| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/NumberFormat.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/Normalizer.Form.html)   [**NEXT CLASS**](http://docs.google.com/java/text/NumberFormat.Field.html) | [**FRAMES**](http://docs.google.com/index.html?java/text/NumberFormat.html)    [**NO FRAMES**](http://docs.google.com/NumberFormat.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#2et92p0) | [FIELD](#tyjcwt) | [CONSTR](#3dy6vkm) | [METHOD](#1t3h5sf) | DETAIL: [FIELD](#17dp8vu) | [CONSTR](#lnxbz9) | [METHOD](#1ksv4uv) |

## **java.text**

Class NumberFormat

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

**All Implemented Interfaces:** [Serializable](http://docs.google.com/java/io/Serializable.html), [Cloneable](http://docs.google.com/java/lang/Cloneable.html) **Direct Known Subclasses:** [ChoiceFormat](http://docs.google.com/java/text/ChoiceFormat.html), [DecimalFormat](http://docs.google.com/java/text/DecimalFormat.html)

public abstract class **NumberFormat**extends [Format](http://docs.google.com/java/text/Format.html)

NumberFormat is the abstract base class for all number formats. This class provides the interface for formatting and parsing numbers. NumberFormat also provides methods for determining which locales have number formats, and what their names are.

NumberFormat helps you to format and parse numbers for any locale. Your code can be completely independent of the locale conventions for decimal points, thousands-separators, or even the particular decimal digits used, or whether the number format is even decimal.

To format a number for the current Locale, use one of the factory class methods:

myString = NumberFormat.getInstance().format(myNumber);

If you are formatting multiple numbers, it is more efficient to get the format and use it multiple times so that the system doesn't have to fetch the information about the local language and country conventions multiple times.

NumberFormat nf = NumberFormat.getInstance();  
 for (int i = 0; i < myNumber.length; ++i) {  
 output.println(nf.format(myNumber[i]) + "; ");  
 }

To format a number for a different Locale, specify it in the call to getInstance.

NumberFormat nf = NumberFormat.getInstance(Locale.FRENCH);

You can also use a NumberFormat to parse numbers:

myNumber = nf.parse(myString);

Use getInstance or getNumberInstance to get the normal number format. Use getIntegerInstance to get an integer number format. Use getCurrencyInstance to get the currency number format. And use getPercentInstance to get a format for displaying percentages. With this format, a fraction like 0.53 is displayed as 53%.

You can also control the display of numbers with such methods as setMinimumFractionDigits. If you want even more control over the format or parsing, or want to give your users more control, you can try casting the NumberFormat you get from the factory methods to a DecimalFormat. This will work for the vast majority of locales; just remember to put it in a try block in case you encounter an unusual one.

NumberFormat and DecimalFormat are designed such that some controls work for formatting and others work for parsing. The following is the detailed description for each these control methods,

setParseIntegerOnly : only affects parsing, e.g. if true, "3456.78" -> 3456 (and leaves the parse position just after index 6) if false, "3456.78" -> 3456.78 (and leaves the parse position just after index 8) This is independent of formatting. If you want to not show a decimal point where there might be no digits after the decimal point, use setDecimalSeparatorAlwaysShown.

setDecimalSeparatorAlwaysShown : only affects formatting, and only where there might be no digits after the decimal point, such as with a pattern like "#,##0.##", e.g., if true, 3456.00 -> "3,456." if false, 3456.00 -> "3456" This is independent of parsing. If you want parsing to stop at the decimal point, use setParseIntegerOnly.

You can also use forms of the parse and format methods with ParsePosition and FieldPosition to allow you to:

* progressively parse through pieces of a string
* align the decimal point and other areas

For example, you can align numbers in two ways:

1. If you are using a monospaced font with spacing for alignment, you can pass the FieldPosition in your format call, with field = INTEGER\_FIELD. On output, getEndIndex will be set to the offset between the last character of the integer and the decimal. Add (desiredSpaceCount - getEndIndex) spaces at the front of the string.
2. If you are using proportional fonts, instead of padding with spaces, measure the width of the string in pixels from the start to getEndIndex. Then move the pen by (desiredPixelWidth - widthToAlignmentPoint) before drawing the text. It also works where there is no decimal, but possibly additional characters at the end, e.g., with parentheses in negative numbers: "(12)" for -12.

#### Synchronization

Number formats are generally not synchronized. It is recommended to create separate format instances for each thread. If multiple threads access a format concurrently, it must be synchronized externally.

