| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/StringBuilder.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/lang/StringBuffer.html)   [**NEXT CLASS**](http://docs.google.com/java/lang/StringIndexOutOfBoundsException.html) | [**FRAMES**](http://docs.google.com/index.html?java/lang/StringBuilder.html)    [**NO FRAMES**](http://docs.google.com/StringBuilder.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#1t3h5sf) | [METHOD](#26in1rg) |

## **java.lang**

Class StringBuilder

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

**All Implemented Interfaces:** [Serializable](http://docs.google.com/java/io/Serializable.html), [Appendable](http://docs.google.com/java/lang/Appendable.html), [CharSequence](http://docs.google.com/java/lang/CharSequence.html)

public final class **StringBuilder**extends [Object](http://docs.google.com/java/lang/Object.html)implements [Serializable](http://docs.google.com/java/io/Serializable.html), [CharSequence](http://docs.google.com/java/lang/CharSequence.html)

A mutable sequence of characters. This class provides an API compatible with StringBuffer, but with no guarantee of synchronization. This class is designed for use as a drop-in replacement for StringBuffer in places where the string buffer was being used by a single thread (as is generally the case). Where possible, it is recommended that this class be used in preference to StringBuffer as it will be faster under most implementations.

The principal operations on a StringBuilder are the append and insert methods, which are overloaded so as to accept data of any type. Each effectively converts a given datum to a string and then appends or inserts the characters of that string to the string builder. The append method always adds these characters at the end of the builder; the insert method adds the characters at a specified point.

For example, if z refers to a string builder object whose current contents are "start", then the method call z.append("le") would cause the string builder to contain "startle", whereas z.insert(4, "le") would alter the string builder to contain "starlet".

In general, if sb refers to an instance of a StringBuilder, then sb.append(x) has the same effect as sb.insert(sb.length(), x). Every string builder has a capacity. As long as the length of the character sequence contained in the string builder does not exceed the capacity, it is not necessary to allocate a new internal buffer. If the internal buffer overflows, it is automatically made larger.

Instances of StringBuilder are not safe for use by multiple threads. If such synchronization is required then it is recommended that [StringBuffer](http://docs.google.com/java/lang/StringBuffer.html) be used.

**Since:** 1.5 **See Also:**[StringBuffer](http://docs.google.com/java/lang/StringBuffer.html), [String](http://docs.google.com/java/lang/String.html), [Serialized Form](http://docs.google.com/serialized-form.html#java.lang.StringBuilder)

| **Constructor Summary** | |
| --- | --- |
| [**StringBuilder**](http://docs.google.com/java/lang/StringBuilder.html#StringBuilder())()            Constructs a string builder with no characters in it and an initial capacity of 16 characters. |
| [**StringBuilder**](http://docs.google.com/java/lang/StringBuilder.html#StringBuilder(java.lang.CharSequence))([CharSequence](http://docs.google.com/java/lang/CharSequence.html) seq)            Constructs a string builder that contains the same characters as the specified CharSequence. |
| [**StringBuilder**](http://docs.google.com/java/lang/StringBuilder.html#StringBuilder(int))(int capacity)            Constructs a string builder with no characters in it and an initial capacity specified by the capacity argument. |
| [**StringBuilder**](http://docs.google.com/java/lang/StringBuilder.html#StringBuilder(java.lang.String))([String](http://docs.google.com/java/lang/String.html) str)            Constructs a string builder initialized to the contents of the specified string. |

