| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/SimpleDateFormat.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/RuleBasedCollator.html)   [**NEXT CLASS**](http://docs.google.com/java/text/StringCharacterIterator.html) | [**FRAMES**](http://docs.google.com/index.html?java/text/SimpleDateFormat.html)    [**NO FRAMES**](http://docs.google.com/SimpleDateFormat.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#26in1rg) | [FIELD](#35nkun2) | [CONSTR](#1ksv4uv) | [METHOD](#44sinio) | DETAIL: FIELD | [CONSTR](#1y810tw) | [METHOD](#2bn6wsx) |

## **java.text**

Class SimpleDateFormat

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

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

public class **SimpleDateFormat**extends [DateFormat](http://docs.google.com/java/text/DateFormat.html)

SimpleDateFormat is a concrete class for formatting and parsing dates in a locale-sensitive manner. It allows for formatting (date -> text), parsing (text -> date), and normalization.

SimpleDateFormat allows you to start by choosing any user-defined patterns for date-time formatting. However, you are encouraged to create a date-time formatter with either getTimeInstance, getDateInstance, or getDateTimeInstance in DateFormat. Each of these class methods can return a date/time formatter initialized with a default format pattern. You may modify the format pattern using the applyPattern methods as desired. For more information on using these methods, see [DateFormat](http://docs.google.com/java/text/DateFormat.html).

#### Date and Time Patterns

Date and time formats are specified by *date and time pattern* strings. Within date and time pattern strings, unquoted letters from 'A' to 'Z' and from 'a' to 'z' are interpreted as pattern letters representing the components of a date or time string. Text can be quoted using single quotes (') to avoid interpretation. "''" represents a single quote. All other characters are not interpreted; they're simply copied into the output string during formatting or matched against the input string during parsing.

The following pattern letters are defined (all other characters from 'A' to 'Z' and from 'a' to 'z' are reserved):

| Letter | Date or Time Component | Presentation | Examples |
| --- | --- | --- | --- |
| G | Era designator | [Text](#3znysh7) | AD |
| y | Year | [Year](#tyjcwt) | 1996; 96 |
| M | Month in year | [Month](#3dy6vkm) | July; Jul; 07 |
| w | Week in year | [Number](#2et92p0) | 27 |
| W | Week in month | [Number](#2et92p0) | 2 |
| D | Day in year | [Number](#2et92p0) | 189 |
| d | Day in month | [Number](#2et92p0) | 10 |
| F | Day of week in month | [Number](#2et92p0) | 2 |
| E | Day in week | [Text](#3znysh7) | Tuesday; Tue |
| a | Am/pm marker | [Text](#3znysh7) | PM |
| H | Hour in day (0-23) | [Number](#2et92p0) | 0 |
| k | Hour in day (1-24) | [Number](#2et92p0) | 24 |
| K | Hour in am/pm (0-11) | [Number](#2et92p0) | 0 |
| h | Hour in am/pm (1-12) | [Number](#2et92p0) | 12 |
| m | Minute in hour | [Number](#2et92p0) | 30 |
| s | Second in minute | [Number](#2et92p0) | 55 |
| S | Millisecond | [Number](#2et92p0) | 978 |
| z | Time zone | [General time zone](#1t3h5sf) | Pacific Standard Time; PST; GMT-08:00 |
| Z | Time zone | [RFC 822 time zone](#2s8eyo1) | -0800 |

Pattern letters are usually repeated, as their number determines the exact presentation:

* **Text:** For formatting, if the number of pattern letters is 4 or more, the full form is used; otherwise a short or abbreviated form is used if available. For parsing, both forms are accepted, independent of the number of pattern letters.

* **Number:** For formatting, the number of pattern letters is the minimum number of digits, and shorter numbers are zero-padded to this amount. For parsing, the number of pattern letters is ignored unless it's needed to separate two adjacent fields.

