| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/CharBuffer.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/ByteOrder.html)   [**NEXT CLASS**](http://docs.google.com/java/nio/DoubleBuffer.html) | [**FRAMES**](http://docs.google.com/index.html?java/nio/CharBuffer.html)    [**NO FRAMES**](http://docs.google.com/CharBuffer.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#3znysh7) | DETAIL: FIELD | CONSTR | [METHOD](#3dy6vkm) |

## **java.nio**

Class CharBuffer

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 [java.nio.Buffer](http://docs.google.com/java/nio/Buffer.html)  
 **java.nio.CharBuffer**

**All Implemented Interfaces:** [Appendable](http://docs.google.com/java/lang/Appendable.html), [CharSequence](http://docs.google.com/java/lang/CharSequence.html), [Comparable](http://docs.google.com/java/lang/Comparable.html)<[CharBuffer](http://docs.google.com/java/nio/CharBuffer.html)>, [Readable](http://docs.google.com/java/lang/Readable.html)

public abstract class **CharBuffer**extends [Buffer](http://docs.google.com/java/nio/Buffer.html)implements [Comparable](http://docs.google.com/java/lang/Comparable.html)<[CharBuffer](http://docs.google.com/java/nio/CharBuffer.html)>, [Appendable](http://docs.google.com/java/lang/Appendable.html), [CharSequence](http://docs.google.com/java/lang/CharSequence.html), [Readable](http://docs.google.com/java/lang/Readable.html)

A character buffer.

This class defines four categories of operations upon character buffers:

* Absolute and relative [*get*](http://docs.google.com/java/nio/CharBuffer.html#get()) and [*put*](http://docs.google.com/java/nio/CharBuffer.html#put(char)) methods that read and write single characters;
* Relative [*bulk get*](http://docs.google.com/java/nio/CharBuffer.html#get(char%5B%5D)) methods that transfer contiguous sequences of characters from this buffer into an array; and
* Relative [*bulk put*](http://docs.google.com/java/nio/CharBuffer.html#put(char%5B%5D)) methods that transfer contiguous sequences of characters from a character array, a string, or some other character buffer into this buffer; and
* Methods for [compacting](http://docs.google.com/java/nio/CharBuffer.html#compact()), [duplicating](http://docs.google.com/java/nio/CharBuffer.html#duplicate()), and [slicing](http://docs.google.com/java/nio/CharBuffer.html#slice()) a character buffer.

Character buffers can be created either by [*allocation*](http://docs.google.com/java/nio/CharBuffer.html#allocate(int)), which allocates space for the buffer's content, by [*wrapping*](http://docs.google.com/java/nio/CharBuffer.html#wrap(char%5B%5D)) an existing character array or string into a buffer, or by creating a [*view*](http://docs.google.com/ByteBuffer.html#views) of an existing byte buffer.

Like a byte buffer, a character buffer is either [*direct* or *non-direct*](http://docs.google.com/ByteBuffer.html#direct). A character buffer created via the wrap methods of this class will be non-direct. A character buffer created as a view of a byte buffer will be direct if, and only if, the byte buffer itself is direct. Whether or not a character buffer is direct may be determined by invoking the [isDirect](http://docs.google.com/java/nio/CharBuffer.html#isDirect()) method.

This class implements the [CharSequence](http://docs.google.com/java/lang/CharSequence.html) interface so that character buffers may be used wherever character sequences are accepted, for example in the regular-expression package [java.util.regex](http://docs.google.com/java/util/regex/package-summary.html).

Methods in this class that do not otherwise have a value to return are specified to return the buffer upon which they are invoked. This allows method invocations to be chained. The sequence of statements

cb.put("text/");  
 cb.put(subtype);  
 cb.put("; charset=");  
 cb.put(enc);

can, for example, be replaced by the single statement

cb.put("text/").put(subtype).put("; charset=").put(enc);