| **Method Summary** | |
| --- | --- |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**append**](http://docs.google.com/java/lang/StringBuilder.html#append(boolean))(boolean b)            Appends the string representation of the boolean argument to the sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**append**](http://docs.google.com/java/lang/StringBuilder.html#append(char))(char c)            Appends the string representation of the char argument to this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**append**](http://docs.google.com/java/lang/StringBuilder.html#append(char%5B%5D))(char[] str)            Appends the string representation of the char array argument to this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**append**](http://docs.google.com/java/lang/StringBuilder.html#append(char%5B%5D,%20int,%20int))(char[] str, int offset, int len)            Appends the string representation of a subarray of the char array argument to this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**append**](http://docs.google.com/java/lang/StringBuilder.html#append(java.lang.CharSequence))([CharSequence](http://docs.google.com/java/lang/CharSequence.html) s)            Appends the specified character sequence to this Appendable. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**append**](http://docs.google.com/java/lang/StringBuilder.html#append(java.lang.CharSequence,%20int,%20int))([CharSequence](http://docs.google.com/java/lang/CharSequence.html) s, int start, int end)            Appends a subsequence of the specified CharSequence to this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**append**](http://docs.google.com/java/lang/StringBuilder.html#append(double))(double d)            Appends the string representation of the double argument to this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**append**](http://docs.google.com/java/lang/StringBuilder.html#append(float))(float f)            Appends the string representation of the float argument to this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**append**](http://docs.google.com/java/lang/StringBuilder.html#append(int))(int i)            Appends the string representation of the int argument to this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**append**](http://docs.google.com/java/lang/StringBuilder.html#append(long))(long lng)            Appends the string representation of the long argument to this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**append**](http://docs.google.com/java/lang/StringBuilder.html#append(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) obj)            Appends the string representation of the Object argument. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**append**](http://docs.google.com/java/lang/StringBuilder.html#append(java.lang.String))([String](http://docs.google.com/java/lang/String.html) str)            Appends the specified string to this character sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**append**](http://docs.google.com/java/lang/StringBuilder.html#append(java.lang.StringBuffer))([StringBuffer](http://docs.google.com/java/lang/StringBuffer.html) sb)            Appends the specified StringBuffer to this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**appendCodePoint**](http://docs.google.com/java/lang/StringBuilder.html#appendCodePoint(int))(int codePoint)            Appends the string representation of the codePoint argument to this sequence. |
| int | [**capacity**](http://docs.google.com/java/lang/StringBuilder.html#capacity())()            Returns the current capacity. |
| char | [**charAt**](http://docs.google.com/java/lang/StringBuilder.html#charAt(int))(int index)            Returns the char value in this sequence at the specified index. |
| int | [**codePointAt**](http://docs.google.com/java/lang/StringBuilder.html#codePointAt(int))(int index)            Returns the character (Unicode code point) at the specified index. |
| int | [**codePointBefore**](http://docs.google.com/java/lang/StringBuilder.html#codePointBefore(int))(int index)            Returns the character (Unicode code point) before the specified index. |
| int | [**codePointCount**](http://docs.google.com/java/lang/StringBuilder.html#codePointCount(int,%20int))(int beginIndex, int endIndex)            Returns the number of Unicode code points in the specified text range of this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**delete**](http://docs.google.com/java/lang/StringBuilder.html#delete(int,%20int))(int start, int end)            Removes the characters in a substring of this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**deleteCharAt**](http://docs.google.com/java/lang/StringBuilder.html#deleteCharAt(int))(int index)            Removes the char at the specified position in this sequence. |
| void | [**ensureCapacity**](http://docs.google.com/java/lang/StringBuilder.html#ensureCapacity(int))(int minimumCapacity)            Ensures that the capacity is at least equal to the specified minimum. |
| void | [**getChars**](http://docs.google.com/java/lang/StringBuilder.html#getChars(int,%20int,%20char%5B%5D,%20int))(int srcBegin, int srcEnd, char[] dst, int dstBegin)            Characters are copied from this sequence into the destination character array dst. |
| int | [**indexOf**](http://docs.google.com/java/lang/StringBuilder.html#indexOf(java.lang.String))([String](http://docs.google.com/java/lang/String.html) str)            Returns the index within this string of the first occurrence of the specified substring. |
| int | [**indexOf**](http://docs.google.com/java/lang/StringBuilder.html#indexOf(java.lang.String,%20int))([String](http://docs.google.com/java/lang/String.html) str, int fromIndex)            Returns the index within this string of the first occurrence of the specified substring, starting at the specified index. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**insert**](http://docs.google.com/java/lang/StringBuilder.html#insert(int,%20boolean))(int offset, boolean b)            Inserts the string representation of the boolean argument into this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**insert**](http://docs.google.com/java/lang/StringBuilder.html#insert(int,%20char))(int offset, char c)            Inserts the string representation of the char argument into this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**insert**](http://docs.google.com/java/lang/StringBuilder.html#insert(int,%20char%5B%5D))(int offset, char[] str)            Inserts the string representation of the char array argument into this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**insert**](http://docs.google.com/java/lang/StringBuilder.html#insert(int,%20char%5B%5D,%20int,%20int))(int index, char[] str, int offset, int len)            Inserts the string representation of a subarray of the str array argument into this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**insert**](http://docs.google.com/java/lang/StringBuilder.html#insert(int,%20java.lang.CharSequence))(int dstOffset, [CharSequence](http://docs.google.com/java/lang/CharSequence.html) s)            Inserts the specified CharSequence into this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**insert**](http://docs.google.com/java/lang/StringBuilder.html#insert(int,%20java.lang.CharSequence,%20int,%20int))(int dstOffset, [CharSequence](http://docs.google.com/java/lang/CharSequence.html) s, int start, int end)            Inserts a subsequence of the specified CharSequence into this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**insert**](http://docs.google.com/java/lang/StringBuilder.html#insert(int,%20double))(int offset, double d)            Inserts the string representation of the double argument into this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**insert**](http://docs.google.com/java/lang/StringBuilder.html#insert(int,%20float))(int offset, float f)            Inserts the string representation of the float argument into this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**insert**](http://docs.google.com/java/lang/StringBuilder.html#insert(int,%20int))(int offset, int i)            Inserts the string representation of the second int argument into this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**insert**](http://docs.google.com/java/lang/StringBuilder.html#insert(int,%20long))(int offset, long l)            Inserts the string representation of the long argument into this sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**insert**](http://docs.google.com/java/lang/StringBuilder.html#insert(int,%20java.lang.Object))(int offset, [Object](http://docs.google.com/java/lang/Object.html) obj)            Inserts the string representation of the Object argument into this character sequence. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**insert**](http://docs.google.com/java/lang/StringBuilder.html#insert(int,%20java.lang.String))(int offset, [String](http://docs.google.com/java/lang/String.html) str)            Inserts the string into this character sequence. |
| int | [**lastIndexOf**](http://docs.google.com/java/lang/StringBuilder.html#lastIndexOf(java.lang.String))([String](http://docs.google.com/java/lang/String.html) str)            Returns the index within this string of the rightmost occurrence of the specified substring. |
| int | [**lastIndexOf**](http://docs.google.com/java/lang/StringBuilder.html#lastIndexOf(java.lang.String,%20int))([String](http://docs.google.com/java/lang/String.html) str, int fromIndex)            Returns the index within this string of the last occurrence of the specified substring. |
| int | [**length**](http://docs.google.com/java/lang/StringBuilder.html#length())()            Returns the length (character count). |
| int | [**offsetByCodePoints**](http://docs.google.com/java/lang/StringBuilder.html#offsetByCodePoints(int,%20int))(int index, int codePointOffset)            Returns the index within this sequence that is offset from the given index by codePointOffset code points. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**replace**](http://docs.google.com/java/lang/StringBuilder.html#replace(int,%20int,%20java.lang.String))(int start, int end, [String](http://docs.google.com/java/lang/String.html) str)            Replaces the characters in a substring of this sequence with characters in the specified String. |
| [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) | [**reverse**](http://docs.google.com/java/lang/StringBuilder.html#reverse())()            Causes this character sequence to be replaced by the reverse of the sequence. |
| void | [**setCharAt**](http://docs.google.com/java/lang/StringBuilder.html#setCharAt(int,%20char))(int index, char ch)            The character at the specified index is set to ch. |
| void | [**setLength**](http://docs.google.com/java/lang/StringBuilder.html#setLength(int))(int newLength)            Sets the length of the character sequence. |
| [CharSequence](http://docs.google.com/java/lang/CharSequence.html) | [**subSequence**](http://docs.google.com/java/lang/StringBuilder.html#subSequence(int,%20int))(int start, int end)            Returns a new character sequence that is a subsequence of this sequence. |
| [String](http://docs.google.com/java/lang/String.html) | [**substring**](http://docs.google.com/java/lang/StringBuilder.html#substring(int))(int start)            Returns a new String that contains a subsequence of characters currently contained in this character sequence. |
| [String](http://docs.google.com/java/lang/String.html) | [**substring**](http://docs.google.com/java/lang/StringBuilder.html#substring(int,%20int))(int start, int end)            Returns a new String that contains a subsequence of characters currently contained in this sequence. |
| [String](http://docs.google.com/java/lang/String.html) | [**toString**](http://docs.google.com/java/lang/StringBuilder.html#toString())()            Returns a string representing the data in this sequence. |
| void | [**trimToSize**](http://docs.google.com/java/lang/StringBuilder.html#trimToSize())()            Attempts to reduce storage used for the character sequence. |