* **Year:** If the formatter's [Calendar](http://docs.google.com/java/text/DateFormat.html#getCalendar()) is the Gregorian calendar, the following rules are applied.
  + For formatting, if the number of pattern letters is 2, the year is truncated to 2 digits; otherwise it is interpreted as a [number](#2et92p0).
  + For parsing, if the number of pattern letters is more than 2, the year is interpreted literally, regardless of the number of digits. So using the pattern "MM/dd/yyyy", "01/11/12" parses to Jan 11, 12 A.D.
  + For parsing with the abbreviated year pattern ("y" or "yy"), SimpleDateFormat must interpret the abbreviated year relative to some century. It does this by adjusting dates to be within 80 years before and 20 years after the time the SimpleDateFormat instance is created. For example, using a pattern of "MM/dd/yy" and a SimpleDateFormat instance created on Jan 1, 1997, the string "01/11/12" would be interpreted as Jan 11, 2012 while the string "05/04/64" would be interpreted as May 4, 1964. During parsing, only strings consisting of exactly two digits, as defined by [Character.isDigit(char)](http://docs.google.com/java/lang/Character.html#isDigit(char)), will be parsed into the default century. Any other numeric string, such as a one digit string, a three or more digit string, or a two digit string that isn't all digits (for example, "-1"), is interpreted literally. So "01/02/3" or "01/02/003" are parsed, using the same pattern, as Jan 2, 3 AD. Likewise, "01/02/-3" is parsed as Jan 2, 4 BC.

Otherwise, calendar system specific forms are applied. For both formatting and parsing, if the number of pattern letters is 4 or more, a calendar specific [long form](http://docs.google.com/java/util/Calendar.html#LONG) is used. Otherwise, a calendar specific [short or abbreviated form](http://docs.google.com/java/util/Calendar.html#SHORT) is used.

* **Month:** If the number of pattern letters is 3 or more, the month is interpreted as [text](#3znysh7); otherwise, it is interpreted as a [number](#2et92p0).

* **General time zone:** Time zones are interpreted as [text](#3znysh7) if they have names. For time zones representing a GMT offset value, the following syntax is used:  
   *GMTOffsetTimeZone:*  
   GMT *Sign* *Hours* : *Minutes*  
   *Sign:* one of  
   + -  
   *Hours:*  
   *Digit*  
   *Digit* *Digit*  
   *Minutes:*  
   *Digit* *Digit*  
   *Digit:* one of  
   0 1 2 3 4 5 6 7 8 9*Hours* must be between 0 and 23, and *Minutes* must be between 00 and 59. The format is locale independent and digits must be taken from the Basic Latin block of the Unicode standard.  
  For parsing, [RFC 822 time zones](#2s8eyo1) are also accepted.

* **RFC 822 time zone:** For formatting, the RFC 822 4-digit time zone format is used:  
   *RFC822TimeZone:*  
   *Sign* *TwoDigitHours* *Minutes*  
   *TwoDigitHours:*  
   *Digit DigitTwoDigitHours* must be between 00 and 23. Other definitions are as for [general time zones](#1t3h5sf).  
  For parsing, [general time zones](#1t3h5sf) are also accepted.

SimpleDateFormat also supports *localized date and time pattern* strings. In these strings, the pattern letters described above may be replaced with other, locale dependent, pattern letters. SimpleDateFormat does not deal with the localization of text other than the pattern letters; that's up to the client of the class.

#### Examples

The following examples show how date and time patterns are interpreted in the U.S. locale. The given date and time are 2001-07-04 12:08:56 local time in the U.S. Pacific Time time zone.

| Date and Time Pattern | Result |
| --- | --- |
| "yyyy.MM.dd G 'at' HH:mm:ss z" | 2001.07.04 AD at 12:08:56 PDT |
| "EEE, MMM d, ''yy" | Wed, Jul 4, '01 |
| "h:mm a" | 12:08 PM |
| "hh 'o''clock' a, zzzz" | 12 o'clock PM, Pacific Daylight Time |
| "K:mm a, z" | 0:08 PM, PDT |
| "yyyyy.MMMMM.dd GGG hh:mm aaa" | 02001.July.04 AD 12:08 PM |
| "EEE, d MMM yyyy HH:mm:ss Z" | Wed, 4 Jul 2001 12:08:56 -0700 |
| "yyMMddHHmmssZ" | 010704120856-0700 |
| "yyyy-MM-dd'T'HH:mm:ss.SSSZ" | 2001-07-04T12:08:56.235-0700 |

