| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Bidi.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/text/AttributedString.html)   [**NEXT CLASS**](http://docs.google.com/java/text/BreakIterator.html) | [**FRAMES**](http://docs.google.com/index.html?java/text/Bidi.html)    [**NO FRAMES**](http://docs.google.com/Bidi.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#26in1rg) | [METHOD](#44sinio) |

## **java.text**

Class Bidi

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

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

This class implements the Unicode Bidirectional Algorithm.

A Bidi object provides information on the bidirectional reordering of the text used to create it. This is required, for example, to properly display Arabic or Hebrew text. These languages are inherently mixed directional, as they order numbers from left-to-right while ordering most other text from right-to-left.

Once created, a Bidi object can be queried to see if the text it represents is all left-to-right or all right-to-left. Such objects are very lightweight and this text is relatively easy to process.

If there are multiple runs of text, information about the runs can be accessed by indexing to get the start, limit, and level of a run. The level represents both the direction and the 'nesting level' of a directional run. Odd levels are right-to-left, while even levels are left-to-right. So for example level 0 represents left-to-right text, while level 1 represents right-to-left text, and level 2 represents left-to-right text embedded in a right-to-left run.

**Since:** 1.4

| **Field Summary** | |
| --- | --- |
| static int | [**DIRECTION\_DEFAULT\_LEFT\_TO\_RIGHT**](http://docs.google.com/java/text/Bidi.html#DIRECTION_DEFAULT_LEFT_TO_RIGHT)            Constant indicating that the base direction depends on the first strong directional character in the text according to the Unicode Bidirectional Algorithm. |
| static int | [**DIRECTION\_DEFAULT\_RIGHT\_TO\_LEFT**](http://docs.google.com/java/text/Bidi.html#DIRECTION_DEFAULT_RIGHT_TO_LEFT)            Constant indicating that the base direction depends on the first strong directional character in the text according to the Unicode Bidirectional Algorithm. |
| static int | [**DIRECTION\_LEFT\_TO\_RIGHT**](http://docs.google.com/java/text/Bidi.html#DIRECTION_LEFT_TO_RIGHT)            Constant indicating base direction is left-to-right. |
| static int | [**DIRECTION\_RIGHT\_TO\_LEFT**](http://docs.google.com/java/text/Bidi.html#DIRECTION_RIGHT_TO_LEFT)            Constant indicating base direction is right-to-left. |

| **Constructor Summary** | |
| --- | --- |
| [**Bidi**](http://docs.google.com/java/text/Bidi.html#Bidi(java.text.AttributedCharacterIterator))([AttributedCharacterIterator](http://docs.google.com/java/text/AttributedCharacterIterator.html) paragraph)            Create Bidi from the given paragraph of text. |
| [**Bidi**](http://docs.google.com/java/text/Bidi.html#Bidi(char%5B%5D,%20int,%20byte%5B%5D,%20int,%20int,%20int))(char[] text, int textStart, byte[] embeddings, int embStart, int paragraphLength, int flags)            Create Bidi from the given text, embedding, and direction information. |
| [**Bidi**](http://docs.google.com/java/text/Bidi.html#Bidi(java.lang.String,%20int))([String](http://docs.google.com/java/lang/String.html) paragraph, int flags)            Create Bidi from the given paragraph of text and base direction. |

| **Method Summary** | |
| --- | --- |
| boolean | [**baseIsLeftToRight**](http://docs.google.com/java/text/Bidi.html#baseIsLeftToRight())()            Return true if the base direction is left-to-right. |
| [Bidi](http://docs.google.com/java/text/Bidi.html) | [**createLineBidi**](http://docs.google.com/java/text/Bidi.html#createLineBidi(int,%20int))(int lineStart, int lineLimit)            Create a Bidi object representing the bidi information on a line of text within the paragraph represented by the current Bidi. |
| int | [**getBaseLevel**](http://docs.google.com/java/text/Bidi.html#getBaseLevel())()            Return the base level (0 if left-to-right, 1 if right-to-left). |
| int | [**getLength**](http://docs.google.com/java/text/Bidi.html#getLength())()            Return the length of text in the line. |
| int | [**getLevelAt**](http://docs.google.com/java/text/Bidi.html#getLevelAt(int))(int offset)            Return the resolved level of the character at offset. |
| int | [**getRunCount**](http://docs.google.com/java/text/Bidi.html#getRunCount())()            Return the number of level runs. |
| int | [**getRunLevel**](http://docs.google.com/java/text/Bidi.html#getRunLevel(int))(int run)            Return the level of the nth logical run in this line. |
| int | [**getRunLimit**](http://docs.google.com/java/text/Bidi.html#getRunLimit(int))(int run)            Return the index of the character past the end of the nth logical run in this line, as an offset from the start of the line. |
| int | [**getRunStart**](http://docs.google.com/java/text/Bidi.html#getRunStart(int))(int run)            Return the index of the character at the start of the nth logical run in this line, as an offset from the start of the line. |
| boolean | [**isLeftToRight**](http://docs.google.com/java/text/Bidi.html#isLeftToRight())()            Return true if the line is all left-to-right text and the base direction is left-to-right. |
| boolean | [**isMixed**](http://docs.google.com/java/text/Bidi.html#isMixed())()            Return true if the line is not left-to-right or right-to-left. |
| boolean | [**isRightToLeft**](http://docs.google.com/java/text/Bidi.html#isRightToLeft())()            Return true if the line is all right-to-left text, and the base direction is right-to-left. |
| static void | [**reorderVisually**](http://docs.google.com/java/text/Bidi.html#reorderVisually(byte%5B%5D,%20int,%20java.lang.Object%5B%5D,%20int,%20int))(byte[] levels, int levelStart, [Object](http://docs.google.com/java/lang/Object.html)[] objects, int objectStart, int count)            Reorder the objects in the array into visual order based on their levels. |
| static boolean | [**requiresBidi**](http://docs.google.com/java/text/Bidi.html#requiresBidi(char%5B%5D,%20int,%20int))(char[] text, int start, int limit)            Return true if the specified text requires bidi analysis. |
| [String](http://docs.google.com/java/lang/String.html) | [**toString**](http://docs.google.com/java/text/Bidi.html#toString())()            Display the bidi internal state, used in debugging. |