| **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()), [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.lang.**[**CharSequence**](http://docs.google.com/java/lang/CharSequence.html) |
| --- |
| [charAt](http://docs.google.com/java/lang/CharSequence.html#charAt(int)), [length](http://docs.google.com/java/lang/CharSequence.html#length()), [subSequence](http://docs.google.com/java/lang/CharSequence.html#subSequence(int,%20int)) |

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

### StringBuilder

public **StringBuilder**()

Constructs a string builder with no characters in it and an initial capacity of 16 characters.

### StringBuilder

public **StringBuilder**(int capacity)

Constructs a string builder with no characters in it and an initial capacity specified by the capacity argument.

**Parameters:**capacity - the initial capacity. **Throws:** [NegativeArraySizeException](http://docs.google.com/java/lang/NegativeArraySizeException.html) - if the capacity argument is less than 0.

### StringBuilder

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

Constructs a string builder initialized to the contents of the specified string. The initial capacity of the string builder is 16 plus the length of the string argument.

**Parameters:**str - the initial contents of the buffer. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if str is null

### StringBuilder

public **StringBuilder**([CharSequence](http://docs.google.com/java/lang/CharSequence.html) seq)

Constructs a string builder that contains the same characters as the specified CharSequence. The initial capacity of the string builder is 16 plus the length of the CharSequence argument.

**Parameters:**seq - the sequence to copy. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if seq is null

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

### append

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **append**([Object](http://docs.google.com/java/lang/Object.html) obj)

Appends the string representation of the Object argument.

The argument is converted to a string as if by the method String.valueOf, and the characters of that string are then appended to this sequence.

**Parameters:**obj - an Object. **Returns:**a reference to this object.**See Also:**[String.valueOf(java.lang.Object)](http://docs.google.com/java/lang/String.html#valueOf(java.lang.Object)), [append(java.lang.String)](http://docs.google.com/java/lang/StringBuilder.html#append(java.lang.String))

### append

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

Appends the specified string to this character sequence.

The characters of the String argument are appended, in order, increasing the length of this sequence by the length of the argument. If str is null, then the four characters "null" are appended.

Let *n* be the length of this character sequence just prior to execution of the append method. Then the character at index *k* in the new character sequence is equal to the character at index *k* in the old character sequence, if *k* is less than *n*; otherwise, it is equal to the character at index *k-n* in the argument str.

**Parameters:**str - a string. **Returns:**a reference to this object.

### append

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **append**([StringBuffer](http://docs.google.com/java/lang/StringBuffer.html) sb)

Appends the specified StringBuffer to this sequence.

The characters of the StringBuffer argument are appended, in order, to this sequence, increasing the length of this sequence by the length of the argument. If sb is null, then the four characters "null" are appended to this sequence.

Let *n* be the length of this character sequence just prior to execution of the append method. Then the character at index *k* in the new character sequence is equal to the character at index *k* in the old character sequence, if *k* is less than *n*; otherwise, it is equal to the character at index *k-n* in the argument sb.

**Parameters:**sb - the StringBuffer to append. **Returns:**a reference to this object.

### append

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **append**([CharSequence](http://docs.google.com/java/lang/CharSequence.html) s)

**Description copied from interface:** [**Appendable**](http://docs.google.com/java/lang/Appendable.html#append(java.lang.CharSequence)) Appends the specified character sequence to this Appendable.

Depending on which class implements the character sequence csq, the entire sequence may not be appended. For instance, if csq is a [CharBuffer](http://docs.google.com/java/nio/CharBuffer.html) then the subsequence to append is defined by 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:**s - The character sequence to append. If csq is null, then the four characters "null" are appended to this Appendable. **Returns:**A reference to this Appendable **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html)

### append

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

Appends a subsequence of the specified CharSequence to this sequence.

Characters of the argument s, starting at index start, are appended, in order, to the contents of this sequence up to the (exclusive) index end. The length of this sequence is increased by the value of end - start.