#### Synchronization

Date formats are 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:**[Java Tutorial](http://java.sun.com/docs/books/tutorial/i18n/format/simpleDateFormat.html), [Calendar](http://docs.google.com/java/util/Calendar.html), [TimeZone](http://docs.google.com/java/util/TimeZone.html), [DateFormat](http://docs.google.com/java/text/DateFormat.html), [DateFormatSymbols](http://docs.google.com/java/text/DateFormatSymbols.html), [Serialized Form](http://docs.google.com/serialized-form.html#java.text.SimpleDateFormat)

| **Nested Class Summary** | |
| --- | --- |

| **Nested classes/interfaces inherited from class java.text.**[**DateFormat**](http://docs.google.com/java/text/DateFormat.html) |
| --- |
| [DateFormat.Field](http://docs.google.com/java/text/DateFormat.Field.html) |

| **Field Summary** | |
| --- | --- |

| **Fields inherited from class java.text.**[**DateFormat**](http://docs.google.com/java/text/DateFormat.html) |
| --- |
| [AM\_PM\_FIELD](http://docs.google.com/java/text/DateFormat.html#AM_PM_FIELD), [calendar](http://docs.google.com/java/text/DateFormat.html#calendar), [DATE\_FIELD](http://docs.google.com/java/text/DateFormat.html#DATE_FIELD), [DAY\_OF\_WEEK\_FIELD](http://docs.google.com/java/text/DateFormat.html#DAY_OF_WEEK_FIELD), [DAY\_OF\_WEEK\_IN\_MONTH\_FIELD](http://docs.google.com/java/text/DateFormat.html#DAY_OF_WEEK_IN_MONTH_FIELD), [DAY\_OF\_YEAR\_FIELD](http://docs.google.com/java/text/DateFormat.html#DAY_OF_YEAR_FIELD), [DEFAULT](http://docs.google.com/java/text/DateFormat.html#DEFAULT), [ERA\_FIELD](http://docs.google.com/java/text/DateFormat.html#ERA_FIELD), [FULL](http://docs.google.com/java/text/DateFormat.html#FULL), [HOUR\_OF\_DAY0\_FIELD](http://docs.google.com/java/text/DateFormat.html#HOUR_OF_DAY0_FIELD), [HOUR\_OF\_DAY1\_FIELD](http://docs.google.com/java/text/DateFormat.html#HOUR_OF_DAY1_FIELD), [HOUR0\_FIELD](http://docs.google.com/java/text/DateFormat.html#HOUR0_FIELD), [HOUR1\_FIELD](http://docs.google.com/java/text/DateFormat.html#HOUR1_FIELD), [LONG](http://docs.google.com/java/text/DateFormat.html#LONG), [MEDIUM](http://docs.google.com/java/text/DateFormat.html#MEDIUM), [MILLISECOND\_FIELD](http://docs.google.com/java/text/DateFormat.html#MILLISECOND_FIELD), [MINUTE\_FIELD](http://docs.google.com/java/text/DateFormat.html#MINUTE_FIELD), [MONTH\_FIELD](http://docs.google.com/java/text/DateFormat.html#MONTH_FIELD), [numberFormat](http://docs.google.com/java/text/DateFormat.html#numberFormat), [SECOND\_FIELD](http://docs.google.com/java/text/DateFormat.html#SECOND_FIELD), [SHORT](http://docs.google.com/java/text/DateFormat.html#SHORT), [TIMEZONE\_FIELD](http://docs.google.com/java/text/DateFormat.html#TIMEZONE_FIELD), [WEEK\_OF\_MONTH\_FIELD](http://docs.google.com/java/text/DateFormat.html#WEEK_OF_MONTH_FIELD), [WEEK\_OF\_YEAR\_FIELD](http://docs.google.com/java/text/DateFormat.html#WEEK_OF_YEAR_FIELD), [YEAR\_FIELD](http://docs.google.com/java/text/DateFormat.html#YEAR_FIELD) |

