Buffer中状态属性和关系图解

Buffer线程不安全的

```
* <blockquote>
*
*   <p> A buffer's <i>capacity</i> is the number of elements it contains.  The
*   capacity of a buffer is never negative and never changes.  </p>
*
*   <p> A buffer's <i>limit</i> is the index of the first element that  should
*   not be read or written.  A buffer's limit is never negative and is never
*   greater than its capacity.  </p>
*
*   <p> A buffer's <i>position</i> is the index of the next element to be
*   read or written.  A buffer's position is never negative and is never
*   greater than its limit.  </p>
*
* </blockquote>
```

1.position

下一个可读写的位置（下一个可读写元素索引），不能超过limit

2.limit

第一个不能被读写的元素索引（最后一个可以读写的下一个元素），不能超过capacity

3.capacity

容量大小

4.mark

标记点

5.reset()

调用后position=mark

6.flip()

翻转状态，将position赋给limit，position(可读写的元素索引)置为0，mark为-1

```
public final Buffer flip() {
 limit = position;
 position = 0;
 mark = -1;
 return this;
}
```

7.clear()

```
public final Buffer clear() {
 position = 0;
 limit = capacity;
 mark = -1;
 return this;
}//position=0,limit=capacity
```

8.rewind()

```
public final Buffer rewind() {
 position = 0;
 mark = -1;
 return this;
}//position=0,limit不变
```

Buffer基本操作

```
public class NioTest_1 {
 public static void main(String[] args) {
 IntBuffer buffer = IntBuffer.allocate(10);//分配10个长度的缓冲区

 for (int i=0;i<buffer.capacity();i++){
 int randomNumber = new SecureRandom().nextInt(20);//生成20以内的随机数
 buffer.put(randomNumber);//放入缓冲区，写入
 }

  buffer.flip();//状态翻转，切换读写状态（标致变化）

  while (buffer.hasRemaining()){//循环取出，读取
  System.out.println(buffer.get());
  }
  }
}
```

Buffer读文件操作

```
public class NioTest_2 {
```

```
2   public static void main(String[] args) throws Exception {
3   try (
4   FileInputStream fileInputStream = new FileInputStream("src/main/java/com/zhangtianyi/nio/resource/NioTest_2.txt");
5   ){
6   FileChannel fileChannel = fileInputStream.getChannel();
7
8   ByteBuffer byteBuffer = ByteBuffer.allocate(512);
9
10   fileChannel.read(byteBuffer);
11
12   byteBuffer.flip();
13
14   while (byteBuffer.remaining()>0){
15   byte b = byteBuffer.get();
16   System.out.println("Character:"+(char)b);
17   }
18
19   }
20   }
21  }
```

Buffer写文件操作

```
1  public class NioTest_3 {
2   public static void main(String[] args) throws Exception{
3
4   try(
5   FileOutputStream fileOutputStream = new FileOutputStream("src/main/java/com/zhangtianyi/nio/resource/NioTest_3.txt");
6   ){
7   FileChannel fileChannel = fileOutputStream.getChannel();
8   ByteBuffer byteBuffer = ByteBuffer.allocate(512);
9   byte[] bytes = "hello nio,ni hao".getBytes(Charset.defaultCharset());
10
11   for (int i = 0; i < bytes.length; i++) {
12   byteBuffer.put(bytes[i]);
13   }
14
15   byteBuffer.flip();
16   fileChannel.write(byteBuffer);
17   }
18   }
```

```
}
```

Buffer关键属性测试

```
public class NioTest_1 {
 public static void main(String[] args) {
 IntBuffer buffer = IntBuffer.allocate(10);//分配10个长度的缓冲区

 for (int i=0;i<5;i++){
 int randomNumber = new SecureRandom().nextInt(20);//生成20以内的随机数
 buffer.put(randomNumber);//放入缓冲区，写入
 }
 System.out.println("翻转前limit:"+buffer.limit());
  buffer.flip();//状态翻转，切换读写状态（标致变化）
  System.out.println("翻转后limit:"+buffer.limit());

  while (buffer.hasRemaining()){//循环取出，读取
  System.out.println("position:"+buffer.position());
  System.out.println("limit:"+buffer.limit());
  System.out.println("capacity:"+buffer.capacity());

  System.out.println(buffer.get());
  }
  }
}
```