| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/GlyphMetrics.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/GlyphJustificationInfo.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/font/GlyphVector.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/font/GlyphMetrics.html)    [**NO FRAMES**](http://docs.google.com/GlyphMetrics.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#lnxbz9) | [METHOD](#44sinio) |

## **java.awt.font**

Class GlyphMetrics

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

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

The GlyphMetrics class represents infomation for a single glyph. A glyph is the visual representation of one or more characters. Many different glyphs can be used to represent a single character or combination of characters. GlyphMetrics instances are produced by [Font](http://docs.google.com/java/awt/Font.html) and are applicable to a specific glyph in a particular Font.

Glyphs are either STANDARD, LIGATURE, COMBINING, or COMPONENT.

* STANDARD glyphs are commonly used to represent single characters.
* LIGATURE glyphs are used to represent sequences of characters.
* COMPONENT glyphs in a [GlyphVector](http://docs.google.com/java/awt/font/GlyphVector.html) do not correspond to a particular character in a text model. Instead, COMPONENT glyphs are added for typographical reasons, such as Arabic justification.
* COMBINING glyphs embellish STANDARD or LIGATURE glyphs, such as accent marks. Carets do not appear before COMBINING glyphs.

Other metrics available through GlyphMetrics are the components of the advance, the visual bounds, and the left and right side bearings.

Glyphs for a rotated font, or obtained from a GlyphVector which has applied a rotation to the glyph, can have advances that contain both X and Y components. Usually the advance only has one component.

The advance of a glyph is the distance from the glyph's origin to the origin of the next glyph along the baseline, which is either vertical or horizontal. Note that, in a GlyphVector, the distance from a glyph to its following glyph might not be the glyph's advance, because of kerning or other positioning adjustments.

The bounds is the smallest rectangle that completely contains the outline of the glyph. The bounds rectangle is relative to the glyph's origin. The left-side bearing is the distance from the glyph origin to the left of its bounds rectangle. If the left-side bearing is negative, part of the glyph is drawn to the left of its origin. The right-side bearing is the distance from the right side of the bounds rectangle to the next glyph origin (the origin plus the advance). If negative, part of the glyph is drawn to the right of the next glyph's origin. Note that the bounds does not necessarily enclose all the pixels affected when rendering the glyph, because of rasterization and pixel adjustment effects.

Although instances of GlyphMetrics can be directly constructed, they are almost always obtained from a GlyphVector. Once constructed, GlyphMetrics objects are immutable.

**Example**:

Querying a Font for glyph information

Font font = ...;  
 int glyphIndex = ...;  
 GlyphMetrics metrics = GlyphVector.getGlyphMetrics(glyphIndex);  
 int isStandard = metrics.isStandard();  
 float glyphAdvance = metrics.getAdvance();

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

| **Field Summary** | |
| --- | --- |
| static byte | [**COMBINING**](http://docs.google.com/java/awt/font/GlyphMetrics.html#COMBINING)            Indicates a glyph that represents a combining character, such as an umlaut. |
| static byte | [**COMPONENT**](http://docs.google.com/java/awt/font/GlyphMetrics.html#COMPONENT)            Indicates a glyph with no corresponding character in the backing store. |
| static byte | [**LIGATURE**](http://docs.google.com/java/awt/font/GlyphMetrics.html#LIGATURE)            Indicates a glyph that represents multiple characters as a ligature, for example 'fi' or 'ffi'. |
| static byte | [**STANDARD**](http://docs.google.com/java/awt/font/GlyphMetrics.html#STANDARD)            Indicates a glyph that represents a single standard character. |
| static byte | [**WHITESPACE**](http://docs.google.com/java/awt/font/GlyphMetrics.html#WHITESPACE)            Indicates a glyph with no visual representation. |

| **Constructor Summary** | |
| --- | --- |
| [**GlyphMetrics**](http://docs.google.com/java/awt/font/GlyphMetrics.html#GlyphMetrics(boolean,%20float,%20float,%20java.awt.geom.Rectangle2D,%20byte))(boolean horizontal, float advanceX, float advanceY, [Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) bounds, byte glyphType)            Constructs a GlyphMetrics object. |
| [**GlyphMetrics**](http://docs.google.com/java/awt/font/GlyphMetrics.html#GlyphMetrics(float,%20java.awt.geom.Rectangle2D,%20byte))(float advance, [Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) bounds, byte glyphType)            Constructs a GlyphMetrics object. |

| **Method Summary** | |
| --- | --- |
| float | [**getAdvance**](http://docs.google.com/java/awt/font/GlyphMetrics.html#getAdvance())()            Returns the advance of the glyph along the baseline (either horizontal or vertical). |
| float | [**getAdvanceX**](http://docs.google.com/java/awt/font/GlyphMetrics.html#getAdvanceX())()            Returns the x-component of the advance of the glyph. |
| float | [**getAdvanceY**](http://docs.google.com/java/awt/font/GlyphMetrics.html#getAdvanceY())()            Returns the y-component of the advance of the glyph. |
| [Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) | [**getBounds2D**](http://docs.google.com/java/awt/font/GlyphMetrics.html#getBounds2D())()            Returns the bounds of the glyph. |
| float | [**getLSB**](http://docs.google.com/java/awt/font/GlyphMetrics.html#getLSB())()            Returns the left (top) side bearing of the glyph. |
| float | [**getRSB**](http://docs.google.com/java/awt/font/GlyphMetrics.html#getRSB())()            Returns the right (bottom) side bearing of the glyph. |
| int | [**getType**](http://docs.google.com/java/awt/font/GlyphMetrics.html#getType())()            Returns the raw glyph type code. |
| boolean | [**isCombining**](http://docs.google.com/java/awt/font/GlyphMetrics.html#isCombining())()            Returns true if this is a combining glyph. |
| boolean | [**isComponent**](http://docs.google.com/java/awt/font/GlyphMetrics.html#isComponent())()            Returns true if this is a component glyph. |
| boolean | [**isLigature**](http://docs.google.com/java/awt/font/GlyphMetrics.html#isLigature())()            Returns true if this is a ligature glyph. |
| boolean | [**isStandard**](http://docs.google.com/java/awt/font/GlyphMetrics.html#isStandard())()            Returns true if this is a standard glyph. |
| boolean | [**isWhitespace**](http://docs.google.com/java/awt/font/GlyphMetrics.html#isWhitespace())()            Returns true if this is a whitespace glyph. |

