| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/TextAttribute.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/ShapeGraphicAttribute.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/font/TextHitInfo.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/font/TextAttribute.html)    [**NO FRAMES**](http://docs.google.com/TextAttribute.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#tyjcwt) | [METHOD](#3dy6vkm) | DETAIL: [FIELD](#2s8eyo1) | [CONSTR](#34g0dwd) | [METHOD](#43ky6rz) |

## **java.awt.font**

Class TextAttribute

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 [java.text.AttributedCharacterIterator.Attribute](http://docs.google.com/java/text/AttributedCharacterIterator.Attribute.html)  
 **java.awt.font.TextAttribute**

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

public final class **TextAttribute**extends [AttributedCharacterIterator.Attribute](http://docs.google.com/java/text/AttributedCharacterIterator.Attribute.html)

The TextAttribute class defines attribute keys and attribute values used for text rendering.

TextAttribute instances are used as attribute keys to identify attributes in [Font](http://docs.google.com/java/awt/Font.html), [TextLayout](http://docs.google.com/java/awt/font/TextLayout.html), [AttributedCharacterIterator](http://docs.google.com/java/text/AttributedCharacterIterator.html), and other classes handling text attributes. Other constants defined in this class can be used as attribute values.

For each text attribute, the documentation provides:

* the type of its value,
* the relevant predefined constants, if any
* the default effect if the attribute is absent
* the valid values if there are limitations
* a description of the effect.

#### Values

* The values of attributes must always be immutable.
* Where value limitations are given, any value outside of that set is reserved for future use; the value will be treated as the default.
* The value null is treated the same as the default value and results in the default behavior.
* If the value is not of the proper type, the attribute will be ignored.
* The identity of the value does not matter, only the actual value. For example, TextAttribute.WEIGHT\_BOLD and new Float(2.0) indicate the same WEIGHT.
* Attribute values of type Number (used for WEIGHT, WIDTH, POSTURE, SIZE, JUSTIFICATION, and TRACKING) can vary along their natural range and are not restricted to the predefined constants. Number.floatValue() is used to get the actual value from the Number.
* The values for WEIGHT, WIDTH, and POSTURE are interpolated by the system, which can select the 'nearest available' font or use other techniques to approximate the user's request.

#### Summary of attributes

| Key | Value Type | Principal Constants | Default Value |
| --- | --- | --- | --- |
| [FAMILY](http://docs.google.com/java/awt/font/TextAttribute.html#FAMILY) | String | See Font [DIALOG](http://docs.google.com/java/awt/Font.html#DIALOG), [DIALOG\_INPUT](http://docs.google.com/java/awt/Font.html#DIALOG_INPUT),  [SERIF](http://docs.google.com/java/awt/Font.html#SERIF), [SANS\_SERIF](http://docs.google.com/java/awt/Font.html#SANS_SERIF), and [MONOSPACED](http://docs.google.com/java/awt/Font.html#MONOSPACED). | "Default" (use platform default) |
| [WEIGHT](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT) | Number | WEIGHT\_REGULAR, WEIGHT\_BOLD | WEIGHT\_REGULAR |
| [WIDTH](http://docs.google.com/java/awt/font/TextAttribute.html#WIDTH) | Number | WIDTH\_CONDENSED, WIDTH\_REGULAR,  WIDTH\_EXTENDED | WIDTH\_REGULAR |
| [POSTURE](http://docs.google.com/java/awt/font/TextAttribute.html#POSTURE) | Number | POSTURE\_REGULAR, POSTURE\_OBLIQUE | POSTURE\_REGULAR |
| [SIZE](http://docs.google.com/java/awt/font/TextAttribute.html#SIZE) | Number | none | 12.0 |
| [TRANSFORM](http://docs.google.com/java/awt/font/TextAttribute.html#TRANSFORM) | [TransformAttribute](http://docs.google.com/java/awt/font/TransformAttribute.html) | See TransformAttribute [IDENTITY](http://docs.google.com/java/awt/font/TransformAttribute.html#IDENTITY) | TransformAttribute.IDENTITY |
| [SUPERSCRIPT](http://docs.google.com/java/awt/font/TextAttribute.html#SUPERSCRIPT) | Integer | SUPERSCRIPT\_SUPER, SUPERSCRIPT\_SUB | 0 (use the standard glyphs and metrics) |
| [FONT](http://docs.google.com/java/awt/font/TextAttribute.html#FONT) | [Font](http://docs.google.com/java/awt/Font.html) | none | null (do not override font resolution) |
| [CHAR\_REPLACEMENT](http://docs.google.com/java/awt/font/TextAttribute.html#CHAR_REPLACEMENT) | [GraphicAttribute](http://docs.google.com/java/awt/font/GraphicAttribute.html) | none | null (draw text using font glyphs) |
| [FOREGROUND](http://docs.google.com/java/awt/font/TextAttribute.html#FOREGROUND) | [Paint](http://docs.google.com/java/awt/Paint.html) | none | null (use current graphics paint) |
| [BACKGROUND](http://docs.google.com/java/awt/font/TextAttribute.html#BACKGROUND) | [Paint](http://docs.google.com/java/awt/Paint.html) | none | null (do not render background) |
| [UNDERLINE](http://docs.google.com/java/awt/font/TextAttribute.html#UNDERLINE) | Integer | UNDERLINE\_ON | -1 (do not render underline) |
| [STRIKETHROUGH](http://docs.google.com/java/awt/font/TextAttribute.html#STRIKETHROUGH) | Boolean | STRIKETHROUGH\_ON | false (do not render strikethrough) |
| [RUN\_DIRECTION](http://docs.google.com/java/awt/font/TextAttribute.html#RUN_DIRECTION) | Boolean | RUN\_DIRECTION\_LTR  RUN\_DIRECTION\_RTL | null (use [Bidi](http://docs.google.com/java/text/Bidi.html) standard default) |
| [BIDI\_EMBEDDING](http://docs.google.com/java/awt/font/TextAttribute.html#BIDI_EMBEDDING) | Integer | none | 0 (use base line direction) |
| [JUSTIFICATION](http://docs.google.com/java/awt/font/TextAttribute.html#JUSTIFICATION) | Number | JUSTIFICATION\_FULL | JUSTIFICATION\_FULL |
| [INPUT\_METHOD\_HIGHLIGHT](http://docs.google.com/java/awt/font/TextAttribute.html#INPUT_METHOD_HIGHLIGHT) | [InputMethodHighlight](http://docs.google.com/java/awt/im/InputMethodHighlight.html),  [Annotation](http://docs.google.com/java/text/Annotation.html) | (see class) | null (do not apply input highlighting) |
| [INPUT\_METHOD\_UNDERLINE](http://docs.google.com/java/awt/font/TextAttribute.html#INPUT_METHOD_UNDERLINE) | Integer | UNDERLINE\_LOW\_ONE\_PIXEL,  UNDERLINE\_LOW\_TWO\_PIXEL | -1 (do not render underline) |
| [SWAP\_COLORS](http://docs.google.com/java/awt/font/TextAttribute.html#SWAP_COLORS) | Boolean | SWAP\_COLORS\_ON | false (do not swap colors) |
| [NUMERIC\_SHAPING](http://docs.google.com/java/awt/font/TextAttribute.html#NUMERIC_SHAPING) | [NumericShaper](http://docs.google.com/java/awt/font/NumericShaper.html) | none | null (do not shape digits) |
| [KERNING](http://docs.google.com/java/awt/font/TextAttribute.html#KERNING) | Integer | KERNING\_ON | 0 (do not request kerning) |
| [LIGATURES](http://docs.google.com/java/awt/font/TextAttribute.html#LIGATURES) | Integer | LIGATURES\_ON | 0 (do not form optional ligatures) |
| [TRACKING](http://docs.google.com/java/awt/font/TextAttribute.html#TRACKING) | Number | TRACKING\_LOOSE, TRACKING\_TIGHT | 0 (do not add tracking) |

