| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Calendar.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/util/BitSet.html)   [**NEXT CLASS**](http://docs.google.com/java/util/Collection.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/Calendar.html)    [**NO FRAMES**](http://docs.google.com/Calendar.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3dy6vkm) | [CONSTR](#1t3h5sf) | [METHOD](#4d34og8) | DETAIL: [FIELD](#17dp8vu) | [CONSTR](#sqyw64) | [METHOD](#4bvk7pj) |

## **java.util**

Class Calendar

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

**All Implemented Interfaces:** [Serializable](http://docs.google.com/java/io/Serializable.html), [Cloneable](http://docs.google.com/java/lang/Cloneable.html), [Comparable](http://docs.google.com/java/lang/Comparable.html)<[Calendar](http://docs.google.com/java/util/Calendar.html)> **Direct Known Subclasses:** [GregorianCalendar](http://docs.google.com/java/util/GregorianCalendar.html)

public abstract class **Calendar**extends [Object](http://docs.google.com/java/lang/Object.html)implements [Serializable](http://docs.google.com/java/io/Serializable.html), [Cloneable](http://docs.google.com/java/lang/Cloneable.html), [Comparable](http://docs.google.com/java/lang/Comparable.html)<[Calendar](http://docs.google.com/java/util/Calendar.html)>

The Calendar class is an abstract class that provides methods for converting between a specific instant in time and a set of [calendar fields](http://docs.google.com/java/util/Calendar.html#fields) such as YEAR, MONTH, DAY\_OF\_MONTH, HOUR, and so on, and for manipulating the calendar fields, such as getting the date of the next week. An instant in time can be represented by a millisecond value that is an offset from the *Epoch*, January 1, 1970 00:00:00.000 GMT (Gregorian).

The class also provides additional fields and methods for implementing a concrete calendar system outside the package. Those fields and methods are defined as protected.

Like other locale-sensitive classes, Calendar provides a class method, getInstance, for getting a generally useful object of this type. Calendar's getInstance method returns a Calendar object whose calendar fields have been initialized with the current date and time:

Calendar rightNow = Calendar.getInstance();

A Calendar object can produce all the calendar field values needed to implement the date-time formatting for a particular language and calendar style (for example, Japanese-Gregorian, Japanese-Traditional). Calendar defines the range of values returned by certain calendar fields, as well as their meaning. For example, the first month of the calendar system has value MONTH == JANUARY for all calendars. Other values are defined by the concrete subclass, such as ERA. See individual field documentation and subclass documentation for details.

#### Getting and Setting Calendar Field Values

The calendar field values can be set by calling the set methods. Any field values set in a Calendar will not be interpreted until it needs to calculate its time value (milliseconds from the Epoch) or values of the calendar fields. Calling the get, getTimeInMillis, getTime, add and roll involves such calculation.

#### Leniency

Calendar has two modes for interpreting the calendar fields, *lenient* and *non-lenient*. When a Calendar is in lenient mode, it accepts a wider range of calendar field values than it produces. When a Calendar recomputes calendar field values for return by get(), all of the calendar fields are normalized. For example, a lenient GregorianCalendar interprets MONTH == JANUARY, DAY\_OF\_MONTH == 32 as February 1.

When a Calendar is in non-lenient mode, it throws an exception if there is any inconsistency in its calendar fields. For example, a GregorianCalendar always produces DAY\_OF\_MONTH values between 1 and the length of the month. A non-lenient GregorianCalendar throws an exception upon calculating its time or calendar field values if any out-of-range field value has been set.

#### First Week

Calendar defines a locale-specific seven day week using two parameters: the first day of the week and the minimal days in first week (from 1 to 7). These numbers are taken from the locale resource data when a Calendar is constructed. They may also be specified explicitly through the methods for setting their values.

When setting or getting the WEEK\_OF\_MONTH or WEEK\_OF\_YEAR fields, Calendar must determine the first week of the month or year as a reference point. The first week of a month or year is defined as the earliest seven day period beginning on getFirstDayOfWeek() and containing at least getMinimalDaysInFirstWeek() days of that month or year. Weeks numbered ..., -1, 0 precede the first week; weeks numbered 2, 3,... follow it. Note that the normalized numbering returned by get() may be different. For example, a specific Calendar subclass may designate the week before week 1 of a year as week *n* of the previous year.

#### Calendar Fields Resolution

When computing a date and time from the calendar fields, there may be insufficient information for the computation (such as only year and month with no day of month), or there may be inconsistent information (such as Tuesday, July 15, 1996 (Gregorian) -- July 15, 1996 is actually a Monday). Calendar will resolve calendar field values to determine the date and time in the following way.

If there is any conflict in calendar field values, Calendar gives priorities to calendar fields that have been set more recently. The following are the default combinations of the calendar fields. The most recent combination, as determined by the most recently set single field, will be used.

For the date fields:

YEAR + MONTH + DAY\_OF\_MONTH  
 YEAR + MONTH + WEEK\_OF\_MONTH + DAY\_OF\_WEEK  
 YEAR + MONTH + DAY\_OF\_WEEK\_IN\_MONTH + DAY\_OF\_WEEK  
 YEAR + DAY\_OF\_YEAR  
 YEAR + DAY\_OF\_WEEK + WEEK\_OF\_YEAR

For the time of day fields:

HOUR\_OF\_DAY  
 AM\_PM + HOUR

If there are any calendar fields whose values haven't been set in the selected field combination, Calendar uses their default values. The default value of each field may vary by concrete calendar systems. For example, in GregorianCalendar, the default of a field is the same as that of the start of the Epoch: i.e., YEAR = 1970, MONTH = JANUARY, DAY\_OF\_MONTH = 1, etc.

**Note:** There are certain possible ambiguities in interpretation of certain singular times, which are resolved in the following ways:

1. 23:59 is the last minute of the day and 00:00 is the first minute of the next day. Thus, 23:59 on Dec 31, 1999 < 00:00 on Jan 1, 2000 < 00:01 on Jan 1, 2000.
2. Although historically not precise, midnight also belongs to "am", and noon belongs to "pm", so on the same day, 12:00 am (midnight) < 12:01 am, and 12:00 pm (noon) < 12:01 pm

The date or time format strings are not part of the definition of a calendar, as those must be modifiable or overridable by the user at runtime. Use [DateFormat](http://docs.google.com/java/text/DateFormat.html) to format dates.

#### Field Manipulation

The calendar fields can be changed using three methods: set(), add(), and roll().

**set(f, value)** changes calendar field f to value. In addition, it sets an internal member variable to indicate that calendar field f has been changed. Although calendar field f is changed immediately, the calendar's time value in milliseconds is not recomputed until the next call to get(), getTime(), getTimeInMillis(), add(), or roll() is made. Thus, multiple calls to set() do not trigger multiple, unnecessary computations. As a result of changing a calendar field using set(), other calendar fields may also change, depending on the calendar field, the calendar field value, and the calendar system. In addition, get(f) will not necessarily return value set by the call to the set method after the calendar fields have been recomputed. The specifics are determined by the concrete calendar class.

*Example*: Consider a GregorianCalendar originally set to August 31, 1999. Calling set(Calendar.MONTH, Calendar.SEPTEMBER) sets the date to September 31, 1999. This is a temporary internal representation that resolves to October 1, 1999 if getTime()is then called. However, a call to set(Calendar.DAY\_OF\_MONTH, 30) before the call to getTime() sets the date to September 30, 1999, since no recomputation occurs after set() itself.

**add(f, delta)** adds delta to field f. This is equivalent to calling set(f, get(f) + delta) with two adjustments:

**Add rule 1**. The value of field f after the call minus the value of field f before the call is delta, modulo any overflow that has occurred in field f. Overflow occurs when a field value exceeds its range and, as a result, the next larger field is incremented or decremented and the field value is adjusted back into its range.

**Add rule 2**. If a smaller field is expected to be invariant, but it is impossible for it to be equal to its prior value because of changes in its minimum or maximum after field f is changed or other constraints, such as time zone offset changes, then its value is adjusted to be as close as possible to its expected value. A smaller field represents a smaller unit of time. HOUR is a smaller field than DAY\_OF\_MONTH. No adjustment is made to smaller fields that are not expected to be invariant. The calendar system determines what fields are expected to be invariant.

In addition, unlike set(), add() forces an immediate recomputation of the calendar's milliseconds and all fields.

