| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/SocketChannel.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/java/nio/channels/ServerSocketChannel.html)   [**NEXT CLASS**](http://docs.google.com/java/nio/channels/UnresolvedAddressException.html) | [**FRAMES**](http://docs.google.com/index.html?java/nio/channels/SocketChannel.html)    [**NO FRAMES**](http://docs.google.com/SocketChannel.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#17dp8vu) | [METHOD](#26in1rg) |

## **java.nio.channels**

Class SocketChannel

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 [java.nio.channels.spi.AbstractInterruptibleChannel](http://docs.google.com/java/nio/channels/spi/AbstractInterruptibleChannel.html)  
 [java.nio.channels.SelectableChannel](http://docs.google.com/java/nio/channels/SelectableChannel.html)  
 [java.nio.channels.spi.AbstractSelectableChannel](http://docs.google.com/java/nio/channels/spi/AbstractSelectableChannel.html)  
 **java.nio.channels.SocketChannel**

**All Implemented Interfaces:** [Closeable](http://docs.google.com/java/io/Closeable.html), [ByteChannel](http://docs.google.com/java/nio/channels/ByteChannel.html), [Channel](http://docs.google.com/java/nio/channels/Channel.html), [GatheringByteChannel](http://docs.google.com/java/nio/channels/GatheringByteChannel.html), [InterruptibleChannel](http://docs.google.com/java/nio/channels/InterruptibleChannel.html), [ReadableByteChannel](http://docs.google.com/java/nio/channels/ReadableByteChannel.html), [ScatteringByteChannel](http://docs.google.com/java/nio/channels/ScatteringByteChannel.html), [WritableByteChannel](http://docs.google.com/java/nio/channels/WritableByteChannel.html)

public abstract class **SocketChannel**extends [AbstractSelectableChannel](http://docs.google.com/java/nio/channels/spi/AbstractSelectableChannel.html)implements [ByteChannel](http://docs.google.com/java/nio/channels/ByteChannel.html), [ScatteringByteChannel](http://docs.google.com/java/nio/channels/ScatteringByteChannel.html), [GatheringByteChannel](http://docs.google.com/java/nio/channels/GatheringByteChannel.html)

A selectable channel for stream-oriented connecting sockets.

Socket channels are not a complete abstraction of connecting network sockets. Binding, shutdown, and the manipulation of socket options must be done through an associated [Socket](http://docs.google.com/java/net/Socket.html) object obtained by invoking the [socket](http://docs.google.com/java/nio/channels/SocketChannel.html#socket()) method. It is not possible to create a channel for an arbitrary, pre-existing socket, nor is it possible to specify the [SocketImpl](http://docs.google.com/java/net/SocketImpl.html) object to be used by a socket associated with a socket channel.

A socket channel is created by invoking one of the [open](http://docs.google.com/java/nio/channels/SocketChannel.html#open()) methods of this class. A newly-created socket channel is open but not yet connected. An attempt to invoke an I/O operation upon an unconnected channel will cause a [NotYetConnectedException](http://docs.google.com/java/nio/channels/NotYetConnectedException.html) to be thrown. A socket channel can be connected by invoking its [connect](http://docs.google.com/java/nio/channels/SocketChannel.html#connect(java.net.SocketAddress)) method; once connected, a socket channel remains connected until it is closed. Whether or not a socket channel is connected may be determined by invoking its [isConnected](http://docs.google.com/java/nio/channels/SocketChannel.html#isConnected()) method.

Socket channels support *non-blocking connection:* A socket channel may be created and the process of establishing the link to the remote socket may be initiated via the [connect](http://docs.google.com/java/nio/channels/SocketChannel.html#connect(java.net.SocketAddress)) method for later completion by the [finishConnect](http://docs.google.com/java/nio/channels/SocketChannel.html#finishConnect()) method. Whether or not a connection operation is in progress may be determined by invoking the [isConnectionPending](http://docs.google.com/java/nio/channels/SocketChannel.html#isConnectionPending()) method.

The input and output sides of a socket channel may independently be *shut down* without actually closing the channel. Shutting down the input side of a channel by invoking the [shutdownInput](http://docs.google.com/java/net/Socket.html#shutdownInput()) method of an associated socket object will cause further reads on the channel to return -1, the end-of-stream indication. Shutting down the output side of the channel by invoking the [shutdownOutput](http://docs.google.com/java/net/Socket.html#shutdownOutput()) method of an associated socket object will cause further writes on the channel to throw a [ClosedChannelException](http://docs.google.com/java/nio/channels/ClosedChannelException.html).

