BigInteger类与BigDecimal类

在Java中有两个类**BigInteger**和**BigDecimal**分别表示**大整数类和大浮点数类**，至于两个类的对象能表示最大范围不清楚，理论上能够表示无限大的数，只要计算机内存足够大。

这两个类都在java.math.\*包中，因此每次必须在开头处引用该包。

# Integer与BigInteger

## BigInteger的主要功能

主要是为了**扩大整数的范围而出现**，并提供了一些常用的数学运算方法。其中**gcd**方法获取最大公约数值得使用。

BigDecimal使用范围比BigInteger更广。

BigInteger相比Integer的确可以用big来形容。它是用于科学计算，Integer只能容纳一个int, 所以最大值也就是2的31次访减去1，十进制为2147483647，如果需要计算更大的数，那么31位显然是不够用了，**BigInteger能够容纳的位数那可就大了**。除了**容量大**之外，BigInteger还封装了一些常见的操作，比如+-\*/的基本操作，还有绝对值，相反数，最大公约数，是否是质数等等的运算。

## BigInteger的主要构造方法：

### 将String转换成BigInteger对象

**BigInteger(String val)：默认十进制**

Translates the decimal String representation of a BigInteger into a BigInteger.

BigInteger(String val, int radix)：指定进制。

Translates the String representation of a BigInteger in the specified radix into a BigInteger.

### 将byte[]字节数组转换成BigInteger对象

**BigInteger(byte[] val)**

Translates a byte array containing the two's-complement binary representation of a BigInteger into a BigInteger.

**BigInteger(int signum, byte[] magnitude)**

Translates the sign-magnitude representation of a BigInteger into a BigInteger.

**没有signum**：默认将val的第一个字节byte的第一个位作为符号位；

**带有signum**：字节数组magnitude不具有符号位，而signum为1表示正数，-1表示负数，0表示0.< -1 for negative, 0 for zero, or 1 for positive>

### 借助Random随机生成一个BigInteger

**BigInteger**(int bitLength, int certainty, Random rnd)

Constructs a randomly generated positive BigInteger that is probably prime, with the specified bitLength.

**BigInteger**(int numBits, Random rnd)

Constructs a randomly generated BigInteger, uniformly distributed over the range 0 to (2numBits - 1), inclusive.

## BigInteger的方法

### gcd方法：提供了一个最大公约数的方法；

### 提供了基本的整数运算：加减乘除、求余、绝对值、乘方等

add、substract、multiply、divide、abs、pow等

### 提供了一些Bit操作的方法，如and、or、not、xor、setBit

从BigInteger的构造方法上就可以看出，BigInteger面向的是比特位bit，因此可以获取关于bit的一些信息，如bit位数、设置bit值等；

### valueOf方法：将long转换为BigInteger；

### 转换为int值、long值：intValue、longValue等；

### 转换为字节数组byte[]：toByteArray。

# BigDecimal

## 功能介绍

**BigDecimal是解决float和double的精度丢失而出现的。**

**一般来说，BigInteger用的不是很多，BigDecimal用的稍微多一点**，就比如说JDBC中，如果一个字段的数据库类型是**Number**, 那么getObject().getClass()的结果是java.math.BigDecimal。

**BigDecimal与BigInteger的区别**：BigDecimal是基于BigInteger，通过增加一个scale参数，指定小数的位数，从而实现一个十进制的小数位，**unscaledVal × 10-scale**。

**BigDecimal的实现利用到了BigInteger, 所不同的是BigDecimal加入了小数位的概念**，比如BigDecimal d = new BigDecimal(new BigInteger(ib)，5);5表示的是5个小数位。BigDecimal可以用来做超大的浮点数的运算，比如+-\*/的运算，其中除法运算是最复杂的，因为商的位数还有除不断的情况下末位小数点的处理都是需要考虑的。

　new BigDecimal(-7.5).divide(new BigDecimal(1),0,BigDecimal.ROUND\_UP);

## BigDecimal的构造方法

### 基于BigInteger构造BigDecimal

**BigDecimal(BigInteger val)**

Translates a BigInteger into a BigDecimal.