*Example*: Consider a GregorianCalendar originally set to August 31, 1999. Calling add(Calendar.MONTH, 13) sets the calendar to September 30, 2000. **Add rule 1** sets the MONTH field to September, since adding 13 months to August gives September of the next year. Since DAY\_OF\_MONTH cannot be 31 in September in a GregorianCalendar, **add rule 2** sets the DAY\_OF\_MONTH to 30, the closest possible value. Although it is a smaller field, DAY\_OF\_WEEK is not adjusted by rule 2, since it is expected to change when the month changes in a GregorianCalendar.

**roll(f, delta)** adds delta to field f without changing larger fields. This is equivalent to calling add(f, delta) with the following adjustment:

**Roll rule**. Larger fields are unchanged after the call. A larger field represents a larger unit of time. DAY\_OF\_MONTH is a larger field than HOUR.

*Example*: See [GregorianCalendar.roll(int, int)](http://docs.google.com/java/util/GregorianCalendar.html#roll(int,%20int)).

**Usage model**. To motivate the behavior of add() and roll(), consider a user interface component with increment and decrement buttons for the month, day, and year, and an underlying GregorianCalendar. If the interface reads January 31, 1999 and the user presses the month increment button, what should it read? If the underlying implementation uses set(), it might read March 3, 1999. A better result would be February 28, 1999. Furthermore, if the user presses the month increment button again, it should read March 31, 1999, not March 28, 1999. By saving the original date and using either add() or roll(), depending on whether larger fields should be affected, the user interface can behave as most users will intuitively expect.

**Since:** JDK1.1 **See Also:**[System.currentTimeMillis()](http://docs.google.com/java/lang/System.html#currentTimeMillis()), [Date](http://docs.google.com/java/util/Date.html), [GregorianCalendar](http://docs.google.com/java/util/GregorianCalendar.html), [TimeZone](http://docs.google.com/java/util/TimeZone.html), [DateFormat](http://docs.google.com/java/text/DateFormat.html), [Serialized Form](http://docs.google.com/serialized-form.html#java.util.Calendar)

| **Field Summary** | |
| --- | --- |
| static int | [**ALL\_STYLES**](http://docs.google.com/java/util/Calendar.html#ALL_STYLES)            A style specifier for [getDisplayNames](http://docs.google.com/java/util/Calendar.html#getDisplayNames(int,%20int,%20java.util.Locale)) indicating names in all styles, such as "January" and "Jan". |
| static int | [**AM**](http://docs.google.com/java/util/Calendar.html#AM)            Value of the [AM\_PM](http://docs.google.com/java/util/Calendar.html#AM_PM) field indicating the period of the day from midnight to just before noon. |
| static int | [**AM\_PM**](http://docs.google.com/java/util/Calendar.html#AM_PM)            Field number for get and set indicating whether the HOUR is before or after noon. |
| static int | [**APRIL**](http://docs.google.com/java/util/Calendar.html#APRIL)            Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the fourth month of the year in the Gregorian and Julian calendars. |
| protected  boolean | [**areFieldsSet**](http://docs.google.com/java/util/Calendar.html#areFieldsSet)            True if fields[] are in sync with the currently set time. |
| static int | [**AUGUST**](http://docs.google.com/java/util/Calendar.html#AUGUST)            Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the eighth month of the year in the Gregorian and Julian calendars. |
| static int | [**DATE**](http://docs.google.com/java/util/Calendar.html#DATE)            Field number for get and set indicating the day of the month. |
| static int | [**DAY\_OF\_MONTH**](http://docs.google.com/java/util/Calendar.html#DAY_OF_MONTH)            Field number for get and set indicating the day of the month. |
| static int | [**DAY\_OF\_WEEK**](http://docs.google.com/java/util/Calendar.html#DAY_OF_WEEK)            Field number for get and set indicating the day of the week. |
| static int | [**DAY\_OF\_WEEK\_IN\_MONTH**](http://docs.google.com/java/util/Calendar.html#DAY_OF_WEEK_IN_MONTH)            Field number for get and set indicating the ordinal number of the day of the week within the current month. |
| static int | [**DAY\_OF\_YEAR**](http://docs.google.com/java/util/Calendar.html#DAY_OF_YEAR)            Field number for get and set indicating the day number within the current year. |
| static int | [**DECEMBER**](http://docs.google.com/java/util/Calendar.html#DECEMBER)            Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the twelfth month of the year in the Gregorian and Julian calendars. |
| static int | [**DST\_OFFSET**](http://docs.google.com/java/util/Calendar.html#DST_OFFSET)            Field number for get and set indicating the daylight savings offset in milliseconds. |
| static int | [**ERA**](http://docs.google.com/java/util/Calendar.html#ERA)            Field number for get and set indicating the era, e.g., AD or BC in the Julian calendar. |
| static int | [**FEBRUARY**](http://docs.google.com/java/util/Calendar.html#FEBRUARY)            Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the second month of the year in the Gregorian and Julian calendars. |
| static int | [**FIELD\_COUNT**](http://docs.google.com/java/util/Calendar.html#FIELD_COUNT)            The number of distinct fields recognized by get and set. |
| protected  int[] | [**fields**](http://docs.google.com/java/util/Calendar.html#fields)            The calendar field values for the currently set time for this calendar. |
| static int | [**FRIDAY**](http://docs.google.com/java/util/Calendar.html#FRIDAY)            Value of the [DAY\_OF\_WEEK](http://docs.google.com/java/util/Calendar.html#DAY_OF_WEEK) field indicating Friday. |
| static int | [**HOUR**](http://docs.google.com/java/util/Calendar.html#HOUR)            Field number for get and set indicating the hour of the morning or afternoon. |
| static int | [**HOUR\_OF\_DAY**](http://docs.google.com/java/util/Calendar.html#HOUR_OF_DAY)            Field number for get and set indicating the hour of the day. |
| protected  boolean[] | [**isSet**](http://docs.google.com/java/util/Calendar.html#isSet)            The flags which tell if a specified calendar field for the calendar is set. |
| protected  boolean | [**isTimeSet**](http://docs.google.com/java/util/Calendar.html#isTimeSet)            True if then the value of time is valid. |
| static int | [**JANUARY**](http://docs.google.com/java/util/Calendar.html#JANUARY)            Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the first month of the year in the Gregorian and Julian calendars. |
| static int | [**JULY**](http://docs.google.com/java/util/Calendar.html#JULY)            Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the seventh month of the year in the Gregorian and Julian calendars. |
| static int | [**JUNE**](http://docs.google.com/java/util/Calendar.html#JUNE)            Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the sixth month of the year in the Gregorian and Julian calendars. |
| static int | [**LONG**](http://docs.google.com/java/util/Calendar.html#LONG)            A style specifier for [getDisplayName](http://docs.google.com/java/util/Calendar.html#getDisplayName(int,%20int,%20java.util.Locale)) and [getDisplayNames](http://docs.google.com/java/util/Calendar.html#getDisplayNames(int,%20int,%20java.util.Locale)) indicating a long name, such as "January". |
| static int | [**MARCH**](http://docs.google.com/java/util/Calendar.html#MARCH)            Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the third month of the year in the Gregorian and Julian calendars. |
| static int | [**MAY**](http://docs.google.com/java/util/Calendar.html#MAY)            Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the fifth month of the year in the Gregorian and Julian calendars. |
| static int | [**MILLISECOND**](http://docs.google.com/java/util/Calendar.html#MILLISECOND)            Field number for get and set indicating the millisecond within the second. |
| static int | [**MINUTE**](http://docs.google.com/java/util/Calendar.html#MINUTE)            Field number for get and set indicating the minute within the hour. |
| static int | [**MONDAY**](http://docs.google.com/java/util/Calendar.html#MONDAY)            Value of the [DAY\_OF\_WEEK](http://docs.google.com/java/util/Calendar.html#DAY_OF_WEEK) field indicating Monday. |
| static int | [**MONTH**](http://docs.google.com/java/util/Calendar.html#MONTH)            Field number for get and set indicating the month. |
| static int | [**NOVEMBER**](http://docs.google.com/java/util/Calendar.html#NOVEMBER)            Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the eleventh month of the year in the Gregorian and Julian calendars. |
| static int | [**OCTOBER**](http://docs.google.com/java/util/Calendar.html#OCTOBER)            Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the tenth month of the year in the Gregorian and Julian calendars. |
| static int | [**PM**](http://docs.google.com/java/util/Calendar.html#PM)            Value of the [AM\_PM](http://docs.google.com/java/util/Calendar.html#AM_PM) field indicating the period of the day from noon to just before midnight. |
| static int | [**SATURDAY**](http://docs.google.com/java/util/Calendar.html#SATURDAY)            Value of the [DAY\_OF\_WEEK](http://docs.google.com/java/util/Calendar.html#DAY_OF_WEEK) field indicating Saturday. |
| static int | [**SECOND**](http://docs.google.com/java/util/Calendar.html#SECOND)            Field number for get and set indicating the second within the minute. |
| static int | [**SEPTEMBER**](http://docs.google.com/java/util/Calendar.html#SEPTEMBER)            Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the ninth month of the year in the Gregorian and Julian calendars. |
| static int | [**SHORT**](http://docs.google.com/java/util/Calendar.html#SHORT)            A style specifier for [getDisplayName](http://docs.google.com/java/util/Calendar.html#getDisplayName(int,%20int,%20java.util.Locale)) and [getDisplayNames](http://docs.google.com/java/util/Calendar.html#getDisplayNames(int,%20int,%20java.util.Locale)) indicating a short name, such as "Jan". |
| static int | [**SUNDAY**](http://docs.google.com/java/util/Calendar.html#SUNDAY)            Value of the [DAY\_OF\_WEEK](http://docs.google.com/java/util/Calendar.html#DAY_OF_WEEK) field indicating Sunday. |
| static int | [**THURSDAY**](http://docs.google.com/java/util/Calendar.html#THURSDAY)            Value of the [DAY\_OF\_WEEK](http://docs.google.com/java/util/Calendar.html#DAY_OF_WEEK) field indicating Thursday. |
| protected  long | [**time**](http://docs.google.com/java/util/Calendar.html#time)            The currently set time for this calendar, expressed in milliseconds after January 1, 1970, 0:00:00 GMT. |
| static int | [**TUESDAY**](http://docs.google.com/java/util/Calendar.html#TUESDAY)            Value of the [DAY\_OF\_WEEK](http://docs.google.com/java/util/Calendar.html#DAY_OF_WEEK) field indicating Tuesday. |
| static int | [**UNDECIMBER**](http://docs.google.com/java/util/Calendar.html#UNDECIMBER)            Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the thirteenth month of the year. |
| static int | [**WEDNESDAY**](http://docs.google.com/java/util/Calendar.html#WEDNESDAY)            Value of the [DAY\_OF\_WEEK](http://docs.google.com/java/util/Calendar.html#DAY_OF_WEEK) field indicating Wednesday. |
| static int | [**WEEK\_OF\_MONTH**](http://docs.google.com/java/util/Calendar.html#WEEK_OF_MONTH)            Field number for get and set indicating the week number within the current month. |
| static int | [**WEEK\_OF\_YEAR**](http://docs.google.com/java/util/Calendar.html#WEEK_OF_YEAR)            Field number for get and set indicating the week number within the current year. |
| static int | [**YEAR**](http://docs.google.com/java/util/Calendar.html#YEAR)            Field number for get and set indicating the year. |
| static int | [**ZONE\_OFFSET**](http://docs.google.com/java/util/Calendar.html#ZONE_OFFSET)            Field number for get and set indicating the raw offset from GMT in milliseconds. |