Socket channels support *asynchronous shutdown,* which is similar to the asynchronous close operation specified in the [Channel](http://docs.google.com/java/nio/channels/Channel.html) class. If the input side of a socket is shut down by one thread while another thread is blocked in a read operation on the socket's channel, then the read operation in the blocked thread will complete without reading any bytes and will return -1. If the output side of a socket is shut down by one thread while another thread is blocked in a write operation on the socket's channel, then the blocked thread will receive an [AsynchronousCloseException](http://docs.google.com/java/nio/channels/AsynchronousCloseException.html).

Socket channels are safe for use by multiple concurrent threads. They support concurrent reading and writing, though at most one thread may be reading and at most one thread may be writing at any given time. The [connect](http://docs.google.com/java/nio/channels/SocketChannel.html#connect(java.net.SocketAddress)) and [finishConnect](http://docs.google.com/java/nio/channels/SocketChannel.html#finishConnect()) methods are mutually synchronized against each other, and an attempt to initiate a read or write operation while an invocation of one of these methods is in progress will block until that invocation is complete.

**Since:** 1.4

| **Constructor Summary** | |
| --- | --- |
| protected | [**SocketChannel**](http://docs.google.com/java/nio/channels/SocketChannel.html#SocketChannel(java.nio.channels.spi.SelectorProvider))([SelectorProvider](http://docs.google.com/java/nio/channels/spi/SelectorProvider.html) provider)            Initializes a new instance of this class. |

| **Method Summary** | |
| --- | --- |
| abstract  boolean | [**connect**](http://docs.google.com/java/nio/channels/SocketChannel.html#connect(java.net.SocketAddress))([SocketAddress](http://docs.google.com/java/net/SocketAddress.html) remote)            Connects this channel's socket. |
| abstract  boolean | [**finishConnect**](http://docs.google.com/java/nio/channels/SocketChannel.html#finishConnect())()            Finishes the process of connecting a socket channel. |
| abstract  boolean | [**isConnected**](http://docs.google.com/java/nio/channels/SocketChannel.html#isConnected())()            Tells whether or not this channel's network socket is connected. |
| abstract  boolean | [**isConnectionPending**](http://docs.google.com/java/nio/channels/SocketChannel.html#isConnectionPending())()            Tells whether or not a connection operation is in progress on this channel. |
| static [SocketChannel](http://docs.google.com/java/nio/channels/SocketChannel.html) | [**open**](http://docs.google.com/java/nio/channels/SocketChannel.html#open())()            Opens a socket channel. |
| static [SocketChannel](http://docs.google.com/java/nio/channels/SocketChannel.html) | [**open**](http://docs.google.com/java/nio/channels/SocketChannel.html#open(java.net.SocketAddress))([SocketAddress](http://docs.google.com/java/net/SocketAddress.html) remote)            Opens a socket channel and connects it to a remote address. |
| abstract  int | [**read**](http://docs.google.com/java/nio/channels/SocketChannel.html#read(java.nio.ByteBuffer))([ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html) dst)            Reads a sequence of bytes from this channel into the given buffer. |
| long | [**read**](http://docs.google.com/java/nio/channels/SocketChannel.html#read(java.nio.ByteBuffer%5B%5D))([ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html)[] dsts)            Reads a sequence of bytes from this channel into the given buffers. |
| abstract  long | [**read**](http://docs.google.com/java/nio/channels/SocketChannel.html#read(java.nio.ByteBuffer%5B%5D,%20int,%20int))([ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html)[] dsts, int offset, int length)            Reads a sequence of bytes from this channel into a subsequence of the given buffers. |
| abstract  [Socket](http://docs.google.com/java/net/Socket.html) | [**socket**](http://docs.google.com/java/nio/channels/SocketChannel.html#socket())()            Retrieves a socket associated with this channel. |
| int | [**validOps**](http://docs.google.com/java/nio/channels/SocketChannel.html#validOps())()            Returns an operation set identifying this channel's supported operations. |
| abstract  int | [**write**](http://docs.google.com/java/nio/channels/SocketChannel.html#write(java.nio.ByteBuffer))([ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html) src)            Writes a sequence of bytes to this channel from the given buffer. |
| long | [**write**](http://docs.google.com/java/nio/channels/SocketChannel.html#write(java.nio.ByteBuffer%5B%5D))([ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html)[] srcs)            Writes a sequence of bytes to this channel from the given buffers. |
| abstract  long | [**write**](http://docs.google.com/java/nio/channels/SocketChannel.html#write(java.nio.ByteBuffer%5B%5D,%20int,%20int))([ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html)[] srcs, int offset, int length)            Writes a sequence of bytes to this channel from a subsequence of the given buffers. |