**BigDecimal(BigInteger unscaledVal, int scale)**

Translates a BigInteger unscaled value and an int scale into a BigDecimal.

BigDecimal(BigInteger unscaledVal, int scale, MathContext mc)

Translates a BigInteger unscaled value and an int scale into a BigDecimal, with rounding according to the context settings.

BigDecimal(BigInteger val, MathContext mc)

Translates a BigInteger into a BigDecimal rounding according to the context settings.

### **基于字符数组char[]实现**

BigDecimal(char[] in)

Translates a character array representation of a BigDecimal into a BigDecimal, accepting the same sequence of characters as the BigDecimal(String) constructor.

BigDecimal(char[] in, int offset, int len)

Translates a character array representation of a BigDecimal into a BigDecimal, accepting the same sequence of characters as the BigDecimal(String) constructor, while allowing a sub-array to be specified.

BigDecimal(char[] in, int offset, int len, MathContext mc)

Translates a character array representation of a BigDecimal into a BigDecimal, accepting the same sequence of characters as the BigDecimal(String) constructor, while allowing a sub-array to be specified and with rounding according to the context settings.

BigDecimal(char[] in, MathContext mc)

Translates a character array representation of a BigDecimal into a BigDecimal, accepting the same sequence of characters as the BigDecimal(String) constructor and with rounding according to the context settings.

### 基于long值

BigDecimal(double val)

Translates a double into a BigDecimal which is the exact decimal representation of the double's binary floating-point value.

BigDecimal(double val, MathContext mc)

Translates a double into a BigDecimal, with rounding according to the context settings.

BigDecimal(int val)

Translates an int into a BigDecimal.

BigDecimal(int val, MathContext mc)

Translates an int into a BigDecimal, with rounding according to the context settings.

BigDecimal(long val)

Translates a long into a BigDecimal.

BigDecimal(long val, MathContext mc)

Translates a long into a BigDecimal, with rounding according to the context settings.

### 基于String类型构造BigDecimal

**BigDecimal(String val)**

Translates the string representation of a BigDecimal into a BigDecimal.

**BigDecimal(String val, MathContext mc)**

Translates the string representation of a BigDecimal into a BigDecimal, accepting the same strings as the BigDecimal(String) constructor, with rounding according to the context settings.

## divide的模式：

　　上面的这个运算中divide的第二个参数表示商的小数点位数，最后一个参数指的是近似处理的模式。一共有一下几个模式:

**BigDecimal.ROUND\_UP** 最后一位如果大于0，则向前进一位，正负数都如此。

**BigDecimal.ROUND\_DOWN** 最后一位不管是什么都会被舍弃。

**BigDecimal.ROUND\_CEILING** 如果是正数，按ROUND\_UP处理，如果是负数，按照ROUND\_DOWN处理。例如7.1->8; -7.1->-7;所以这种近似的结果都会>=实际值。

**BigDecimal.ROUND\_FLOOR** 跟BigDecimal\_ROUND\_CEILING相反。例如7.1->7;-7.1->-8。这种处理的结果<=实际值。

**BigDecimal.ROUND\_HALF\_DOWN** 如果最后一位<=5则舍弃，如果>5， 向前进一位。如7.5->7;7.6->8;-7.5->-7

**BigDecimal.ROUND\_HALF\_UP** 如果最后一位<5则舍弃，如果>=5， 向前进一位。反之舍弃。如7.5->8;7.4->7;-7.5->-8

**BigDecimal.ROUND\_HALF\_EVEN** 如果倒数第二位是奇数，按照BigDecimal.ROUND\_HALF\_UP处理，如果是偶数，按照BigDecimal.ROUND\_HALF\_DOWN来处理。如7.5->8;8.5->8;7.4->7;-7.5->-8

## BigDecimal的常用方法

### 提供了基本的数学运算方法：如加减乘除、取舍方法

add、substrace特别是divide方法比较复杂，提供了很多模式。

### 提供了一些获取整数与小数信息的方法：

如scale方法、signum方法。

### 转换方法：toBigInteger、valueOf方法等。