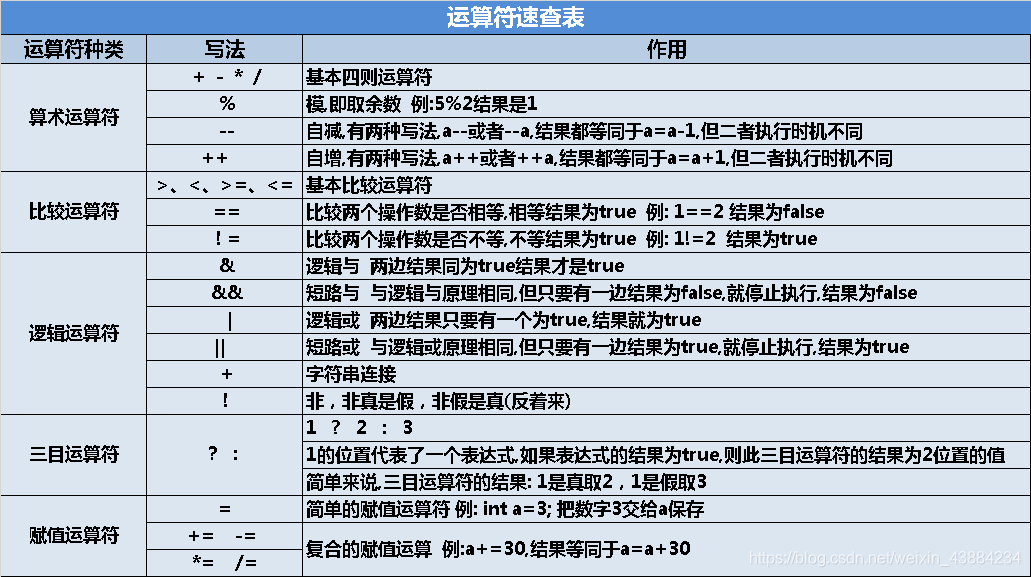
只能包含字母、数字、\_和$符，不能以数字开头

严格区分大小写 不能使用关键字

允许中文命名，但不建议，建议"英文的见名知意"、"小驼峰命名法"

第三个知识点运算符

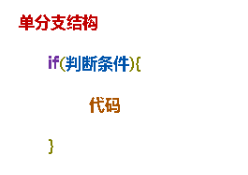
--运算符



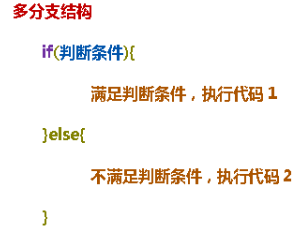
自增（++）：将变量的值加1  
分前缀式（如++a）和后缀式（如a++）。前缀式是先加1再使用；后缀式是先使用再加1。  
自减（- -）：将变量的值减1  
分前缀式（如- -a）和后缀式（如a- -）。前缀式是先减1再使用；后缀式是先使用再减1。

第四个知识点分支结构

--if eles

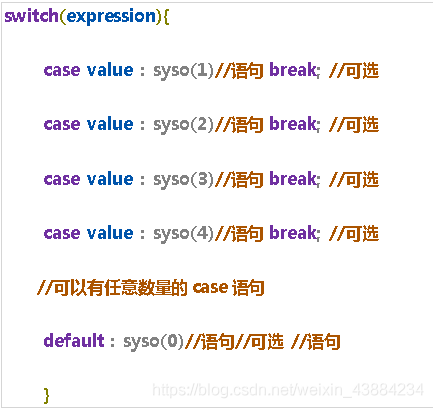


**if（boolean）{ … } 判断条件为ture则进入语句执行语句里的代码，为false则结束下面有代码继续执行没有代码则结束。**



**--switch（）{ case }**

switch case 语句用来判断一个变量与一系列值中某个值是否相等，每个值称为一个分支。  
当一个case成立，从这个case向后穿透所有case，包括default，直到程序结束或者遇到break程序才结束

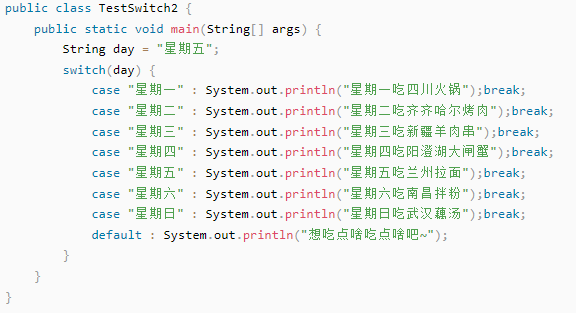


既然已经有了if else分支语句，那为什么还要switch 语句呢？

switch 语句也是一种分支语句，常常用于多分支的情况。这种多分支，一般指的是很多很多分支，而且判定条件主要以整型为主：

case本质是进行判定功能，与switch里面的语句相比较，如果相同，则执行语句，不相同则找下一个case比较，直至所有case语句都不与switch相同，此时执行default语句。