| **Methods inherited from class java.nio.channels.spi.**[**AbstractSelectableChannel**](http://docs.google.com/java/nio/channels/spi/AbstractSelectableChannel.html) |
| --- |
| [blockingLock](http://docs.google.com/java/nio/channels/spi/AbstractSelectableChannel.html#blockingLock()), [configureBlocking](http://docs.google.com/java/nio/channels/spi/AbstractSelectableChannel.html#configureBlocking(boolean)), [implCloseChannel](http://docs.google.com/java/nio/channels/spi/AbstractSelectableChannel.html#implCloseChannel()), [implCloseSelectableChannel](http://docs.google.com/java/nio/channels/spi/AbstractSelectableChannel.html#implCloseSelectableChannel()), [implConfigureBlocking](http://docs.google.com/java/nio/channels/spi/AbstractSelectableChannel.html#implConfigureBlocking(boolean)), [isBlocking](http://docs.google.com/java/nio/channels/spi/AbstractSelectableChannel.html#isBlocking()), [isRegistered](http://docs.google.com/java/nio/channels/spi/AbstractSelectableChannel.html#isRegistered()), [keyFor](http://docs.google.com/java/nio/channels/spi/AbstractSelectableChannel.html#keyFor(java.nio.channels.Selector)), [provider](http://docs.google.com/java/nio/channels/spi/AbstractSelectableChannel.html#provider()), [register](http://docs.google.com/java/nio/channels/spi/AbstractSelectableChannel.html#register(java.nio.channels.Selector,%20int,%20java.lang.Object)) |

| **Methods inherited from class java.nio.channels.**[**SelectableChannel**](http://docs.google.com/java/nio/channels/SelectableChannel.html) |
| --- |
| [register](http://docs.google.com/java/nio/channels/SelectableChannel.html#register(java.nio.channels.Selector,%20int)) |

| **Methods inherited from class java.nio.channels.spi.**[**AbstractInterruptibleChannel**](http://docs.google.com/java/nio/channels/spi/AbstractInterruptibleChannel.html) |
| --- |
| [begin](http://docs.google.com/java/nio/channels/spi/AbstractInterruptibleChannel.html#begin()), [close](http://docs.google.com/java/nio/channels/spi/AbstractInterruptibleChannel.html#close()), [end](http://docs.google.com/java/nio/channels/spi/AbstractInterruptibleChannel.html#end(boolean)), [isOpen](http://docs.google.com/java/nio/channels/spi/AbstractInterruptibleChannel.html#isOpen()) |

| **Methods inherited from class java.lang.**[**Object**](http://docs.google.com/java/lang/Object.html) |
| --- |
| [clone](http://docs.google.com/java/lang/Object.html#clone()), [equals](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [finalize](http://docs.google.com/java/lang/Object.html#finalize()), [getClass](http://docs.google.com/java/lang/Object.html#getClass()), [hashCode](http://docs.google.com/java/lang/Object.html#hashCode()), [notify](http://docs.google.com/java/lang/Object.html#notify()), [notifyAll](http://docs.google.com/java/lang/Object.html#notifyAll()), [toString](http://docs.google.com/java/lang/Object.html#toString()), [wait](http://docs.google.com/java/lang/Object.html#wait()), [wait](http://docs.google.com/java/lang/Object.html#wait(long)), [wait](http://docs.google.com/java/lang/Object.html#wait(long,%20int)) |

| **Methods inherited from interface java.nio.channels.**[**Channel**](http://docs.google.com/java/nio/channels/Channel.html) |
| --- |
| [close](http://docs.google.com/java/nio/channels/Channel.html#close()), [isOpen](http://docs.google.com/java/nio/channels/Channel.html#isOpen()) |

| **Constructor Detail** |
| --- |

### SocketChannel

protected **SocketChannel**([SelectorProvider](http://docs.google.com/java/nio/channels/spi/SelectorProvider.html) provider)

Initializes a new instance of this class.

| **Method Detail** |
| --- |

### open

public static [SocketChannel](http://docs.google.com/java/nio/channels/SocketChannel.html) **open**()  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Opens a socket channel.