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

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

### STANDARD

public static final byte **STANDARD**

Indicates a glyph that represents a single standard character.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.awt.font.GlyphMetrics.STANDARD)

### LIGATURE

public static final byte **LIGATURE**

Indicates a glyph that represents multiple characters as a ligature, for example 'fi' or 'ffi'. It is followed by filler glyphs for the remaining characters. Filler and combining glyphs can be intermixed to control positioning of accent marks on the logically preceeding ligature.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.awt.font.GlyphMetrics.LIGATURE)

### COMBINING

public static final byte **COMBINING**

Indicates a glyph that represents a combining character, such as an umlaut. There is no caret position between this glyph and the preceeding glyph.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.awt.font.GlyphMetrics.COMBINING)

### COMPONENT

public static final byte **COMPONENT**

Indicates a glyph with no corresponding character in the backing store. The glyph is associated with the character represented by the logicaly preceeding non-component glyph. This is used for kashida justification or other visual modifications to existing glyphs. There is no caret position between this glyph and the preceeding glyph.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.awt.font.GlyphMetrics.COMPONENT)

### WHITESPACE

public static final byte **WHITESPACE**

Indicates a glyph with no visual representation. It can be added to the other code values to indicate an invisible glyph.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.awt.font.GlyphMetrics.WHITESPACE)

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

### GlyphMetrics

public **GlyphMetrics**(float advance,  
 [Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) bounds,  
 byte glyphType)

Constructs a GlyphMetrics object.

**Parameters:**advance - the advance width of the glyphbounds - the black box bounds of the glyphglyphType - the type of the glyph

### GlyphMetrics

public **GlyphMetrics**(boolean horizontal,  
 float advanceX,  
 float advanceY,  
 [Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) bounds,  
 byte glyphType)

Constructs a GlyphMetrics object.

**Parameters:**horizontal - if true, metrics are for a horizontal baseline, otherwise they are for a vertical baselineadvanceX - the X-component of the glyph's advanceadvanceY - the Y-component of the glyph's advancebounds - the visual bounds of the glyphglyphType - the type of the glyph**Since:** 1.4

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

### getAdvance

public float **getAdvance**()

Returns the advance of the glyph along the baseline (either horizontal or vertical).

**Returns:**the advance of the glyph

### getAdvanceX

public float **getAdvanceX**()

Returns the x-component of the advance of the glyph.

**Returns:**the x-component of the advance of the glyph**Since:** 1.4

### getAdvanceY

public float **getAdvanceY**()

Returns the y-component of the advance of the glyph.

**Returns:**the y-component of the advance of the glyph**Since:** 1.4

### getBounds2D

public [Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) **getBounds2D**()

Returns the bounds of the glyph. This is the bounding box of the glyph outline. Because of rasterization and pixel alignment effects, it does not necessarily enclose the pixels that are affected when rendering the glyph.

**Returns:**a [Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) that is the bounds of the glyph.

### getLSB

public float **getLSB**()

Returns the left (top) side bearing of the glyph.

This is the distance from 0, 0 to the left (top) of the glyph bounds. If the bounds of the glyph is to the left of (above) the origin, the LSB is negative.

**Returns:**the left side bearing of the glyph.

### getRSB

public float **getRSB**()

Returns the right (bottom) side bearing of the glyph.

This is the distance from the right (bottom) of the glyph bounds to the advance. If the bounds of the glyph is to the right of (below) the advance, the RSB is negative.

**Returns:**the right side bearing of the glyph.

### getType

public int **getType**()

Returns the raw glyph type code.

**Returns:**the raw glyph type code.

### isStandard

public boolean **isStandard**()

Returns true if this is a standard glyph.

**Returns:**true if this is a standard glyph; false otherwise.

### isLigature

public boolean **isLigature**()

Returns true if this is a ligature glyph.

**Returns:**true if this is a ligature glyph; false otherwise.

### isCombining

public boolean **isCombining**()

Returns true if this is a combining glyph.

**Returns:**true if this is a combining glyph; false otherwise.

### isComponent

public boolean **isComponent**()

Returns true if this is a component glyph.

**Returns:**true if this is a component glyph; false otherwise.

### isWhitespace

public boolean **isWhitespace**()

Returns true if this is a whitespace glyph.

**Returns:**true if this is a whitespace glyph; false otherwise.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/GlyphMetrics.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/GlyphJustificationInfo.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/font/GlyphVector.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/font/GlyphMetrics.html)    [**NO FRAMES**](http://docs.google.com/GlyphMetrics.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#lnxbz9) | [METHOD](#44sinio) |

[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).