案例

****

**switch结**构的注意事项

**--switch 语句中的变量类型可以是： byte、short、int 、char、String(jdk1.7以后支持)**

**--switch 语句可以拥有多个 case 语句**

**--每个 case 后面跟一个要比较的值和冒号,且此值的数据类型必须与变量的数据类型一致**

**--当变量值与 case 语句值相等时，开始执行此case 语句的内容,执行完会判断此行代码是否有break,如果有,结束执行,如果没有,会继续向后执行穿透所有case,包括default**

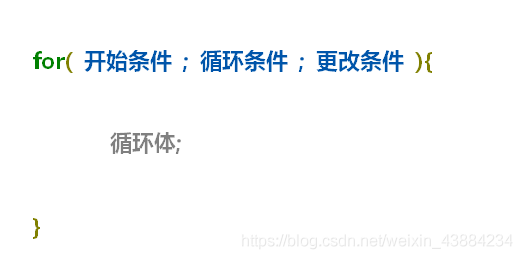
**--switch 语句可以包含一个 default 分支，该分支一般是写在switch 语句的最后**

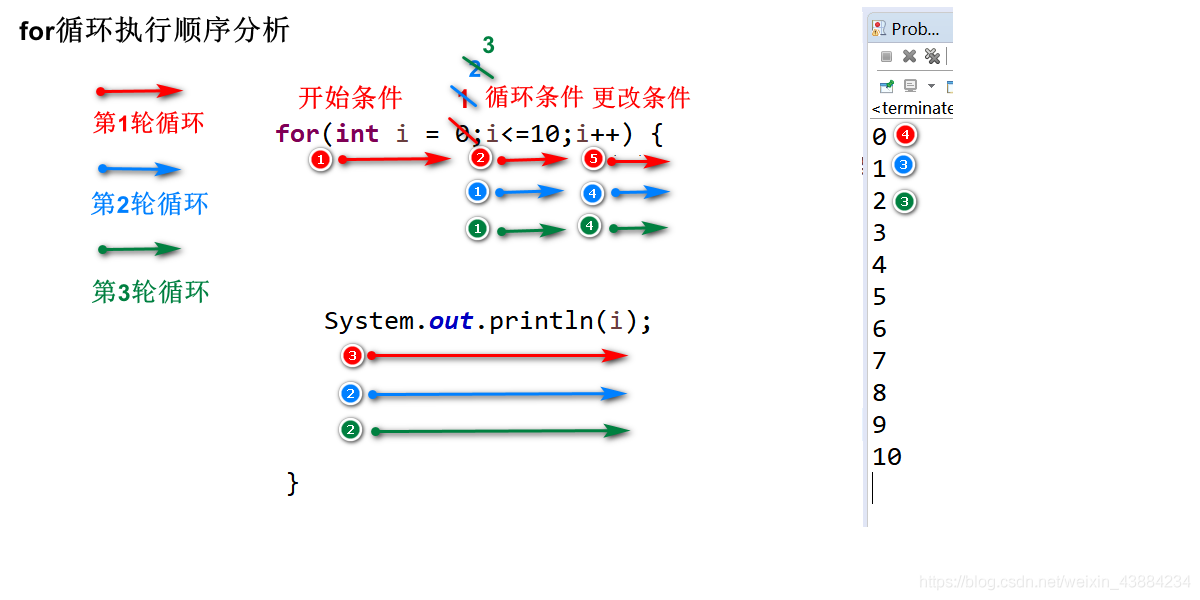
**如果在default之前的case有break,则default不会执行**