The new channel is created by invoking the [openSocketChannel](http://docs.google.com/java/nio/channels/spi/SelectorProvider.html#openSocketChannel()) method of the system-wide default [SelectorProvider](http://docs.google.com/java/nio/channels/spi/SelectorProvider.html) object.

**Returns:**A new socket channel **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - If an I/O error occurs

### open

public static [SocketChannel](http://docs.google.com/java/nio/channels/SocketChannel.html) **open**([SocketAddress](http://docs.google.com/java/net/SocketAddress.html) remote)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Opens a socket channel and connects it to a remote address.

This convenience method works as if by invoking the [open()](http://docs.google.com/java/nio/channels/SocketChannel.html#open()) method, invoking the [connect](http://docs.google.com/java/nio/channels/SocketChannel.html#connect(java.net.SocketAddress)) method upon the resulting socket channel, passing it remote, and then returning that channel.

**Parameters:**remote - The remote address to which the new channel is to be connected **Throws:** [AsynchronousCloseException](http://docs.google.com/java/nio/channels/AsynchronousCloseException.html) - If another thread closes this channel while the connect operation is in progress [ClosedByInterruptException](http://docs.google.com/java/nio/channels/ClosedByInterruptException.html) - If another thread interrupts the current thread while the connect operation is in progress, thereby closing the channel and setting the current thread's interrupt status [UnresolvedAddressException](http://docs.google.com/java/nio/channels/UnresolvedAddressException.html) - If the given remote address is not fully resolved [UnsupportedAddressTypeException](http://docs.google.com/java/nio/channels/UnsupportedAddressTypeException.html) - If the type of the given remote address is not supported [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - If a security manager has been installed and it does not permit access to the given remote endpoint [IOException](http://docs.google.com/java/io/IOException.html) - If some other I/O error occurs

### validOps

public final int **validOps**()

Returns an operation set identifying this channel's supported operations.

Socket channels support connecting, reading, and writing, so this method returns ([SelectionKey.OP\_CONNECT](http://docs.google.com/java/nio/channels/SelectionKey.html#OP_CONNECT) | [SelectionKey.OP\_READ](http://docs.google.com/java/nio/channels/SelectionKey.html#OP_READ) | [SelectionKey.OP\_WRITE](http://docs.google.com/java/nio/channels/SelectionKey.html#OP_WRITE)).

**Specified by:**[validOps](http://docs.google.com/java/nio/channels/SelectableChannel.html#validOps()) in class [SelectableChannel](http://docs.google.com/java/nio/channels/SelectableChannel.html) **Returns:**The valid-operation set

### socket

public abstract [Socket](http://docs.google.com/java/net/Socket.html) **socket**()

Retrieves a socket associated with this channel.

The returned object will not declare any public methods that are not declared in the [Socket](http://docs.google.com/java/net/Socket.html) class.

**Returns:**A socket associated with this channel

### isConnected

public abstract boolean **isConnected**()

Tells whether or not this channel's network socket is connected.

**Returns:**true if, and only if, this channel's network socket is connected

### isConnectionPending

public abstract boolean **isConnectionPending**()

Tells whether or not a connection operation is in progress on this channel.

**Returns:**true if, and only if, a connection operation has been initiated on this channel but not yet completed by invoking the [finishConnect](http://docs.google.com/java/nio/channels/SocketChannel.html#finishConnect()) method

### connect

public abstract boolean **connect**([SocketAddress](http://docs.google.com/java/net/SocketAddress.html) remote)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Connects this channel's socket.

If this channel is in non-blocking mode then an invocation of this method initiates a non-blocking connection operation. If the connection is established immediately, as can happen with a local connection, then this method returns true. Otherwise this method returns false and the connection operation must later be completed by invoking the [finishConnect](http://docs.google.com/java/nio/channels/SocketChannel.html#finishConnect()) method.

If this channel is in blocking mode then an invocation of this method will block until the connection is established or an I/O error occurs.

This method performs exactly the same security checks as the [Socket](http://docs.google.com/java/net/Socket.html) class. That is, if a security manager has been installed then this method verifies that its [checkConnect](http://docs.google.com/java/lang/SecurityManager.html#checkConnect(java.lang.String,%20int)) method permits connecting to the address and port number of the given remote endpoint.

This method may be invoked at any time. If a read or write operation upon this channel is invoked while an invocation of this method is in progress then that operation will first block until this invocation is complete. If a connection attempt is initiated but fails, that is, if an invocation of this method throws a checked exception, then the channel will be closed.