| **Constructor Summary** | |
| --- | --- |
| [**SimpleDateFormat**](http://docs.google.com/java/text/SimpleDateFormat.html#SimpleDateFormat())()            Constructs a SimpleDateFormat using the default pattern and date format symbols for the default locale. |
| [**SimpleDateFormat**](http://docs.google.com/java/text/SimpleDateFormat.html#SimpleDateFormat(java.lang.String))([String](http://docs.google.com/java/lang/String.html) pattern)            Constructs a SimpleDateFormat using the given pattern and the default date format symbols for the default locale. |
| [**SimpleDateFormat**](http://docs.google.com/java/text/SimpleDateFormat.html#SimpleDateFormat(java.lang.String,%20java.text.DateFormatSymbols))([String](http://docs.google.com/java/lang/String.html) pattern, [DateFormatSymbols](http://docs.google.com/java/text/DateFormatSymbols.html) formatSymbols)            Constructs a SimpleDateFormat using the given pattern and date format symbols. |
| [**SimpleDateFormat**](http://docs.google.com/java/text/SimpleDateFormat.html#SimpleDateFormat(java.lang.String,%20java.util.Locale))([String](http://docs.google.com/java/lang/String.html) pattern, [Locale](http://docs.google.com/java/util/Locale.html) locale)            Constructs a SimpleDateFormat using the given pattern and the default date format symbols for the given locale. |

| **Method Summary** | |
| --- | --- |
| void | [**applyLocalizedPattern**](http://docs.google.com/java/text/SimpleDateFormat.html#applyLocalizedPattern(java.lang.String))([String](http://docs.google.com/java/lang/String.html) pattern)            Applies the given localized pattern string to this date format. |
| void | [**applyPattern**](http://docs.google.com/java/text/SimpleDateFormat.html#applyPattern(java.lang.String))([String](http://docs.google.com/java/lang/String.html) pattern)            Applies the given pattern string to this date format. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**clone**](http://docs.google.com/java/text/SimpleDateFormat.html#clone())()            Creates a copy of this SimpleDateFormat. |
| boolean | [**equals**](http://docs.google.com/java/text/SimpleDateFormat.html#equals(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) obj)            Compares the given object with this SimpleDateFormat for equality. |
| [StringBuffer](http://docs.google.com/java/lang/StringBuffer.html) | [**format**](http://docs.google.com/java/text/SimpleDateFormat.html#format(java.util.Date,%20java.lang.StringBuffer,%20java.text.FieldPosition))([Date](http://docs.google.com/java/util/Date.html) date, [StringBuffer](http://docs.google.com/java/lang/StringBuffer.html) toAppendTo, [FieldPosition](http://docs.google.com/java/text/FieldPosition.html) pos)            Formats the given Date into a date/time string and appends the result to the given StringBuffer. |
| [AttributedCharacterIterator](http://docs.google.com/java/text/AttributedCharacterIterator.html) | [**formatToCharacterIterator**](http://docs.google.com/java/text/SimpleDateFormat.html#formatToCharacterIterator(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) obj)            Formats an Object producing an AttributedCharacterIterator. |
| [Date](http://docs.google.com/java/util/Date.html) | [**get2DigitYearStart**](http://docs.google.com/java/text/SimpleDateFormat.html#get2DigitYearStart())()            Returns the beginning date of the 100-year period 2-digit years are interpreted as being within. |
| [DateFormatSymbols](http://docs.google.com/java/text/DateFormatSymbols.html) | [**getDateFormatSymbols**](http://docs.google.com/java/text/SimpleDateFormat.html#getDateFormatSymbols())()            Gets a copy of the date and time format symbols of this date format. |
| int | [**hashCode**](http://docs.google.com/java/text/SimpleDateFormat.html#hashCode())()            Returns the hash code value for this SimpleDateFormat object. |
| [Date](http://docs.google.com/java/util/Date.html) | [**parse**](http://docs.google.com/java/text/SimpleDateFormat.html#parse(java.lang.String,%20java.text.ParsePosition))([String](http://docs.google.com/java/lang/String.html) text, [ParsePosition](http://docs.google.com/java/text/ParsePosition.html) pos)            Parses text from a string to produce a Date. |
| void | [**set2DigitYearStart**](http://docs.google.com/java/text/SimpleDateFormat.html#set2DigitYearStart(java.util.Date))([Date](http://docs.google.com/java/util/Date.html) startDate)            Sets the 100-year period 2-digit years will be interpreted as being in to begin on the date the user specifies. |
| void | [**setDateFormatSymbols**](http://docs.google.com/java/text/SimpleDateFormat.html#setDateFormatSymbols(java.text.DateFormatSymbols))([DateFormatSymbols](http://docs.google.com/java/text/DateFormatSymbols.html) newFormatSymbols)            Sets the date and time format symbols of this date format. |
| [String](http://docs.google.com/java/lang/String.html) | [**toLocalizedPattern**](http://docs.google.com/java/text/SimpleDateFormat.html#toLocalizedPattern())()            Returns a localized pattern string describing this date format. |
| [String](http://docs.google.com/java/lang/String.html) | [**toPattern**](http://docs.google.com/java/text/SimpleDateFormat.html#toPattern())()            Returns a pattern string describing this date format. |