| **Constructor Summary** | |
| --- | --- |
| protected | [**Calendar**](http://docs.google.com/java/util/Calendar.html#Calendar())()            Constructs a Calendar with the default time zone and locale. |
| protected | [**Calendar**](http://docs.google.com/java/util/Calendar.html#Calendar(java.util.TimeZone,%20java.util.Locale))([TimeZone](http://docs.google.com/java/util/TimeZone.html) zone, [Locale](http://docs.google.com/java/util/Locale.html) aLocale)            Constructs a calendar with the specified time zone and locale. |

| **Method Summary** | |
| --- | --- |
| abstract  void | [**add**](http://docs.google.com/java/util/Calendar.html#add(int,%20int))(int field, int amount)            Adds or subtracts the specified amount of time to the given calendar field, based on the calendar's rules. |
| boolean | [**after**](http://docs.google.com/java/util/Calendar.html#after(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) when)            Returns whether this Calendar represents a time after the time represented by the specified Object. |
| boolean | [**before**](http://docs.google.com/java/util/Calendar.html#before(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) when)            Returns whether this Calendar represents a time before the time represented by the specified Object. |
| void | [**clear**](http://docs.google.com/java/util/Calendar.html#clear())()            Sets all the calendar field values and the time value (millisecond offset from the [Epoch](#3znysh7)) of this Calendar undefined. |
| void | [**clear**](http://docs.google.com/java/util/Calendar.html#clear(int))(int field)            Sets the given calendar field value and the time value (millisecond offset from the [Epoch](#3znysh7)) of this Calendar undefined. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**clone**](http://docs.google.com/java/util/Calendar.html#clone())()            Creates and returns a copy of this object. |
| int | [**compareTo**](http://docs.google.com/java/util/Calendar.html#compareTo(java.util.Calendar))([Calendar](http://docs.google.com/java/util/Calendar.html) anotherCalendar)            Compares the time values (millisecond offsets from the [Epoch](#3znysh7)) represented by two Calendar objects. |
| protected  void | [**complete**](http://docs.google.com/java/util/Calendar.html#complete())()            Fills in any unset fields in the calendar fields. |
| protected abstract  void | [**computeFields**](http://docs.google.com/java/util/Calendar.html#computeFields())()            Converts the current millisecond time value [time](http://docs.google.com/java/util/Calendar.html#time) to calendar field values in [fields[]](http://docs.google.com/java/util/Calendar.html#fields). |
| protected abstract  void | [**computeTime**](http://docs.google.com/java/util/Calendar.html#computeTime())()            Converts the current calendar field values in [fields[]](http://docs.google.com/java/util/Calendar.html#fields) to the millisecond time value [time](http://docs.google.com/java/util/Calendar.html#time). |
| boolean | [**equals**](http://docs.google.com/java/util/Calendar.html#equals(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) obj)            Compares this Calendar to the specified Object. |
| int | [**get**](http://docs.google.com/java/util/Calendar.html#get(int))(int field)            Returns the value of the given calendar field. |
| int | [**getActualMaximum**](http://docs.google.com/java/util/Calendar.html#getActualMaximum(int))(int field)            Returns the maximum value that the specified calendar field could have, given the time value of this Calendar. |
| int | [**getActualMinimum**](http://docs.google.com/java/util/Calendar.html#getActualMinimum(int))(int field)            Returns the minimum value that the specified calendar field could have, given the time value of this Calendar. |
| static [Locale](http://docs.google.com/java/util/Locale.html)[] | [**getAvailableLocales**](http://docs.google.com/java/util/Calendar.html#getAvailableLocales())()            Returns an array of all locales for which the getInstance methods of this class can return localized instances. |
| [String](http://docs.google.com/java/lang/String.html) | [**getDisplayName**](http://docs.google.com/java/util/Calendar.html#getDisplayName(int,%20int,%20java.util.Locale))(int field, int style, [Locale](http://docs.google.com/java/util/Locale.html) locale)            Returns the string representation of the calendar field value in the given style and locale. |
| [Map](http://docs.google.com/java/util/Map.html)<[String](http://docs.google.com/java/lang/String.html),[Integer](http://docs.google.com/java/lang/Integer.html)> | [**getDisplayNames**](http://docs.google.com/java/util/Calendar.html#getDisplayNames(int,%20int,%20java.util.Locale))(int field, int style, [Locale](http://docs.google.com/java/util/Locale.html) locale)            Returns a Map containing all names of the calendar field in the given style and locale and their corresponding field values. |
| int | [**getFirstDayOfWeek**](http://docs.google.com/java/util/Calendar.html#getFirstDayOfWeek())()            Gets what the first day of the week is; e.g., SUNDAY in the U.S., MONDAY in France. |
| abstract  int | [**getGreatestMinimum**](http://docs.google.com/java/util/Calendar.html#getGreatestMinimum(int))(int field)            Returns the highest minimum value for the given calendar field of this Calendar instance. |
| static [Calendar](http://docs.google.com/java/util/Calendar.html) | [**getInstance**](http://docs.google.com/java/util/Calendar.html#getInstance())()            Gets a calendar using the default time zone and locale. |
| static [Calendar](http://docs.google.com/java/util/Calendar.html) | [**getInstance**](http://docs.google.com/java/util/Calendar.html#getInstance(java.util.Locale))([Locale](http://docs.google.com/java/util/Locale.html) aLocale)            Gets a calendar using the default time zone and specified locale. |
| static [Calendar](http://docs.google.com/java/util/Calendar.html) | [**getInstance**](http://docs.google.com/java/util/Calendar.html#getInstance(java.util.TimeZone))([TimeZone](http://docs.google.com/java/util/TimeZone.html) zone)            Gets a calendar using the specified time zone and default locale. |
| static [Calendar](http://docs.google.com/java/util/Calendar.html) | [**getInstance**](http://docs.google.com/java/util/Calendar.html#getInstance(java.util.TimeZone,%20java.util.Locale))([TimeZone](http://docs.google.com/java/util/TimeZone.html) zone, [Locale](http://docs.google.com/java/util/Locale.html) aLocale)            Gets a calendar with the specified time zone and locale. |
| abstract  int | [**getLeastMaximum**](http://docs.google.com/java/util/Calendar.html#getLeastMaximum(int))(int field)            Returns the lowest maximum value for the given calendar field of this Calendar instance. |
| abstract  int | [**getMaximum**](http://docs.google.com/java/util/Calendar.html#getMaximum(int))(int field)            Returns the maximum value for the given calendar field of this Calendar instance. |
