**新IO简介：**

1. 如果一个程序需要用户输入数据，则可以通过System.in来完成，但这样一来，在使 用时就会出现一个问题：如果用户没有输入信息，则程序会一直等待用户输入，大 量的系统资源都要被白白浪费。所以为了提升传统IO的操作性能，Java又提供了新 IO—NIO。
2. NIO并没有在原来的IO基础上进行开发，而是采用了全新的类和接口，它们都定义 在java.nio包中。
3. 在传统IO操作总所有的操作都是以流的形式完成的，而在NIO中所有的操作都要使 用到缓冲区。

**Buffer类：**

在传统IO操作中所有的操作都是流的方式完成的，而在NIO操作中所有的操作都要使 用到缓冲区，且所有的读写操作都是通过缓冲区完成的。缓冲区是一个线性的、有序的 数据集，只能容纳指定数据类型的数据。在Java中使用java.nio.Buffer类表示缓冲区类。

java.nio.Buffer类本身是一个抽象类，在新IO中每一个基本数据类型都有一种对应的缓

|  |  |
| --- | --- |
| 缓冲区类 | 描述 |
| java.nio.ByteBuffer | 存储字节的Buffer |
| java.nio.CharBuffer | 存储字符的Buffer |
| java.nio.ShortBuffer | 存储短整型的Buffer |
| java.nio.IntBuffer | 存储整型的Buffer |
| java.nio.LongBuffer | 存储长整型的Buffer |
| java.nio.FloatBuffer | 存储单精度浮点型的Buffer |
| java.nio.DoubleBuffer | 存储双精度浮动型的Buffer |

冲区操作类。

**缓冲区状态变量：**

在Buffer类中存在一系列的状态变量，这些状态变量随着写入或读取都有可能会被改变。 在缓冲区中可以使用3个值表示缓冲区的状态。

1. position：表示下一个读取或写入的操作指针，当向缓冲区中写入数据时此指针就会 改变。
2. limit：表示还有多少个数据需要存储或者需要读取
3. capacity：表示缓冲区的最大容量，向缓冲区写入内容不能超过容量，否则会出现 异常。

当调用flip( )方法后，limit设置为position，而position将重新设置为0

**创建子缓冲区：** 1）可以调用缓冲区类中的slice( )方法创建该缓冲区的子缓冲区，子缓冲区与原缓 冲区中的部分数据可以共享，即子缓冲区的内容改变后主缓冲区中的内容也随之改 变。

2）使用slice( )方法创建子缓冲区时是把position与limit之间的内容给子缓冲区。

**创建只读缓冲区：** 1）如果需要使用到缓冲区的内容，但又不希望其内容被修改，则可以通过 asReadOnlyBuffer( )方法创建该缓冲区的只读缓冲区。

1. 只读缓冲区只可以被读取，不可以被修改，否该将出现异常。

**创建直接缓冲区：** 1）在所有的缓冲区操作类中，只有ByteBuffer类可以创建直接缓冲区，直接缓冲区更 能提高操作效率。

2）创建直接缓冲区直接使用ByteBuffer类中定义的allocateDirect( )方法，其定义为：

**public static ByteBuffer allocateDirect(int capacity);**

**通道：**

1）通道是用来读取和写入数据的，类似与输入输出流。但不能直接操作通道，通道必 须通过缓冲区完成读取和写入操作。

2）IO流分为输入输出流，而通道是双向的，既可以完成输入操作也可以完成输出操作。

3）在Java中使用Channel接口表示通道，Channel接口定义在java.nio.channels包中。

**FileChannel类：**

FileChannel类是Channel类的子类，用来进行文件的读写操作。

如果要使用FileChannel类，则必须依靠FileInputStream类或FileOutputStream类中 getChannel( )方法取的输入或输出的通道。

**内存映射：**

内存映射可以进行文件的读写操作，并且使用内存映射来读取文件的速度是最快的。

要想将实现内存映射的操作，则需要依靠MappedByteBuffer类，通过FinalChannel类提 供的map( )方法为MappedByteBuffer类对象实例化，在调用map( )方法时需要指定映射 模式、指针位置，读取长度。

**文件锁：**

当一个线程将文件锁定后，其他线程是无法操作此文件的。要想实现文件锁，则需要使 用FileLock类完成，此类对象需要依靠FileChannel类进行实例化操作。

文件锁有两种：

·共享锁：允许多个线程进行文件的读取操作

·独占锁：只允许一个文件进行文件的读/写操作

**字符集Charset类：**

1. 在Java中所有的信息都是以UNICODE进行编码。
2. 在Java的NIO中提供了Charset类来负责编码的问题，该类对象可以创建解码器 （CharsetEncoder类对象）和编码器（CharsetDecoder类对象），

3）使用Charset类中的静态方法availableCharset( )可以取得Charset类支持的全部编码，返回一个SortedMap<String,Charset>类集合。

3）编码器的功能是将一个字符型缓冲区进行编码并返回一个Byte型缓冲区；解码器的 功能是将编码后返回的Byte型缓冲区进行解码并返回一个字符型缓冲区。

**选择器Selector类：**

在使用IO构造网络服务时，所有的网络服务将使用阻塞的方式进行客户端的连接，而 使用新IO则可以构造一个非阻塞的网络服务，建立一个异步的、非阻塞的且可以多个 客户端可以连接的服务器。