**Parameters:**remote - The remote address to which this channel is to be connected **Returns:**true if a connection was established, false if this channel is in non-blocking mode and the connection operation is in progress **Throws:** [AlreadyConnectedException](http://docs.google.com/java/nio/channels/AlreadyConnectedException.html) - If this channel is already connected [ConnectionPendingException](http://docs.google.com/java/nio/channels/ConnectionPendingException.html) - If a non-blocking connection operation is already in progress on this channel [ClosedChannelException](http://docs.google.com/java/nio/channels/ClosedChannelException.html) - If this channel is closed [AsynchronousCloseException](http://docs.google.com/java/nio/channels/AsynchronousCloseException.html) - If another thread closes this channel while the connect operation is in progress [ClosedByInterruptException](http://docs.google.com/java/nio/channels/ClosedByInterruptException.html) - If another thread interrupts the current thread while the connect operation is in progress, thereby closing the channel and setting the current thread's interrupt status [UnresolvedAddressException](http://docs.google.com/java/nio/channels/UnresolvedAddressException.html) - If the given remote address is not fully resolved [UnsupportedAddressTypeException](http://docs.google.com/java/nio/channels/UnsupportedAddressTypeException.html) - If the type of the given remote address is not supported [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - If a security manager has been installed and it does not permit access to the given remote endpoint [IOException](http://docs.google.com/java/io/IOException.html) - If some other I/O error occurs

### finishConnect

public abstract boolean **finishConnect**()  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Finishes the process of connecting a socket channel.

A non-blocking connection operation is initiated by placing a socket channel in non-blocking mode and then invoking its [connect](http://docs.google.com/java/nio/channels/SocketChannel.html#connect(java.net.SocketAddress)) method. Once the connection is established, or the attempt has failed, the socket channel will become connectable and this method may be invoked to complete the connection sequence. If the connection operation failed then invoking this method will cause an appropriate [IOException](http://docs.google.com/java/io/IOException.html) to be thrown.

If this channel is already connected then this method will not block and will immediately return true. If this channel is in non-blocking mode then this method will return false if the connection process is not yet complete. If this channel is in blocking mode then this method will block until the connection either completes or fails, and will always either return true or throw a checked exception describing the failure.

This method may be invoked at any time. If a read or write operation upon this channel is invoked while an invocation of this method is in progress then that operation will first block until this invocation is complete. If a connection attempt fails, that is, if an invocation of this method throws a checked exception, then the channel will be closed.

**Returns:**true if, and only if, this channel's socket is now connected **Throws:** [NoConnectionPendingException](http://docs.google.com/java/nio/channels/NoConnectionPendingException.html) - If this channel is not connected and a connection operation has not been initiated [ClosedChannelException](http://docs.google.com/java/nio/channels/ClosedChannelException.html) - If this channel is closed [AsynchronousCloseException](http://docs.google.com/java/nio/channels/AsynchronousCloseException.html) - If another thread closes this channel while the connect operation is in progress [ClosedByInterruptException](http://docs.google.com/java/nio/channels/ClosedByInterruptException.html) - If another thread interrupts the current thread while the connect operation is in progress, thereby closing the channel and setting the current thread's interrupt status [IOException](http://docs.google.com/java/io/IOException.html) - If some other I/O error occurs

### read

public abstract int **read**([ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html) dst)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

**Description copied from interface:** [**ReadableByteChannel**](http://docs.google.com/java/nio/channels/ReadableByteChannel.html#read(java.nio.ByteBuffer)) Reads a sequence of bytes from this channel into the given buffer.

An attempt is made to read up to *r* bytes from the channel, where *r* is the number of bytes remaining in the buffer, that is, dst.remaining(), at the moment this method is invoked.

Suppose that a byte sequence of length *n* is read, where 0 <= *n* <= *r*. This byte sequence will be transferred into the buffer so that the first byte in the sequence is at index *p* and the last byte is at index *p* + *n* - 1, where *p* is the buffer's position at the moment this method is invoked. Upon return the buffer's position will be equal to *p* + *n*; its limit will not have changed.

A read operation might not fill the buffer, and in fact it might not read any bytes at all. Whether or not it does so depends upon the nature and state of the channel. A socket channel in non-blocking mode, for example, cannot read any more bytes than are immediately available from the socket's input buffer; similarly, a file channel cannot read any more bytes than remain in the file. It is guaranteed, however, that if a channel is in blocking mode and there is at least one byte remaining in the buffer then this method will block until at least one byte is read.