| **Methods inherited from class java.text.**[**DateFormat**](http://docs.google.com/java/text/DateFormat.html) |
| --- |
| [format](http://docs.google.com/java/text/DateFormat.html#format(java.util.Date)), [format](http://docs.google.com/java/text/DateFormat.html#format(java.lang.Object,%20java.lang.StringBuffer,%20java.text.FieldPosition)), [getAvailableLocales](http://docs.google.com/java/text/DateFormat.html#getAvailableLocales()), [getCalendar](http://docs.google.com/java/text/DateFormat.html#getCalendar()), [getDateInstance](http://docs.google.com/java/text/DateFormat.html#getDateInstance()), [getDateInstance](http://docs.google.com/java/text/DateFormat.html#getDateInstance(int)), [getDateInstance](http://docs.google.com/java/text/DateFormat.html#getDateInstance(int,%20java.util.Locale)), [getDateTimeInstance](http://docs.google.com/java/text/DateFormat.html#getDateTimeInstance()), [getDateTimeInstance](http://docs.google.com/java/text/DateFormat.html#getDateTimeInstance(int,%20int)), [getDateTimeInstance](http://docs.google.com/java/text/DateFormat.html#getDateTimeInstance(int,%20int,%20java.util.Locale)), [getInstance](http://docs.google.com/java/text/DateFormat.html#getInstance()), [getNumberFormat](http://docs.google.com/java/text/DateFormat.html#getNumberFormat()), [getTimeInstance](http://docs.google.com/java/text/DateFormat.html#getTimeInstance()), [getTimeInstance](http://docs.google.com/java/text/DateFormat.html#getTimeInstance(int)), [getTimeInstance](http://docs.google.com/java/text/DateFormat.html#getTimeInstance(int,%20java.util.Locale)), [getTimeZone](http://docs.google.com/java/text/DateFormat.html#getTimeZone()), [isLenient](http://docs.google.com/java/text/DateFormat.html#isLenient()), [parse](http://docs.google.com/java/text/DateFormat.html#parse(java.lang.String)), [parseObject](http://docs.google.com/java/text/DateFormat.html#parseObject(java.lang.String,%20java.text.ParsePosition)), [setCalendar](http://docs.google.com/java/text/DateFormat.html#setCalendar(java.util.Calendar)), [setLenient](http://docs.google.com/java/text/DateFormat.html#setLenient(boolean)), [setNumberFormat](http://docs.google.com/java/text/DateFormat.html#setNumberFormat(java.text.NumberFormat)), [setTimeZone](http://docs.google.com/java/text/DateFormat.html#setTimeZone(java.util.TimeZone)) |

| **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)), [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)) |