**See Also:**[Font](http://docs.google.com/java/awt/Font.html), [TextLayout](http://docs.google.com/java/awt/font/TextLayout.html), [AttributedCharacterIterator](http://docs.google.com/java/text/AttributedCharacterIterator.html), [Serialized Form](http://docs.google.com/serialized-form.html#java.awt.font.TextAttribute)

| **Field Summary** | |
| --- | --- |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**BACKGROUND**](http://docs.google.com/java/awt/font/TextAttribute.html#BACKGROUND)            Attribute key for the paint used to render the background of the text. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**BIDI\_EMBEDDING**](http://docs.google.com/java/awt/font/TextAttribute.html#BIDI_EMBEDDING)            Attribute key for the embedding level of the text. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**CHAR\_REPLACEMENT**](http://docs.google.com/java/awt/font/TextAttribute.html#CHAR_REPLACEMENT)            Attribute key for a user-defined glyph to display in lieu of the font's standard glyph for a character. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**FAMILY**](http://docs.google.com/java/awt/font/TextAttribute.html#FAMILY)            Attribute key for the font name. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**FONT**](http://docs.google.com/java/awt/font/TextAttribute.html#FONT)            Attribute key used to provide the font to use to render text. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**FOREGROUND**](http://docs.google.com/java/awt/font/TextAttribute.html#FOREGROUND)            Attribute key for the paint used to render the text. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**INPUT\_METHOD\_HIGHLIGHT**](http://docs.google.com/java/awt/font/TextAttribute.html#INPUT_METHOD_HIGHLIGHT)            Attribute key for input method highlight styles. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**INPUT\_METHOD\_UNDERLINE**](http://docs.google.com/java/awt/font/TextAttribute.html#INPUT_METHOD_UNDERLINE)            Attribute key for input method underlines. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**JUSTIFICATION**](http://docs.google.com/java/awt/font/TextAttribute.html#JUSTIFICATION)            Attribute key for the justification of a paragraph. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**JUSTIFICATION\_FULL**](http://docs.google.com/java/awt/font/TextAttribute.html#JUSTIFICATION_FULL)            Justify the line to the full requested width. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**JUSTIFICATION\_NONE**](http://docs.google.com/java/awt/font/TextAttribute.html#JUSTIFICATION_NONE)            Do not allow the line to be justified. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**KERNING**](http://docs.google.com/java/awt/font/TextAttribute.html#KERNING)            Attribute key to request kerning. |
| static [Integer](http://docs.google.com/java/lang/Integer.html) | [**KERNING\_ON**](http://docs.google.com/java/awt/font/TextAttribute.html#KERNING_ON)            Request standard kerning. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**LIGATURES**](http://docs.google.com/java/awt/font/TextAttribute.html#LIGATURES)            Attribute key for enabling optional ligatures. |
| static [Integer](http://docs.google.com/java/lang/Integer.html) | [**LIGATURES\_ON**](http://docs.google.com/java/awt/font/TextAttribute.html#LIGATURES_ON)            Request standard optional ligatures. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**NUMERIC\_SHAPING**](http://docs.google.com/java/awt/font/TextAttribute.html#NUMERIC_SHAPING)            Attribute key for converting ASCII decimal digits to other decimal ranges. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**POSTURE**](http://docs.google.com/java/awt/font/TextAttribute.html#POSTURE)            Attribute key for the posture of a font. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**POSTURE\_OBLIQUE**](http://docs.google.com/java/awt/font/TextAttribute.html#POSTURE_OBLIQUE)            The standard italic posture. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**POSTURE\_REGULAR**](http://docs.google.com/java/awt/font/TextAttribute.html#POSTURE_REGULAR)            The standard posture, upright. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**RUN\_DIRECTION**](http://docs.google.com/java/awt/font/TextAttribute.html#RUN_DIRECTION)            Attribute key for the run direction of the line. |
| static [Boolean](http://docs.google.com/java/lang/Boolean.html) | [**RUN\_DIRECTION\_LTR**](http://docs.google.com/java/awt/font/TextAttribute.html#RUN_DIRECTION_LTR)            Left-to-right run direction. |
| static [Boolean](http://docs.google.com/java/lang/Boolean.html) | [**RUN\_DIRECTION\_RTL**](http://docs.google.com/java/awt/font/TextAttribute.html#RUN_DIRECTION_RTL)            Right-to-left run direction. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**SIZE**](http://docs.google.com/java/awt/font/TextAttribute.html#SIZE)            Attribute key for the font size. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**STRIKETHROUGH**](http://docs.google.com/java/awt/font/TextAttribute.html#STRIKETHROUGH)            Attribute key for strikethrough. |
| static [Boolean](http://docs.google.com/java/lang/Boolean.html) | [**STRIKETHROUGH\_ON**](http://docs.google.com/java/awt/font/TextAttribute.html#STRIKETHROUGH_ON)            A single strikethrough. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**SUPERSCRIPT**](http://docs.google.com/java/awt/font/TextAttribute.html#SUPERSCRIPT)            Attribute key for superscripting and subscripting. |
| static [Integer](http://docs.google.com/java/lang/Integer.html) | [**SUPERSCRIPT\_SUB**](http://docs.google.com/java/awt/font/TextAttribute.html#SUPERSCRIPT_SUB)            Standard subscript. |
| static [Integer](http://docs.google.com/java/lang/Integer.html) | [**SUPERSCRIPT\_SUPER**](http://docs.google.com/java/awt/font/TextAttribute.html#SUPERSCRIPT_SUPER)            Standard superscript. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**SWAP\_COLORS**](http://docs.google.com/java/awt/font/TextAttribute.html#SWAP_COLORS)            Attribute key for swapping foreground and background Paints. |
| static [Boolean](http://docs.google.com/java/lang/Boolean.html) | [**SWAP\_COLORS\_ON**](http://docs.google.com/java/awt/font/TextAttribute.html#SWAP_COLORS_ON)            Swap foreground and background. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**TRACKING**](http://docs.google.com/java/awt/font/TextAttribute.html#TRACKING)            Attribute key to control tracking. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**TRACKING\_LOOSE**](http://docs.google.com/java/awt/font/TextAttribute.html#TRACKING_LOOSE)            Perform loose tracking. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**TRACKING\_TIGHT**](http://docs.google.com/java/awt/font/TextAttribute.html#TRACKING_TIGHT)            Perform tight tracking. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**TRANSFORM**](http://docs.google.com/java/awt/font/TextAttribute.html#TRANSFORM)            Attribute key for the transform of a font. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**UNDERLINE**](http://docs.google.com/java/awt/font/TextAttribute.html#UNDERLINE)            Attribute key for underline. |