Let *n* be the length of this character sequence just prior to execution of the append method. Then the character at index *k* in this character sequence becomes equal to the character at index *k* in this sequence, if *k* is less than *n*; otherwise, it is equal to the character at index *k+start-n* in the argument s.

If s is null, then this method appends characters as if the s parameter was a sequence containing the four characters "null".

**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:**s - the sequence to append.start - the starting index of the subsequence to be appended.end - the end index of the subsequence to be appended. **Returns:**a reference to this object. **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if start or end are negative, or start is greater than end or end is greater than s.length()

### append

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **append**(char[] str)

Appends the string representation of the char array argument to this sequence.

The characters of the array argument are appended, in order, to the contents of this sequence. The length of this sequence increases by the length of the argument.

The overall effect is exactly as if the argument were converted to a string by the method [String.valueOf(char[])](http://docs.google.com/java/lang/String.html#valueOf(char%5B%5D)) and the characters of that string were then [appended](http://docs.google.com/java/lang/StringBuilder.html#append(java.lang.String)) to this character sequence.

**Parameters:**str - the characters to be appended. **Returns:**a reference to this object.

### append

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **append**(char[] str,  
 int offset,  
 int len)

Appends the string representation of a subarray of the char array argument to this sequence.

Characters of the char array str, starting at index offset, are appended, in order, to the contents of this sequence. The length of this sequence increases by the value of len.

The overall effect is exactly as if the arguments were converted to a string by the method [String.valueOf(char[],int,int)](http://docs.google.com/java/lang/String.html#valueOf(char%5B%5D,%20int,%20int)) and the characters of that string were then [appended](http://docs.google.com/java/lang/StringBuilder.html#append(java.lang.String)) to this character sequence.

**Parameters:**str - the characters to be appended.offset - the index of the first char to append.len - the number of chars to append. **Returns:**a reference to this object.

### append

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **append**(boolean b)

Appends the string representation of the boolean argument to the sequence.

The argument is converted to a string as if by the method String.valueOf, and the characters of that string are then appended to this sequence.

**Parameters:**b - a boolean. **Returns:**a reference to this object.**See Also:**[String.valueOf(boolean)](http://docs.google.com/java/lang/String.html#valueOf(boolean)), [append(java.lang.String)](http://docs.google.com/java/lang/StringBuilder.html#append(java.lang.String))

### append

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

Appends the string representation of the char argument to this sequence.

The argument is appended to the contents of this sequence. The length of this sequence increases by 1.

The overall effect is exactly as if the argument were converted to a string by the method [String.valueOf(char)](http://docs.google.com/java/lang/String.html#valueOf(char)) and the character in that string were then [appended](http://docs.google.com/java/lang/StringBuilder.html#append(java.lang.String)) to this character sequence.

**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 - a char. **Returns:**a reference to this object.

### append

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **append**(int i)

Appends the string representation of the int argument to this sequence.

The argument is converted to a string as if by the method String.valueOf, and the characters of that string are then appended to this sequence.

**Parameters:**i - an int. **Returns:**a reference to this object.**See Also:**[String.valueOf(int)](http://docs.google.com/java/lang/String.html#valueOf(int)), [append(java.lang.String)](http://docs.google.com/java/lang/StringBuilder.html#append(java.lang.String))

### append

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **append**(long lng)

Appends the string representation of the long argument to this sequence.

The argument is converted to a string as if by the method String.valueOf, and the characters of that string are then appended to this sequence.

**Parameters:**lng - a long. **Returns:**a reference to this object.**See Also:**[String.valueOf(long)](http://docs.google.com/java/lang/String.html#valueOf(long)), [append(java.lang.String)](http://docs.google.com/java/lang/StringBuilder.html#append(java.lang.String))

### append

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **append**(float f)

Appends the string representation of the float argument to this sequence.

The argument is converted to a string as if by the method String.valueOf, and the characters of that string are then appended to this string sequence.

**Parameters:**f - a float. **Returns:**a reference to this object.**See Also:**[String.valueOf(float)](http://docs.google.com/java/lang/String.html#valueOf(float)), [append(java.lang.String)](http://docs.google.com/java/lang/StringBuilder.html#append(java.lang.String))

### append

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **append**(double d)

Appends the string representation of the double argument to this sequence.

The argument is converted to a string as if by the method String.valueOf, and the characters of that string are then appended to this sequence.

**Parameters:**d - a double. **Returns:**a reference to this object.**See Also:**[String.valueOf(double)](http://docs.google.com/java/lang/String.html#valueOf(double)), [append(java.lang.String)](http://docs.google.com/java/lang/StringBuilder.html#append(java.lang.String))

### appendCodePoint

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **appendCodePoint**(int codePoint)

Appends the string representation of the codePoint argument to this sequence.

The argument is appended to the contents of this sequence. The length of this sequence increases by [Character.charCount(codePoint)](http://docs.google.com/java/lang/Character.html#charCount(int)).

The overall effect is exactly as if the argument were converted to a char array by the method [Character.toChars(int)](http://docs.google.com/java/lang/Character.html#toChars(int)) and the character in that array were then [appended](http://docs.google.com/java/lang/StringBuilder.html#append(char%5B%5D)) to this character sequence.

**Parameters:**codePoint - a Unicode code point **Returns:**a reference to this object.**Since:** 1.5

### delete

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

Removes the characters in a substring of this sequence. The substring begins at the specified start and extends to the character at index end - 1 or to the end of the sequence if no such character exists. If start is equal to end, no changes are made.

**Parameters:**start - The beginning index, inclusive.end - The ending index, exclusive. **Returns:**This object. **Throws:** [StringIndexOutOfBoundsException](http://docs.google.com/java/lang/StringIndexOutOfBoundsException.html) - if start is negative, greater than length(), or greater than end.