**See Also:**[DecimalFormat](http://docs.google.com/java/text/DecimalFormat.html), [ChoiceFormat](http://docs.google.com/java/text/ChoiceFormat.html), [Serialized Form](http://docs.google.com/serialized-form.html#java.text.NumberFormat)

| **Nested Class Summary** | |
| --- | --- |
| static class | [**NumberFormat.Field**](http://docs.google.com/java/text/NumberFormat.Field.html)            Defines constants that are used as attribute keys in the AttributedCharacterIterator returned from NumberFormat.formatToCharacterIterator and as field identifiers in FieldPosition. |

| **Field Summary** | |
| --- | --- |
| static int | [**FRACTION\_FIELD**](http://docs.google.com/java/text/NumberFormat.html#FRACTION_FIELD)            Field constant used to construct a FieldPosition object. |
| static int | [**INTEGER\_FIELD**](http://docs.google.com/java/text/NumberFormat.html#INTEGER_FIELD)            Field constant used to construct a FieldPosition object. |

| **Constructor Summary** | |
| --- | --- |
| protected | [**NumberFormat**](http://docs.google.com/java/text/NumberFormat.html#NumberFormat())()            Sole constructor. |

| **Method Summary** | |
| --- | --- |
| [Object](http://docs.google.com/java/lang/Object.html) | [**clone**](http://docs.google.com/java/text/NumberFormat.html#clone())()            Overrides Cloneable |
| boolean | [**equals**](http://docs.google.com/java/text/NumberFormat.html#equals(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) obj)            Overrides equals |
| [String](http://docs.google.com/java/lang/String.html) | [**format**](http://docs.google.com/java/text/NumberFormat.html#format(double))(double number)            Specialization of format. |
| abstract  [StringBuffer](http://docs.google.com/java/lang/StringBuffer.html) | [**format**](http://docs.google.com/java/text/NumberFormat.html#format(double,%20java.lang.StringBuffer,%20java.text.FieldPosition))(double number, [StringBuffer](http://docs.google.com/java/lang/StringBuffer.html) toAppendTo, [FieldPosition](http://docs.google.com/java/text/FieldPosition.html) pos)            Specialization of format. |
| [String](http://docs.google.com/java/lang/String.html) | [**format**](http://docs.google.com/java/text/NumberFormat.html#format(long))(long number)            Specialization of format. |
| abstract  [StringBuffer](http://docs.google.com/java/lang/StringBuffer.html) | [**format**](http://docs.google.com/java/text/NumberFormat.html#format(long,%20java.lang.StringBuffer,%20java.text.FieldPosition))(long number, [StringBuffer](http://docs.google.com/java/lang/StringBuffer.html) toAppendTo, [FieldPosition](http://docs.google.com/java/text/FieldPosition.html) pos)            Specialization of format. |
| [StringBuffer](http://docs.google.com/java/lang/StringBuffer.html) | [**format**](http://docs.google.com/java/text/NumberFormat.html#format(java.lang.Object,%20java.lang.StringBuffer,%20java.text.FieldPosition))([Object](http://docs.google.com/java/lang/Object.html) number, [StringBuffer](http://docs.google.com/java/lang/StringBuffer.html) toAppendTo, [FieldPosition](http://docs.google.com/java/text/FieldPosition.html) pos)            Formats a number and appends the resulting text to the given string buffer. |
| static [Locale](http://docs.google.com/java/util/Locale.html)[] | [**getAvailableLocales**](http://docs.google.com/java/text/NumberFormat.html#getAvailableLocales())()            Returns an array of all locales for which the get\*Instance methods of this class can return localized instances. |
| [Currency](http://docs.google.com/java/util/Currency.html) | [**getCurrency**](http://docs.google.com/java/text/NumberFormat.html#getCurrency())()            Gets the currency used by this number format when formatting currency values. |
| static [NumberFormat](http://docs.google.com/java/text/NumberFormat.html) | [**getCurrencyInstance**](http://docs.google.com/java/text/NumberFormat.html#getCurrencyInstance())()            Returns a currency format for the current default locale. |
| static [NumberFormat](http://docs.google.com/java/text/NumberFormat.html) | [**getCurrencyInstance**](http://docs.google.com/java/text/NumberFormat.html#getCurrencyInstance(java.util.Locale))([Locale](http://docs.google.com/java/util/Locale.html) inLocale)            Returns a currency format for the specified locale. |
| static [NumberFormat](http://docs.google.com/java/text/NumberFormat.html) | [**getInstance**](http://docs.google.com/java/text/NumberFormat.html#getInstance())()            Returns a general-purpose number format for the current default locale. |
| static [NumberFormat](http://docs.google.com/java/text/NumberFormat.html) | [**getInstance**](http://docs.google.com/java/text/NumberFormat.html#getInstance(java.util.Locale))([Locale](http://docs.google.com/java/util/Locale.html) inLocale)            Returns a general-purpose number format for the specified locale. |
| static [NumberFormat](http://docs.google.com/java/text/NumberFormat.html) | [**getIntegerInstance**](http://docs.google.com/java/text/NumberFormat.html#getIntegerInstance())()            Returns an integer number format for the current default locale. |
| static [NumberFormat](http://docs.google.com/java/text/NumberFormat.html) | [**getIntegerInstance**](http://docs.google.com/java/text/NumberFormat.html#getIntegerInstance(java.util.Locale))([Locale](http://docs.google.com/java/util/Locale.html) inLocale)            Returns an integer number format for the specified locale. |
| int | [**getMaximumFractionDigits**](http://docs.google.com/java/text/NumberFormat.html#getMaximumFractionDigits())()            Returns the maximum number of digits allowed in the fraction portion of a number. |