**第五个知识点循环机构**

**--for循环**

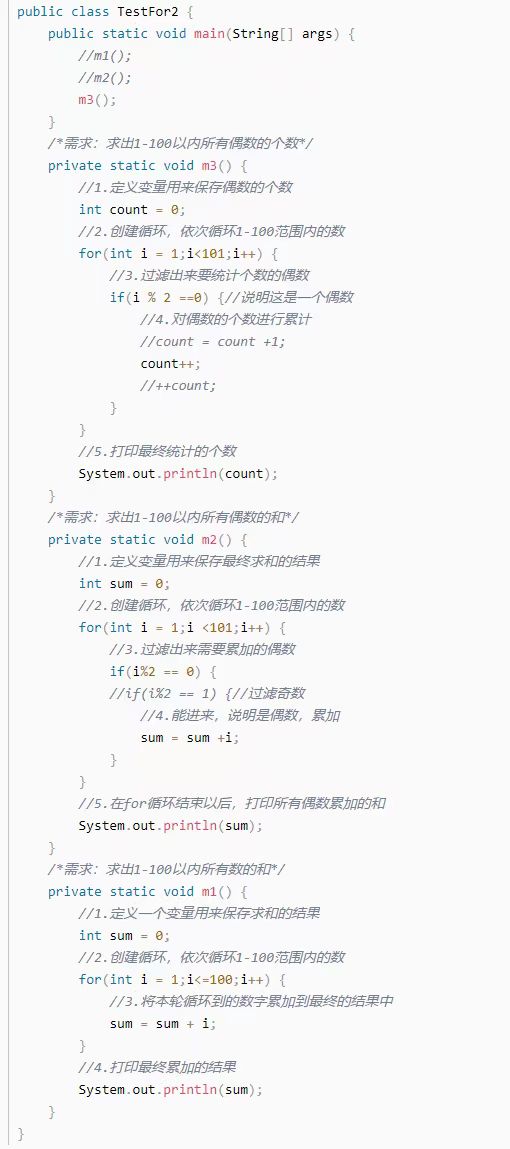
循环结构是指在程序中需要反复执行某个功能而设置的一种程序结构。  
它由循环体中的条件，判断继续执行某个功能还是退出循环。  
根据判断条件,循环结构又可细分为先判断后执行的循环结构和先执行后判断的循环结构。





--案例

↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓



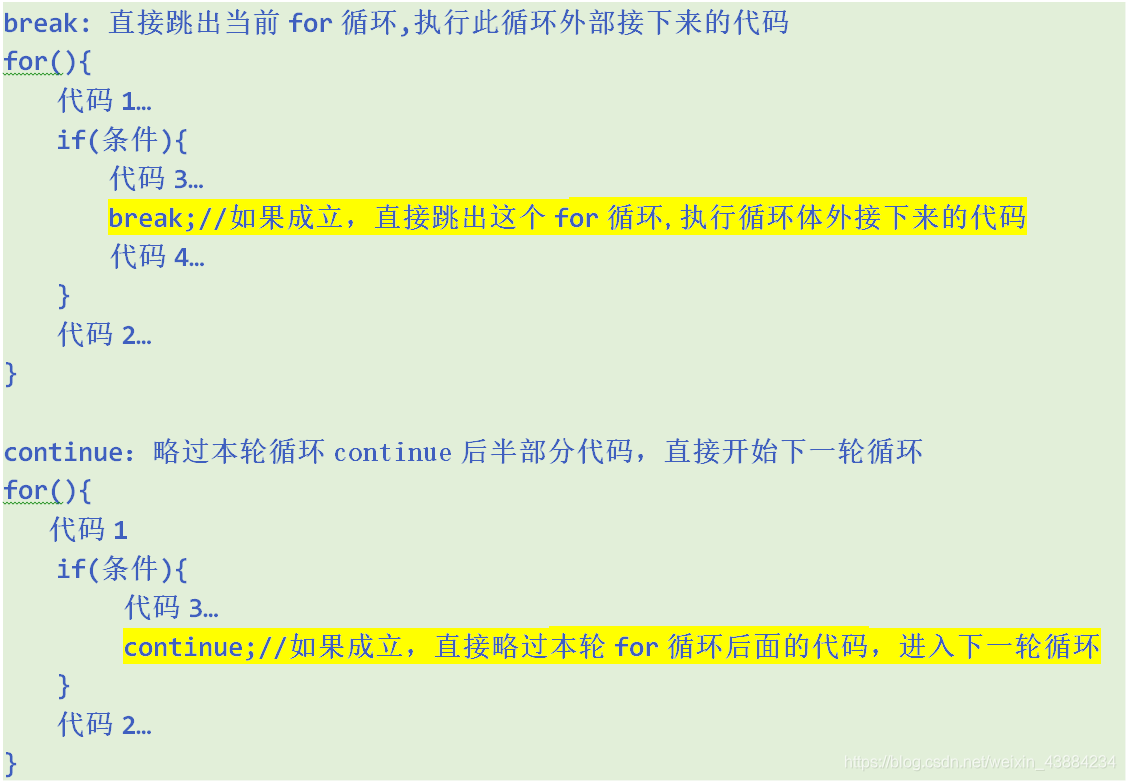
-- break continue

break以后的循环体中的语句不会继续执行,循环体外的会执行  
注意如果是嵌套for循环,在内层循环遇到了break,只会跳出当前这一层内循环哦

continue后本轮循环体中的语句不会继续执行,但是会继续执行下轮循环,循环体外的也会执行

break: 直接结束当前循环,跳出循环体,简单粗暴

continue: 跳出本轮循环,继续下一轮循环



--案例

↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓

↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓↓

