Java 核心技术(高阶)

第一章 Java语法糖

第四节 语法糖(3)

华东师范大学 陈良育

自动装箱和拆箱(1)

- 自动装箱和拆箱(auto-boxing/auto-unboxing)
  - 从JDK5.0开始引入，简化基本类型和对象转换的写法
  - 基本类型：boolean/byte/char/int/short/long/float/double
  - 对象：Boolean/Byte/Character/Integer/Short/Long/Float/Double

```
Integer obj1 = 5;  //自动装箱
Integer obj2 = Integer.valueOf(5);

int a1 = obj1;     //自动拆箱
int a2 = obj1.intValue();
```

```
ArrayList<Integer> list = new ArrayList<>();
list.add(1);
list.add(Integer.valueOf(2));

int a3 = list.get(1);
int a4 = list.get(1).intValue();
```

自动装箱和拆箱(2)

- 自动装箱和拆箱的注意事项
  - 装箱和拆箱是编译器的工作，在class中已经添加转化。虚拟机没有自动装箱和拆箱的语句。
  - ==：基本类型是内容相同，对象是指针是否相同(内存同一个区域)
  - 基本类型没有空值，对象有null，可能触发NullPointerException。
  - 当一个基础数据类型与封装类进行==、+、-、*、/运算时，会将封装类进行拆箱，对基础数据类型进行运算。
  - 谨慎使用多个非同类的数值类对象进行运算。

多异常并列(1)

- 多个异常并列在一个catch中
  - 从JDK7.0开始引入，简化写法

```
try
{
    test();
}
catch(IOException ex)
{
    //异常处理
}
catch(SQLException ex)
{
    //异常处理
}
```

```
try
{
    test();
}
catch(IOException | SQLException ex)
{
    //JDK7开始，支持一个catch写多个异常
    //异常处理
}
```

多异常并列(2)

- 多个异常并列在一个catch中
  - 多个异常之间不能有(直接/间接)继承关系，如果有，则报错

```
try
{
    test2();
}
catch(IOException | FileNotFoundException ex)
{
    //JDK7开始，支持一个catch写多个异常
    //异常处理
}
```

整数类型用二进制数赋值

- Java 7的新语法：整数类型用二进制数赋值
  - 避免二进制计算
  - byte/short/int/long

```
byte a1 = (byte) 0b00100001;
short a2 = (short) 0b1010000101000101;
int a3 = 0b10100001010001011010000101000101;
int a4 = 0b101;
int a5 = 0B101; //B可以大小写
long a6 = 0b10100001010001011010000101000101101000010100010110100001010001010L;

final int[] s1 = { 0b00110001, 0b01100010, 0b11000100, 0b10000100 };
```

数字中的下划线

- Java 7的新语法：在数值字面量(literal)中使用下划线
  - 增加数字的可读性和纠错功能
  - short/int/long/float/double
  - 下划线只能出现数字中间，前后必须有数字
  - 允许在二/八/十/十六进制的数字中使用

```
long a1 = 9999999999L;
long a2 = 9_999_999_999L;
```

```
int a3 = 0b0111_1011_0001; //二进制，0b开头
int a4 = 02_014;           //八进制，0开头
int a5 = 123__45;          //可以多个下划线
int a6 = 0x7_B_1;          //十六进制
float a7 = 3.56_78f;       //float
double a8 = 1.3_45__67;    //double
```

```
int b1 = 0b_123_4;   //_必须在数字之间
int b2 = 0123_4_;    //_不能在末尾
int b3 = _123;       //_不能在开头
int b4 = 0_x_123;    //不能拆开0x
int b5 = 0x_51;      //_必须在数字之间
long b6 = 1000_L;    //_必须在数字之间
float b7 = 1.34f_;   //_不能在末尾
```

总结

- 自动装箱和拆箱(5)：避免多个不同类型对象共同运算
- 多异常并列(7)：不建议使用，针对每一类异常单独处理
- 数值类型赋值优化(7)：优化显示，避免错误，建议使用

谢 谢！