| int | [**getMaximumIntegerDigits**](http://docs.google.com/java/text/NumberFormat.html#getMaximumIntegerDigits())()            Returns the maximum number of digits allowed in the integer portion of a number. |
| int | [**getMinimumFractionDigits**](http://docs.google.com/java/text/NumberFormat.html#getMinimumFractionDigits())()            Returns the minimum number of digits allowed in the fraction portion of a number. |
| int | [**getMinimumIntegerDigits**](http://docs.google.com/java/text/NumberFormat.html#getMinimumIntegerDigits())()            Returns the minimum number of digits allowed in the integer portion of a number. |
| static [NumberFormat](http://docs.google.com/java/text/NumberFormat.html) | [**getNumberInstance**](http://docs.google.com/java/text/NumberFormat.html#getNumberInstance())()            Returns a general-purpose number format for the current default locale. |
| static [NumberFormat](http://docs.google.com/java/text/NumberFormat.html) | [**getNumberInstance**](http://docs.google.com/java/text/NumberFormat.html#getNumberInstance(java.util.Locale))([Locale](http://docs.google.com/java/util/Locale.html) inLocale)            Returns a general-purpose number format for the specified locale. |
| static [NumberFormat](http://docs.google.com/java/text/NumberFormat.html) | [**getPercentInstance**](http://docs.google.com/java/text/NumberFormat.html#getPercentInstance())()            Returns a percentage format for the current default locale. |
| static [NumberFormat](http://docs.google.com/java/text/NumberFormat.html) | [**getPercentInstance**](http://docs.google.com/java/text/NumberFormat.html#getPercentInstance(java.util.Locale))([Locale](http://docs.google.com/java/util/Locale.html) inLocale)            Returns a percentage format for the specified locale. |
| [RoundingMode](http://docs.google.com/java/math/RoundingMode.html) | [**getRoundingMode**](http://docs.google.com/java/text/NumberFormat.html#getRoundingMode())()            Gets the [RoundingMode](http://docs.google.com/java/math/RoundingMode.html) used in this NumberFormat. |
| int | [**hashCode**](http://docs.google.com/java/text/NumberFormat.html#hashCode())()            Overrides hashCode |
| boolean | [**isGroupingUsed**](http://docs.google.com/java/text/NumberFormat.html#isGroupingUsed())()            Returns true if grouping is used in this format. |
| boolean | [**isParseIntegerOnly**](http://docs.google.com/java/text/NumberFormat.html#isParseIntegerOnly())()            Returns true if this format will parse numbers as integers only. |
| [Number](http://docs.google.com/java/lang/Number.html) | [**parse**](http://docs.google.com/java/text/NumberFormat.html#parse(java.lang.String))([String](http://docs.google.com/java/lang/String.html) source)            Parses text from the beginning of the given string to produce a number. |
| abstract  [Number](http://docs.google.com/java/lang/Number.html) | [**parse**](http://docs.google.com/java/text/NumberFormat.html#parse(java.lang.String,%20java.text.ParsePosition))([String](http://docs.google.com/java/lang/String.html) source, [ParsePosition](http://docs.google.com/java/text/ParsePosition.html) parsePosition)            Returns a Long if possible (e.g., within the range [Long.MIN\_VALUE, Long.MAX\_VALUE] and with no decimals), otherwise a Double. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**parseObject**](http://docs.google.com/java/text/NumberFormat.html#parseObject(java.lang.String,%20java.text.ParsePosition))([String](http://docs.google.com/java/lang/String.html) source, [ParsePosition](http://docs.google.com/java/text/ParsePosition.html) pos)            Parses text from a string to produce a Number. |
| void | [**setCurrency**](http://docs.google.com/java/text/NumberFormat.html#setCurrency(java.util.Currency))([Currency](http://docs.google.com/java/util/Currency.html) currency)            Sets the currency used by this number format when formatting currency values. |
| void | [**setGroupingUsed**](http://docs.google.com/java/text/NumberFormat.html#setGroupingUsed(boolean))(boolean newValue)            Set whether or not grouping will be used in this format. |
| void | [**setMaximumFractionDigits**](http://docs.google.com/java/text/NumberFormat.html#setMaximumFractionDigits(int))(int newValue)            Sets the maximum number of digits allowed in the fraction portion of a number. |
| void | [**setMaximumIntegerDigits**](http://docs.google.com/java/text/NumberFormat.html#setMaximumIntegerDigits(int))(int newValue)            Sets the maximum number of digits allowed in the integer portion of a number. |
| void | [**setMinimumFractionDigits**](http://docs.google.com/java/text/NumberFormat.html#setMinimumFractionDigits(int))(int newValue)            Sets the minimum number of digits allowed in the fraction portion of a number. |
| void | [**setMinimumIntegerDigits**](http://docs.google.com/java/text/NumberFormat.html#setMinimumIntegerDigits(int))(int newValue)            Sets the minimum number of digits allowed in the integer portion of a number. |
| void | [**setParseIntegerOnly**](http://docs.google.com/java/text/NumberFormat.html#setParseIntegerOnly(boolean))(boolean value)            Sets whether or not numbers should be parsed as integers only. |
| void | [**setRoundingMode**](http://docs.google.com/java/text/NumberFormat.html#setRoundingMode(java.math.RoundingMode))([RoundingMode](http://docs.google.com/java/math/RoundingMode.html) roundingMode)            Sets the [RoundingMode](http://docs.google.com/java/math/RoundingMode.html) used in this NumberFormat. |