| **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)) |

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

### DIRECTION\_LEFT\_TO\_RIGHT

public static final int **DIRECTION\_LEFT\_TO\_RIGHT**

Constant indicating base direction is left-to-right.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.text.Bidi.DIRECTION_LEFT_TO_RIGHT)

### DIRECTION\_RIGHT\_TO\_LEFT

public static final int **DIRECTION\_RIGHT\_TO\_LEFT**

Constant indicating base direction is right-to-left.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.text.Bidi.DIRECTION_RIGHT_TO_LEFT)

### DIRECTION\_DEFAULT\_LEFT\_TO\_RIGHT

public static final int **DIRECTION\_DEFAULT\_LEFT\_TO\_RIGHT**

Constant indicating that the base direction depends on the first strong directional character in the text according to the Unicode Bidirectional Algorithm. If no strong directional character is present, the base direction is left-to-right.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.text.Bidi.DIRECTION_DEFAULT_LEFT_TO_RIGHT)

### DIRECTION\_DEFAULT\_RIGHT\_TO\_LEFT

public static final int **DIRECTION\_DEFAULT\_RIGHT\_TO\_LEFT**

Constant indicating that the base direction depends on the first strong directional character in the text according to the Unicode Bidirectional Algorithm. If no strong directional character is present, the base direction is right-to-left.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.text.Bidi.DIRECTION_DEFAULT_RIGHT_TO_LEFT)

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

### Bidi

public **Bidi**([String](http://docs.google.com/java/lang/String.html) paragraph,  
 int flags)

Create Bidi from the given paragraph of text and base direction.

**Parameters:**paragraph - a paragraph of textflags - a collection of flags that control the algorithm. The algorithm understands the flags DIRECTION\_LEFT\_TO\_RIGHT, DIRECTION\_RIGHT\_TO\_LEFT, DIRECTION\_DEFAULT\_LEFT\_TO\_RIGHT, and DIRECTION\_DEFAULT\_RIGHT\_TO\_LEFT. Other values are reserved.

### Bidi

public **Bidi**([AttributedCharacterIterator](http://docs.google.com/java/text/AttributedCharacterIterator.html) paragraph)

Create Bidi from the given paragraph of text.

The RUN\_DIRECTION attribute in the text, if present, determines the base direction (left-to-right or right-to-left). If not present, the base direction is computes using the Unicode Bidirectional Algorithm, defaulting to left-to-right if there are no strong directional characters in the text. This attribute, if present, must be applied to all the text in the paragraph.

The BIDI\_EMBEDDING attribute in the text, if present, represents embedding level information. Negative values from -1 to -62 indicate overrides at the absolute value of the level. Positive values from 1 to 62 indicate embeddings. Where values are zero or not defined, the base embedding level as determined by the base direction is assumed.

The NUMERIC\_SHAPING attribute in the text, if present, converts European digits to other decimal digits before running the bidi algorithm. This attribute, if present, must be applied to all the text in the paragraph.

**Parameters:**paragraph - a paragraph of text with optional character and paragraph attribute information**See Also:**[TextAttribute.BIDI\_EMBEDDING](http://docs.google.com/java/awt/font/TextAttribute.html#BIDI_EMBEDDING), [TextAttribute.NUMERIC\_SHAPING](http://docs.google.com/java/awt/font/TextAttribute.html#NUMERIC_SHAPING), [TextAttribute.RUN\_DIRECTION](http://docs.google.com/java/awt/font/TextAttribute.html#RUN_DIRECTION)

### Bidi

public **Bidi**(char[] text,  
 int textStart,  
 byte[] embeddings,  
 int embStart,  
 int paragraphLength,  
 int flags)

Create Bidi from the given text, embedding, and direction information. The embeddings array may be null. If present, the values represent embedding level information. Negative values from -1 to -61 indicate overrides at the absolute value of the level. Positive values from 1 to 61 indicate embeddings. Where values are zero, the base embedding level as determined by the base direction is assumed.