**Since:** 1.4

| **Method Summary** | |
| --- | --- |
| static [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**allocate**](http://docs.google.com/java/nio/CharBuffer.html#allocate(int))(int capacity)            Allocates a new character buffer. |
| [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**append**](http://docs.google.com/java/nio/CharBuffer.html#append(char))(char c)            Appends the specified character to this buffer  *(optional operation)*. |
| [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**append**](http://docs.google.com/java/nio/CharBuffer.html#append(java.lang.CharSequence))([CharSequence](http://docs.google.com/java/lang/CharSequence.html) csq)            Appends the specified character sequence to this buffer  *(optional operation)*. |
| [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**append**](http://docs.google.com/java/nio/CharBuffer.html#append(java.lang.CharSequence,%20int,%20int))([CharSequence](http://docs.google.com/java/lang/CharSequence.html) csq, int start, int end)            Appends a subsequence of the specified character sequence to this buffer  *(optional operation)*. |
| char[] | [**array**](http://docs.google.com/java/nio/CharBuffer.html#array())()            Returns the character array that backs this buffer  *(optional operation)*. |
| int | [**arrayOffset**](http://docs.google.com/java/nio/CharBuffer.html#arrayOffset())()            Returns the offset within this buffer's backing array of the first element of the buffer  *(optional operation)*. |
| abstract  [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**asReadOnlyBuffer**](http://docs.google.com/java/nio/CharBuffer.html#asReadOnlyBuffer())()            Creates a new, read-only character buffer that shares this buffer's content. |
| char | [**charAt**](http://docs.google.com/java/nio/CharBuffer.html#charAt(int))(int index)            Reads the character at the given index relative to the current position. |
| abstract  [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**compact**](http://docs.google.com/java/nio/CharBuffer.html#compact())()            Compacts this buffer  *(optional operation)*. |
| int | [**compareTo**](http://docs.google.com/java/nio/CharBuffer.html#compareTo(java.nio.CharBuffer))([CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) that)            Compares this buffer to another. |
| abstract  [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**duplicate**](http://docs.google.com/java/nio/CharBuffer.html#duplicate())()            Creates a new character buffer that shares this buffer's content. |
| boolean | [**equals**](http://docs.google.com/java/nio/CharBuffer.html#equals(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) ob)            Tells whether or not this buffer is equal to another object. |
| abstract  char | [**get**](http://docs.google.com/java/nio/CharBuffer.html#get())()            Relative *get* method. |
| [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**get**](http://docs.google.com/java/nio/CharBuffer.html#get(char%5B%5D))(char[] dst)            Relative bulk *get* method. |
| [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**get**](http://docs.google.com/java/nio/CharBuffer.html#get(char%5B%5D,%20int,%20int))(char[] dst, int offset, int length)            Relative bulk *get* method. |
| abstract  char | [**get**](http://docs.google.com/java/nio/CharBuffer.html#get(int))(int index)            Absolute *get* method. |
| boolean | [**hasArray**](http://docs.google.com/java/nio/CharBuffer.html#hasArray())()            Tells whether or not this buffer is backed by an accessible character array. |
| int | [**hashCode**](http://docs.google.com/java/nio/CharBuffer.html#hashCode())()            Returns the current hash code of this buffer. |
| abstract  boolean | [**isDirect**](http://docs.google.com/java/nio/CharBuffer.html#isDirect())()            Tells whether or not this character buffer is direct. |
| int | [**length**](http://docs.google.com/java/nio/CharBuffer.html#length())()            Returns the length of this character buffer. |
| abstract  [ByteOrder](http://docs.google.com/java/nio/ByteOrder.html) | [**order**](http://docs.google.com/java/nio/CharBuffer.html#order())()            Retrieves this buffer's byte order. |
| abstract  [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**put**](http://docs.google.com/java/nio/CharBuffer.html#put(char))(char c)            Relative *put* method  *(optional operation)*. |
| [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**put**](http://docs.google.com/java/nio/CharBuffer.html#put(char%5B%5D))(char[] src)            Relative bulk *put* method  *(optional operation)*. |
| [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**put**](http://docs.google.com/java/nio/CharBuffer.html#put(char%5B%5D,%20int,%20int))(char[] src, int offset, int length)            Relative bulk *put* method  *(optional operation)*. |
| [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**put**](http://docs.google.com/java/nio/CharBuffer.html#put(java.nio.CharBuffer))([CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) src)            Relative bulk *put* method  *(optional operation)*. |
| abstract  [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**put**](http://docs.google.com/java/nio/CharBuffer.html#put(int,%20char))(int index, char c)            Absolute *put* method  *(optional operation)*. |
| [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**put**](http://docs.google.com/java/nio/CharBuffer.html#put(java.lang.String))([String](http://docs.google.com/java/lang/String.html) src)            Relative bulk *put* method  *(optional operation)*. |
| [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**put**](http://docs.google.com/java/nio/CharBuffer.html#put(java.lang.String,%20int,%20int))([String](http://docs.google.com/java/lang/String.html) src, int start, int end)            Relative bulk *put* method  *(optional operation)*. |
| int | [**read**](http://docs.google.com/java/nio/CharBuffer.html#read(java.nio.CharBuffer))([CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) target)            Attempts to read characters into the specified character buffer. |
| abstract  [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**slice**](http://docs.google.com/java/nio/CharBuffer.html#slice())()            Creates a new character buffer whose content is a shared subsequence of this buffer's content. |
| abstract  [CharSequence](http://docs.google.com/java/lang/CharSequence.html) | [**subSequence**](http://docs.google.com/java/nio/CharBuffer.html#subSequence(int,%20int))(int start, int end)            Creates a new character buffer that represents the specified subsequence of this buffer, relative to the current position. |
| [String](http://docs.google.com/java/lang/String.html) | [**toString**](http://docs.google.com/java/nio/CharBuffer.html#toString())()            Returns a string containing the characters in this buffer. |
| static [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**wrap**](http://docs.google.com/java/nio/CharBuffer.html#wrap(char%5B%5D))(char[] array)            Wraps a character array into a buffer. |
| static [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**wrap**](http://docs.google.com/java/nio/CharBuffer.html#wrap(char%5B%5D,%20int,%20int))(char[] array, int offset, int length)            Wraps a character array into a buffer. |
| static [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**wrap**](http://docs.google.com/java/nio/CharBuffer.html#wrap(java.lang.CharSequence))([CharSequence](http://docs.google.com/java/lang/CharSequence.html) csq)            Wraps a character sequence into a buffer. |
| static [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) | [**wrap**](http://docs.google.com/java/nio/CharBuffer.html#wrap(java.lang.CharSequence,%20int,%20int))([CharSequence](http://docs.google.com/java/lang/CharSequence.html) csq, int start, int end)            Wraps a character sequence into a buffer. |