| int | [**getMinimalDaysInFirstWeek**](http://docs.google.com/java/util/Calendar.html#getMinimalDaysInFirstWeek())()            Gets what the minimal days required in the first week of the year are; e.g., if the first week is defined as one that contains the first day of the first month of a year, this method returns 1. |
| abstract  int | [**getMinimum**](http://docs.google.com/java/util/Calendar.html#getMinimum(int))(int field)            Returns the minimum value for the given calendar field of this Calendar instance. |
| [Date](http://docs.google.com/java/util/Date.html) | [**getTime**](http://docs.google.com/java/util/Calendar.html#getTime())()            Returns a Date object representing this Calendar's time value (millisecond offset from the [Epoch](#3znysh7)"). |
| long | [**getTimeInMillis**](http://docs.google.com/java/util/Calendar.html#getTimeInMillis())()            Returns this Calendar's time value in milliseconds. |
| [TimeZone](http://docs.google.com/java/util/TimeZone.html) | [**getTimeZone**](http://docs.google.com/java/util/Calendar.html#getTimeZone())()            Gets the time zone. |
| int | [**hashCode**](http://docs.google.com/java/util/Calendar.html#hashCode())()            Returns a hash code for this calendar. |
| protected  int | [**internalGet**](http://docs.google.com/java/util/Calendar.html#internalGet(int))(int field)            Returns the value of the given calendar field. |
| boolean | [**isLenient**](http://docs.google.com/java/util/Calendar.html#isLenient())()            Tells whether date/time interpretation is to be lenient. |
| boolean | [**isSet**](http://docs.google.com/java/util/Calendar.html#isSet(int))(int field)            Determines if the given calendar field has a value set, including cases that the value has been set by internal fields calculations triggered by a get method call. |
| abstract  void | [**roll**](http://docs.google.com/java/util/Calendar.html#roll(int,%20boolean))(int field, boolean up)            Adds or subtracts (up/down) a single unit of time on the given time field without changing larger fields. |
| void | [**roll**](http://docs.google.com/java/util/Calendar.html#roll(int,%20int))(int field, int amount)            Adds the specified (signed) amount to the specified calendar field without changing larger fields. |
| void | [**set**](http://docs.google.com/java/util/Calendar.html#set(int,%20int))(int field, int value)            Sets the given calendar field to the given value. |
| void | [**set**](http://docs.google.com/java/util/Calendar.html#set(int,%20int,%20int))(int year, int month, int date)            Sets the values for the calendar fields YEAR, MONTH, and DAY\_OF\_MONTH. |
| void | [**set**](http://docs.google.com/java/util/Calendar.html#set(int,%20int,%20int,%20int,%20int))(int year, int month, int date, int hourOfDay, int minute)            Sets the values for the calendar fields YEAR, MONTH, DAY\_OF\_MONTH, HOUR\_OF\_DAY, and MINUTE. |
| void | [**set**](http://docs.google.com/java/util/Calendar.html#set(int,%20int,%20int,%20int,%20int,%20int))(int year, int month, int date, int hourOfDay, int minute, int second)            Sets the values for the fields YEAR, MONTH, DAY\_OF\_MONTH, HOUR, MINUTE, and SECOND. |
| void | [**setFirstDayOfWeek**](http://docs.google.com/java/util/Calendar.html#setFirstDayOfWeek(int))(int value)            Sets what the first day of the week is; e.g., SUNDAY in the U.S., MONDAY in France. |
| void | [**setLenient**](http://docs.google.com/java/util/Calendar.html#setLenient(boolean))(boolean lenient)            Specifies whether or not date/time interpretation is to be lenient. |
| void | [**setMinimalDaysInFirstWeek**](http://docs.google.com/java/util/Calendar.html#setMinimalDaysInFirstWeek(int))(int value)            Sets what the minimal days required in the first week of the year are; For example, if the first week is defined as one that contains the first day of the first month of a year, call this method with value 1. |
| void | [**setTime**](http://docs.google.com/java/util/Calendar.html#setTime(java.util.Date))([Date](http://docs.google.com/java/util/Date.html) date)            Sets this Calendar's time with the given Date. |
| void | [**setTimeInMillis**](http://docs.google.com/java/util/Calendar.html#setTimeInMillis(long))(long millis)            Sets this Calendar's current time from the given long value. |
| void | [**setTimeZone**](http://docs.google.com/java/util/Calendar.html#setTimeZone(java.util.TimeZone))([TimeZone](http://docs.google.com/java/util/TimeZone.html) value)            Sets the time zone with the given time zone value. |
| [String](http://docs.google.com/java/lang/String.html) | [**toString**](http://docs.google.com/java/util/Calendar.html#toString())()            Return a string representation of this calendar. |

| **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()), [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** |
| --- |

### ERA

public static final int **ERA**

Field number for get and set indicating the era, e.g., AD or BC in the Julian calendar. This is a calendar-specific value; see subclass documentation.

**See Also:**[GregorianCalendar.AD](http://docs.google.com/java/util/GregorianCalendar.html#AD), [GregorianCalendar.BC](http://docs.google.com/java/util/GregorianCalendar.html#BC), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.ERA)

### YEAR

public static final int **YEAR**

Field number for get and set indicating the year. This is a calendar-specific value; see subclass documentation.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.YEAR)

### MONTH

public static final int **MONTH**

Field number for get and set indicating the month. This is a calendar-specific value. The first month of the year in the Gregorian and Julian calendars is JANUARY which is 0; the last depends on the number of months in a year.

**See Also:**[JANUARY](http://docs.google.com/java/util/Calendar.html#JANUARY), [FEBRUARY](http://docs.google.com/java/util/Calendar.html#FEBRUARY), [MARCH](http://docs.google.com/java/util/Calendar.html#MARCH), [APRIL](http://docs.google.com/java/util/Calendar.html#APRIL), [MAY](http://docs.google.com/java/util/Calendar.html#MAY), [JUNE](http://docs.google.com/java/util/Calendar.html#JUNE), [JULY](http://docs.google.com/java/util/Calendar.html#JULY), [AUGUST](http://docs.google.com/java/util/Calendar.html#AUGUST), [SEPTEMBER](http://docs.google.com/java/util/Calendar.html#SEPTEMBER), [OCTOBER](http://docs.google.com/java/util/Calendar.html#OCTOBER), [NOVEMBER](http://docs.google.com/java/util/Calendar.html#NOVEMBER), [DECEMBER](http://docs.google.com/java/util/Calendar.html#DECEMBER), [UNDECIMBER](http://docs.google.com/java/util/Calendar.html#UNDECIMBER), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.MONTH)

### WEEK\_OF\_YEAR

public static final int **WEEK\_OF\_YEAR**

Field number for get and set indicating the week number within the current year. The first week of the year, as defined by getFirstDayOfWeek() and getMinimalDaysInFirstWeek(), has value 1. Subclasses define the value of WEEK\_OF\_YEAR for days before the first week of the year.