**Parameters:**text - an array containing the paragraph of text to process.textStart - the index into the text array of the start of the paragraph.embeddings - an array containing embedding values for each character in the paragraph. This can be null, in which case it is assumed that there is no external embedding information.embStart - the index into the embedding array of the start of the paragraph.paragraphLength - the length of the paragraph in the text and embeddings arrays.flags - a collection of flags that control the algorithm. The algorithm understands the flags DIRECTION\_LEFT\_TO\_RIGHT, DIRECTION\_RIGHT\_TO\_LEFT, DIRECTION\_DEFAULT\_LEFT\_TO\_RIGHT, and DIRECTION\_DEFAULT\_RIGHT\_TO\_LEFT. Other values are reserved.

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

### createLineBidi

public [Bidi](http://docs.google.com/java/text/Bidi.html) **createLineBidi**(int lineStart,  
 int lineLimit)

Create a Bidi object representing the bidi information on a line of text within the paragraph represented by the current Bidi. This call is not required if the entire paragraph fits on one line.

**Parameters:**lineStart - the offset from the start of the paragraph to the start of the line.lineLimit - the offset from the start of the paragraph to the limit of the line.

### isMixed

public boolean **isMixed**()

Return true if the line is not left-to-right or right-to-left. This means it either has mixed runs of left-to-right and right-to-left text, or the base direction differs from the direction of the only run of text.

**Returns:**true if the line is not left-to-right or right-to-left.

### isLeftToRight

public boolean **isLeftToRight**()

Return true if the line is all left-to-right text and the base direction is left-to-right.

**Returns:**true if the line is all left-to-right text and the base direction is left-to-right

### isRightToLeft

public boolean **isRightToLeft**()

Return true if the line is all right-to-left text, and the base direction is right-to-left.

**Returns:**true if the line is all right-to-left text, and the base direction is right-to-left

### getLength

public int **getLength**()

Return the length of text in the line.

**Returns:**the length of text in the line

### baseIsLeftToRight

public boolean **baseIsLeftToRight**()

Return true if the base direction is left-to-right.

**Returns:**true if the base direction is left-to-right

### getBaseLevel

public int **getBaseLevel**()

Return the base level (0 if left-to-right, 1 if right-to-left).

**Returns:**the base level

### getLevelAt

public int **getLevelAt**(int offset)

Return the resolved level of the character at offset. If offset is <0 or >= the length of the line, return the base direction level.

**Parameters:**offset - the index of the character for which to return the level **Returns:**the resolved level of the character at offset

### getRunCount

public int **getRunCount**()

Return the number of level runs.

**Returns:**the number of level runs

### getRunLevel

public int **getRunLevel**(int run)

Return the level of the nth logical run in this line.

**Parameters:**run - the index of the run, between 0 and getRunCount() **Returns:**the level of the run

### getRunStart

public int **getRunStart**(int run)

Return the index of the character at the start of the nth logical run in this line, as an offset from the start of the line.

**Parameters:**run - the index of the run, between 0 and getRunCount() **Returns:**the start of the run

### getRunLimit

public int **getRunLimit**(int run)

Return the index of the character past the end of the nth logical run in this line, as an offset from the start of the line. For example, this will return the length of the line for the last run on the line.

**Parameters:**run - the index of the run, between 0 and getRunCount() **Returns:**limit the limit of the run

### requiresBidi

public static boolean **requiresBidi**(char[] text,  
 int start,  
 int limit)

Return true if the specified text requires bidi analysis. If this returns false, the text will display left-to-right. Clients can then avoid constructing a Bidi object. Text in the Arabic Presentation Forms area of Unicode is presumed to already be shaped and ordered for display, and so will not cause this function to return true.

**Parameters:**text - the text containing the characters to teststart - the start of the range of characters to testlimit - the limit of the range of characters to test **Returns:**true if the range of characters requires bidi analysis

### reorderVisually

public static void **reorderVisually**(byte[] levels,  
 int levelStart,  
 [Object](http://docs.google.com/java/lang/Object.html)[] objects,  
 int objectStart,  
 int count)

Reorder the objects in the array into visual order based on their levels. This is a utility function to use when you have a collection of objects representing runs of text in logical order, each run containing text at a single level. The elements at index from objectStart up to objectStart + count in the objects array will be reordered into visual order assuming each run of text has the level indicated by the corresponding element in the levels array (at index - objectStart + levelStart).

**Parameters:**levels - an array representing the bidi level of each objectlevelStart - the start position in the levels arrayobjects - the array of objects to be reordered into visual orderobjectStart - the start position in the objects arraycount - the number of objects to reorder

### toString

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

Display the bidi internal state, used in debugging.

**Overrides:**[toString](http://docs.google.com/java/lang/Object.html#toString()) in class [Object](http://docs.google.com/java/lang/Object.html) **Returns:**a string representation of the object.

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