| static [Integer](http://docs.google.com/java/lang/Integer.html) | [**UNDERLINE\_LOW\_DASHED**](http://docs.google.com/java/awt/font/TextAttribute.html#UNDERLINE_LOW_DASHED)            Single pixel dashed low underline. |
| static [Integer](http://docs.google.com/java/lang/Integer.html) | [**UNDERLINE\_LOW\_DOTTED**](http://docs.google.com/java/awt/font/TextAttribute.html#UNDERLINE_LOW_DOTTED)            Single pixel dotted low underline. |
| static [Integer](http://docs.google.com/java/lang/Integer.html) | [**UNDERLINE\_LOW\_GRAY**](http://docs.google.com/java/awt/font/TextAttribute.html#UNDERLINE_LOW_GRAY)            Double pixel gray low underline. |
| static [Integer](http://docs.google.com/java/lang/Integer.html) | [**UNDERLINE\_LOW\_ONE\_PIXEL**](http://docs.google.com/java/awt/font/TextAttribute.html#UNDERLINE_LOW_ONE_PIXEL)            Single pixel solid low underline. |
| static [Integer](http://docs.google.com/java/lang/Integer.html) | [**UNDERLINE\_LOW\_TWO\_PIXEL**](http://docs.google.com/java/awt/font/TextAttribute.html#UNDERLINE_LOW_TWO_PIXEL)            Double pixel solid low underline. |
| static [Integer](http://docs.google.com/java/lang/Integer.html) | [**UNDERLINE\_ON**](http://docs.google.com/java/awt/font/TextAttribute.html#UNDERLINE_ON)            Standard underline. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**WEIGHT**](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT)            Attribute key for the weight of a font. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**WEIGHT\_BOLD**](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_BOLD)            The standard bold weight. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**WEIGHT\_DEMIBOLD**](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_DEMIBOLD)            A moderately lighter weight than WEIGHT\_BOLD. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**WEIGHT\_DEMILIGHT**](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_DEMILIGHT)            An intermediate weight between WEIGHT\_LIGHT and WEIGHT\_STANDARD. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**WEIGHT\_EXTRA\_LIGHT**](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_EXTRA_LIGHT)            The lightest predefined weight. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**WEIGHT\_EXTRABOLD**](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_EXTRABOLD)            An extra heavy weight. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**WEIGHT\_HEAVY**](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_HEAVY)            A moderately heavier weight than WEIGHT\_BOLD. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**WEIGHT\_LIGHT**](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_LIGHT)            The standard light weight. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**WEIGHT\_MEDIUM**](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_MEDIUM)            An intermediate weight between WEIGHT\_REGULAR and WEIGHT\_BOLD. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**WEIGHT\_REGULAR**](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_REGULAR)            The standard weight. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**WEIGHT\_SEMIBOLD**](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_SEMIBOLD)            A moderately heavier weight than WEIGHT\_REGULAR. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**WEIGHT\_ULTRABOLD**](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_ULTRABOLD)            The heaviest predefined weight. |
| static [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) | [**WIDTH**](http://docs.google.com/java/awt/font/TextAttribute.html#WIDTH)            Attribute key for the width of a font. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**WIDTH\_CONDENSED**](http://docs.google.com/java/awt/font/TextAttribute.html#WIDTH_CONDENSED)            The most condensed predefined width. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**WIDTH\_EXTENDED**](http://docs.google.com/java/awt/font/TextAttribute.html#WIDTH_EXTENDED)            The most extended predefined width. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**WIDTH\_REGULAR**](http://docs.google.com/java/awt/font/TextAttribute.html#WIDTH_REGULAR)            The standard width. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**WIDTH\_SEMI\_CONDENSED**](http://docs.google.com/java/awt/font/TextAttribute.html#WIDTH_SEMI_CONDENSED)            A moderately condensed width. |
| static [Float](http://docs.google.com/java/lang/Float.html) | [**WIDTH\_SEMI\_EXTENDED**](http://docs.google.com/java/awt/font/TextAttribute.html#WIDTH_SEMI_EXTENDED)            A moderately extended width. |

| **Fields inherited from class java.text.**[**AttributedCharacterIterator.Attribute**](http://docs.google.com/java/text/AttributedCharacterIterator.Attribute.html) |
| --- |
| [INPUT\_METHOD\_SEGMENT](http://docs.google.com/java/text/AttributedCharacterIterator.Attribute.html#INPUT_METHOD_SEGMENT), [LANGUAGE](http://docs.google.com/java/text/AttributedCharacterIterator.Attribute.html#LANGUAGE), [READING](http://docs.google.com/java/text/AttributedCharacterIterator.Attribute.html#READING) |

| **Constructor Summary** | |
| --- | --- |
| protected | [**TextAttribute**](http://docs.google.com/java/awt/font/TextAttribute.html#TextAttribute(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)            Constructs a TextAttribute with the specified name. |

| **Method Summary** | |
| --- | --- |
| protected  [Object](http://docs.google.com/java/lang/Object.html) | [**readResolve**](http://docs.google.com/java/awt/font/TextAttribute.html#readResolve())()            Resolves instances being deserialized to the predefined constants. |

| **Methods inherited from class java.text.**[**AttributedCharacterIterator.Attribute**](http://docs.google.com/java/text/AttributedCharacterIterator.Attribute.html) |
| --- |
| [equals](http://docs.google.com/java/text/AttributedCharacterIterator.Attribute.html#equals(java.lang.Object)), [getName](http://docs.google.com/java/text/AttributedCharacterIterator.Attribute.html#getName()), [hashCode](http://docs.google.com/java/text/AttributedCharacterIterator.Attribute.html#hashCode()), [toString](http://docs.google.com/java/text/AttributedCharacterIterator.Attribute.html#toString()) |

| **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)) |

| **Field Detail** |
| --- |

### FAMILY

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **FAMILY**

Attribute key for the font name. Values are instances of **String**. The default value is "Default", which causes the platform default font family to be used.

