第 02 章 语言基础

Java 的三类注释

- 单行注释：// 注释；
- 多行注释：/* …*/；
- 文档注释：/** … */ ~~此类注释最后再使用~~。

Java 数据类型划分:

- 基本数据类型（只包含具体的数据，不牵扯到内存的关联）：
  - 数值型：
    - 整型：byte、short、int、long； 默认值：0
    - 实型（浮点型）：float、double； 默认值：0.0
  - 字符型：char； 默认值：'\u0000'空格
  - 布尔型：boolean； 默认值：false
- 引用数据类型（指的内存的关联数据，类似于指针的概念）：类、数组、接口。 默认值：null

~~表示整数永恒使用 int、表示小数永恒使用 double；~~
~~在进行数据传输和字符转码的过程之中都使用 byte；~~
~~表示日期时间或者是文件大小的时候都使用 long；~~
~~表示逻辑关系的时候往往都使用 boolean。~~

2.3.1 整型数据（byte、short、int、long）

Int 的数据范围 最大值：2147483647 最小值：-2147483648

自动的数据类型转换是由位数低的向位数高的转换，但是如果现在要想将位数高的向位数低的转换，则必须采用强制手段

2.3.2 实型数据（float、double）

在程序之中默认的小数类型就是 double，double 是范围最大的数据类型。

字母和数值之间的转换是有一定的数据联系的：

2.3.4 布尔型数据（boolean）

Java 里面的布尔是绝对不可能使用数字来表示的

2.3.5 字符串类型：String（~~大写的 S 非基本数据类型~~）

在数学计算之中“+”表示的是加法计算，而在字符串之中“+”表示字符串连接操作

运算符

Java 中提供了丰富的运算符，如赋值运算符、算术运算符、比较运算符等。

赋值运算符：

int a,b,c=11;// 多个变量赋值

c=b=a+4;//能否直接多个变量赋值

算术运算符

自增、自减运算符：++，--

单目运算符，可以放在操作元之前，也可以放在操作元之后。

操作元必须是一个整型或浮点型变量。作用是使变量的值增 1 或减 1，如：

++x（--x）表示在使用 x 之前，先使 x 的值增（减）1。

x++（x--）表示在使用 x 之后，使 x 的值增（减）1。

关系运算符：== != > < >= <=

逻辑运算

短路与“&&” 短路或||

复合运算符 ：c += a

三目运算符

数据类型 变量 = 布尔表达式 ? 值 1 : 值 2 ;

如果布尔表达式成立，将值 1 的数据赋予变量，否则将值 2 的数据赋予变量。

程序结构

基本控制结构：：顺序、分支、循环。

分支语句：

<table>
<tr><td colspan="3">条件分支</td></tr>
<tr><td>if 语句：</td><td>if…else 语句：</td><td>if…else if…else 语句：</td></tr>
<tr><td>if(布尔表达式) {
　　// 条件满足时执行
}</td><td>if(布尔表达式) {
　　// 条件满足时执行
} else {
　　// 条件不满足时执行
}</td><td>if(布尔表达式) {
　　// 条件满足时执行
} else if (布尔表达式) {
　　// 条件满足时执行
} ..... else {
　　// 条件不满足时执行
}</td></tr>
<tr><td colspan="3">语句：可以是一条或多条语句，当布尔表达式的值为 true 是执行这些语句。若为一条语句，则可以省略条件语句的{}。</td></tr>
</table>

<table>
<tr><td colspan="2">开关语句</td></tr>
<tr><td>switch(变量 | 常量) {
　　case 值 :
　　　　满足时执行 ;
　　　　[break] ;
　　case 值 :
　　　　满足时执行 ;
　　　　[break] ;
　　...
　　[default :
　　　　默认执行 ;
　　　　[break]]
}</td><td>switch 之中变量可以是 int 或 char 型数据、enum 类型、字符串数据。
在 switch 之中如果每一个 case 之后没有 break 语句，那么就会在第一个满足的 case 之后的所有语句都执行，一直到 break 或者是执行完毕。</td></tr>
</table>

循环语句

<table>
<tr><td>while 循环：</td><td>do…while 循环：</td></tr>
<tr><td>// 循环初始化条件
while(循环结束条件判断) {
　　// 执行语句
　　// 修改循环条件或者是循环次数的统计
}</td><td>// 循环初始化条件
do {
　　// 执行语句
　　// 修改循环条件或者是循环次数的统计
} while(循环结束条件判断) ;</td></tr>
<tr><td></td><td>至少执行一次</td></tr>
<tr><td>for (循环初始化条件;循环结束条件判断;循环条件变更) {
　　// 语句
}</td><td>循环初始化条件：通常是一个赋值语句，负责设置循环的初始值，即给循环变量赋值
循环结束条件判断：通常为一个关系表达式，用来控制循环变量和循环变量允许的范围进行比较
循环条件变更：通常为赋值语句，对控制循环的变量进行增大和减小</td></tr>
<tr><td colspan="2">关于循环的选择问题（个人总结，99%适用）
• 如果在已经明确知道循环次数的情况下，一定使用 for 循环；
• 如果不知道循环次数，但是知道循环结束条件，一定使用 while 循环。</td></tr>
</table>

方法的定义和使用

~~方法就是一段可以被重复调用的代码段，在 C 语言之中将方法又称为函数，但是在 java 开发之中千万不能说函数，永远只能说方法。~~

方法是一段可以被重复调用的代码块，利用方法可以对一些重复使用的功能进行包装，并且统一维护。

方法的定义结构如下：

```
public static 返回值类型 方法名称([数据类型 参数名称 , ...]) {
    [return [返回值] ;]
}
```

对于方法的返回值类型主要是以之前讲解的数据类型划分中的类型为主，如果现在某一个不需要返回内容，那么就使用 void 表示。

方法重载（Overload）

一些方法拥有同类功能，往往可以将其定义为一个名字，而一旦多个方法是一个名字，那么这种形式就被称为方法的重载。

方法重载：方法名称相同、参数的类型及个数、次序不同。

方法重载的概念之中，并没有针对于方法的返回值类型是否统一有明确的要求

循环语句