| **Methods inherited from class java.text.**[**Format**](http://docs.google.com/java/text/Format.html) |
| --- |
| [format](http://docs.google.com/java/text/Format.html#format(java.lang.Object)), [formatToCharacterIterator](http://docs.google.com/java/text/Format.html#formatToCharacterIterator(java.lang.Object)), [parseObject](http://docs.google.com/java/text/Format.html#parseObject(java.lang.String)) |

| **Methods inherited from class java.lang.**[**Object**](http://docs.google.com/java/lang/Object.html) |
| --- |
| [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()), [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** |
| --- |

### INTEGER\_FIELD

public static final int **INTEGER\_FIELD**

Field constant used to construct a FieldPosition object. Signifies that the position of the integer part of a formatted number should be returned.

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

### FRACTION\_FIELD

public static final int **FRACTION\_FIELD**

Field constant used to construct a FieldPosition object. Signifies that the position of the fraction part of a formatted number should be returned.

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

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

### NumberFormat

protected **NumberFormat**()

Sole constructor. (For invocation by subclass constructors, typically implicit.)

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

### format

public [StringBuffer](http://docs.google.com/java/lang/StringBuffer.html) **format**([Object](http://docs.google.com/java/lang/Object.html) number,  
 [StringBuffer](http://docs.google.com/java/lang/StringBuffer.html) toAppendTo,  
 [FieldPosition](http://docs.google.com/java/text/FieldPosition.html) pos)

Formats a number and appends the resulting text to the given string buffer. The number can be of any subclass of [Number](http://docs.google.com/java/lang/Number.html).

This implementation extracts the number's value using [Number.longValue()](http://docs.google.com/java/lang/Number.html#longValue()) for all integral type values that can be converted to long without loss of information, including BigInteger values with a [bit length](http://docs.google.com/java/math/BigInteger.html#bitLength()) of less than 64, and [Number.doubleValue()](http://docs.google.com/java/lang/Number.html#doubleValue()) for all other types. It then calls [format(long,java.lang.StringBuffer,java.text.FieldPosition)](http://docs.google.com/java/text/NumberFormat.html#format(long,%20java.lang.StringBuffer,%20java.text.FieldPosition)) or [format(double,java.lang.StringBuffer,java.text.FieldPosition)](http://docs.google.com/java/text/NumberFormat.html#format(double,%20java.lang.StringBuffer,%20java.text.FieldPosition)). This may result in loss of magnitude information and precision for BigInteger and BigDecimal values.