The Font class defines constants for the logical font names [DIALOG](http://docs.google.com/java/awt/Font.html#DIALOG), [DIALOG\_INPUT](http://docs.google.com/java/awt/Font.html#DIALOG_INPUT), [SANS\_SERIF](http://docs.google.com/java/awt/Font.html#SANS_SERIF), [SERIF](http://docs.google.com/java/awt/Font.html#SERIF), and [MONOSPACED](http://docs.google.com/java/awt/Font.html#MONOSPACED).

This defines the value passed as name to the Font constructor. Both logical and physical font names are allowed. If a font with the requested name is not found, the default font is used.

*Note:* This attribute is unfortunately misnamed, as it specifies the face name and not just the family. Thus values such as "Lucida Sans Bold" will select that face if it exists. Note, though, that if the requested face does not exist, the default will be used with *regular* weight. The "Bold" in the name is part of the face name, not a separate request that the font's weight be bold.

### WEIGHT

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **WEIGHT**

Attribute key for the weight of a font. Values are instances of **Number**. The default value is WEIGHT\_REGULAR.

Several constant values are provided, see [WEIGHT\_EXTRA\_LIGHT](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_EXTRA_LIGHT), [WEIGHT\_LIGHT](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_LIGHT), [WEIGHT\_DEMILIGHT](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_DEMILIGHT), [WEIGHT\_REGULAR](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_REGULAR), [WEIGHT\_SEMIBOLD](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_SEMIBOLD), [WEIGHT\_MEDIUM](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_MEDIUM), [WEIGHT\_DEMIBOLD](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_DEMIBOLD), [WEIGHT\_BOLD](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_BOLD), [WEIGHT\_HEAVY](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_HEAVY), [WEIGHT\_EXTRABOLD](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_EXTRABOLD), and [WEIGHT\_ULTRABOLD](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT_ULTRABOLD). The value WEIGHT\_BOLD corresponds to the style value Font.BOLD as passed to the Font constructor.

The value is roughly the ratio of the stem width to that of the regular weight.

The system can interpolate the provided value.

### WEIGHT\_EXTRA\_LIGHT

public static final [Float](http://docs.google.com/java/lang/Float.html) **WEIGHT\_EXTRA\_LIGHT**

The lightest predefined weight.

**See Also:**[WEIGHT](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT)

### WEIGHT\_LIGHT

public static final [Float](http://docs.google.com/java/lang/Float.html) **WEIGHT\_LIGHT**

The standard light weight.

**See Also:**[WEIGHT](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT)

### WEIGHT\_DEMILIGHT

public static final [Float](http://docs.google.com/java/lang/Float.html) **WEIGHT\_DEMILIGHT**

An intermediate weight between WEIGHT\_LIGHT and WEIGHT\_STANDARD.

**See Also:**[WEIGHT](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT)

### WEIGHT\_REGULAR

public static final [Float](http://docs.google.com/java/lang/Float.html) **WEIGHT\_REGULAR**

The standard weight. This is the default value for WEIGHT.

**See Also:**[WEIGHT](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT)

### WEIGHT\_SEMIBOLD

public static final [Float](http://docs.google.com/java/lang/Float.html) **WEIGHT\_SEMIBOLD**

A moderately heavier weight than WEIGHT\_REGULAR.

**See Also:**[WEIGHT](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT)

### WEIGHT\_MEDIUM

public static final [Float](http://docs.google.com/java/lang/Float.html) **WEIGHT\_MEDIUM**

An intermediate weight between WEIGHT\_REGULAR and WEIGHT\_BOLD.

**See Also:**[WEIGHT](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT)

### WEIGHT\_DEMIBOLD

public static final [Float](http://docs.google.com/java/lang/Float.html) **WEIGHT\_DEMIBOLD**

A moderately lighter weight than WEIGHT\_BOLD.

**See Also:**[WEIGHT](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT)

### WEIGHT\_BOLD

public static final [Float](http://docs.google.com/java/lang/Float.html) **WEIGHT\_BOLD**

The standard bold weight.

**See Also:**[WEIGHT](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT)

### WEIGHT\_HEAVY

public static final [Float](http://docs.google.com/java/lang/Float.html) **WEIGHT\_HEAVY**

A moderately heavier weight than WEIGHT\_BOLD.

**See Also:**[WEIGHT](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT)

### WEIGHT\_EXTRABOLD

public static final [Float](http://docs.google.com/java/lang/Float.html) **WEIGHT\_EXTRABOLD**

An extra heavy weight.

**See Also:**[WEIGHT](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT)

### WEIGHT\_ULTRABOLD

public static final [Float](http://docs.google.com/java/lang/Float.html) **WEIGHT\_ULTRABOLD**

The heaviest predefined weight.

**See Also:**[WEIGHT](http://docs.google.com/java/awt/font/TextAttribute.html#WEIGHT)

### WIDTH

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **WIDTH**

Attribute key for the width of a font. Values are instances of **Number**. The default value is WIDTH\_REGULAR.

Several constant values are provided, see [WIDTH\_CONDENSED](http://docs.google.com/java/awt/font/TextAttribute.html#WIDTH_CONDENSED), [WIDTH\_SEMI\_CONDENSED](http://docs.google.com/java/awt/font/TextAttribute.html#WIDTH_SEMI_CONDENSED), [WIDTH\_REGULAR](http://docs.google.com/java/awt/font/TextAttribute.html#WIDTH_REGULAR), [WIDTH\_SEMI\_EXTENDED](http://docs.google.com/java/awt/font/TextAttribute.html#WIDTH_SEMI_EXTENDED), [WIDTH\_EXTENDED](http://docs.google.com/java/awt/font/TextAttribute.html#WIDTH_EXTENDED).