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

### SimpleDateFormat

public **SimpleDateFormat**()

Constructs a SimpleDateFormat using the default pattern and date format symbols for the default locale. **Note:** This constructor may not support all locales. For full coverage, use the factory methods in the [DateFormat](http://docs.google.com/java/text/DateFormat.html) class.

### SimpleDateFormat

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

Constructs a SimpleDateFormat using the given pattern and the default date format symbols for the default locale. **Note:** This constructor may not support all locales. For full coverage, use the factory methods in the [DateFormat](http://docs.google.com/java/text/DateFormat.html) class.

**Parameters:**pattern - the pattern describing the date and time format **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the given pattern is null [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the given pattern is invalid

### SimpleDateFormat

public **SimpleDateFormat**([String](http://docs.google.com/java/lang/String.html) pattern,  
 [Locale](http://docs.google.com/java/util/Locale.html) locale)

Constructs a SimpleDateFormat using the given pattern and the default date format symbols for the given locale. **Note:** This constructor may not support all locales. For full coverage, use the factory methods in the [DateFormat](http://docs.google.com/java/text/DateFormat.html) class.

**Parameters:**pattern - the pattern describing the date and time formatlocale - the locale whose date format symbols should be used **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the given pattern or locale is null [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the given pattern is invalid

### SimpleDateFormat

public **SimpleDateFormat**([String](http://docs.google.com/java/lang/String.html) pattern,  
 [DateFormatSymbols](http://docs.google.com/java/text/DateFormatSymbols.html) formatSymbols)

Constructs a SimpleDateFormat using the given pattern and date format symbols.

**Parameters:**pattern - the pattern describing the date and time formatformatSymbols - the date format symbols to be used for formatting **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the given pattern or formatSymbols is null [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the given pattern is invalid

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

### set2DigitYearStart

public void **set2DigitYearStart**([Date](http://docs.google.com/java/util/Date.html) startDate)

Sets the 100-year period 2-digit years will be interpreted as being in to begin on the date the user specifies.

**Parameters:**startDate - During parsing, two digit years will be placed in the range startDate to startDate + 100 years.**Since:** 1.2 **See Also:**[get2DigitYearStart()](http://docs.google.com/java/text/SimpleDateFormat.html#get2DigitYearStart())

### get2DigitYearStart

public [Date](http://docs.google.com/java/util/Date.html) **get2DigitYearStart**()

Returns the beginning date of the 100-year period 2-digit years are interpreted as being within.

**Returns:**the start of the 100-year period into which two digit years are parsed**Since:** 1.2 **See Also:**[set2DigitYearStart(java.util.Date)](http://docs.google.com/java/text/SimpleDateFormat.html#set2DigitYearStart(java.util.Date))

### format

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

Formats the given Date into a date/time string and appends the result to the given StringBuffer.

**Specified by:**[format](http://docs.google.com/java/text/DateFormat.html#format(java.util.Date,%20java.lang.StringBuffer,%20java.text.FieldPosition)) in class [DateFormat](http://docs.google.com/java/text/DateFormat.html) **Parameters:**date - the date-time value to be formatted into a date-time string.toAppendTo - where the new date-time text is to be appended.pos - the formatting position. On input: an alignment field, if desired. On output: the offsets of the alignment field. **Returns:**the formatted date-time string. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the given date is null

### formatToCharacterIterator

public [AttributedCharacterIterator](http://docs.google.com/java/text/AttributedCharacterIterator.html) **formatToCharacterIterator**([Object](http://docs.google.com/java/lang/Object.html) obj)

Formats an Object producing an AttributedCharacterIterator. You can use the returned AttributedCharacterIterator to build the resulting String, as well as to determine information about the resulting String.