| **Methods inherited from class java.nio.**[**Buffer**](http://docs.google.com/java/nio/Buffer.html) |
| --- |
| [capacity](http://docs.google.com/java/nio/Buffer.html#capacity()), [clear](http://docs.google.com/java/nio/Buffer.html#clear()), [flip](http://docs.google.com/java/nio/Buffer.html#flip()), [hasRemaining](http://docs.google.com/java/nio/Buffer.html#hasRemaining()), [isReadOnly](http://docs.google.com/java/nio/Buffer.html#isReadOnly()), [limit](http://docs.google.com/java/nio/Buffer.html#limit()), [limit](http://docs.google.com/java/nio/Buffer.html#limit(int)), [mark](http://docs.google.com/java/nio/Buffer.html#mark()), [position](http://docs.google.com/java/nio/Buffer.html#position()), [position](http://docs.google.com/java/nio/Buffer.html#position(int)), [remaining](http://docs.google.com/java/nio/Buffer.html#remaining()), [reset](http://docs.google.com/java/nio/Buffer.html#reset()), [rewind](http://docs.google.com/java/nio/Buffer.html#rewind()) |

| **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()), [finalize](http://docs.google.com/java/lang/Object.html#finalize()), [getClass](http://docs.google.com/java/lang/Object.html#getClass()), [notify](http://docs.google.com/java/lang/Object.html#notify()), [notifyAll](http://docs.google.com/java/lang/Object.html#notifyAll()), [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)) |

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

### allocate

public static [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **allocate**(int capacity)

Allocates a new character buffer.

The new buffer's position will be zero, its limit will be its capacity, and its mark will be undefined. It will have a [backing array](http://docs.google.com/java/nio/CharBuffer.html#array()), and its [array offset](http://docs.google.com/java/nio/CharBuffer.html#arrayOffset()) will be zero.

**Parameters:**capacity - The new buffer's capacity, in characters **Returns:**The new character buffer **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - If the capacity is a negative integer

### wrap

public static [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **wrap**(char[] array,  
 int offset,  
 int length)

Wraps a character array into a buffer.

The new buffer will be backed by the given character array; that is, modifications to the buffer will cause the array to be modified and vice versa. The new buffer's capacity will be array.length, its position will be offset, its limit will be offset + length, and its mark will be undefined. Its [backing array](http://docs.google.com/java/nio/CharBuffer.html#array()) will be the given array, and its [array offset](http://docs.google.com/java/nio/CharBuffer.html#arrayOffset()) will be zero.

**Parameters:**array - The array that will back the new bufferoffset - The offset of the subarray to be used; must be non-negative and no larger than array.length. The new buffer's position will be set to this value.length - The length of the subarray to be used; must be non-negative and no larger than array.length - offset. The new buffer's limit will be set to offset + length. **Returns:**The new character buffer **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - If the preconditions on the offset and length parameters do not hold

### wrap

public static [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **wrap**(char[] array)

Wraps a character array into a buffer.

The new buffer will be backed by the given character array; that is, modifications to the buffer will cause the array to be modified and vice versa. The new buffer's capacity and limit will be array.length, its position will be zero, and its mark will be undefined. Its [backing array](http://docs.google.com/java/nio/CharBuffer.html#array()) will be the given array, and its [array offset](http://docs.google.com/java/nio/CharBuffer.html#arrayOffset()) will be zero.

**Parameters:**array - The array that will back this buffer **Returns:**The new character buffer

### read

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

Attempts to read characters into the specified character buffer. The buffer is used as a repository of characters as-is: the only changes made are the results of a put operation. No flipping or rewinding of the buffer is performed.