The value is roughly the ratio of the advance width to that of the regular width.

The system can interpolate the provided value.

### WIDTH\_CONDENSED

public static final [Float](http://docs.google.com/java/lang/Float.html) **WIDTH\_CONDENSED**

The most condensed predefined width.

**See Also:**[WIDTH](http://docs.google.com/java/awt/font/TextAttribute.html#WIDTH)

### WIDTH\_SEMI\_CONDENSED

public static final [Float](http://docs.google.com/java/lang/Float.html) **WIDTH\_SEMI\_CONDENSED**

A moderately condensed width.

**See Also:**[WIDTH](http://docs.google.com/java/awt/font/TextAttribute.html#WIDTH)

### WIDTH\_REGULAR

public static final [Float](http://docs.google.com/java/lang/Float.html) **WIDTH\_REGULAR**

The standard width. This is the default value for WIDTH.

**See Also:**[WIDTH](http://docs.google.com/java/awt/font/TextAttribute.html#WIDTH)

### WIDTH\_SEMI\_EXTENDED

public static final [Float](http://docs.google.com/java/lang/Float.html) **WIDTH\_SEMI\_EXTENDED**

A moderately extended width.

**See Also:**[WIDTH](http://docs.google.com/java/awt/font/TextAttribute.html#WIDTH)

### WIDTH\_EXTENDED

public static final [Float](http://docs.google.com/java/lang/Float.html) **WIDTH\_EXTENDED**

The most extended predefined width.

**See Also:**[WIDTH](http://docs.google.com/java/awt/font/TextAttribute.html#WIDTH)

### POSTURE

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **POSTURE**

Attribute key for the posture of a font. Values are instances of **Number**. The default value is POSTURE\_REGULAR.

Two constant values are provided, [POSTURE\_REGULAR](http://docs.google.com/java/awt/font/TextAttribute.html#POSTURE_REGULAR) and [POSTURE\_OBLIQUE](http://docs.google.com/java/awt/font/TextAttribute.html#POSTURE_OBLIQUE). The value POSTURE\_OBLIQUE corresponds to the style value Font.ITALIC as passed to the Font constructor.

The value is roughly the slope of the stems of the font, expressed as the run over the rise. Positive values lean right.

The system can interpolate the provided value.

This will affect the font's italic angle as returned by Font.getItalicAngle.

**See Also:**[Font.getItalicAngle()](http://docs.google.com/java/awt/Font.html#getItalicAngle())

### POSTURE\_REGULAR

public static final [Float](http://docs.google.com/java/lang/Float.html) **POSTURE\_REGULAR**

The standard posture, upright. This is the default value for POSTURE.

**See Also:**[POSTURE](http://docs.google.com/java/awt/font/TextAttribute.html#POSTURE)

### POSTURE\_OBLIQUE

public static final [Float](http://docs.google.com/java/lang/Float.html) **POSTURE\_OBLIQUE**

The standard italic posture.

**See Also:**[POSTURE](http://docs.google.com/java/awt/font/TextAttribute.html#POSTURE)

### SIZE

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **SIZE**

Attribute key for the font size. Values are instances of **Number**. The default value is 12pt.

This corresponds to the size parameter to the Font constructor.

Very large or small sizes will impact rendering performance, and the rendering system might not render text at these sizes. Negative sizes are illegal and result in the default size.

Note that the appearance and metrics of a 12pt font with a 2x transform might be different than that of a 24 point font with no transform.

### TRANSFORM

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **TRANSFORM**

Attribute key for the transform of a font. Values are instances of **TransformAttribute**. The default value is TransformAttribute.IDENTITY.

The TransformAttribute class defines the constant [IDENTITY](http://docs.google.com/java/awt/font/TransformAttribute.html#IDENTITY).

This corresponds to the transform passed to Font.deriveFont(AffineTransform). Since that transform is mutable and TextAttribute values must not be, the TransformAttribute wrapper class is used.

The primary intent is to support scaling and skewing, though other effects are possible.

Some transforms will cause the baseline to be rotated and/or shifted. The text and the baseline are transformed together so that the text follows the new baseline. For example, with text on a horizontal baseline, the new baseline follows the direction of the unit x vector passed through the transform. Text metrics are measured against this new baseline. So, for example, with other things being equal, text rendered with a rotated TRANSFORM and an unrotated TRANSFORM will measure as having the same ascent, descent, and advance.

In styled text, the baselines for each such run are aligned one after the other to potentially create a non-linear baseline for the entire run of text. For more information, see [TextLayout.getLayoutPath()](http://docs.google.com/java/awt/font/TextLayout.html#getLayoutPath()).

**See Also:**[TransformAttribute](http://docs.google.com/java/awt/font/TransformAttribute.html), [AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html)

### SUPERSCRIPT

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **SUPERSCRIPT**

Attribute key for superscripting and subscripting. Values are instances of **Integer**. The default value is 0, which means that no superscript or subscript is used.

Two constant values are provided, see [SUPERSCRIPT\_SUPER](http://docs.google.com/java/awt/font/TextAttribute.html#SUPERSCRIPT_SUPER) and [SUPERSCRIPT\_SUB](http://docs.google.com/java/awt/font/TextAttribute.html#SUPERSCRIPT_SUB). These have the values 1 and -1 respectively. Values of greater magnitude define greater levels of superscript or subscripting, for example, 2 corresponds to super-superscript, 3 to super-super-superscript, and similarly for negative values and subscript, up to a level of 7 (or -7). Values beyond this range are reserved; behavior is platform-dependent.

SUPERSCRIPT can impact the ascent and descent of a font. The ascent and descent can never become negative, however.

### SUPERSCRIPT\_SUPER

public static final [Integer](http://docs.google.com/java/lang/Integer.html) **SUPERSCRIPT\_SUPER**

Standard superscript.

**See Also:**[SUPERSCRIPT](http://docs.google.com/java/awt/font/TextAttribute.html#SUPERSCRIPT)

### SUPERSCRIPT\_SUB

public static final [Integer](http://docs.google.com/java/lang/Integer.html) **SUPERSCRIPT\_SUB**

Standard subscript.

**See Also:**[SUPERSCRIPT](http://docs.google.com/java/awt/font/TextAttribute.html#SUPERSCRIPT)

### FONT

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **FONT**

Attribute key used to provide the font to use to render text. Values are instances of [Font](http://docs.google.com/java/awt/Font.html). The default value is null, indicating that normal resolution of a Font from attributes should be performed.