**Specified by:**[format](http://docs.google.com/java/text/Format.html#format(java.lang.Object,%20java.lang.StringBuffer,%20java.text.FieldPosition)) in class [Format](http://docs.google.com/java/text/Format.html) **Parameters:**number - the number to formattoAppendTo - the StringBuffer to which the formatted text is to be appendedpos - On input: an alignment field, if desired. On output: the offsets of the alignment field. **Returns:**the value passed in as toAppendTo **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if number is null or not an instance of Number. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if toAppendTo or pos is null [ArithmeticException](http://docs.google.com/java/lang/ArithmeticException.html) - if rounding is needed with rounding mode being set to RoundingMode.UNNECESSARY**See Also:**[FieldPosition](http://docs.google.com/java/text/FieldPosition.html)

### parseObject

public final [Object](http://docs.google.com/java/lang/Object.html) **parseObject**([String](http://docs.google.com/java/lang/String.html) source,  
 [ParsePosition](http://docs.google.com/java/text/ParsePosition.html) pos)

Parses text from a string to produce a Number.

The method attempts to parse text starting at the index given by pos. If parsing succeeds, then the index of pos is updated to the index after the last character used (parsing does not necessarily use all characters up to the end of the string), and the parsed number is returned. The updated pos can be used to indicate the starting point for the next call to this method. If an error occurs, then the index of pos is not changed, the error index of pos is set to the index of the character where the error occurred, and null is returned.

See the [parse(String, ParsePosition)](http://docs.google.com/java/text/NumberFormat.html#parse(java.lang.String,%20java.text.ParsePosition)) method for more information on number parsing.

**Specified by:**[parseObject](http://docs.google.com/java/text/Format.html#parseObject(java.lang.String,%20java.text.ParsePosition)) in class [Format](http://docs.google.com/java/text/Format.html) **Parameters:**source - A String, part of which should be parsed.pos - A ParsePosition object with index and error index information as described above. **Returns:**A Number parsed from the string. In case of error, returns null. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if pos is null.

### format

public final [String](http://docs.google.com/java/lang/String.html) **format**(double number)

Specialization of format.

**Throws:** [ArithmeticException](http://docs.google.com/java/lang/ArithmeticException.html) - if rounding is needed with rounding mode being set to RoundingMode.UNNECESSARY**See Also:**[Format.format(java.lang.Object)](http://docs.google.com/java/text/Format.html#format(java.lang.Object))

### format

public final [String](http://docs.google.com/java/lang/String.html) **format**(long number)

Specialization of format.

**Throws:** [ArithmeticException](http://docs.google.com/java/lang/ArithmeticException.html) - if rounding is needed with rounding mode being set to RoundingMode.UNNECESSARY**See Also:**[Format.format(java.lang.Object)](http://docs.google.com/java/text/Format.html#format(java.lang.Object))

### format

public abstract [StringBuffer](http://docs.google.com/java/lang/StringBuffer.html) **format**(double number,  
 [StringBuffer](http://docs.google.com/java/lang/StringBuffer.html) toAppendTo,  
 [FieldPosition](http://docs.google.com/java/text/FieldPosition.html) pos)

Specialization of format.

**Throws:** [ArithmeticException](http://docs.google.com/java/lang/ArithmeticException.html) - if rounding is needed with rounding mode being set to RoundingMode.UNNECESSARY**See Also:**[Format.format(java.lang.Object)](http://docs.google.com/java/text/Format.html#format(java.lang.Object))

### format

public abstract [StringBuffer](http://docs.google.com/java/lang/StringBuffer.html) **format**(long number,  
 [StringBuffer](http://docs.google.com/java/lang/StringBuffer.html) toAppendTo,  
 [FieldPosition](http://docs.google.com/java/text/FieldPosition.html) pos)

Specialization of format.

**Throws:** [ArithmeticException](http://docs.google.com/java/lang/ArithmeticException.html) - if rounding is needed with rounding mode being set to RoundingMode.UNNECESSARY**See Also:**[Format.format(java.lang.Object)](http://docs.google.com/java/text/Format.html#format(java.lang.Object))

### parse

public abstract [Number](http://docs.google.com/java/lang/Number.html) **parse**([String](http://docs.google.com/java/lang/String.html) source,  
 [ParsePosition](http://docs.google.com/java/text/ParsePosition.html) parsePosition)

Returns a Long if possible (e.g., within the range [Long.MIN\_VALUE, Long.MAX\_VALUE] and with no decimals), otherwise a Double. If IntegerOnly is set, will stop at a decimal point (or equivalent; e.g., for rational numbers "1 2/3", will stop after the 1). Does not throw an exception; if no object can be parsed, index is unchanged!

