| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Normalizer.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/MessageFormat.Field.html)   [**NEXT CLASS**](http://docs.google.com/java/text/Normalizer.Form.html) | [**FRAMES**](http://docs.google.com/index.html?java/text/Normalizer.html)    [**NO FRAMES**](http://docs.google.com/Normalizer.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | FIELD | CONSTR | [METHOD](#2et92p0) | DETAIL: FIELD | CONSTR | [METHOD](#3dy6vkm) |

## **java.text**

Class Normalizer

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

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

This class provides the method normalize which transforms Unicode text into an equivalent composed or decomposed form, allowing for easier sorting and searching of text. The normalize method supports the standard normalization forms described in  [Unicode Standard Annex #15 — Unicode Normalization Forms](http://www.unicode.org/unicode/reports/tr15/tr15-23.html).

Characters with accents or other adornments can be encoded in several different ways in Unicode. For example, take the character A-acute. In Unicode, this can be encoded as a single character (the "composed" form):

U+00C1 LATIN CAPITAL LETTER A WITH ACUTE

or as two separate characters (the "decomposed" form):

U+0041 LATIN CAPITAL LETTER A  
 U+0301 COMBINING ACUTE ACCENT

To a user of your program, however, both of these sequences should be treated as the same "user-level" character "A with acute accent". When you are searching or comparing text, you must ensure that these two sequences are treated as equivalent. In addition, you must handle characters with more than one accent. Sometimes the order of a character's combining accents is significant, while in other cases accent sequences in different orders are really equivalent.

Similarly, the string "ffi" can be encoded as three separate letters:

U+0066 LATIN SMALL LETTER F  
 U+0066 LATIN SMALL LETTER F  
 U+0069 LATIN SMALL LETTER I

or as the single character

U+FB03 LATIN SMALL LIGATURE FFI

The ffi ligature is not a distinct semantic character, and strictly speaking it shouldn't be in Unicode at all, but it was included for compatibility with existing character sets that already provided it. The Unicode standard identifies such characters by giving them "compatibility" decompositions into the corresponding semantic characters. When sorting and searching, you will often want to use these mappings.

The normalize method helps solve these problems by transforming text into the canonical composed and decomposed forms as shown in the first example above. In addition, you can have it perform compatibility decompositions so that you can treat compatibility characters the same as their equivalents. Finally, the normalize method rearranges accents into the proper canonical order, so that you do not have to worry about accent rearrangement on your own.

The W3C generally recommends to exchange texts in NFC. Note also that most legacy character encodings use only precomposed forms and often do not encode any combining marks by themselves. For conversion to such character encodings the Unicode text needs to be normalized to NFC. For more usage examples, see the Unicode Standard Annex.

**Since:** 1.6

| **Nested Class Summary** | |
| --- | --- |
| static class | [**Normalizer.Form**](http://docs.google.com/java/text/Normalizer.Form.html)            This enum provides constants of the four Unicode normalization forms that are described in  [Unicode Standard Annex #15 — Unicode Normalization Forms](http://www.unicode.org/unicode/reports/tr15/tr15-23.html) and two methods to access them. |

| **Method Summary** | |
| --- | --- |
| static boolean | [**isNormalized**](http://docs.google.com/java/text/Normalizer.html#isNormalized(java.lang.CharSequence,%20java.text.Normalizer.Form))([CharSequence](http://docs.google.com/java/lang/CharSequence.html) src, [Normalizer.Form](http://docs.google.com/java/text/Normalizer.Form.html) form)            Determines if the given sequence of char values is normalized. |
| static [String](http://docs.google.com/java/lang/String.html) | [**normalize**](http://docs.google.com/java/text/Normalizer.html#normalize(java.lang.CharSequence,%20java.text.Normalizer.Form))([CharSequence](http://docs.google.com/java/lang/CharSequence.html) src, [Normalizer.Form](http://docs.google.com/java/text/Normalizer.Form.html) form)            Normalize a sequence of char values. |

| **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)) |

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

### normalize

public static [String](http://docs.google.com/java/lang/String.html) **normalize**([CharSequence](http://docs.google.com/java/lang/CharSequence.html) src,  
 [Normalizer.Form](http://docs.google.com/java/text/Normalizer.Form.html) form)

Normalize a sequence of char values. The sequence will be normalized according to the specified normalization from.

**Parameters:**src - The sequence of char values to normalize.form - The normalization form; one of [Normalizer.Form.NFC](http://docs.google.com/java/text/Normalizer.Form.html#NFC), [Normalizer.Form.NFD](http://docs.google.com/java/text/Normalizer.Form.html#NFD), [Normalizer.Form.NFKC](http://docs.google.com/java/text/Normalizer.Form.html#NFKC), [Normalizer.Form.NFKD](http://docs.google.com/java/text/Normalizer.Form.html#NFKD) **Returns:**The normalized String **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - If src or form is null.

### isNormalized

public static boolean **isNormalized**([CharSequence](http://docs.google.com/java/lang/CharSequence.html) src,  
 [Normalizer.Form](http://docs.google.com/java/text/Normalizer.Form.html) form)

Determines if the given sequence of char values is normalized.

**Parameters:**src - The sequence of char values to be checked.form - The normalization form; one of [Normalizer.Form.NFC](http://docs.google.com/java/text/Normalizer.Form.html#NFC), [Normalizer.Form.NFD](http://docs.google.com/java/text/Normalizer.Form.html#NFD), [Normalizer.Form.NFKC](http://docs.google.com/java/text/Normalizer.Form.html#NFKC), [Normalizer.Form.NFKD](http://docs.google.com/java/text/Normalizer.Form.html#NFKD) **Returns:**true if the sequence of char values is normalized; false otherwise. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - If src or form is null.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Normalizer.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/MessageFormat.Field.html)   [**NEXT CLASS**](http://docs.google.com/java/text/Normalizer.Form.html) | [**FRAMES**](http://docs.google.com/index.html?java/text/Normalizer.html)    [**NO FRAMES**](http://docs.google.com/Normalizer.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | FIELD | CONSTR | [METHOD](#2et92p0) | DETAIL: FIELD | CONSTR | [METHOD](#3dy6vkm) |

[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).