**Specified by:**[read](http://docs.google.com/java/lang/Readable.html#read(java.nio.CharBuffer)) in interface [Readable](http://docs.google.com/java/lang/Readable.html) **Parameters:**target - the buffer to read characters into **Returns:**The number of characters added to the buffer, or -1 if this source of characters is at its end **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if an I/O error occurs [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if target is null [ReadOnlyBufferException](http://docs.google.com/java/nio/ReadOnlyBufferException.html) - if target is a read only buffer**Since:** 1.5

### wrap

public static [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **wrap**([CharSequence](http://docs.google.com/java/lang/CharSequence.html) csq,  
 int start,  
 int end)

Wraps a character sequence into a buffer.

The content of the new, read-only buffer will be the content of the given character sequence. The buffer's capacity will be csq.length(), its position will be start, its limit will be end, and its mark will be undefined.

**Parameters:**csq - The character sequence from which the new character buffer is to be createdstart - The index of the first character to be used; must be non-negative and no larger than csq.length(). The new buffer's position will be set to this value.end - The index of the character following the last character to be used; must be no smaller than start and no larger than csq.length(). The new buffer's limit will be set to this value. **Returns:**The new character buffer **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - If the preconditions on the start and end parameters do not hold

### wrap

public static [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **wrap**([CharSequence](http://docs.google.com/java/lang/CharSequence.html) csq)

Wraps a character sequence into a buffer.

The content of the new, read-only buffer will be the content of the given character sequence. The new buffer's capacity and limit will be csq.length(), its position will be zero, and its mark will be undefined.

**Parameters:**csq - The character sequence from which the new character buffer is to be created **Returns:**The new character buffer

### slice

public abstract [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **slice**()

Creates a new character buffer whose content is a shared subsequence of this buffer's content.

The content of the new buffer will start at this buffer's current position. Changes to this buffer's content will be visible in the new buffer, and vice versa; the two buffers' position, limit, and mark values will be independent.

The new buffer's position will be zero, its capacity and its limit will be the number of characters remaining in this buffer, and its mark will be undefined. The new buffer will be direct if, and only if, this buffer is direct, and it will be read-only if, and only if, this buffer is read-only.

**Returns:**The new character buffer

### duplicate

public abstract [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **duplicate**()

Creates a new character buffer that shares this buffer's content.

The content of the new buffer will be that of this buffer. Changes to this buffer's content will be visible in the new buffer, and vice versa; the two buffers' position, limit, and mark values will be independent.

The new buffer's capacity, limit, position, and mark values will be identical to those of this buffer. The new buffer will be direct if, and only if, this buffer is direct, and it will be read-only if, and only if, this buffer is read-only.

**Returns:**The new character buffer

### asReadOnlyBuffer

public abstract [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **asReadOnlyBuffer**()

Creates a new, read-only character buffer that shares this buffer's content.

The content of the new buffer will be that of this buffer. Changes to this buffer's content will be visible in the new buffer; the new buffer itself, however, will be read-only and will not allow the shared content to be modified. The two buffers' position, limit, and mark values will be independent.

The new buffer's capacity, limit, position, and mark values will be identical to those of this buffer.

If this buffer is itself read-only then this method behaves in exactly the same way as the [duplicate](http://docs.google.com/java/nio/CharBuffer.html#duplicate()) method.

**Returns:**The new, read-only character buffer

### get

public abstract char **get**()

Relative *get* method. Reads the character at this buffer's current position, and then increments the position.

**Returns:**The character at the buffer's current position **Throws:** [BufferUnderflowException](http://docs.google.com/java/nio/BufferUnderflowException.html) - If the buffer's current position is not smaller than its limit

### put

public abstract [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **put**(char c)

Relative *put* method  *(optional operation)*.

Writes the given character into this buffer at the current position, and then increments the position.

**Parameters:**c - The character to be written **Returns:**This buffer **Throws:** [BufferOverflowException](http://docs.google.com/java/nio/BufferOverflowException.html) - If this buffer's current position is not smaller than its limit [ReadOnlyBufferException](http://docs.google.com/java/nio/ReadOnlyBufferException.html) - If this buffer is read-only

### get

public abstract char **get**(int index)

Absolute *get* method. Reads the character at the given index.

**Parameters:**index - The index from which the character will be read **Returns:**The character at the given index **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - If index is negative or not smaller than the buffer's limit

### put

public abstract [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **put**(int index,  
 char c)

Absolute *put* method  *(optional operation)*.

Writes the given character into this buffer at the given index.