**See Also:**[isParseIntegerOnly()](http://docs.google.com/java/text/NumberFormat.html#isParseIntegerOnly()), [Format.parseObject(java.lang.String, java.text.ParsePosition)](http://docs.google.com/java/text/Format.html#parseObject(java.lang.String,%20java.text.ParsePosition))

### parse

public [Number](http://docs.google.com/java/lang/Number.html) **parse**([String](http://docs.google.com/java/lang/String.html) source)  
 throws [ParseException](http://docs.google.com/java/text/ParseException.html)

Parses text from the beginning of the given string to produce a number. The method may not use the entire text of the given string.

See the [parse(String, ParsePosition)](http://docs.google.com/java/text/NumberFormat.html#parse(java.lang.String,%20java.text.ParsePosition)) method for more information on number parsing.

**Parameters:**source - A String whose beginning should be parsed. **Returns:**A Number parsed from the string. **Throws:** [ParseException](http://docs.google.com/java/text/ParseException.html) - if the beginning of the specified string cannot be parsed.

### isParseIntegerOnly

public boolean **isParseIntegerOnly**()

Returns true if this format will parse numbers as integers only. For example in the English locale, with ParseIntegerOnly true, the string "1234." would be parsed as the integer value 1234 and parsing would stop at the "." character. Of course, the exact format accepted by the parse operation is locale dependant and determined by sub-classes of NumberFormat.

### setParseIntegerOnly

public void **setParseIntegerOnly**(boolean value)

Sets whether or not numbers should be parsed as integers only.

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

### getInstance

public static final [NumberFormat](http://docs.google.com/java/text/NumberFormat.html) **getInstance**()

Returns a general-purpose number format for the current default locale. This is the same as calling [getNumberInstance()](http://docs.google.com/java/text/NumberFormat.html#getNumberInstance()).

### getInstance

public static [NumberFormat](http://docs.google.com/java/text/NumberFormat.html) **getInstance**([Locale](http://docs.google.com/java/util/Locale.html) inLocale)

Returns a general-purpose number format for the specified locale. This is the same as calling [getNumberInstance(inLocale)](http://docs.google.com/java/text/NumberFormat.html#getNumberInstance(java.util.Locale)).

### getNumberInstance

public static final [NumberFormat](http://docs.google.com/java/text/NumberFormat.html) **getNumberInstance**()

Returns a general-purpose number format for the current default locale.

### getNumberInstance

public static [NumberFormat](http://docs.google.com/java/text/NumberFormat.html) **getNumberInstance**([Locale](http://docs.google.com/java/util/Locale.html) inLocale)

Returns a general-purpose number format for the specified locale.

### getIntegerInstance

public static final [NumberFormat](http://docs.google.com/java/text/NumberFormat.html) **getIntegerInstance**()

Returns an integer number format for the current default locale. The returned number format is configured to round floating point numbers to the nearest integer using half-even rounding (see [RoundingMode.HALF\_EVEN](http://docs.google.com/java/math/RoundingMode.html#HALF_EVEN)) for formatting, and to parse only the integer part of an input string (see [isParseIntegerOnly](http://docs.google.com/java/text/NumberFormat.html#isParseIntegerOnly())).

**Returns:**a number format for integer values**Since:** 1.4 **See Also:**[getRoundingMode()](http://docs.google.com/java/text/NumberFormat.html#getRoundingMode())

### getIntegerInstance

public static [NumberFormat](http://docs.google.com/java/text/NumberFormat.html) **getIntegerInstance**([Locale](http://docs.google.com/java/util/Locale.html) inLocale)

Returns an integer number format for the specified locale. The returned number format is configured to round floating point numbers to the nearest integer using half-even rounding (see [RoundingMode.HALF\_EVEN](http://docs.google.com/java/math/RoundingMode.html#HALF_EVEN)) for formatting, and to parse only the integer part of an input string (see [isParseIntegerOnly](http://docs.google.com/java/text/NumberFormat.html#isParseIntegerOnly())).

**Returns:**a number format for integer values**Since:** 1.4 **See Also:**[getRoundingMode()](http://docs.google.com/java/text/NumberFormat.html#getRoundingMode())

### getCurrencyInstance

public static final [NumberFormat](http://docs.google.com/java/text/NumberFormat.html) **getCurrencyInstance**()

Returns a currency format for the current default locale.

### getCurrencyInstance

public static [NumberFormat](http://docs.google.com/java/text/NumberFormat.html) **getCurrencyInstance**([Locale](http://docs.google.com/java/util/Locale.html) inLocale)

Returns a currency format for the specified locale.

