| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/TextMeasurer.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/awt/font/TextLayout.CaretPolicy.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/font/TransformAttribute.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/font/TextMeasurer.html)    [**NO FRAMES**](http://docs.google.com/TextMeasurer.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#4d34og8) |

## **java.awt.font**

Class TextMeasurer

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

**All Implemented Interfaces:** [Cloneable](http://docs.google.com/java/lang/Cloneable.html)

public final class **TextMeasurer**extends [Object](http://docs.google.com/java/lang/Object.html)implements [Cloneable](http://docs.google.com/java/lang/Cloneable.html)

The TextMeasurer class provides the primitive operations needed for line break: measuring up to a given advance, determining the advance of a range of characters, and generating a TextLayout for a range of characters. It also provides methods for incremental editing of paragraphs.

A TextMeasurer object is constructed with an [AttributedCharacterIterator](http://docs.google.com/java/text/AttributedCharacterIterator.html) representing a single paragraph of text. The value returned by the [getBeginIndex](http://docs.google.com/java/text/CharacterIterator.html#getBeginIndex()) method of AttributedCharacterIterator defines the absolute index of the first character. The value returned by the [getEndIndex](http://docs.google.com/java/text/CharacterIterator.html#getEndIndex()) method of AttributedCharacterIterator defines the index past the last character. These values define the range of indexes to use in calls to the TextMeasurer. For example, calls to get the advance of a range of text or the line break of a range of text must use indexes between the beginning and end index values. Calls to [insertChar](http://docs.google.com/java/awt/font/TextMeasurer.html#insertChar(java.text.AttributedCharacterIterator,%20int)) and [deleteChar](http://docs.google.com/java/awt/font/TextMeasurer.html#deleteChar(java.text.AttributedCharacterIterator,%20int)) reset the TextMeasurer to use the beginning index and end index of the AttributedCharacterIterator passed in those calls.

Most clients will use the more convenient LineBreakMeasurer, which implements the standard line break policy (placing as many words as will fit on each line).

**Since:** 1.3 **See Also:**[LineBreakMeasurer](http://docs.google.com/java/awt/font/LineBreakMeasurer.html)

| **Constructor Summary** | |
| --- | --- |
| [**TextMeasurer**](http://docs.google.com/java/awt/font/TextMeasurer.html#TextMeasurer(java.text.AttributedCharacterIterator,%20java.awt.font.FontRenderContext))([AttributedCharacterIterator](http://docs.google.com/java/text/AttributedCharacterIterator.html) text, [FontRenderContext](http://docs.google.com/java/awt/font/FontRenderContext.html) frc)            Constructs a TextMeasurer from the source text. |

| **Method Summary** | |
| --- | --- |
| protected  [Object](http://docs.google.com/java/lang/Object.html) | [**clone**](http://docs.google.com/java/awt/font/TextMeasurer.html#clone())()            Creates and returns a copy of this object. |
| void | [**deleteChar**](http://docs.google.com/java/awt/font/TextMeasurer.html#deleteChar(java.text.AttributedCharacterIterator,%20int))([AttributedCharacterIterator](http://docs.google.com/java/text/AttributedCharacterIterator.html) newParagraph, int deletePos)            Updates the TextMeasurer after a single character has been deleted from the paragraph currently represented by this TextMeasurer. |
| float | [**getAdvanceBetween**](http://docs.google.com/java/awt/font/TextMeasurer.html#getAdvanceBetween(int,%20int))(int start, int limit)            Returns the graphical width of a line beginning at start and including characters up to limit. |
| [TextLayout](http://docs.google.com/java/awt/font/TextLayout.html) | [**getLayout**](http://docs.google.com/java/awt/font/TextMeasurer.html#getLayout(int,%20int))(int start, int limit)            Returns a TextLayout on the given character range. |
| int | [**getLineBreakIndex**](http://docs.google.com/java/awt/font/TextMeasurer.html#getLineBreakIndex(int,%20float))(int start, float maxAdvance)            Returns the index of the first character which will not fit on on a line beginning at start and possible measuring up to maxAdvance in graphical width. |
| void | [**insertChar**](http://docs.google.com/java/awt/font/TextMeasurer.html#insertChar(java.text.AttributedCharacterIterator,%20int))([AttributedCharacterIterator](http://docs.google.com/java/text/AttributedCharacterIterator.html) newParagraph, int insertPos)            Updates the TextMeasurer after a single character has been inserted into the paragraph currently represented by this TextMeasurer. |

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

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

### TextMeasurer

public **TextMeasurer**([AttributedCharacterIterator](http://docs.google.com/java/text/AttributedCharacterIterator.html) text,  
 [FontRenderContext](http://docs.google.com/java/awt/font/FontRenderContext.html) frc)

Constructs a TextMeasurer from the source text. The source text should be a single entire paragraph.

**Parameters:**text - the source paragraph. Cannot be null.frc - the information about a graphics device which is needed to measure the text correctly. Cannot be null.