**Parameters:**index - The index at which the character will be writtenc - The character value to be written **Returns:**This buffer **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - If index is negative or not smaller than the buffer's limit [ReadOnlyBufferException](http://docs.google.com/java/nio/ReadOnlyBufferException.html) - If this buffer is read-only

### get

public [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **get**(char[] dst,  
 int offset,  
 int length)

Relative bulk *get* method.

This method transfers characters from this buffer into the given destination array. If there are fewer characters remaining in the buffer than are required to satisfy the request, that is, if length > remaining(), then no characters are transferred and a [BufferUnderflowException](http://docs.google.com/java/nio/BufferUnderflowException.html) is thrown.

Otherwise, this method copies length characters from this buffer into the given array, starting at the current position of this buffer and at the given offset in the array. The position of this buffer is then incremented by length.

In other words, an invocation of this method of the form src.get(dst, off, len) has exactly the same effect as the loop

for (int i = off; i < off + len; i++)  
 dst[i] = src.get();

except that it first checks that there are sufficient characters in this buffer and it is potentially much more efficient.

**Parameters:**dst - The array into which characters are to be writtenoffset - The offset within the array of the first character to be written; must be non-negative and no larger than dst.lengthlength - The maximum number of characters to be written to the given array; must be non-negative and no larger than dst.length - offset **Returns:**This buffer **Throws:** [BufferUnderflowException](http://docs.google.com/java/nio/BufferUnderflowException.html) - If there are fewer than length characters remaining in this buffer [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - If the preconditions on the offset and length parameters do not hold

### get

public [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **get**(char[] dst)

Relative bulk *get* method.

This method transfers characters from this buffer into the given destination array. An invocation of this method of the form src.get(a) behaves in exactly the same way as the invocation

src.get(a, 0, a.length)

**Returns:**This buffer **Throws:** [BufferUnderflowException](http://docs.google.com/java/nio/BufferUnderflowException.html) - If there are fewer than length characters remaining in this buffer

### put

public [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **put**([CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) src)

Relative bulk *put* method  *(optional operation)*.

This method transfers the characters remaining in the given source buffer into this buffer. If there are more characters remaining in the source buffer than in this buffer, that is, if src.remaining() > remaining(), then no characters are transferred and a [BufferOverflowException](http://docs.google.com/java/nio/BufferOverflowException.html) is thrown.

Otherwise, this method copies *n* = src.remaining() characters from the given buffer into this buffer, starting at each buffer's current position. The positions of both buffers are then incremented by *n*.

In other words, an invocation of this method of the form dst.put(src) has exactly the same effect as the loop

while (src.hasRemaining())  
 dst.put(src.get());

except that it first checks that there is sufficient space in this buffer and it is potentially much more efficient.

**Parameters:**src - The source buffer from which characters are to be read; must not be this buffer **Returns:**This buffer **Throws:** [BufferOverflowException](http://docs.google.com/java/nio/BufferOverflowException.html) - If there is insufficient space in this buffer for the remaining characters in the source buffer [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - If the source buffer is this buffer [ReadOnlyBufferException](http://docs.google.com/java/nio/ReadOnlyBufferException.html) - If this buffer is read-only

### put

public [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **put**(char[] src,  
 int offset,  
 int length)

Relative bulk *put* method  *(optional operation)*.

This method transfers characters into this buffer from the given source array. If there are more characters to be copied from the array than remain in this buffer, that is, if length > remaining(), then no characters are transferred and a [BufferOverflowException](http://docs.google.com/java/nio/BufferOverflowException.html) is thrown.

Otherwise, this method copies length characters from the given array into this buffer, starting at the given offset in the array and at the current position of this buffer. The position of this buffer is then incremented by length.

In other words, an invocation of this method of the form dst.put(src, off, len) has exactly the same effect as the loop

for (int i = off; i < off + len; i++)  
 dst.put(a[i]);

except that it first checks that there is sufficient space in this buffer and it is potentially much more efficient.

**Parameters:**src - The array from which characters are to be readoffset - The offset within the array of the first character to be read; must be non-negative and no larger than array.lengthlength - The number of characters to be read from the given array; must be non-negative and no larger than array.length - offset **Returns:**This buffer **Throws:** [BufferOverflowException](http://docs.google.com/java/nio/BufferOverflowException.html) - If there is insufficient space in this buffer [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - If the preconditions on the offset and length parameters do not hold [ReadOnlyBufferException](http://docs.google.com/java/nio/ReadOnlyBufferException.html) - If this buffer is read-only

### put

public final [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **put**(char[] src)

Relative bulk *put* method  *(optional operation)*.