### getPercentInstance

public static final [NumberFormat](http://docs.google.com/java/text/NumberFormat.html) **getPercentInstance**()

Returns a percentage format for the current default locale.

### getPercentInstance

public static [NumberFormat](http://docs.google.com/java/text/NumberFormat.html) **getPercentInstance**([Locale](http://docs.google.com/java/util/Locale.html) inLocale)

Returns a percentage format for the specified locale.

### getAvailableLocales

public static [Locale](http://docs.google.com/java/util/Locale.html)[] **getAvailableLocales**()

Returns an array of all locales for which the get\*Instance methods of this class can return localized instances. The returned array represents the union of locales supported by the Java runtime and by installed [NumberFormatProvider](http://docs.google.com/java/text/spi/NumberFormatProvider.html) implementations. It must contain at least a Locale instance equal to [Locale.US](http://docs.google.com/java/util/Locale.html#US).

**Returns:**An array of locales for which localized NumberFormat instances are available.

### hashCode

public int **hashCode**()

Overrides hashCode

**Overrides:**[hashCode](http://docs.google.com/java/lang/Object.html#hashCode()) in class [Object](http://docs.google.com/java/lang/Object.html) **Returns:**a hash code value for this object.**See Also:**[Object.equals(java.lang.Object)](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [Hashtable](http://docs.google.com/java/util/Hashtable.html)

### equals

public boolean **equals**([Object](http://docs.google.com/java/lang/Object.html) obj)

Overrides equals

**Overrides:**[equals](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)) in class [Object](http://docs.google.com/java/lang/Object.html) **Parameters:**obj - the reference object with which to compare. **Returns:**true if this object is the same as the obj argument; false otherwise.**See Also:**[Object.hashCode()](http://docs.google.com/java/lang/Object.html#hashCode()), [Hashtable](http://docs.google.com/java/util/Hashtable.html)

### clone

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

Overrides Cloneable

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

### isGroupingUsed

public boolean **isGroupingUsed**()

Returns true if grouping is used in this format. For example, in the English locale, with grouping on, the number 1234567 might be formatted as "1,234,567". The grouping separator as well as the size of each group is locale dependant and is determined by sub-classes of NumberFormat.

**See Also:**[setGroupingUsed(boolean)](http://docs.google.com/java/text/NumberFormat.html#setGroupingUsed(boolean))

### setGroupingUsed

public void **setGroupingUsed**(boolean newValue)

Set whether or not grouping will be used in this format.

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

### getMaximumIntegerDigits

public int **getMaximumIntegerDigits**()

Returns the maximum number of digits allowed in the integer portion of a number.

**See Also:**[setMaximumIntegerDigits(int)](http://docs.google.com/java/text/NumberFormat.html#setMaximumIntegerDigits(int))

### setMaximumIntegerDigits

public void **setMaximumIntegerDigits**(int newValue)

Sets the maximum number of digits allowed in the integer portion of a number. maximumIntegerDigits must be >= minimumIntegerDigits. If the new value for maximumIntegerDigits is less than the current value of minimumIntegerDigits, then minimumIntegerDigits will also be set to the new value.

**Parameters:**newValue - the maximum number of integer digits to be shown; if less than zero, then zero is used. The concrete subclass may enforce an upper limit to this value appropriate to the numeric type being formatted.**See Also:**[getMaximumIntegerDigits()](http://docs.google.com/java/text/NumberFormat.html#getMaximumIntegerDigits())

### getMinimumIntegerDigits

public int **getMinimumIntegerDigits**()

Returns the minimum number of digits allowed in the integer portion of a number.

**See Also:**[setMinimumIntegerDigits(int)](http://docs.google.com/java/text/NumberFormat.html#setMinimumIntegerDigits(int))

### setMinimumIntegerDigits

public void **setMinimumIntegerDigits**(int newValue)

Sets the minimum number of digits allowed in the integer portion of a number. minimumIntegerDigits must be <= maximumIntegerDigits. If the new value for minimumIntegerDigits exceeds the current value of maximumIntegerDigits, then maximumIntegerDigits will also be set to the new value

**Parameters:**newValue - the minimum number of integer digits to be shown; if less than zero, then zero is used. The concrete subclass may enforce an upper limit to this value appropriate to the numeric type being formatted.**See Also:**[getMinimumIntegerDigits()](http://docs.google.com/java/text/NumberFormat.html#getMinimumIntegerDigits())

### getMaximumFractionDigits

public int **getMaximumFractionDigits**()

Returns the maximum number of digits allowed in the fraction portion of a number.

