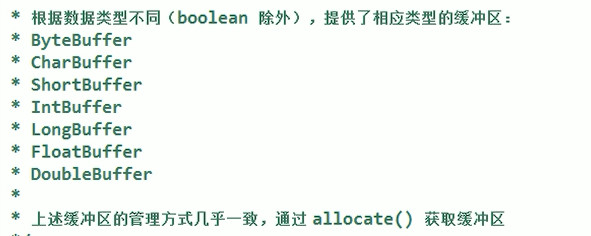
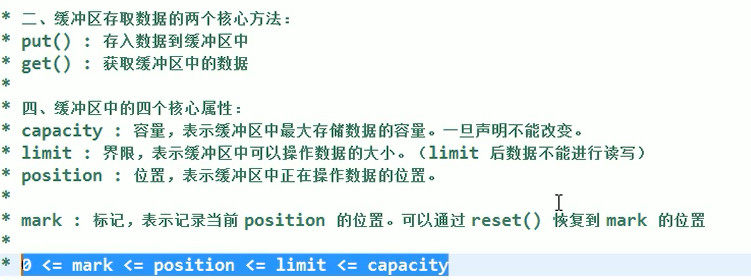
一、缓冲区：在JavaNIO中负责数据的存取。缓冲区就是数组。用于存储不同类型的数据。

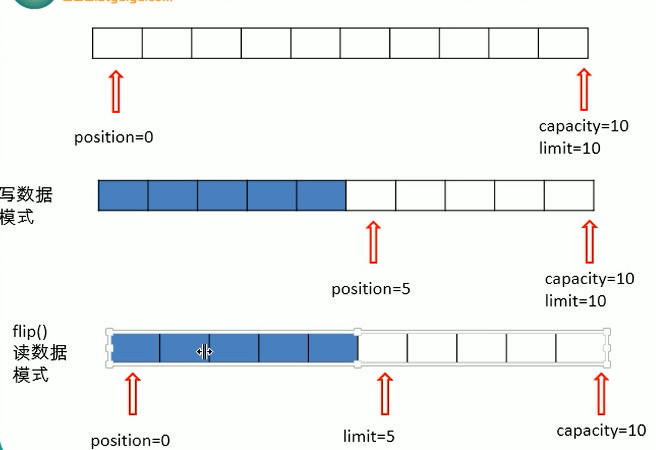
根据数据类型不同，提供了相应类型的缓冲区（boolean类型除外）：

感觉使用缓冲区后，我们能够一定的自主的进行读写控制。



问题：NIO为什么需要这么多缓冲区。





Flip()是从写模式切换到读模式，rewind()是从读模式切换到重读模式。

**问题：不都是改变指针的位置吗，为什么还要成两个方法，而且如果是从读模式切换到写模式呢？**

答：我觉得是面向对象的关系，即使他们底层调用的是同一个方法，但是也通过不同接口进行调用，从而更加清楚的表达上层逻辑。第二个问题我有些不确定。

**Clear()方法，清空缓冲区数据，但是并不是真正的清除，只是重置指针，将指针的状态转换为新建时候的状态（position、limit、capacity）位置，但是看它指针的位置，还是能进行读取的，只是你不太可能准确的读取到自己想要的数据（因为你无法准确的知道缓冲区中数据的大小，唯一的limit标识已经被重置了，我是这么觉得的，因为它只是改变了指针的位置，其实我们的新建的缓冲区应该也是有数据的。）**



**Mark（）方法将当前的position指针进行存储，当使用reset（）方法时，缓冲区会将position的值复原。（所以Buffer中应该还有一个存储最近一次标识position的值的变量）**

**hasRemaining()方法进行将判断缓冲区中是否还有没有剩余的数据返回Boolean值。如果有，可以通过remaining（）方法得到剩余的数据的数量个数（这个数量根据什么进行计算，要不要考虑buffer的类型？）。**