TextLayout and AttributedCharacterIterator work in terms of Maps of TextAttributes. Normally, all the attributes are examined and used to select and configure a Font instance. If a FONT attribute is present, though, its associated Font will be used. This provides a way for users to override the resolution of font attributes into a Font, or force use of a particular Font instance. This also allows users to specify subclasses of Font in cases where a Font can be subclassed.

FONT is used for special situations where clients already have a Font instance but still need to use Map-based APIs. Typically, there will be no other attributes in the Map except the FONT attribute. With Map-based APIs the common case is to specify all attributes individually, so FONT is not needed or desireable.

However, if both FONT and other attributes are present in the Map, the rendering system will merge the attributes defined in the Font with the additional attributes. This merging process classifies TextAttributes into two groups. One group, the 'primary' group, is considered fundamental to the selection and metric behavior of a font. These attributes are FAMILY, WEIGHT, WIDTH, POSTURE, SIZE, TRANSFORM, SUPERSCRIPT, and TRACKING. The other group, the 'secondary' group, consists of all other defined attributes, with the exception of FONT itself.

To generate the new Map, first the Font is obtained from the FONT attribute, and *all* of its attributes extracted into a new Map. Then only the *secondary* attributes from the original Map are added to those in the new Map. Thus the values of primary attributes come solely from the Font, and the values of secondary attributes originate with the Font but can be overridden by other values in the Map.

*Note:*Font's Map-based constructor and deriveFont methods do not process the FONT attribute, as these are used to create new Font objects. Instead, [Font.getFont(Map)](http://docs.google.com/java/awt/Font.html#getFont(java.util.Map)) should be used to handle the FONT attribute.

**See Also:**[Font](http://docs.google.com/java/awt/Font.html)

### CHAR\_REPLACEMENT

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **CHAR\_REPLACEMENT**

Attribute key for a user-defined glyph to display in lieu of the font's standard glyph for a character. Values are intances of GraphicAttribute. The default value is null, indicating that the standard glyphs provided by the font should be used.

This attribute is used to reserve space for a graphic or other component embedded in a line of text. It is required for correct positioning of 'inline' components within a line when bidirectional reordering (see [Bidi](http://docs.google.com/java/text/Bidi.html)) is performed. Each character (Unicode code point) will be rendered using the provided GraphicAttribute. Typically, the characters to which this attribute is applied should be \uFFFC.

The GraphicAttribute determines the logical and visual bounds of the text; the actual Font values are ignored.

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

### FOREGROUND

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **FOREGROUND**

Attribute key for the paint used to render the text. Values are instances of **Paint**. The default value is null, indicating that the Paint set on the Graphics2D at the time of rendering is used.

Glyphs will be rendered using this Paint regardless of the Paint value set on the Graphics (but see [SWAP\_COLORS](http://docs.google.com/java/awt/font/TextAttribute.html#SWAP_COLORS)).

**See Also:**[Paint](http://docs.google.com/java/awt/Paint.html), [SWAP\_COLORS](http://docs.google.com/java/awt/font/TextAttribute.html#SWAP_COLORS)

### BACKGROUND

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **BACKGROUND**

Attribute key for the paint used to render the background of the text. Values are instances of **Paint**. The default value is null, indicating that the background should not be rendered.

The logical bounds of the text will be filled using this Paint, and then the text will be rendered on top of it (but see [SWAP\_COLORS](http://docs.google.com/java/awt/font/TextAttribute.html#SWAP_COLORS)).

The visual bounds of the text is extended to include the logical bounds, if necessary. The outline is not affected.

**See Also:**[Paint](http://docs.google.com/java/awt/Paint.html), [SWAP\_COLORS](http://docs.google.com/java/awt/font/TextAttribute.html#SWAP_COLORS)

### UNDERLINE

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **UNDERLINE**

Attribute key for underline. Values are instances of **Integer**. The default value is -1, which means no underline.

The constant value [UNDERLINE\_ON](http://docs.google.com/java/awt/font/TextAttribute.html#UNDERLINE_ON) is provided.

The underline affects both the visual bounds and the outline of the text.

### UNDERLINE\_ON

public static final [Integer](http://docs.google.com/java/lang/Integer.html) **UNDERLINE\_ON**

Standard underline.

**See Also:**[UNDERLINE](http://docs.google.com/java/awt/font/TextAttribute.html#UNDERLINE)

### STRIKETHROUGH

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **STRIKETHROUGH**

Attribute key for strikethrough. Values are instances of **Boolean**. The default value is false, which means no strikethrough.

The constant value [STRIKETHROUGH\_ON](http://docs.google.com/java/awt/font/TextAttribute.html#STRIKETHROUGH_ON) is provided.

The strikethrough affects both the visual bounds and the outline of the text.

### STRIKETHROUGH\_ON

public static final [Boolean](http://docs.google.com/java/lang/Boolean.html) **STRIKETHROUGH\_ON**

A single strikethrough.

**See Also:**[STRIKETHROUGH](http://docs.google.com/java/awt/font/TextAttribute.html#STRIKETHROUGH)

### RUN\_DIRECTION

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **RUN\_DIRECTION**

Attribute key for the run direction of the line. Values are instances of **Boolean**. The default value is null, which indicates that the standard Bidi algorithm for determining run direction should be used with the value [Bidi.DIRECTION\_DEFAULT\_LEFT\_TO\_RIGHT](http://docs.google.com/java/text/Bidi.html#DIRECTION_DEFAULT_LEFT_TO_RIGHT).

The constants [RUN\_DIRECTION\_RTL](http://docs.google.com/java/awt/font/TextAttribute.html#RUN_DIRECTION_RTL) and [RUN\_DIRECTION\_LTR](http://docs.google.com/java/awt/font/TextAttribute.html#RUN_DIRECTION_LTR) are provided.

This determines the value passed to the [Bidi](http://docs.google.com/java/text/Bidi.html) constructor to select the primary direction of the text in the paragraph.

*Note:* This attribute should have the same value for all the text in a paragraph, otherwise the behavior is undetermined.

**See Also:**[Bidi](http://docs.google.com/java/text/Bidi.html)

### RUN\_DIRECTION\_LTR

public static final [Boolean](http://docs.google.com/java/lang/Boolean.html) **RUN\_DIRECTION\_LTR**

Left-to-right run direction.

**See Also:**[RUN\_DIRECTION](http://docs.google.com/java/awt/font/TextAttribute.html#RUN_DIRECTION)

### RUN\_DIRECTION\_RTL

public static final [Boolean](http://docs.google.com/java/lang/Boolean.html) **RUN\_DIRECTION\_RTL**

Right-to-left run direction.