This method transfers the entire content of the given source character array into this buffer. An invocation of this method of the form dst.put(a) behaves in exactly the same way as the invocation

dst.put(a, 0, a.length)

**Returns:**This buffer **Throws:** [BufferOverflowException](http://docs.google.com/java/nio/BufferOverflowException.html) - If there is insufficient space in this buffer [ReadOnlyBufferException](http://docs.google.com/java/nio/ReadOnlyBufferException.html) - If this buffer is read-only

### put

public [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **put**([String](http://docs.google.com/java/lang/String.html) src,  
 int start,  
 int end)

Relative bulk *put* method  *(optional operation)*.

This method transfers characters from the given string into this buffer. If there are more characters to be copied from the string than remain in this buffer, that is, if end - start > remaining(), then no characters are transferred and a [BufferOverflowException](http://docs.google.com/java/nio/BufferOverflowException.html) is thrown.

Otherwise, this method copies *n* = end - start characters from the given string into this buffer, starting at the given start index and at the current position of this buffer. The position of this buffer is then incremented by *n*.

In other words, an invocation of this method of the form dst.put(src, start, end) has exactly the same effect as the loop

for (int i = start; i < end; i++)  
 dst.put(src.charAt(i));

except that it first checks that there is sufficient space in this buffer and it is potentially much more efficient.

**Parameters:**src - The string from which characters are to be readstart - The offset within the string of the first character to be read; must be non-negative and no larger than string.length()end - The offset within the string of the last character to be read, plus one; must be non-negative and no larger than string.length() **Returns:**This buffer **Throws:** [BufferOverflowException](http://docs.google.com/java/nio/BufferOverflowException.html) - If there is insufficient space in this buffer [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - If the preconditions on the start and end parameters do not hold [ReadOnlyBufferException](http://docs.google.com/java/nio/ReadOnlyBufferException.html) - If this buffer is read-only

### put

public final [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **put**([String](http://docs.google.com/java/lang/String.html) src)

Relative bulk *put* method  *(optional operation)*.

This method transfers the entire content of the given source string into this buffer. An invocation of this method of the form dst.put(s) behaves in exactly the same way as the invocation

dst.put(s, 0, s.length())

**Returns:**This buffer **Throws:** [BufferOverflowException](http://docs.google.com/java/nio/BufferOverflowException.html) - If there is insufficient space in this buffer [ReadOnlyBufferException](http://docs.google.com/java/nio/ReadOnlyBufferException.html) - If this buffer is read-only

### hasArray

public final boolean **hasArray**()

Tells whether or not this buffer is backed by an accessible character array.

If this method returns true then the [array](http://docs.google.com/java/nio/CharBuffer.html#array()) and [arrayOffset](http://docs.google.com/java/nio/CharBuffer.html#arrayOffset()) methods may safely be invoked.

**Specified by:**[hasArray](http://docs.google.com/java/nio/Buffer.html#hasArray()) in class [Buffer](http://docs.google.com/java/nio/Buffer.html) **Returns:**true if, and only if, this buffer is backed by an array and is not read-only

### array

public final char[] **array**()

Returns the character array that backs this buffer  *(optional operation)*.

Modifications to this buffer's content will cause the returned array's content to be modified, and vice versa.

Invoke the [hasArray](http://docs.google.com/java/nio/CharBuffer.html#hasArray()) method before invoking this method in order to ensure that this buffer has an accessible backing array.

**Specified by:**[array](http://docs.google.com/java/nio/Buffer.html#array()) in class [Buffer](http://docs.google.com/java/nio/Buffer.html) **Returns:**The array that backs this buffer **Throws:** [ReadOnlyBufferException](http://docs.google.com/java/nio/ReadOnlyBufferException.html) - If this buffer is backed by an array but is read-only [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - If this buffer is not backed by an accessible array

### arrayOffset

public final int **arrayOffset**()

Returns the offset within this buffer's backing array of the first element of the buffer  *(optional operation)*.

If this buffer is backed by an array then buffer position *p* corresponds to array index *p* + arrayOffset().

Invoke the [hasArray](http://docs.google.com/java/nio/CharBuffer.html#hasArray()) method before invoking this method in order to ensure that this buffer has an accessible backing array.

**Specified by:**[arrayOffset](http://docs.google.com/java/nio/Buffer.html#arrayOffset()) in class [Buffer](http://docs.google.com/java/nio/Buffer.html) **Returns:**The offset within this buffer's array of the first element of the buffer **Throws:** [ReadOnlyBufferException](http://docs.google.com/java/nio/ReadOnlyBufferException.html) - If this buffer is backed by an array but is read-only [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - If this buffer is not backed by an accessible array

### compact

public abstract [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **compact**()

Compacts this buffer  *(optional operation)*.

The characters between the buffer's current position and its limit, if any, are copied to the beginning of the buffer. That is, the character at index *p* = position() is copied to index zero, the character at index *p* + 1 is copied to index one, and so forth until the character at index limit() - 1 is copied to index *n* = limit() - 1 - *p*. The buffer's position is then set to *n+1* and its limit is set to its capacity. The mark, if defined, is discarded.