Each attribute key of the AttributedCharacterIterator will be of type DateFormat.Field, with the corresponding attribute value being the same as the attribute key.

**Overrides:**[formatToCharacterIterator](http://docs.google.com/java/text/Format.html#formatToCharacterIterator(java.lang.Object)) in class [Format](http://docs.google.com/java/text/Format.html) **Parameters:**obj - The object to format **Returns:**AttributedCharacterIterator describing the formatted value. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if obj is null. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the Format cannot format the given object, or if the Format's pattern string is invalid.**Since:** 1.4

### parse

public [Date](http://docs.google.com/java/util/Date.html) **parse**([String](http://docs.google.com/java/lang/String.html) text,  
 [ParsePosition](http://docs.google.com/java/text/ParsePosition.html) pos)

Parses text from a string to produce a Date.

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 date 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.

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

### toPattern

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

Returns a pattern string describing this date format.

**Returns:**a pattern string describing this date format.

### toLocalizedPattern

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

Returns a localized pattern string describing this date format.

**Returns:**a localized pattern string describing this date format.

### applyPattern

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

Applies the given pattern string to this date format.

**Parameters:**pattern - the new date and time pattern for this date format **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the given pattern is null [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the given pattern is invalid

### applyLocalizedPattern

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

Applies the given localized pattern string to this date format.

**Parameters:**pattern - a String to be mapped to the new date and time format pattern for this format **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the given pattern is null [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the given pattern is invalid

### getDateFormatSymbols

public [DateFormatSymbols](http://docs.google.com/java/text/DateFormatSymbols.html) **getDateFormatSymbols**()

Gets a copy of the date and time format symbols of this date format.

**Returns:**the date and time format symbols of this date format**See Also:**[setDateFormatSymbols(java.text.DateFormatSymbols)](http://docs.google.com/java/text/SimpleDateFormat.html#setDateFormatSymbols(java.text.DateFormatSymbols))

### setDateFormatSymbols

public void **setDateFormatSymbols**([DateFormatSymbols](http://docs.google.com/java/text/DateFormatSymbols.html) newFormatSymbols)

Sets the date and time format symbols of this date format.

**Parameters:**newFormatSymbols - the new date and time format symbols **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the given newFormatSymbols is null**See Also:**[getDateFormatSymbols()](http://docs.google.com/java/text/SimpleDateFormat.html#getDateFormatSymbols())

### clone

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

Creates a copy of this SimpleDateFormat. This also clones the format's date format symbols.

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

### hashCode

public int **hashCode**()

Returns the hash code value for this SimpleDateFormat object.

**Overrides:**[hashCode](http://docs.google.com/java/text/DateFormat.html#hashCode()) in class [DateFormat](http://docs.google.com/java/text/DateFormat.html) **Returns:**the hash code value for this SimpleDateFormat 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)

Compares the given object with this SimpleDateFormat for equality.

**Overrides:**[equals](http://docs.google.com/java/text/DateFormat.html#equals(java.lang.Object)) in class [DateFormat](http://docs.google.com/java/text/DateFormat.html) **Parameters:**obj - the reference object with which to compare. **Returns:**true if the given object is equal to this SimpleDateFormat**See Also:**[Object.hashCode()](http://docs.google.com/java/lang/Object.html#hashCode()), [Hashtable](http://docs.google.com/java/util/Hashtable.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/SimpleDateFormat.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/RuleBasedCollator.html)   [**NEXT CLASS**](http://docs.google.com/java/text/StringCharacterIterator.html) | [**FRAMES**](http://docs.google.com/index.html?java/text/SimpleDateFormat.html)    [**NO FRAMES**](http://docs.google.com/SimpleDateFormat.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#26in1rg) | [FIELD](#35nkun2) | [CONSTR](#1ksv4uv) | [METHOD](#44sinio) | DETAIL: FIELD | [CONSTR](#1y810tw) | [METHOD](#2bn6wsx) |

[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).