**See Also:**[getFirstDayOfWeek()](http://docs.google.com/java/util/Calendar.html#getFirstDayOfWeek()), [getMinimalDaysInFirstWeek()](http://docs.google.com/java/util/Calendar.html#getMinimalDaysInFirstWeek()), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.WEEK_OF_YEAR)

### WEEK\_OF\_MONTH

public static final int **WEEK\_OF\_MONTH**

Field number for get and set indicating the week number within the current month. The first week of the month, as defined by getFirstDayOfWeek() and getMinimalDaysInFirstWeek(), has value 1. Subclasses define the value of WEEK\_OF\_MONTH for days before the first week of the month.

**See Also:**[getFirstDayOfWeek()](http://docs.google.com/java/util/Calendar.html#getFirstDayOfWeek()), [getMinimalDaysInFirstWeek()](http://docs.google.com/java/util/Calendar.html#getMinimalDaysInFirstWeek()), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.WEEK_OF_MONTH)

### DATE

public static final int **DATE**

Field number for get and set indicating the day of the month. This is a synonym for DAY\_OF\_MONTH. The first day of the month has value 1.

**See Also:**[DAY\_OF\_MONTH](http://docs.google.com/java/util/Calendar.html#DAY_OF_MONTH), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.DATE)

### DAY\_OF\_MONTH

public static final int **DAY\_OF\_MONTH**

Field number for get and set indicating the day of the month. This is a synonym for DATE. The first day of the month has value 1.

**See Also:**[DATE](http://docs.google.com/java/util/Calendar.html#DATE), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.DAY_OF_MONTH)

### DAY\_OF\_YEAR

public static final int **DAY\_OF\_YEAR**

Field number for get and set indicating the day number within the current year. The first day of the year has value 1.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.DAY_OF_YEAR)

### DAY\_OF\_WEEK

public static final int **DAY\_OF\_WEEK**

Field number for get and set indicating the day of the week. This field takes values SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, and SATURDAY.

**See Also:**[SUNDAY](http://docs.google.com/java/util/Calendar.html#SUNDAY), [MONDAY](http://docs.google.com/java/util/Calendar.html#MONDAY), [TUESDAY](http://docs.google.com/java/util/Calendar.html#TUESDAY), [WEDNESDAY](http://docs.google.com/java/util/Calendar.html#WEDNESDAY), [THURSDAY](http://docs.google.com/java/util/Calendar.html#THURSDAY), [FRIDAY](http://docs.google.com/java/util/Calendar.html#FRIDAY), [SATURDAY](http://docs.google.com/java/util/Calendar.html#SATURDAY), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.DAY_OF_WEEK)

### DAY\_OF\_WEEK\_IN\_MONTH

public static final int **DAY\_OF\_WEEK\_IN\_MONTH**

Field number for get and set indicating the ordinal number of the day of the week within the current month. Together with the DAY\_OF\_WEEK field, this uniquely specifies a day within a month. Unlike WEEK\_OF\_MONTH and WEEK\_OF\_YEAR, this field's value does *not* depend on getFirstDayOfWeek() or getMinimalDaysInFirstWeek(). DAY\_OF\_MONTH 1 through 7 always correspond to DAY\_OF\_WEEK\_IN\_MONTH 1; 8 through 14 correspond to DAY\_OF\_WEEK\_IN\_MONTH 2, and so on. DAY\_OF\_WEEK\_IN\_MONTH 0 indicates the week before DAY\_OF\_WEEK\_IN\_MONTH 1. Negative values count back from the end of the month, so the last Sunday of a month is specified as DAY\_OF\_WEEK = SUNDAY, DAY\_OF\_WEEK\_IN\_MONTH = -1. Because negative values count backward they will usually be aligned differently within the month than positive values. For example, if a month has 31 days, DAY\_OF\_WEEK\_IN\_MONTH -1 will overlap DAY\_OF\_WEEK\_IN\_MONTH 5 and the end of 4.

**See Also:**[DAY\_OF\_WEEK](http://docs.google.com/java/util/Calendar.html#DAY_OF_WEEK), [WEEK\_OF\_MONTH](http://docs.google.com/java/util/Calendar.html#WEEK_OF_MONTH), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.DAY_OF_WEEK_IN_MONTH)

### AM\_PM

public static final int **AM\_PM**

Field number for get and set indicating whether the HOUR is before or after noon. E.g., at 10:04:15.250 PM the AM\_PM is PM.

**See Also:**[AM](http://docs.google.com/java/util/Calendar.html#AM), [PM](http://docs.google.com/java/util/Calendar.html#PM), [HOUR](http://docs.google.com/java/util/Calendar.html#HOUR), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.AM_PM)

### HOUR

public static final int **HOUR**

Field number for get and set indicating the hour of the morning or afternoon. HOUR is used for the 12-hour clock (0 - 11). Noon and midnight are represented by 0, not by 12. E.g., at 10:04:15.250 PM the HOUR is 10.

**See Also:**[AM\_PM](http://docs.google.com/java/util/Calendar.html#AM_PM), [HOUR\_OF\_DAY](http://docs.google.com/java/util/Calendar.html#HOUR_OF_DAY), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.HOUR)

### HOUR\_OF\_DAY

public static final int **HOUR\_OF\_DAY**

Field number for get and set indicating the hour of the day. HOUR\_OF\_DAY is used for the 24-hour clock. E.g., at 10:04:15.250 PM the HOUR\_OF\_DAY is 22.

**See Also:**[HOUR](http://docs.google.com/java/util/Calendar.html#HOUR), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.HOUR_OF_DAY)

### MINUTE

public static final int **MINUTE**

Field number for get and set indicating the minute within the hour. E.g., at 10:04:15.250 PM the MINUTE is 4.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.MINUTE)

### SECOND

public static final int **SECOND**

Field number for get and set indicating the second within the minute. E.g., at 10:04:15.250 PM the SECOND is 15.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.SECOND)

### MILLISECOND

public static final int **MILLISECOND**

Field number for get and set indicating the millisecond within the second. E.g., at 10:04:15.250 PM the MILLISECOND is 250.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.MILLISECOND)

### ZONE\_OFFSET

public static final int **ZONE\_OFFSET**

Field number for get and set indicating the raw offset from GMT in milliseconds.

This field reflects the correct GMT offset value of the time zone of this Calendar if the TimeZone implementation subclass supports historical GMT offset changes.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.ZONE_OFFSET)

### DST\_OFFSET

public static final int **DST\_OFFSET**

Field number for get and set indicating the daylight savings offset in milliseconds.

This field reflects the correct daylight saving offset value of the time zone of this Calendar if the TimeZone implementation subclass supports historical Daylight Saving Time schedule changes.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.DST_OFFSET)

### FIELD\_COUNT

public static final int **FIELD\_COUNT**

The number of distinct fields recognized by get and set. Field numbers range from 0..FIELD\_COUNT-1.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.FIELD_COUNT)

### SUNDAY

public static final int **SUNDAY**

Value of the [DAY\_OF\_WEEK](http://docs.google.com/java/util/Calendar.html#DAY_OF_WEEK) field indicating Sunday.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.SUNDAY)

### MONDAY

public static final int **MONDAY**

Value of the [DAY\_OF\_WEEK](http://docs.google.com/java/util/Calendar.html#DAY_OF_WEEK) field indicating Monday.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.MONDAY)

### TUESDAY

public static final int **TUESDAY**

Value of the [DAY\_OF\_WEEK](http://docs.google.com/java/util/Calendar.html#DAY_OF_WEEK) field indicating Tuesday.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.TUESDAY)

### WEDNESDAY

public static final int **WEDNESDAY**

Value of the [DAY\_OF\_WEEK](http://docs.google.com/java/util/Calendar.html#DAY_OF_WEEK) field indicating Wednesday.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.WEDNESDAY)

### THURSDAY

public static final int **THURSDAY**

Value of the [DAY\_OF\_WEEK](http://docs.google.com/java/util/Calendar.html#DAY_OF_WEEK) field indicating Thursday.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.THURSDAY)

### FRIDAY

public static final int **FRIDAY**

Value of the [DAY\_OF\_WEEK](http://docs.google.com/java/util/Calendar.html#DAY_OF_WEEK) field indicating Friday.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.FRIDAY)

### SATURDAY

public static final int **SATURDAY**

Value of the [DAY\_OF\_WEEK](http://docs.google.com/java/util/Calendar.html#DAY_OF_WEEK) field indicating Saturday.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.SATURDAY)

### JANUARY

public static final int **JANUARY**

Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the first month of the year in the Gregorian and Julian calendars.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.JANUARY)

### FEBRUARY

public static final int **FEBRUARY**

Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the second month of the year in the Gregorian and Julian calendars.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.FEBRUARY)