**See Also:**[RUN\_DIRECTION](http://docs.google.com/java/awt/font/TextAttribute.html#RUN_DIRECTION)

### BIDI\_EMBEDDING

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **BIDI\_EMBEDDING**

Attribute key for the embedding level of the text. Values are instances of **Integer**. The default value is null, indicating that the the Bidirectional algorithm should run without explicit embeddings.

Positive values 1 through 61 are *embedding* levels, negative values -1 through -61 are *override* levels. The value 0 means that the base line direction is used. These levels are passed in the embedding levels array to the [Bidi](http://docs.google.com/java/text/Bidi.html) constructor.

*Note:* When this attribute is present anywhere in a paragraph, then any Unicode bidi control characters (RLO, LRO, RLE, LRE, and PDF) in the paragraph are disregarded, and runs of text where this attribute is not present are treated as though it were present and had the value 0.

**See Also:**[Bidi](http://docs.google.com/java/text/Bidi.html)

### JUSTIFICATION

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **JUSTIFICATION**

Attribute key for the justification of a paragraph. Values are instances of **Number**. The default value is 1, indicating that justification should use the full width provided. Values are pinned to the range [0..1].

The constants [JUSTIFICATION\_FULL](http://docs.google.com/java/awt/font/TextAttribute.html#JUSTIFICATION_FULL) and [JUSTIFICATION\_NONE](http://docs.google.com/java/awt/font/TextAttribute.html#JUSTIFICATION_NONE) are provided.

Specifies the fraction of the extra space to use when justification is requested on a TextLayout. For example, if the line is 50 points wide and it is requested to justify to 70 points, a value of 0.75 will pad to use three-quarters of the remaining space, or 15 points, so that the resulting line will be 65 points in length.

*Note:* This should have the same value for all the text in a paragraph, otherwise the behavior is undetermined.

**See Also:**[TextLayout.getJustifiedLayout(float)](http://docs.google.com/java/awt/font/TextLayout.html#getJustifiedLayout(float))

### JUSTIFICATION\_FULL

public static final [Float](http://docs.google.com/java/lang/Float.html) **JUSTIFICATION\_FULL**

Justify the line to the full requested width. This is the default value for JUSTIFICATION.

**See Also:**[JUSTIFICATION](http://docs.google.com/java/awt/font/TextAttribute.html#JUSTIFICATION)

### JUSTIFICATION\_NONE

public static final [Float](http://docs.google.com/java/lang/Float.html) **JUSTIFICATION\_NONE**

Do not allow the line to be justified.

**See Also:**[JUSTIFICATION](http://docs.google.com/java/awt/font/TextAttribute.html#JUSTIFICATION)

### INPUT\_METHOD\_HIGHLIGHT

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **INPUT\_METHOD\_HIGHLIGHT**

Attribute key for input method highlight styles.

Values are instances of [InputMethodHighlight](http://docs.google.com/java/awt/im/InputMethodHighlight.html) or [Annotation](http://docs.google.com/java/text/Annotation.html). The default value is null, which means that input method styles should not be applied before rendering.

If adjacent runs of text with the same InputMethodHighlight need to be rendered separately, the InputMethodHighlights should be wrapped in Annotation instances.

Input method highlights are used while text is being composed by an input method. Text editing components should retain them even if they generally only deal with unstyled text, and make them available to the drawing routines.

**See Also:**[Font](http://docs.google.com/java/awt/Font.html), [InputMethodHighlight](http://docs.google.com/java/awt/im/InputMethodHighlight.html), [Annotation](http://docs.google.com/java/text/Annotation.html)

### INPUT\_METHOD\_UNDERLINE

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **INPUT\_METHOD\_UNDERLINE**

Attribute key for input method underlines. Values are instances of **Integer**. The default value is -1, which means no underline.

Several constant values are provided, see [UNDERLINE\_LOW\_ONE\_PIXEL](http://docs.google.com/java/awt/font/TextAttribute.html#UNDERLINE_LOW_ONE_PIXEL), [UNDERLINE\_LOW\_TWO\_PIXEL](http://docs.google.com/java/awt/font/TextAttribute.html#UNDERLINE_LOW_TWO_PIXEL), [UNDERLINE\_LOW\_DOTTED](http://docs.google.com/java/awt/font/TextAttribute.html#UNDERLINE_LOW_DOTTED), [UNDERLINE\_LOW\_GRAY](http://docs.google.com/java/awt/font/TextAttribute.html#UNDERLINE_LOW_GRAY), and [UNDERLINE\_LOW\_DASHED](http://docs.google.com/java/awt/font/TextAttribute.html#UNDERLINE_LOW_DASHED).

This may be used in conjunction with [UNDERLINE](http://docs.google.com/java/awt/font/TextAttribute.html#UNDERLINE) if desired. The primary purpose is for use by input methods. Other use of these underlines for simple ornamentation might confuse users.

The input method underline affects both the visual bounds and the outline of the text.

**Since:** 1.3

### UNDERLINE\_LOW\_ONE\_PIXEL

public static final [Integer](http://docs.google.com/java/lang/Integer.html) **UNDERLINE\_LOW\_ONE\_PIXEL**

Single pixel solid low underline.

**Since:** 1.3 **See Also:**[INPUT\_METHOD\_UNDERLINE](http://docs.google.com/java/awt/font/TextAttribute.html#INPUT_METHOD_UNDERLINE)

### UNDERLINE\_LOW\_TWO\_PIXEL

public static final [Integer](http://docs.google.com/java/lang/Integer.html) **UNDERLINE\_LOW\_TWO\_PIXEL**

Double pixel solid low underline.

**Since:** 1.3 **See Also:**[INPUT\_METHOD\_UNDERLINE](http://docs.google.com/java/awt/font/TextAttribute.html#INPUT_METHOD_UNDERLINE)

### UNDERLINE\_LOW\_DOTTED

public static final [Integer](http://docs.google.com/java/lang/Integer.html) **UNDERLINE\_LOW\_DOTTED**

Single pixel dotted low underline.

**Since:** 1.3 **See Also:**[INPUT\_METHOD\_UNDERLINE](http://docs.google.com/java/awt/font/TextAttribute.html#INPUT_METHOD_UNDERLINE)

### UNDERLINE\_LOW\_GRAY

public static final [Integer](http://docs.google.com/java/lang/Integer.html) **UNDERLINE\_LOW\_GRAY**

Double pixel gray low underline.