This method may be invoked at any time. If another thread has already initiated a read operation upon this channel, however, then an invocation of this method will block until the first operation is complete.

**Specified by:**[read](http://docs.google.com/java/nio/channels/ReadableByteChannel.html#read(java.nio.ByteBuffer)) in interface [ReadableByteChannel](http://docs.google.com/java/nio/channels/ReadableByteChannel.html) **Parameters:**dst - The buffer into which bytes are to be transferred **Returns:**The number of bytes read, possibly zero, or -1 if the channel has reached end-of-stream **Throws:** [NotYetConnectedException](http://docs.google.com/java/nio/channels/NotYetConnectedException.html) - If this channel is not yet connected [ClosedChannelException](http://docs.google.com/java/nio/channels/ClosedChannelException.html) - If this channel is closed [AsynchronousCloseException](http://docs.google.com/java/nio/channels/AsynchronousCloseException.html) - If another thread closes this channel while the read operation is in progress [ClosedByInterruptException](http://docs.google.com/java/nio/channels/ClosedByInterruptException.html) - If another thread interrupts the current thread while the read operation is in progress, thereby closing the channel and setting the current thread's interrupt status [IOException](http://docs.google.com/java/io/IOException.html) - If some other I/O error occurs

### read

public abstract long **read**([ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html)[] dsts,  
 int offset,  
 int length)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

**Description copied from interface:** [**ScatteringByteChannel**](http://docs.google.com/java/nio/channels/ScatteringByteChannel.html#read(java.nio.ByteBuffer%5B%5D,%20int,%20int)) Reads a sequence of bytes from this channel into a subsequence of the given buffers.

An invocation of this method attempts to read up to *r* bytes from this channel, where *r* is the total number of bytes remaining the specified subsequence of the given buffer array, that is,

dsts[offset].remaining()  
 + dsts[offset+1].remaining()  
 + ... + dsts[offset+length-1].remaining()

at the moment that this method is invoked.

Suppose that a byte sequence of length *n* is read, where 0 <= *n* <= *r*. Up to the first dsts[offset].remaining() bytes of this sequence are transferred into buffer dsts[offset], up to the next dsts[offset+1].remaining() bytes are transferred into buffer dsts[offset+1], and so forth, until the entire byte sequence is transferred into the given buffers. As many bytes as possible are transferred into each buffer, hence the final position of each updated buffer, except the last updated buffer, is guaranteed to be equal to that buffer's limit.

This method may be invoked at any time. If another thread has already initiated a read operation upon this channel, however, then an invocation of this method will block until the first operation is complete.

**Specified by:**[read](http://docs.google.com/java/nio/channels/ScatteringByteChannel.html#read(java.nio.ByteBuffer%5B%5D,%20int,%20int)) in interface [ScatteringByteChannel](http://docs.google.com/java/nio/channels/ScatteringByteChannel.html) **Parameters:**dsts - The buffers into which bytes are to be transferredoffset - The offset within the buffer array of the first buffer into which bytes are to be transferred; must be non-negative and no larger than dsts.lengthlength - The maximum number of buffers to be accessed; must be non-negative and no larger than dsts.length - offset **Returns:**The number of bytes read, possibly zero, or -1 if the channel has reached end-of-stream **Throws:** [NotYetConnectedException](http://docs.google.com/java/nio/channels/NotYetConnectedException.html) - If this channel is not yet connected [ClosedChannelException](http://docs.google.com/java/nio/channels/ClosedChannelException.html) - If this channel is closed [AsynchronousCloseException](http://docs.google.com/java/nio/channels/AsynchronousCloseException.html) - If another thread closes this channel while the read operation is in progress [ClosedByInterruptException](http://docs.google.com/java/nio/channels/ClosedByInterruptException.html) - If another thread interrupts the current thread while the read operation is in progress, thereby closing the channel and setting the current thread's interrupt status [IOException](http://docs.google.com/java/io/IOException.html) - If some other I/O error occurs

### read

public final long **read**([ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html)[] dsts)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

**Description copied from interface:** [**ScatteringByteChannel**](http://docs.google.com/java/nio/channels/ScatteringByteChannel.html#read(java.nio.ByteBuffer%5B%5D)) Reads a sequence of bytes from this channel into the given buffers.