### MARCH

public static final int **MARCH**

Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the third month of the year in the Gregorian and Julian calendars.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.MARCH)

### APRIL

public static final int **APRIL**

Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the fourth month of the year in the Gregorian and Julian calendars.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.APRIL)

### MAY

public static final int **MAY**

Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the fifth month of the year in the Gregorian and Julian calendars.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.MAY)

### JUNE

public static final int **JUNE**

Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the sixth month of the year in the Gregorian and Julian calendars.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.JUNE)

### JULY

public static final int **JULY**

Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the seventh month of the year in the Gregorian and Julian calendars.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.JULY)

### AUGUST

public static final int **AUGUST**

Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the eighth month of the year in the Gregorian and Julian calendars.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.AUGUST)

### SEPTEMBER

public static final int **SEPTEMBER**

Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the ninth month of the year in the Gregorian and Julian calendars.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.SEPTEMBER)

### OCTOBER

public static final int **OCTOBER**

Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the tenth month of the year in the Gregorian and Julian calendars.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.OCTOBER)

### NOVEMBER

public static final int **NOVEMBER**

Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the eleventh month of the year in the Gregorian and Julian calendars.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.NOVEMBER)

### DECEMBER

public static final int **DECEMBER**

Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the twelfth month of the year in the Gregorian and Julian calendars.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.DECEMBER)

### UNDECIMBER

public static final int **UNDECIMBER**

Value of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field indicating the thirteenth month of the year. Although GregorianCalendar does not use this value, lunar calendars do.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.UNDECIMBER)

### AM

public static final int **AM**

Value of the [AM\_PM](http://docs.google.com/java/util/Calendar.html#AM_PM) field indicating the period of the day from midnight to just before noon.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.AM)

### PM

public static final int **PM**

Value of the [AM\_PM](http://docs.google.com/java/util/Calendar.html#AM_PM) field indicating the period of the day from noon to just before midnight.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.PM)

### ALL\_STYLES

public static final int **ALL\_STYLES**

A style specifier for [getDisplayNames](http://docs.google.com/java/util/Calendar.html#getDisplayNames(int,%20int,%20java.util.Locale)) indicating names in all styles, such as "January" and "Jan".

**Since:** 1.6 **See Also:**[SHORT](http://docs.google.com/java/util/Calendar.html#SHORT), [LONG](http://docs.google.com/java/util/Calendar.html#LONG), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.ALL_STYLES)

### SHORT

public static final int **SHORT**

A style specifier for [getDisplayName](http://docs.google.com/java/util/Calendar.html#getDisplayName(int,%20int,%20java.util.Locale)) and [getDisplayNames](http://docs.google.com/java/util/Calendar.html#getDisplayNames(int,%20int,%20java.util.Locale)) indicating a short name, such as "Jan".

**Since:** 1.6 **See Also:**[LONG](http://docs.google.com/java/util/Calendar.html#LONG), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.SHORT)

### LONG

public static final int **LONG**

A style specifier for [getDisplayName](http://docs.google.com/java/util/Calendar.html#getDisplayName(int,%20int,%20java.util.Locale)) and [getDisplayNames](http://docs.google.com/java/util/Calendar.html#getDisplayNames(int,%20int,%20java.util.Locale)) indicating a long name, such as "January".

**Since:** 1.6 **See Also:**[SHORT](http://docs.google.com/java/util/Calendar.html#SHORT), [Constant Field Values](http://docs.google.com/constant-values.html#java.util.Calendar.LONG)

### fields

protected int[] **fields**

The calendar field values for the currently set time for this calendar. This is an array of FIELD\_COUNT integers, with index values ERA through DST\_OFFSET.

### isSet

protected boolean[] **isSet**

The flags which tell if a specified calendar field for the calendar is set. A new object has no fields set. After the first call to a method which generates the fields, they all remain set after that. This is an array of FIELD\_COUNT booleans, with index values ERA through DST\_OFFSET.

### time

protected long **time**

The currently set time for this calendar, expressed in milliseconds after January 1, 1970, 0:00:00 GMT.

**See Also:**[isTimeSet](http://docs.google.com/java/util/Calendar.html#isTimeSet)

### isTimeSet

protected boolean **isTimeSet**

True if then the value of time is valid. The time is made invalid by a change to an item of field[].

**See Also:**[time](http://docs.google.com/java/util/Calendar.html#time)

### areFieldsSet

protected boolean **areFieldsSet**

True if fields[] are in sync with the currently set time. If false, then the next attempt to get the value of a field will force a recomputation of all fields from the current value of time.

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

### Calendar

protected **Calendar**()

Constructs a Calendar with the default time zone and locale.

**See Also:**[TimeZone.getDefault()](http://docs.google.com/java/util/TimeZone.html#getDefault())

### Calendar

protected **Calendar**([TimeZone](http://docs.google.com/java/util/TimeZone.html) zone,  
 [Locale](http://docs.google.com/java/util/Locale.html) aLocale)

Constructs a calendar with the specified time zone and locale.

**Parameters:**zone - the time zone to useaLocale - the locale for the week data

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

### getInstance

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

Gets a calendar using the default time zone and locale. The Calendar returned is based on the current time in the default time zone with the default locale.

**Returns:**a Calendar.

### getInstance

public static [Calendar](http://docs.google.com/java/util/Calendar.html) **getInstance**([TimeZone](http://docs.google.com/java/util/TimeZone.html) zone)

Gets a calendar using the specified time zone and default locale. The Calendar returned is based on the current time in the given time zone with the default locale.

**Parameters:**zone - the time zone to use **Returns:**a Calendar.

### getInstance

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

Gets a calendar using the default time zone and specified locale. The Calendar returned is based on the current time in the default time zone with the given locale.

**Parameters:**aLocale - the locale for the week data **Returns:**a Calendar.

### getInstance

public static [Calendar](http://docs.google.com/java/util/Calendar.html) **getInstance**([TimeZone](http://docs.google.com/java/util/TimeZone.html) zone,  
 [Locale](http://docs.google.com/java/util/Locale.html) aLocale)

Gets a calendar with the specified time zone and locale. The Calendar returned is based on the current time in the given time zone with the given locale.

**Parameters:**zone - the time zone to useaLocale - the locale for the week data **Returns:**a Calendar.

### getAvailableLocales

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

Returns an array of all locales for which the getInstance methods of this class can return localized instances. The array returned 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 Calendar instances are available.

### computeTime

protected abstract void **computeTime**()

Converts the current calendar field values in [fields[]](http://docs.google.com/java/util/Calendar.html#fields) to the millisecond time value [time](http://docs.google.com/java/util/Calendar.html#time).

**See Also:**[complete()](http://docs.google.com/java/util/Calendar.html#complete()), [computeFields()](http://docs.google.com/java/util/Calendar.html#computeFields())

### computeFields

protected abstract void **computeFields**()

Converts the current millisecond time value [time](http://docs.google.com/java/util/Calendar.html#time) to calendar field values in [fields[]](http://docs.google.com/java/util/Calendar.html#fields). This allows you to sync up the calendar field values with a new time that is set for the calendar. The time is *not* recomputed first; to recompute the time, then the fields, call the [complete()](http://docs.google.com/java/util/Calendar.html#complete()) method.

**See Also:**[computeTime()](http://docs.google.com/java/util/Calendar.html#computeTime())

### getTime

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

Returns a Date object representing this Calendar's time value (millisecond offset from the [Epoch](#3znysh7)").

**Returns:**a Date representing the time value.**See Also:**[setTime(Date)](http://docs.google.com/java/util/Calendar.html#setTime(java.util.Date)), [getTimeInMillis()](http://docs.google.com/java/util/Calendar.html#getTimeInMillis())

### setTime

public final void **setTime**([Date](http://docs.google.com/java/util/Date.html) date)

Sets this Calendar's time with the given Date.

Note: Calling setTime() with Date(Long.MAX\_VALUE) or Date(Long.MIN\_VALUE) may yield incorrect field values from get().

**Parameters:**date - the given Date.**See Also:**[getTime()](http://docs.google.com/java/util/Calendar.html#getTime()), [setTimeInMillis(long)](http://docs.google.com/java/util/Calendar.html#setTimeInMillis(long))

### getTimeInMillis

public long **getTimeInMillis**()

Returns this Calendar's time value in milliseconds.