**Since:** 1.3 **See Also:**[INPUT\_METHOD\_UNDERLINE](http://docs.google.com/java/awt/font/TextAttribute.html#INPUT_METHOD_UNDERLINE)

### UNDERLINE\_LOW\_DASHED

public static final [Integer](http://docs.google.com/java/lang/Integer.html) **UNDERLINE\_LOW\_DASHED**

Single pixel dashed low underline.

**Since:** 1.3 **See Also:**[INPUT\_METHOD\_UNDERLINE](http://docs.google.com/java/awt/font/TextAttribute.html#INPUT_METHOD_UNDERLINE)

### SWAP\_COLORS

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **SWAP\_COLORS**

Attribute key for swapping foreground and background Paints. Values are instances of **Boolean**. The default value is false, which means do not swap colors.

The constant value [SWAP\_COLORS\_ON](http://docs.google.com/java/awt/font/TextAttribute.html#SWAP_COLORS_ON) is defined.

If the [FOREGROUND](http://docs.google.com/java/awt/font/TextAttribute.html#FOREGROUND) attribute is set, its Paint will be used as the background, otherwise the Paint currently on the Graphics will be used. If the [BACKGROUND](http://docs.google.com/java/awt/font/TextAttribute.html#BACKGROUND) attribute is set, its Paint will be used as the foreground, otherwise the system will find a contrasting color to the (resolved) background so that the text will be visible.

**See Also:**[FOREGROUND](http://docs.google.com/java/awt/font/TextAttribute.html#FOREGROUND), [BACKGROUND](http://docs.google.com/java/awt/font/TextAttribute.html#BACKGROUND)

### SWAP\_COLORS\_ON

public static final [Boolean](http://docs.google.com/java/lang/Boolean.html) **SWAP\_COLORS\_ON**

Swap foreground and background.

**Since:** 1.3 **See Also:**[SWAP\_COLORS](http://docs.google.com/java/awt/font/TextAttribute.html#SWAP_COLORS)

### NUMERIC\_SHAPING

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **NUMERIC\_SHAPING**

Attribute key for converting ASCII decimal digits to other decimal ranges. Values are instances of [NumericShaper](http://docs.google.com/java/awt/font/NumericShaper.html). The default is null, which means do not perform numeric shaping.

When a numeric shaper is defined, the text is first processed by the shaper before any other analysis of the text is performed.

*Note:* This should have the same value for all the text in the paragraph, otherwise the behavior is undetermined.

**Since:** 1.4 **See Also:**[NumericShaper](http://docs.google.com/java/awt/font/NumericShaper.html)

### KERNING

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **KERNING**

Attribute key to request kerning. Values are instances of **Integer**. The default value is 0, which does not request kerning.

The constant value [KERNING\_ON](http://docs.google.com/java/awt/font/TextAttribute.html#KERNING_ON) is provided.

The default advances of single characters are not appropriate for some character sequences, for example "To" or "AWAY". Without kerning the adjacent characters appear to be separated by too much space. Kerning causes selected sequences of characters to be spaced differently for a more pleasing visual appearance.

**Since:** 1.6

### KERNING\_ON

public static final [Integer](http://docs.google.com/java/lang/Integer.html) **KERNING\_ON**

Request standard kerning.

**Since:** 1.6 **See Also:**[KERNING](http://docs.google.com/java/awt/font/TextAttribute.html#KERNING)

### LIGATURES

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **LIGATURES**

Attribute key for enabling optional ligatures. Values are instances of **Integer**. The default value is 0, which means do not use optional ligatures.

The constant value [LIGATURES\_ON](http://docs.google.com/java/awt/font/TextAttribute.html#LIGATURES_ON) is defined.

Ligatures required by the writing system are always enabled.

**Since:** 1.6

### LIGATURES\_ON

public static final [Integer](http://docs.google.com/java/lang/Integer.html) **LIGATURES\_ON**

Request standard optional ligatures.

**Since:** 1.6 **See Also:**[LIGATURES](http://docs.google.com/java/awt/font/TextAttribute.html#LIGATURES)

### TRACKING

public static final [TextAttribute](http://docs.google.com/java/awt/font/TextAttribute.html) **TRACKING**

Attribute key to control tracking. Values are instances of **Number**. The default value is 0, which means no additional tracking.

The constant values [TRACKING\_TIGHT](http://docs.google.com/java/awt/font/TextAttribute.html#TRACKING_TIGHT) and [TRACKING\_LOOSE](http://docs.google.com/java/awt/font/TextAttribute.html#TRACKING_LOOSE) are provided.

The tracking value is multiplied by the font point size and passed through the font transform to determine an additional amount to add to the advance of each glyph cluster. Positive tracking values will inhibit formation of optional ligatures. Tracking values are typically between -0.1 and 0.3; values outside this range are generally not desireable.

**Since:** 1.6

### TRACKING\_TIGHT

public static final [Float](http://docs.google.com/java/lang/Float.html) **TRACKING\_TIGHT**

Perform tight tracking.

**Since:** 1.6 **See Also:**[TRACKING](http://docs.google.com/java/awt/font/TextAttribute.html#TRACKING)

### TRACKING\_LOOSE

public static final [Float](http://docs.google.com/java/lang/Float.html) **TRACKING\_LOOSE**

Perform loose tracking.

**Since:** 1.6 **See Also:**[TRACKING](http://docs.google.com/java/awt/font/TextAttribute.html#TRACKING)

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

### TextAttribute

protected **TextAttribute**([String](http://docs.google.com/java/lang/String.html) name)

Constructs a TextAttribute with the specified name.

**Parameters:**name - the attribute name to assign to this TextAttribute

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

### readResolve

protected [Object](http://docs.google.com/java/lang/Object.html) **readResolve**()  
 throws [InvalidObjectException](http://docs.google.com/java/io/InvalidObjectException.html)

Resolves instances being deserialized to the predefined constants.

**Overrides:**[readResolve](http://docs.google.com/java/text/AttributedCharacterIterator.Attribute.html#readResolve()) in class [AttributedCharacterIterator.Attribute](http://docs.google.com/java/text/AttributedCharacterIterator.Attribute.html) **Throws:** [InvalidObjectException](http://docs.google.com/java/io/InvalidObjectException.html)

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/TextAttribute.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/ShapeGraphicAttribute.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/font/TextHitInfo.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/font/TextAttribute.html)    [**NO FRAMES**](http://docs.google.com/TextAttribute.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#tyjcwt) | [METHOD](#3dy6vkm) | DETAIL: [FIELD](#2s8eyo1) | [CONSTR](#34g0dwd) | [METHOD](#43ky6rz) |

[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).