An invocation of this method of the form c.read(dsts) behaves in exactly the same manner as the invocation

c.read(dsts, 0, dsts.length);

**Specified by:**[read](http://docs.google.com/java/nio/channels/ScatteringByteChannel.html#read(java.nio.ByteBuffer%5B%5D)) in interface [ScatteringByteChannel](http://docs.google.com/java/nio/channels/ScatteringByteChannel.html) **Parameters:**dsts - The buffers into which bytes are to be transferred **Returns:**The number of bytes read, possibly zero, or -1 if the channel has reached end-of-stream **Throws:** [NotYetConnectedException](http://docs.google.com/java/nio/channels/NotYetConnectedException.html) - If this channel is not yet connected [ClosedChannelException](http://docs.google.com/java/nio/channels/ClosedChannelException.html) - If this channel is closed [AsynchronousCloseException](http://docs.google.com/java/nio/channels/AsynchronousCloseException.html) - If another thread closes this channel while the read operation is in progress [ClosedByInterruptException](http://docs.google.com/java/nio/channels/ClosedByInterruptException.html) - If another thread interrupts the current thread while the read operation is in progress, thereby closing the channel and setting the current thread's interrupt status [IOException](http://docs.google.com/java/io/IOException.html) - If some other I/O error occurs

### write

public abstract int **write**([ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html) src)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

**Description copied from interface:** [**WritableByteChannel**](http://docs.google.com/java/nio/channels/WritableByteChannel.html#write(java.nio.ByteBuffer)) Writes a sequence of bytes to this channel from the given buffer.

An attempt is made to write up to *r* bytes to the channel, where *r* is the number of bytes remaining in the buffer, that is, src.remaining(), at the moment this method is invoked.

Suppose that a byte sequence of length *n* is written, where 0 <= *n* <= *r*. This byte sequence will be transferred from the buffer starting at index *p*, where *p* is the buffer's position at the moment this method is invoked; the index of the last byte written will be *p* + *n* - 1. Upon return the buffer's position will be equal to *p* + *n*; its limit will not have changed.

Unless otherwise specified, a write operation will return only after writing all of the *r* requested bytes. Some types of channels, depending upon their state, may write only some of the bytes or possibly none at all. A socket channel in non-blocking mode, for example, cannot write any more bytes than are free in the socket's output buffer.

This method may be invoked at any time. If another thread has already initiated a write operation upon this channel, however, then an invocation of this method will block until the first operation is complete.

**Specified by:**[write](http://docs.google.com/java/nio/channels/WritableByteChannel.html#write(java.nio.ByteBuffer)) in interface [WritableByteChannel](http://docs.google.com/java/nio/channels/WritableByteChannel.html) **Parameters:**src - The buffer from which bytes are to be retrieved **Returns:**The number of bytes written, possibly zero **Throws:** [NotYetConnectedException](http://docs.google.com/java/nio/channels/NotYetConnectedException.html) - If this channel is not yet connected [ClosedChannelException](http://docs.google.com/java/nio/channels/ClosedChannelException.html) - If this channel is closed [AsynchronousCloseException](http://docs.google.com/java/nio/channels/AsynchronousCloseException.html) - If another thread closes this channel while the write operation is in progress [ClosedByInterruptException](http://docs.google.com/java/nio/channels/ClosedByInterruptException.html) - If another thread interrupts the current thread while the write operation is in progress, thereby closing the channel and setting the current thread's interrupt status [IOException](http://docs.google.com/java/io/IOException.html) - If some other I/O error occurs

### write

public abstract long **write**([ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html)[] srcs,  
 int offset,  
 int length)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

**Description copied from interface:** [**GatheringByteChannel**](http://docs.google.com/java/nio/channels/GatheringByteChannel.html#write(java.nio.ByteBuffer%5B%5D,%20int,%20int)) Writes a sequence of bytes to this channel from a subsequence of the given buffers.

An attempt is made to write up to *r* bytes to this channel, where *r* is the total number of bytes remaining in the specified subsequence of the given buffer array, that is,

srcs[offset].remaining()  
 + srcs[offset+1].remaining()  
 + ... + srcs[offset+length-1].remaining()

at the moment that this method is invoked.

Suppose that a byte sequence of length *n* is written, where 0 <= *n* <= *r*. Up to the first srcs[offset].remaining() bytes of this sequence are written from buffer srcs[offset], up to the next srcs[offset+1].remaining() bytes are written from buffer srcs[offset+1], and so forth, until the entire byte sequence is written. As many bytes as possible are written from each buffer, hence the final position of each updated buffer, except the last updated buffer, is guaranteed to be equal to that buffer's limit.