### deleteCharAt

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **deleteCharAt**(int index)

Removes the char at the specified position in this sequence. This sequence is shortened by one char.

Note: If the character at the given index is a supplementary character, this method does not remove the entire character. If correct handling of supplementary characters is required, determine the number of chars to remove by calling Character.charCount(thisSequence.codePointAt(index)), where thisSequence is this sequence.

**Parameters:**index - Index of char to remove **Returns:**This object. **Throws:** [StringIndexOutOfBoundsException](http://docs.google.com/java/lang/StringIndexOutOfBoundsException.html) - if the index is negative or greater than or equal to length().

### replace

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **replace**(int start,  
 int end,  
 [String](http://docs.google.com/java/lang/String.html) str)

Replaces the characters in a substring of this sequence with characters in the specified String. The substring begins at the specified start and extends to the character at index end - 1 or to the end of the sequence if no such character exists. First the characters in the substring are removed and then the specified String is inserted at start. (This sequence will be lengthened to accommodate the specified String if necessary.)

**Parameters:**start - The beginning index, inclusive.end - The ending index, exclusive.str - String that will replace previous contents. **Returns:**This object. **Throws:** [StringIndexOutOfBoundsException](http://docs.google.com/java/lang/StringIndexOutOfBoundsException.html) - if start is negative, greater than length(), or greater than end.

### insert

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **insert**(int index,  
 char[] str,  
 int offset,  
 int len)

Inserts the string representation of a subarray of the str array argument into this sequence. The subarray begins at the specified offset and extends len chars. The characters of the subarray are inserted into this sequence at the position indicated by index. The length of this sequence increases by len chars.

**Parameters:**index - position at which to insert subarray.str - A char array.offset - the index of the first char in subarray to be inserted.len - the number of chars in the subarray to be inserted. **Returns:**This object **Throws:** [StringIndexOutOfBoundsException](http://docs.google.com/java/lang/StringIndexOutOfBoundsException.html) - if index is negative or greater than length(), or offset or len are negative, or (offset+len) is greater than str.length.

### insert

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **insert**(int offset,  
 [Object](http://docs.google.com/java/lang/Object.html) obj)

Inserts the string representation of the Object argument into this character sequence.

The second argument is converted to a string as if by the method String.valueOf, and the characters of that string are then inserted into this sequence at the indicated offset.

The offset argument must be greater than or equal to 0, and less than or equal to the length of this sequence.

**Parameters:**offset - the offset.obj - an Object. **Returns:**a reference to this object. **Throws:** [StringIndexOutOfBoundsException](http://docs.google.com/java/lang/StringIndexOutOfBoundsException.html) - if the offset is invalid.**See Also:**[String.valueOf(java.lang.Object)](http://docs.google.com/java/lang/String.html#valueOf(java.lang.Object)), [insert(int, java.lang.String)](http://docs.google.com/java/lang/StringBuilder.html#insert(int,%20java.lang.String)), [length()](http://docs.google.com/java/lang/StringBuilder.html#length())

### insert

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **insert**(int offset,  
 [String](http://docs.google.com/java/lang/String.html) str)

Inserts the string into this character sequence.

The characters of the String argument are inserted, in order, into this sequence at the indicated offset, moving up any characters originally above that position and increasing the length of this sequence by the length of the argument. If str is null, then the four characters "null" are inserted into this sequence.

The character at index *k* in the new character sequence is equal to:

* the character at index *k* in the old character sequence, if *k* is less than offset
* the character at index *k*-offset in the argument str, if *k* is not less than offset but is less than offset+str.length()
* the character at index *k*-str.length() in the old character sequence, if *k* is not less than offset+str.length()

The offset argument must be greater than or equal to 0, and less than or equal to the length of this sequence.

**Parameters:**offset - the offset.str - a string. **Returns:**a reference to this object. **Throws:** [StringIndexOutOfBoundsException](http://docs.google.com/java/lang/StringIndexOutOfBoundsException.html) - if the offset is invalid.**See Also:**[length()](http://docs.google.com/java/lang/StringBuilder.html#length())

### insert

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **insert**(int offset,  
 char[] str)

Inserts the string representation of the char array argument into this sequence.

The characters of the array argument are inserted into the contents of this sequence at the position indicated by offset. The length of this sequence increases by the length of the argument.

The overall effect is exactly as if the argument were converted to a string by the method [String.valueOf(char[])](http://docs.google.com/java/lang/String.html#valueOf(char%5B%5D)) and the characters of that string were then [inserted](http://docs.google.com/java/lang/StringBuilder.html#insert(int,%20java.lang.String)) into this character sequence at the position indicated by offset.

**Parameters:**offset - the offset.str - a character array. **Returns:**a reference to this object. **Throws:** [StringIndexOutOfBoundsException](http://docs.google.com/java/lang/StringIndexOutOfBoundsException.html) - if the offset is invalid.

### insert

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **insert**(int dstOffset,  
 [CharSequence](http://docs.google.com/java/lang/CharSequence.html) s)

Inserts the specified CharSequence into this sequence.

The characters of the CharSequence argument are inserted, in order, into this sequence at the indicated offset, moving up any characters originally above that position and increasing the length of this sequence by the length of the argument s.

The result of this method is exactly the same as if it were an invocation of this object's insert(dstOffset, s, 0, s.length()) method.

If s is null, then the four characters "null" are inserted into this sequence.

**Parameters:**dstOffset - the offset.s - the sequence to be inserted **Returns:**a reference to this object. **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if the offset is invalid.

### insert

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **insert**(int dstOffset,  
 [CharSequence](http://docs.google.com/java/lang/CharSequence.html) s,  
 int start,  
 int end)

Inserts a subsequence of the specified CharSequence into this sequence.