**Returns:**the current time as UTC milliseconds from the epoch.**See Also:**[getTime()](http://docs.google.com/java/util/Calendar.html#getTime()), [setTimeInMillis(long)](http://docs.google.com/java/util/Calendar.html#setTimeInMillis(long))

### setTimeInMillis

public void **setTimeInMillis**(long millis)

Sets this Calendar's current time from the given long value.

**Parameters:**millis - the new time in UTC milliseconds from the epoch.**See Also:**[setTime(Date)](http://docs.google.com/java/util/Calendar.html#setTime(java.util.Date)), [getTimeInMillis()](http://docs.google.com/java/util/Calendar.html#getTimeInMillis())

### get

public int **get**(int field)

Returns the value of the given calendar field. In lenient mode, all calendar fields are normalized. In non-lenient mode, all calendar fields are validated and this method throws an exception if any calendar fields have out-of-range values. The normalization and validation are handled by the [complete()](http://docs.google.com/java/util/Calendar.html#complete()) method, which process is calendar system dependent.

**Parameters:**field - the given calendar field. **Returns:**the value for the given calendar field. **Throws:** [ArrayIndexOutOfBoundsException](http://docs.google.com/java/lang/ArrayIndexOutOfBoundsException.html) - if the specified field is out of range (field < 0 || field >= FIELD\_COUNT).**See Also:**[set(int,int)](http://docs.google.com/java/util/Calendar.html#set(int,%20int)), [complete()](http://docs.google.com/java/util/Calendar.html#complete())

### internalGet

protected final int **internalGet**(int field)

Returns the value of the given calendar field. This method does not involve normalization or validation of the field value.

**Parameters:**field - the given calendar field. **Returns:**the value for the given calendar field.**See Also:**[get(int)](http://docs.google.com/java/util/Calendar.html#get(int))

### set

public void **set**(int field,  
 int value)

Sets the given calendar field to the given value. The value is not interpreted by this method regardless of the leniency mode.

**Parameters:**field - the given calendar field.value - the value to be set for the given calendar field. **Throws:** [ArrayIndexOutOfBoundsException](http://docs.google.com/java/lang/ArrayIndexOutOfBoundsException.html) - if the specified field is out of range (field < 0 || field >= FIELD\_COUNT). in non-lenient mode.**See Also:**[set(int,int,int)](http://docs.google.com/java/util/Calendar.html#set(int,%20int,%20int)), [set(int,int,int,int,int)](http://docs.google.com/java/util/Calendar.html#set(int,%20int,%20int,%20int,%20int)), [set(int,int,int,int,int,int)](http://docs.google.com/java/util/Calendar.html#set(int,%20int,%20int,%20int,%20int,%20int)), [get(int)](http://docs.google.com/java/util/Calendar.html#get(int))

### set

public final void **set**(int year,  
 int month,  
 int date)

Sets the values for the calendar fields YEAR, MONTH, and DAY\_OF\_MONTH. Previous values of other calendar fields are retained. If this is not desired, call [clear()](http://docs.google.com/java/util/Calendar.html#clear()) first.

**Parameters:**year - the value used to set the YEAR calendar field.month - the value used to set the MONTH calendar field. Month value is 0-based. e.g., 0 for January.date - the value used to set the DAY\_OF\_MONTH calendar field.**See Also:**[set(int,int)](http://docs.google.com/java/util/Calendar.html#set(int,%20int)), [set(int,int,int,int,int)](http://docs.google.com/java/util/Calendar.html#set(int,%20int,%20int,%20int,%20int)), [set(int,int,int,int,int,int)](http://docs.google.com/java/util/Calendar.html#set(int,%20int,%20int,%20int,%20int,%20int))

### set

public final void **set**(int year,  
 int month,  
 int date,  
 int hourOfDay,  
 int minute)

Sets the values for the calendar fields YEAR, MONTH, DAY\_OF\_MONTH, HOUR\_OF\_DAY, and MINUTE. Previous values of other fields are retained. If this is not desired, call [clear()](http://docs.google.com/java/util/Calendar.html#clear()) first.

**Parameters:**year - the value used to set the YEAR calendar field.month - the value used to set the MONTH calendar field. Month value is 0-based. e.g., 0 for January.date - the value used to set the DAY\_OF\_MONTH calendar field.hourOfDay - the value used to set the HOUR\_OF\_DAY calendar field.minute - the value used to set the MINUTE calendar field.**See Also:**[set(int,int)](http://docs.google.com/java/util/Calendar.html#set(int,%20int)), [set(int,int,int)](http://docs.google.com/java/util/Calendar.html#set(int,%20int,%20int)), [set(int,int,int,int,int,int)](http://docs.google.com/java/util/Calendar.html#set(int,%20int,%20int,%20int,%20int,%20int))

### set

public final void **set**(int year,  
 int month,  
 int date,  
 int hourOfDay,  
 int minute,  
 int second)

Sets the values for the fields YEAR, MONTH, DAY\_OF\_MONTH, HOUR, MINUTE, and SECOND. Previous values of other fields are retained. If this is not desired, call [clear()](http://docs.google.com/java/util/Calendar.html#clear()) first.

**Parameters:**year - the value used to set the YEAR calendar field.month - the value used to set the MONTH calendar field. Month value is 0-based. e.g., 0 for January.date - the value used to set the DAY\_OF\_MONTH calendar field.hourOfDay - the value used to set the HOUR\_OF\_DAY calendar field.minute - the value used to set the MINUTE calendar field.second - the value used to set the SECOND calendar field.**See Also:**[set(int,int)](http://docs.google.com/java/util/Calendar.html#set(int,%20int)), [set(int,int,int)](http://docs.google.com/java/util/Calendar.html#set(int,%20int,%20int)), [set(int,int,int,int,int)](http://docs.google.com/java/util/Calendar.html#set(int,%20int,%20int,%20int,%20int))

### clear

public final void **clear**()

Sets all the calendar field values and the time value (millisecond offset from the [Epoch](#3znysh7)) of this Calendar undefined. This means that [isSet()](http://docs.google.com/java/util/Calendar.html#isSet(int)) will return false for all the calendar fields, and the date and time calculations will treat the fields as if they had never been set. A Calendar implementation class may use its specific default field values for date/time calculations. For example, GregorianCalendar uses 1970 if the YEAR field value is undefined.

**See Also:**[clear(int)](http://docs.google.com/java/util/Calendar.html#clear(int))

### clear

public final void **clear**(int field)

Sets the given calendar field value and the time value (millisecond offset from the [Epoch](#3znysh7)) of this Calendar undefined. This means that [isSet(field)](http://docs.google.com/java/util/Calendar.html#isSet(int)) will return false, and the date and time calculations will treat the field as if it had never been set. A Calendar implementation class may use the field's specific default value for date and time calculations.

The [HOUR\_OF\_DAY](http://docs.google.com/java/util/Calendar.html#HOUR_OF_DAY), [HOUR](http://docs.google.com/java/util/Calendar.html#HOUR) and [AM\_PM](http://docs.google.com/java/util/Calendar.html#AM_PM) fields are handled independently and the [the resolution rule for the time of day](#tyjcwt) is applied. Clearing one of the fields doesn't reset the hour of day value of this Calendar. Use [set(Calendar.HOUR\_OF\_DAY, 0)](http://docs.google.com/java/util/Calendar.html#set(int,%20int)) to reset the hour value.

**Parameters:**field - the calendar field to be cleared.**See Also:**[clear()](http://docs.google.com/java/util/Calendar.html#clear())

### isSet

public final boolean **isSet**(int field)

Determines if the given calendar field has a value set, including cases that the value has been set by internal fields calculations triggered by a get method call.

**Returns:**true if the given calendar field has a value set; false otherwise.

### getDisplayName

public [String](http://docs.google.com/java/lang/String.html) **getDisplayName**(int field,  
 int style,  
 [Locale](http://docs.google.com/java/util/Locale.html) locale)

Returns the string representation of the calendar field value in the given style and locale. If no string representation is applicable, null is returned. This method calls [get(field)](http://docs.google.com/java/util/Calendar.html#get(int)) to get the calendar field value if the string representation is applicable to the given calendar field.

For example, if this Calendar is a GregorianCalendar and its date is 2005-01-01, then the string representation of the [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) field would be "January" in the long style in an English locale or "Jan" in the short style. However, no string representation would be available for the [DAY\_OF\_MONTH](http://docs.google.com/java/util/Calendar.html#DAY_OF_MONTH) field, and this method would return null.

