Module java.base

Package java.nio

`package java.nio`

Defines buffers, which are containers for data, and provides an overview of the other NIO packages.

The central abstractions of the NIO APIs are:

- *Buffers*, which are containers for data;
- *Charsets* and their associated *decoders* and *encoders*, which translate between bytes and Unicode characters;
- *Channels* of various types, which represent connections to entities capable of performing I/O operations; and
- *Selectors* and *selection keys*, which together with *selectable channels* define a multiplexed, non-blocking I/O facility.
- *Path*, which together with the *Files* class provides access to files.

The `java.nio` package defines the buffer classes, which are used throughout the NIO APIs. The charset API is defined in the `java.nio.charset` package, the channel and selector APIs in the `java.nio.channels` package, and the files and path APIs in the `java.nio.file` package. Each of these subpackages has its own service-provider interface (SPI) subpackage, the contents of which can be used to extend the platform's default implementations or to construct alternative implementations.

Buffers	Description
Buffer	Position, limit, and capacity; clear, flip, rewind, and mark/reset
ByteBuffer	Get/put, compact, views; allocate, wrap
MappedByteBuffer	A byte buffer mapped to a file
CharBuffer	Get/put, compact; allocate, wrap
DoubleBuffer	Get/put, compact; allocate, wrap
FloatBuffer	Get/put, compact; allocate, wrap
IntBuffer	Get/put, compact; allocate, wrap

Buffers	Description
LongBuffer	Get/put, compact; allocate, wrap
ShortBuffer	Get/put, compact; allocate, wrap
ByteOrder	Typesafe enumeration for byte orders

A *buffer* is a container for a fixed amount of data of a specific primitive type. In addition to its content a buffer has a *position*, which is the index of the next element to be read or written, and a *limit*, which is the index of the first element that should not be read or written. The base `Buffer` class defines these properties as well as methods for *clearing*, *flipping*, and *rewinding*, for *marking* the current position, and for *resetting* the position to the previous mark.

There is a buffer class for each non-boolean primitive type. Each class defines a family of *get* and *put* methods for moving data out of and in to a buffer, methods for *compacting*, *duplicating*, and *slicing* a buffer, and static methods for *allocating* a new buffer as well as for *wrapping* an existing array into a buffer.

Byte buffers are distinguished in that they can be used as the sources and targets of I/O operations. They also support several features not found in the other buffer classes:

- A byte buffer can be allocated as a *direct* buffer, in which case the Java virtual machine will make a best effort to perform native I/O operations directly upon it.
- A byte buffer can be created by *`mapping`* a region of a file directly into memory, in which case a few additional file-related operations defined in the `MappedByteBuffer` class are available.
- A byte buffer provides access to its content as either a heterogeneous or homogeneous sequence of *binary data* of any non-boolean primitive type, in either big-endian or little-endian byte order.

Unless otherwise noted, passing a `null` argument to a constructor or method in any class or interface in this package will cause a `NullPointerException` to be thrown.

Since:

1.4

Related Packages

Package	Description
java.nio.channels	Defines channels, which represent connections to entities that are capable of performing I/O operations, such as files and sockets; defines selectors, for multiplexed, non-blocking I/O operations.
java.nio.charset	Defines charsets, decoders, and encoders, for translating between bytes and Unicode characters.

java.nio.file	Defines interfaces and classes for the Java virtual machine to access files, file attributes, and file systems.

All Classes and Interfaces | **Classes** | **Exception Classes**

Class	Description
Buffer	A container for data of a specific primitive type.
BufferOverflowException	Unchecked exception thrown when a relative *put* operation reaches the target buffer's limit.
BufferUnderflowException	Unchecked exception thrown when a relative *get* operation reaches the source buffer's limit.
ByteBuffer	A byte buffer.
ByteOrder	A typesafe enumeration for byte orders.
CharBuffer	A char buffer.
DoubleBuffer	A double buffer.
FloatBuffer	A float buffer.
IntBuffer	An int buffer.
InvalidMarkException	Unchecked exception thrown when an attempt is made to reset a buffer when its mark is not defined.
LongBuffer	A long buffer.
MappedByteBuffer	A direct byte buffer whose content is a memory-mapped region of a file.
ReadOnlyBufferException	Unchecked exception thrown when a content-mutation method such as `put` or `compact` is invoked upon a read-only buffer.
ShortBuffer	A short buffer.