The subsequence of the argument s specified by start and end are inserted, in order, into this sequence at the specified destination offset, moving up any characters originally above that position. The length of this sequence is increased by end - start.

The character at index *k* in this sequence becomes equal to:

* the character at index *k* in this sequence, if *k* is less than dstOffset
* the character at index *k*+start-dstOffset in the argument s, if *k* is greater than or equal to dstOffset but is less than dstOffset+end-start
* the character at index *k*-(end-start) in this sequence, if *k* is greater than or equal to dstOffset+end-start

The dstOffset argument must be greater than or equal to 0, and less than or equal to the length of this sequence.

The start argument must be nonnegative, and not greater than end.

The end argument must be greater than or equal to start, and less than or equal to the length of s.

If s is null, then this method inserts characters as if the s parameter was a sequence containing the four characters "null".

**Parameters:**dstOffset - the offset in this sequence.s - the sequence to be inserted.start - the starting index of the subsequence to be inserted.end - the end index of the subsequence to be inserted. **Returns:**a reference to this object. **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if dstOffset is negative or greater than this.length(), or start or end are negative, or start is greater than end or end is greater than s.length()

### insert

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **insert**(int offset,  
 boolean b)

Inserts the string representation of the boolean argument into this sequence.

The second argument is converted to a string as if by the method String.valueOf, and the characters of that string are then inserted into this sequence at the indicated offset.

The offset argument must be greater than or equal to 0, and less than or equal to the length of this sequence.

**Parameters:**offset - the offset.b - a boolean. **Returns:**a reference to this object. **Throws:** [StringIndexOutOfBoundsException](http://docs.google.com/java/lang/StringIndexOutOfBoundsException.html) - if the offset is invalid.**See Also:**[String.valueOf(boolean)](http://docs.google.com/java/lang/String.html#valueOf(boolean)), [insert(int, java.lang.String)](http://docs.google.com/java/lang/StringBuilder.html#insert(int,%20java.lang.String)), [length()](http://docs.google.com/java/lang/StringBuilder.html#length())

### insert

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **insert**(int offset,  
 char c)

Inserts the string representation of the char argument into this sequence.

The second argument is inserted into the contents of this sequence at the position indicated by offset. The length of this sequence increases by one.

The overall effect is exactly as if the argument were converted to a string by the method [String.valueOf(char)](http://docs.google.com/java/lang/String.html#valueOf(char)) and the character in that string were then [inserted](http://docs.google.com/java/lang/StringBuilder.html#insert(int,%20java.lang.String)) into this character sequence at the position indicated by offset.

The offset argument must be greater than or equal to 0, and less than or equal to the length of this sequence.

**Parameters:**offset - the offset.c - a char. **Returns:**a reference to this object. **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if the offset is invalid.**See Also:**[length()](http://docs.google.com/java/lang/StringBuilder.html#length())

### insert

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **insert**(int offset,  
 int i)

Inserts the string representation of the second int argument into this sequence.

The second argument is converted to a string as if by the method String.valueOf, and the characters of that string are then inserted into this sequence at the indicated offset.

The offset argument must be greater than or equal to 0, and less than or equal to the length of this sequence.

**Parameters:**offset - the offset.i - an int. **Returns:**a reference to this object. **Throws:** [StringIndexOutOfBoundsException](http://docs.google.com/java/lang/StringIndexOutOfBoundsException.html) - if the offset is invalid.**See Also:**[String.valueOf(int)](http://docs.google.com/java/lang/String.html#valueOf(int)), [insert(int, java.lang.String)](http://docs.google.com/java/lang/StringBuilder.html#insert(int,%20java.lang.String)), [length()](http://docs.google.com/java/lang/StringBuilder.html#length())

### insert

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **insert**(int offset,  
 long l)

Inserts the string representation of the long argument into this sequence.

The second argument is converted to a string as if by the method String.valueOf, and the characters of that string are then inserted into this sequence at the position indicated by offset.

The offset argument must be greater than or equal to 0, and less than or equal to the length of this sequence.

**Parameters:**offset - the offset.l - a long. **Returns:**a reference to this object. **Throws:** [StringIndexOutOfBoundsException](http://docs.google.com/java/lang/StringIndexOutOfBoundsException.html) - if the offset is invalid.**See Also:**[String.valueOf(long)](http://docs.google.com/java/lang/String.html#valueOf(long)), [insert(int, java.lang.String)](http://docs.google.com/java/lang/StringBuilder.html#insert(int,%20java.lang.String)), [length()](http://docs.google.com/java/lang/StringBuilder.html#length())

### insert

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **insert**(int offset,  
 float f)

Inserts the string representation of the float argument into this sequence.

The second argument is converted to a string as if by the method String.valueOf, and the characters of that string are then inserted into this sequence at the indicated offset.

The offset argument must be greater than or equal to 0, and less than or equal to the length of this sequence.

**Parameters:**offset - the offset.f - a float. **Returns:**a reference to this object. **Throws:** [StringIndexOutOfBoundsException](http://docs.google.com/java/lang/StringIndexOutOfBoundsException.html) - if the offset is invalid.**See Also:**[String.valueOf(float)](http://docs.google.com/java/lang/String.html#valueOf(float)), [insert(int, java.lang.String)](http://docs.google.com/java/lang/StringBuilder.html#insert(int,%20java.lang.String)), [length()](http://docs.google.com/java/lang/StringBuilder.html#length())

### insert

public [StringBuilder](http://docs.google.com/java/lang/StringBuilder.html) **insert**(int offset,  
 double d)

Inserts the string representation of the double argument into this sequence.