The buffer's position is set to the number of characters copied, rather than to zero, so that an invocation of this method can be followed immediately by an invocation of another relative *put* method.

**Returns:**This buffer **Throws:** [ReadOnlyBufferException](http://docs.google.com/java/nio/ReadOnlyBufferException.html) - If this buffer is read-only

### isDirect

public abstract boolean **isDirect**()

Tells whether or not this character buffer is direct.

**Specified by:**[isDirect](http://docs.google.com/java/nio/Buffer.html#isDirect()) in class [Buffer](http://docs.google.com/java/nio/Buffer.html) **Returns:**true if, and only if, this buffer is direct

### hashCode

public int **hashCode**()

Returns the current hash code of this buffer.

The hash code of a char buffer depends only upon its remaining elements; that is, upon the elements from position() up to, and including, the element at limit() - 1.

Because buffer hash codes are content-dependent, it is inadvisable to use buffers as keys in hash maps or similar data structures unless it is known that their contents will not change.

**Overrides:**[hashCode](http://docs.google.com/java/lang/Object.html#hashCode()) in class [Object](http://docs.google.com/java/lang/Object.html) **Returns:**The current hash code of this buffer**See Also:**[Object.equals(java.lang.Object)](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [Hashtable](http://docs.google.com/java/util/Hashtable.html)

### equals

public boolean **equals**([Object](http://docs.google.com/java/lang/Object.html) ob)

Tells whether or not this buffer is equal to another object.

Two char buffers are equal if, and only if,

1. They have the same element type,
2. They have the same number of remaining elements, and
3. The two sequences of remaining elements, considered independently of their starting positions, are pointwise equal.

A char buffer is not equal to any other type of object.

**Overrides:**[equals](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)) in class [Object](http://docs.google.com/java/lang/Object.html) **Parameters:**ob - The object to which this buffer is to be compared **Returns:**true if, and only if, this buffer is equal to the given object**See Also:**[Object.hashCode()](http://docs.google.com/java/lang/Object.html#hashCode()), [Hashtable](http://docs.google.com/java/util/Hashtable.html)

### compareTo

public int **compareTo**([CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) that)

Compares this buffer to another.

Two char buffers are compared by comparing their sequences of remaining elements lexicographically, without regard to the starting position of each sequence within its corresponding buffer.

A char buffer is not comparable to any other type of object.

**Specified by:**[compareTo](http://docs.google.com/java/lang/Comparable.html#compareTo(T)) in interface [Comparable](http://docs.google.com/java/lang/Comparable.html)<[CharBuffer](http://docs.google.com/java/nio/CharBuffer.html)> **Parameters:**that - the object to be compared. **Returns:**A negative integer, zero, or a positive integer as this buffer is less than, equal to, or greater than the given buffer

### toString

public [String](http://docs.google.com/java/lang/String.html) **toString**()

Returns a string containing the characters in this buffer.

The first character of the resulting string will be the character at this buffer's position, while the last character will be the character at index limit() - 1. Invoking this method does not change the buffer's position.

**Specified by:**[toString](http://docs.google.com/java/lang/CharSequence.html#toString()) in interface [CharSequence](http://docs.google.com/java/lang/CharSequence.html)**Overrides:**[toString](http://docs.google.com/java/lang/Object.html#toString()) in class [Object](http://docs.google.com/java/lang/Object.html) **Returns:**The specified string

### length

public final int **length**()

Returns the length of this character buffer.

When viewed as a character sequence, the length of a character buffer is simply the number of characters between the position (inclusive) and the limit (exclusive); that is, it is equivalent to remaining().

**Specified by:**[length](http://docs.google.com/java/lang/CharSequence.html#length()) in interface [CharSequence](http://docs.google.com/java/lang/CharSequence.html) **Returns:**The length of this character buffer

### charAt

public final char **charAt**(int index)

Reads the character at the given index relative to the current position.

**Specified by:**[charAt](http://docs.google.com/java/lang/CharSequence.html#charAt(int)) in interface [CharSequence](http://docs.google.com/java/lang/CharSequence.html) **Parameters:**index - The index of the character to be read, relative to the position; must be non-negative and smaller than remaining() **Returns:**The character at index position() + index **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - If the preconditions on index do not hold

### subSequence

public abstract [CharSequence](http://docs.google.com/java/lang/CharSequence.html) **subSequence**(int start,  
 int end)

Creates a new character buffer that represents the specified subsequence of this buffer, relative to the current position.