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

### clone

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

**Description copied from class:** [**Object**](http://docs.google.com/java/lang/Object.html#clone()) Creates and returns a copy of this object. The precise meaning of "copy" may depend on the class of the object. The general intent is that, for any object x, the expression:

x.clone() != x

will be true, and that the expression:

x.clone().getClass() == x.getClass()

will be true, but these are not absolute requirements. While it is typically the case that:

x.clone().equals(x)

will be true, this is not an absolute requirement.

By convention, the returned object should be obtained by calling super.clone. If a class and all of its superclasses (except Object) obey this convention, it will be the case that x.clone().getClass() == x.getClass().

By convention, the object returned by this method should be independent of this object (which is being cloned). To achieve this independence, it may be necessary to modify one or more fields of the object returned by super.clone before returning it. Typically, this means copying any mutable objects that comprise the internal "deep structure" of the object being cloned and replacing the references to these objects with references to the copies. If a class contains only primitive fields or references to immutable objects, then it is usually the case that no fields in the object returned by super.clone need to be modified.

The method clone for class Object performs a specific cloning operation. First, if the class of this object does not implement the interface Cloneable, then a CloneNotSupportedException is thrown. Note that all arrays are considered to implement the interface Cloneable. Otherwise, this method creates a new instance of the class of this object and initializes all its fields with exactly the contents of the corresponding fields of this object, as if by assignment; the contents of the fields are not themselves cloned. Thus, this method performs a "shallow copy" of this object, not a "deep copy" operation.

The class Object does not itself implement the interface Cloneable, so calling the clone method on an object whose class is Object will result in throwing an exception at run time.

**Overrides:**[clone](http://docs.google.com/java/lang/Object.html#clone()) in class [Object](http://docs.google.com/java/lang/Object.html) **Returns:**a clone of this instance.**See Also:**[Cloneable](http://docs.google.com/java/lang/Cloneable.html)

### getLineBreakIndex

public int **getLineBreakIndex**(int start,  
 float maxAdvance)

Returns the index of the first character which will not fit on on a line beginning at start and possible measuring up to maxAdvance in graphical width.

**Parameters:**start - the character index at which to start measuring. start is an absolute index, not relative to the start of the paragraphmaxAdvance - the graphical width in which the line must fit **Returns:**the index after the last character that will fit on a line beginning at start, which is not longer than maxAdvance in graphical width **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if start is less than the beginning of the paragraph.

### getAdvanceBetween

public float **getAdvanceBetween**(int start,  
 int limit)

Returns the graphical width of a line beginning at start and including characters up to limit. start and limit are absolute indices, not relative to the start of the paragraph.

**Parameters:**start - the character index at which to start measuringlimit - the character index at which to stop measuring **Returns:**the graphical width of a line beginning at start and including characters up to limit **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if limit is less than start [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if start or limit is not between the beginning of the paragraph and the end of the paragraph.

### getLayout

public [TextLayout](http://docs.google.com/java/awt/font/TextLayout.html) **getLayout**(int start,  
 int limit)

Returns a TextLayout on the given character range.

**Parameters:**start - the index of the first characterlimit - the index after the last character. Must be greater than start **Returns:**a TextLayout for the characters beginning at start up to (but not including) limit **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if limit is less than start [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if start or limit is not between the beginning of the paragraph and the end of the paragraph.

### insertChar

public void **insertChar**([AttributedCharacterIterator](http://docs.google.com/java/text/AttributedCharacterIterator.html) newParagraph,  
 int insertPos)

Updates the TextMeasurer after a single character has been inserted into the paragraph currently represented by this TextMeasurer. After this call, this TextMeasurer is equivalent to a new TextMeasurer created from the text; however, it will usually be more efficient to update an existing TextMeasurer than to create a new one from scratch.

**Parameters:**newParagraph - the text of the paragraph after performing the insertion. Cannot be null.insertPos - the position in the text where the character was inserted. Must not be less than the start of newParagraph, and must be less than the end of newParagraph. **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if insertPos is less than the start of newParagraph or greater than or equal to the end of newParagraph [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if newParagraph is null

### deleteChar

public void **deleteChar**([AttributedCharacterIterator](http://docs.google.com/java/text/AttributedCharacterIterator.html) newParagraph,  
 int deletePos)

Updates the TextMeasurer after a single character has been deleted from the paragraph currently represented by this TextMeasurer. After this call, this TextMeasurer is equivalent to a new TextMeasurer created from the text; however, it will usually be more efficient to update an existing TextMeasurer than to create a new one from scratch.

**Parameters:**newParagraph - the text of the paragraph after performing the deletion. Cannot be null.deletePos - the position in the text where the character was removed. Must not be less than the start of newParagraph, and must not be greater than the end of newParagraph. **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if deletePos is less than the start of newParagraph or greater than the end of newParagraph [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if newParagraph is null

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/TextMeasurer.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/awt/font/TextLayout.CaretPolicy.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/font/TransformAttribute.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/font/TextMeasurer.html)    [**NO FRAMES**](http://docs.google.com/TextMeasurer.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#4d34og8) |

[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).