**See Also:**[setMaximumFractionDigits(int)](http://docs.google.com/java/text/NumberFormat.html#setMaximumFractionDigits(int))

### setMaximumFractionDigits

public void **setMaximumFractionDigits**(int newValue)

Sets the maximum number of digits allowed in the fraction portion of a number. maximumFractionDigits must be >= minimumFractionDigits. If the new value for maximumFractionDigits is less than the current value of minimumFractionDigits, then minimumFractionDigits will also be set to the new value.

**Parameters:**newValue - the maximum number of fraction digits to be shown; if less than zero, then zero is used. The concrete subclass may enforce an upper limit to this value appropriate to the numeric type being formatted.**See Also:**[getMaximumFractionDigits()](http://docs.google.com/java/text/NumberFormat.html#getMaximumFractionDigits())

### getMinimumFractionDigits

public int **getMinimumFractionDigits**()

Returns the minimum number of digits allowed in the fraction portion of a number.

**See Also:**[setMinimumFractionDigits(int)](http://docs.google.com/java/text/NumberFormat.html#setMinimumFractionDigits(int))

### setMinimumFractionDigits

public void **setMinimumFractionDigits**(int newValue)

Sets the minimum number of digits allowed in the fraction portion of a number. minimumFractionDigits must be <= maximumFractionDigits. If the new value for minimumFractionDigits exceeds the current value of maximumFractionDigits, then maximumIntegerDigits will also be set to the new value

**Parameters:**newValue - the minimum number of fraction digits to be shown; if less than zero, then zero is used. The concrete subclass may enforce an upper limit to this value appropriate to the numeric type being formatted.**See Also:**[getMinimumFractionDigits()](http://docs.google.com/java/text/NumberFormat.html#getMinimumFractionDigits())

### getCurrency

public [Currency](http://docs.google.com/java/util/Currency.html) **getCurrency**()

Gets the currency used by this number format when formatting currency values. The initial value is derived in a locale dependent way. The returned value may be null if no valid currency could be determined and no currency has been set using [setCurrency](http://docs.google.com/java/text/NumberFormat.html#setCurrency(java.util.Currency)).

The default implementation throws UnsupportedOperationException.

**Returns:**the currency used by this number format, or null **Throws:** [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if the number format class doesn't implement currency formatting**Since:** 1.4

### setCurrency

public void **setCurrency**([Currency](http://docs.google.com/java/util/Currency.html) currency)

Sets the currency used by this number format when formatting currency values. This does not update the minimum or maximum number of fraction digits used by the number format.

The default implementation throws UnsupportedOperationException.

**Parameters:**currency - the new currency to be used by this number format **Throws:** [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if the number format class doesn't implement currency formatting [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if currency is null**Since:** 1.4

### getRoundingMode

public [RoundingMode](http://docs.google.com/java/math/RoundingMode.html) **getRoundingMode**()

Gets the [RoundingMode](http://docs.google.com/java/math/RoundingMode.html) used in this NumberFormat. The default implementation of this method in NumberFormat always throws [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html). Subclasses which handle different rounding modes should override this method.

**Returns:**The RoundingMode used for this NumberFormat. **Throws:** [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - The default implementation always throws this exception**Since:** 1.6 **See Also:**[setRoundingMode(RoundingMode)](http://docs.google.com/java/text/NumberFormat.html#setRoundingMode(java.math.RoundingMode))

### setRoundingMode

public void **setRoundingMode**([RoundingMode](http://docs.google.com/java/math/RoundingMode.html) roundingMode)

Sets the [RoundingMode](http://docs.google.com/java/math/RoundingMode.html) used in this NumberFormat. The default implementation of this method in NumberFormat always throws [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html). Subclasses which handle different rounding modes should override this method.

**Parameters:**roundingMode - The RoundingMode to be used **Throws:** [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - The default implementation always throws this exception [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if roundingMode is null**Since:** 1.6 **See Also:**[getRoundingMode()](http://docs.google.com/java/text/NumberFormat.html#getRoundingMode())

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/NumberFormat.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/Normalizer.Form.html)   [**NEXT CLASS**](http://docs.google.com/java/text/NumberFormat.Field.html) | [**FRAMES**](http://docs.google.com/index.html?java/text/NumberFormat.html)    [**NO FRAMES**](http://docs.google.com/NumberFormat.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#2et92p0) | [FIELD](#tyjcwt) | [CONSTR](#3dy6vkm) | [METHOD](#1t3h5sf) | DETAIL: [FIELD](#17dp8vu) | [CONSTR](#lnxbz9) | [METHOD](#1ksv4uv) |

[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).