The second argument is converted to a string as if by the method String.valueOf, and the characters of that string are then inserted into this sequence at the indicated offset.

The offset argument must be greater than or equal to 0, and less than or equal to the length of this sequence.

**Parameters:**offset - the offset.d - a double. **Returns:**a reference to this object. **Throws:** [StringIndexOutOfBoundsException](http://docs.google.com/java/lang/StringIndexOutOfBoundsException.html) - if the offset is invalid.**See Also:**[String.valueOf(double)](http://docs.google.com/java/lang/String.html#valueOf(double)), [insert(int, java.lang.String)](http://docs.google.com/java/lang/StringBuilder.html#insert(int,%20java.lang.String)), [length()](http://docs.google.com/java/lang/StringBuilder.html#length())

### indexOf

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

Returns the index within this string of the first occurrence of the specified substring. The integer returned is the smallest value *k* such that:

this.toString().startsWith(str, *k*)

is true.

**Parameters:**str - any string. **Returns:**if the string argument occurs as a substring within this object, then the index of the first character of the first such substring is returned; if it does not occur as a substring, -1 is returned. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if str is null.

### indexOf

public int **indexOf**([String](http://docs.google.com/java/lang/String.html) str,  
 int fromIndex)

Returns the index within this string of the first occurrence of the specified substring, starting at the specified index. The integer returned is the smallest value k for which:

k >= Math.min(fromIndex, str.length()) &&  
 this.toString().startsWith(str, k)

If no such value of *k* exists, then -1 is returned.

**Parameters:**str - the substring for which to search.fromIndex - the index from which to start the search. **Returns:**the index within this string of the first occurrence of the specified substring, starting at the specified index. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if str is null.

### lastIndexOf

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

Returns the index within this string of the rightmost occurrence of the specified substring. The rightmost empty string "" is considered to occur at the index value this.length(). The returned index is the largest value *k* such that

this.toString().startsWith(str, k)

is true.

**Parameters:**str - the substring to search for. **Returns:**if the string argument occurs one or more times as a substring within this object, then the index of the first character of the last such substring is returned. If it does not occur as a substring, -1 is returned. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if str is null.

### lastIndexOf

public int **lastIndexOf**([String](http://docs.google.com/java/lang/String.html) str,  
 int fromIndex)

Returns the index within this string of the last occurrence of the specified substring. The integer returned is the largest value *k* such that:

k <= Math.min(fromIndex, str.length()) &&  
 this.toString().startsWith(str, k)

If no such value of *k* exists, then -1 is returned.

**Parameters:**str - the substring to search for.fromIndex - the index to start the search from. **Returns:**the index within this sequence of the last occurrence of the specified substring. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if str is null.

### reverse

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

Causes this character sequence to be replaced by the reverse of the sequence. If there are any surrogate pairs included in the sequence, these are treated as single characters for the reverse operation. Thus, the order of the high-low surrogates is never reversed. Let *n* be the character length of this character sequence (not the length in char values) just prior to execution of the reverse method. Then the character at index *k* in the new character sequence is equal to the character at index *n-k-1* in the old character sequence.

Note that the reverse operation may result in producing surrogate pairs that were unpaired low-surrogates and high-surrogates before the operation. For example, reversing "\uDC00\uD800" produces "\uD800\uDC00" which is a valid surrogate pair.

**Returns:**a reference to this object.

### toString

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

Returns a string representing the data in this sequence. A new String object is allocated and initialized to contain the character sequence currently represented by this object. This String is then returned. Subsequent changes to this sequence do not affect the contents of the String.

**Specified by:**[toString](http://docs.google.com/java/lang/CharSequence.html#toString()) in interface [CharSequence](http://docs.google.com/java/lang/CharSequence.html) **Returns:**a string representation of this sequence of characters.

### length

public int **length**()

Returns the length (character count).

**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 the sequence of characters currently represented by this object

### capacity

public int **capacity**()

Returns the current capacity. The capacity is the amount of storage available for newly inserted characters, beyond which an allocation will occur.

**Returns:**the current capacity

### ensureCapacity

public void **ensureCapacity**(int minimumCapacity)

Ensures that the capacity is at least equal to the specified minimum. If the current capacity is less than the argument, then a new internal array is allocated with greater capacity. The new capacity is the larger of:

* The minimumCapacity argument.
* Twice the old capacity, plus 2.

If the minimumCapacity argument is nonpositive, this method takes no action and simply returns.

**Parameters:**minimumCapacity - the minimum desired capacity.

### trimToSize

public void **trimToSize**()

Attempts to reduce storage used for the character sequence. If the buffer is larger than necessary to hold its current sequence of characters, then it may be resized to become more space efficient. Calling this method may, but is not required to, affect the value returned by a subsequent call to the [capacity()](http://docs.google.com/java/lang/StringBuilder.html#capacity()) method.

### setLength

public void **setLength**(int newLength)

Sets the length of the character sequence. The sequence is changed to a new character sequence whose length is specified by the argument. For every nonnegative index *k* less than newLength, the character at index *k* in the new character sequence is the same as the character at index *k* in the old sequence if *k* is less than the length of the old character sequence; otherwise, it is the null character '\u0000'. In other words, if the newLength argument is less than the current length, the length is changed to the specified length.

If the newLength argument is greater than or equal to the current length, sufficient null characters ('\u0000') are appended so that length becomes the newLength argument.

The newLength argument must be greater than or equal to 0.

**Parameters:**newLength - the new length **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if the newLength argument is negative.

### charAt

public char **charAt**(int index)

Returns the char value in this sequence at the specified index. The first char value is at index 0, the next at index 1, and so on, as in array indexing.

The index argument must be greater than or equal to 0, and less than the length of this sequence.