Unless otherwise specified, a write operation will return only after writing all of the *r* requested bytes. Some types of channels, depending upon their state, may write only some of the bytes or possibly none at all. A socket channel in non-blocking mode, for example, cannot write any more bytes than are free in the socket's output buffer.

This method may be invoked at any time. If another thread has already initiated a write operation upon this channel, however, then an invocation of this method will block until the first operation is complete.

**Specified by:**[write](http://docs.google.com/java/nio/channels/GatheringByteChannel.html#write(java.nio.ByteBuffer%5B%5D,%20int,%20int)) in interface [GatheringByteChannel](http://docs.google.com/java/nio/channels/GatheringByteChannel.html) **Parameters:**srcs - The buffers from which bytes are to be retrievedoffset - The offset within the buffer array of the first buffer from which bytes are to be retrieved; must be non-negative and no larger than srcs.lengthlength - The maximum number of buffers to be accessed; must be non-negative and no larger than srcs.length - offset **Returns:**The number of bytes written, possibly zero **Throws:** [NotYetConnectedException](http://docs.google.com/java/nio/channels/NotYetConnectedException.html) - If this channel is not yet connected [ClosedChannelException](http://docs.google.com/java/nio/channels/ClosedChannelException.html) - If this channel is closed [AsynchronousCloseException](http://docs.google.com/java/nio/channels/AsynchronousCloseException.html) - If another thread closes this channel while the write operation is in progress [ClosedByInterruptException](http://docs.google.com/java/nio/channels/ClosedByInterruptException.html) - If another thread interrupts the current thread while the write operation is in progress, thereby closing the channel and setting the current thread's interrupt status [IOException](http://docs.google.com/java/io/IOException.html) - If some other I/O error occurs

### write

public final long **write**([ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html)[] srcs)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

**Description copied from interface:** [**GatheringByteChannel**](http://docs.google.com/java/nio/channels/GatheringByteChannel.html#write(java.nio.ByteBuffer%5B%5D)) Writes a sequence of bytes to this channel from the given buffers.

An invocation of this method of the form c.write(srcs) behaves in exactly the same manner as the invocation

c.write(srcs, 0, srcs.length);

**Specified by:**[write](http://docs.google.com/java/nio/channels/GatheringByteChannel.html#write(java.nio.ByteBuffer%5B%5D)) in interface [GatheringByteChannel](http://docs.google.com/java/nio/channels/GatheringByteChannel.html) **Parameters:**srcs - The buffers from which bytes are to be retrieved **Returns:**The number of bytes written, possibly zero **Throws:** [NotYetConnectedException](http://docs.google.com/java/nio/channels/NotYetConnectedException.html) - If this channel is not yet connected [ClosedChannelException](http://docs.google.com/java/nio/channels/ClosedChannelException.html) - If this channel is closed [AsynchronousCloseException](http://docs.google.com/java/nio/channels/AsynchronousCloseException.html) - If another thread closes this channel while the write operation is in progress [ClosedByInterruptException](http://docs.google.com/java/nio/channels/ClosedByInterruptException.html) - If another thread interrupts the current thread while the write operation is in progress, thereby closing the channel and setting the current thread's interrupt status [IOException](http://docs.google.com/java/io/IOException.html) - If some other I/O error occurs

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/SocketChannel.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/java/nio/channels/ServerSocketChannel.html)   [**NEXT CLASS**](http://docs.google.com/java/nio/channels/UnresolvedAddressException.html) | [**FRAMES**](http://docs.google.com/index.html?java/nio/channels/SocketChannel.html)    [**NO FRAMES**](http://docs.google.com/SocketChannel.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#17dp8vu) | [METHOD](#26in1rg) |

[Submit a bug or feature](http://bugs.sun.com/services/bugreport/index.jsp)

For further API reference and developer documentation, see [Java SE Developer Documentation](http://docs.google.com/webnotes/devdocs-vs-specs.html). That documentation contains more detailed, developer-targeted descriptions, with conceptual overviews, definitions of terms, workarounds, and working code examples.

Copyright 2006 Sun Microsystems, Inc. All rights reserved. Use is subject to [license terms](http://docs.google.com/legal/license.html). Also see the [documentation redistribution policy](http://java.sun.com/docs/redist.html).