The new buffer will share this buffer's content; that is, if the content of this buffer is mutable then modifications to one buffer will cause the other to be modified. The new buffer's capacity will be that of this buffer, its position will be position() + start, and its limit will be position() + end. The new buffer will be direct if, and only if, this buffer is direct, and it will be read-only if, and only if, this buffer is read-only.

**Specified by:**[subSequence](http://docs.google.com/java/lang/CharSequence.html#subSequence(int,%20int)) in interface [CharSequence](http://docs.google.com/java/lang/CharSequence.html) **Parameters:**start - The index, relative to the current position, of the first character in the subsequence; must be non-negative and no larger than remaining()end - The index, relative to the current position, of the character following the last character in the subsequence; must be no smaller than start and no larger than remaining() **Returns:**The new character sequence **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - If the preconditions on start and end do not hold

### append

public [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **append**([CharSequence](http://docs.google.com/java/lang/CharSequence.html) csq)

Appends the specified character sequence to this buffer  *(optional operation)*.

An invocation of this method of the form dst.append(csq) behaves in exactly the same way as the invocation

dst.put(csq.toString())

Depending on the specification of toString for the character sequence csq, the entire sequence may not be appended. For instance, invoking the [toString](http://docs.google.com/java/nio/CharBuffer.html#toString()) method of a character buffer will return a subsequence whose content depends upon the buffer's position and limit.

**Specified by:**[append](http://docs.google.com/java/lang/Appendable.html#append(java.lang.CharSequence)) in interface [Appendable](http://docs.google.com/java/lang/Appendable.html) **Parameters:**csq - The character sequence to append. If csq is null, then the four characters "null" are appended to this character buffer. **Returns:**This buffer **Throws:** [BufferOverflowException](http://docs.google.com/java/nio/BufferOverflowException.html) - If there is insufficient space in this buffer [ReadOnlyBufferException](http://docs.google.com/java/nio/ReadOnlyBufferException.html) - If this buffer is read-only**Since:** 1.5

### append

public [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **append**([CharSequence](http://docs.google.com/java/lang/CharSequence.html) csq,  
 int start,  
 int end)

Appends a subsequence of the specified character sequence to this buffer  *(optional operation)*.

An invocation of this method of the form dst.append(csq, start, end) when csq is not null, behaves in exactly the same way as the invocation

dst.put(csq.subSequence(start, end).toString())

**Specified by:**[append](http://docs.google.com/java/lang/Appendable.html#append(java.lang.CharSequence,%20int,%20int)) in interface [Appendable](http://docs.google.com/java/lang/Appendable.html) **Parameters:**csq - The character sequence from which a subsequence will be appended. If csq is null, then characters will be appended as if csq contained the four characters "null".start - The index of the first character in the subsequenceend - The index of the character following the last character in the subsequence **Returns:**This buffer **Throws:** [BufferOverflowException](http://docs.google.com/java/nio/BufferOverflowException.html) - If there is insufficient space in this buffer [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - If start or end are negative, start is greater than end, or end is greater than csq.length() [ReadOnlyBufferException](http://docs.google.com/java/nio/ReadOnlyBufferException.html) - If this buffer is read-only**Since:** 1.5

### append

public [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) **append**(char c)

Appends the specified character to this buffer  *(optional operation)*.

An invocation of this method of the form dst.append(c) behaves in exactly the same way as the invocation

dst.put(c)

**Specified by:**[append](http://docs.google.com/java/lang/Appendable.html#append(char)) in interface [Appendable](http://docs.google.com/java/lang/Appendable.html) **Parameters:**c - The 16-bit character to append **Returns:**This buffer **Throws:** [BufferOverflowException](http://docs.google.com/java/nio/BufferOverflowException.html) - If there is insufficient space in this buffer [ReadOnlyBufferException](http://docs.google.com/java/nio/ReadOnlyBufferException.html) - If this buffer is read-only**Since:** 1.5

### order

public abstract [ByteOrder](http://docs.google.com/java/nio/ByteOrder.html) **order**()

Retrieves this buffer's byte order.

The byte order of a character buffer created by allocation or by wrapping an existing char array is the [native order](http://docs.google.com/java/nio/ByteOrder.html#nativeOrder()) of the underlying hardware. The byte order of a character buffer created as a [view](http://docs.google.com/ByteBuffer.html#views) of a byte buffer is that of the byte buffer at the moment that the view is created.

**Returns:**This buffer's byte order

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/CharBuffer.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/ByteOrder.html)   [**NEXT CLASS**](http://docs.google.com/java/nio/DoubleBuffer.html) | [**FRAMES**](http://docs.google.com/index.html?java/nio/CharBuffer.html)    [**NO FRAMES**](http://docs.google.com/CharBuffer.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#3znysh7) | DETAIL: FIELD | CONSTR | [METHOD](#3dy6vkm) |

[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).