If the char value specified by the index is a [surrogate](http://docs.google.com/Character.html#unicode), the surrogate value is returned.

**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 desired char value. **Returns:**the char value at the specified index. **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is negative or greater than or equal to length().

### codePointAt

public int **codePointAt**(int index)

Returns the character (Unicode code point) at the specified index. The index refers to char values (Unicode code units) and ranges from 0 to [length()](http://docs.google.com/java/lang/StringBuilder.html#length()) - 1.

If the char value specified at the given index is in the high-surrogate range, the following index is less than the length of this sequence, and the char value at the following index is in the low-surrogate range, then the supplementary code point corresponding to this surrogate pair is returned. Otherwise, the char value at the given index is returned.

**Parameters:**index - the index to the char values **Returns:**the code point value of the character at the index **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if the index argument is negative or not less than the length of this sequence.

### codePointBefore

public int **codePointBefore**(int index)

Returns the character (Unicode code point) before the specified index. The index refers to char values (Unicode code units) and ranges from 1 to [length()](http://docs.google.com/java/lang/StringBuilder.html#length()).

If the char value at (index - 1) is in the low-surrogate range, (index - 2) is not negative, and the char value at (index - 2) is in the high-surrogate range, then the supplementary code point value of the surrogate pair is returned. If the char value at index - 1 is an unpaired low-surrogate or a high-surrogate, the surrogate value is returned.

**Parameters:**index - the index following the code point that should be returned **Returns:**the Unicode code point value before the given index. **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if the index argument is less than 1 or greater than the length of this sequence.

### codePointCount

public int **codePointCount**(int beginIndex,  
 int endIndex)

Returns the number of Unicode code points in the specified text range of this sequence. The text range begins at the specified beginIndex and extends to the char at index endIndex - 1. Thus the length (in chars) of the text range is endIndex-beginIndex. Unpaired surrogates within this sequence count as one code point each.

**Parameters:**beginIndex - the index to the first char of the text range.endIndex - the index after the last char of the text range. **Returns:**the number of Unicode code points in the specified text range **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if the beginIndex is negative, or endIndex is larger than the length of this sequence, or beginIndex is larger than endIndex.

### offsetByCodePoints

public int **offsetByCodePoints**(int index,  
 int codePointOffset)

Returns the index within this sequence that is offset from the given index by codePointOffset code points. Unpaired surrogates within the text range given by index and codePointOffset count as one code point each.

**Parameters:**index - the index to be offsetcodePointOffset - the offset in code points **Returns:**the index within this sequence **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is negative or larger then the length of this sequence, or if codePointOffset is positive and the subsequence starting with index has fewer than codePointOffset code points, or if codePointOffset is negative and the subsequence before index has fewer than the absolute value of codePointOffset code points.

### getChars

public void **getChars**(int srcBegin,  
 int srcEnd,  
 char[] dst,  
 int dstBegin)

Characters are copied from this sequence into the destination character array dst. The first character to be copied is at index srcBegin; the last character to be copied is at index srcEnd-1. The total number of characters to be copied is srcEnd-srcBegin. The characters are copied into the subarray of dst starting at index dstBegin and ending at index:

dstbegin + (srcEnd-srcBegin) - 1

**Parameters:**srcBegin - start copying at this offset.srcEnd - stop copying at this offset.dst - the array to copy the data into.dstBegin - offset into dst. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if dst is null. [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if any of the following is true:

* srcBegin is negative
* dstBegin is negative
* the srcBegin argument is greater than the srcEnd argument.
* srcEnd is greater than this.length().
* dstBegin+srcEnd-srcBegin is greater than dst.length

### setCharAt

public void **setCharAt**(int index,  
 char ch)

The character at the specified index is set to ch. This sequence is altered to represent a new character sequence that is identical to the old character sequence, except that it contains the character ch at position index.

The index argument must be greater than or equal to 0, and less than the length of this sequence.

**Parameters:**index - the index of the character to modify.ch - the new character. **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is negative or greater than or equal to length().

### substring

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

Returns a new String that contains a subsequence of characters currently contained in this character sequence. The substring begins at the specified index and extends to the end of this sequence.

**Parameters:**start - The beginning index, inclusive. **Returns:**The new string. **Throws:** [StringIndexOutOfBoundsException](http://docs.google.com/java/lang/StringIndexOutOfBoundsException.html) - if start is less than zero, or greater than the length of this object.

### subSequence

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

Returns a new character sequence that is a subsequence of this sequence.

An invocation of this method of the form

sb.subSequence(begin, end)

behaves in exactly the same way as the invocation

sb.substring(begin, end)

This method is provided so that this class can implement the [CharSequence](http://docs.google.com/java/lang/CharSequence.html) interface.

**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 start index, inclusive.end - the end index, exclusive. **Returns:**the specified subsequence. **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if start or end are negative, if end is greater than length(), or if start is greater than end

### substring

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

Returns a new String that contains a subsequence of characters currently contained in this sequence. The substring begins at the specified start and extends to the character at index end - 1.

**Parameters:**start - The beginning index, inclusive.end - The ending index, exclusive. **Returns:**The new string. **Throws:** [StringIndexOutOfBoundsException](http://docs.google.com/java/lang/StringIndexOutOfBoundsException.html) - if start or end are negative or greater than length(), or start is greater than end.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/StringBuilder.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/lang/StringBuffer.html)   [**NEXT CLASS**](http://docs.google.com/java/lang/StringIndexOutOfBoundsException.html) | [**FRAMES**](http://docs.google.com/index.html?java/lang/StringBuilder.html)    [**NO FRAMES**](http://docs.google.com/StringBuilder.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#1t3h5sf) | [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).