The default implementation supports the calendar fields for which a [DateFormatSymbols](http://docs.google.com/java/text/DateFormatSymbols.html) has names in the given locale.

**Parameters:**field - the calendar field for which the string representation is returnedstyle - the style applied to the string representation; one of [SHORT](http://docs.google.com/java/util/Calendar.html#SHORT) or [LONG](http://docs.google.com/java/util/Calendar.html#LONG).locale - the locale for the string representation **Returns:**the string representation of the given field in the given style, or null if no string representation is applicable. **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if field or style is invalid, or if this Calendar is non-lenient and any of the calendar fields have invalid values [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if locale is null**Since:** 1.6

### getDisplayNames

public [Map](http://docs.google.com/java/util/Map.html)<[String](http://docs.google.com/java/lang/String.html),[Integer](http://docs.google.com/java/lang/Integer.html)> **getDisplayNames**(int field,  
 int style,  
 [Locale](http://docs.google.com/java/util/Locale.html) locale)

Returns a Map containing all names of the calendar field in the given style and locale and their corresponding field values. For example, if this Calendar is a [GregorianCalendar](http://docs.google.com/java/util/GregorianCalendar.html), the returned map would contain "Jan" to [JANUARY](http://docs.google.com/java/util/Calendar.html#JANUARY), "Feb" to [FEBRUARY](http://docs.google.com/java/util/Calendar.html#FEBRUARY), and so on, in the [short](http://docs.google.com/java/util/Calendar.html#SHORT) style in an English locale.

The values of other calendar fields may be taken into account to determine a set of display names. For example, if this Calendar is a lunisolar calendar system and the year value given by the [YEAR](http://docs.google.com/java/util/Calendar.html#YEAR) field has a leap month, this method would return month names containing the leap month name, and month names are mapped to their values specific for the year.

The default implementation supports display names contained in a [DateFormatSymbols](http://docs.google.com/java/text/DateFormatSymbols.html). For example, if field is [MONTH](http://docs.google.com/java/util/Calendar.html#MONTH) and style is [ALL\_STYLES](http://docs.google.com/java/util/Calendar.html#ALL_STYLES), this method returns a Map containing all strings returned by [DateFormatSymbols.getShortMonths()](http://docs.google.com/java/text/DateFormatSymbols.html#getShortMonths()) and [DateFormatSymbols.getMonths()](http://docs.google.com/java/text/DateFormatSymbols.html#getMonths()).

**Parameters:**field - the calendar field for which the display names are returnedstyle - the style applied to the display names; one of [SHORT](http://docs.google.com/java/util/Calendar.html#SHORT), [LONG](http://docs.google.com/java/util/Calendar.html#LONG), or [ALL\_STYLES](http://docs.google.com/java/util/Calendar.html#ALL_STYLES).locale - the locale for the display names **Returns:**a Map containing all display names in style and locale and their field values, or null if no display names are defined for field **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if field or style is invalid, or if this Calendar is non-lenient and any of the calendar fields have invalid values [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if locale is null**Since:** 1.6

### complete

protected void **complete**()

Fills in any unset fields in the calendar fields. First, the [computeTime()](http://docs.google.com/java/util/Calendar.html#computeTime()) method is called if the time value (millisecond offset from the [Epoch](#3znysh7)) has not been calculated from calendar field values. Then, the [computeFields()](http://docs.google.com/java/util/Calendar.html#computeFields()) method is called to calculate all calendar field values.

### equals

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

Compares this Calendar to the specified Object. The result is true if and only if the argument is a Calendar object of the same calendar system that represents the same time value (millisecond offset from the [Epoch](#3znysh7)) under the same Calendar parameters as this object.

The Calendar parameters are the values represented by the isLenient, getFirstDayOfWeek, getMinimalDaysInFirstWeek and getTimeZone methods. If there is any difference in those parameters between the two Calendars, this method returns false.

Use the [compareTo](http://docs.google.com/java/util/Calendar.html#compareTo(java.util.Calendar)) method to compare only the time values.

**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 object to compare with. **Returns:**true if this object is equal to obj; false otherwise.**See Also:**[Object.hashCode()](http://docs.google.com/java/lang/Object.html#hashCode()), [Hashtable](http://docs.google.com/java/util/Hashtable.html)

### hashCode

public int **hashCode**()

Returns a hash code for this calendar.

**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.**Since:** 1.2 **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)

### before

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

Returns whether this Calendar represents a time before the time represented by the specified Object. This method is equivalent to:

compareTo(when) < 0

if and only if when is a Calendar instance. Otherwise, the method returns false.

**Parameters:**when - the Object to be compared **Returns:**true if the time of this Calendar is before the time represented by when; false otherwise.**See Also:**[compareTo(Calendar)](http://docs.google.com/java/util/Calendar.html#compareTo(java.util.Calendar))

### after

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

Returns whether this Calendar represents a time after the time represented by the specified Object. This method is equivalent to:

compareTo(when) > 0

if and only if when is a Calendar instance. Otherwise, the method returns false.

**Parameters:**when - the Object to be compared **Returns:**true if the time of this Calendar is after the time represented by when; false otherwise.**See Also:**[compareTo(Calendar)](http://docs.google.com/java/util/Calendar.html#compareTo(java.util.Calendar))

### compareTo

public int **compareTo**([Calendar](http://docs.google.com/java/util/Calendar.html) anotherCalendar)

Compares the time values (millisecond offsets from the [Epoch](#3znysh7)) represented by two Calendar objects.

**Specified by:**[compareTo](http://docs.google.com/java/lang/Comparable.html#compareTo(T)) in interface [Comparable](http://docs.google.com/java/lang/Comparable.html)<[Calendar](http://docs.google.com/java/util/Calendar.html)> **Parameters:**anotherCalendar - the Calendar to be compared. **Returns:**the value 0 if the time represented by the argument is equal to the time represented by this Calendar; a value less than 0 if the time of this Calendar is before the time represented by the argument; and a value greater than 0 if the time of this Calendar is after the time represented by the argument. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified Calendar is null. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the time value of the specified Calendar object can't be obtained due to any invalid calendar values.**Since:** 1.5

### add

public abstract void **add**(int field,  
 int amount)

Adds or subtracts the specified amount of time to the given calendar field, based on the calendar's rules. For example, to subtract 5 days from the current time of the calendar, you can achieve it by calling:

add(Calendar.DAY\_OF\_MONTH, -5).

**Parameters:**field - the calendar field.amount - the amount of date or time to be added to the field.**See Also:**[roll(int,int)](http://docs.google.com/java/util/Calendar.html#roll(int,%20int)), [set(int,int)](http://docs.google.com/java/util/Calendar.html#set(int,%20int))

### roll

public abstract void **roll**(int field,  
 boolean up)

Adds or subtracts (up/down) a single unit of time on the given time field without changing larger fields. For example, to roll the current date up by one day, you can achieve it by calling:

roll(Calendar.DATE, true). When rolling on the year or Calendar.YEAR field, it will roll the year value in the range between 1 and the value returned by calling getMaximum(Calendar.YEAR). When rolling on the month or Calendar.MONTH field, other fields like date might conflict and, need to be changed. For instance, rolling the month on the date 01/31/96 will result in 02/29/96. When rolling on the hour-in-day or Calendar.HOUR\_OF\_DAY field, it will roll the hour value in the range between 0 and 23, which is zero-based.

**Parameters:**field - the time field.up - indicates if the value of the specified time field is to be rolled up or rolled down. Use true if rolling up, false otherwise.**See Also:**[add(int,int)](http://docs.google.com/java/util/Calendar.html#add(int,%20int)), [set(int,int)](http://docs.google.com/java/util/Calendar.html#set(int,%20int))

### roll

public void **roll**(int field,  
 int amount)

Adds the specified (signed) amount to the specified calendar field without changing larger fields. A negative amount means to roll down.

NOTE: This default implementation on Calendar just repeatedly calls the version of [roll()](http://docs.google.com/java/util/Calendar.html#roll(int,%20boolean)) that rolls by one unit. This may not always do the right thing. For example, if the DAY\_OF\_MONTH field is 31, rolling through February will leave it set to 28. The GregorianCalendar version of this function takes care of this problem. Other subclasses should also provide overrides of this function that do the right thing.

**Parameters:**field - the calendar field.amount - the signed amount to add to the calendar field.**Since:** 1.2 **See Also:**[roll(int,boolean)](http://docs.google.com/java/util/Calendar.html#roll(int,%20boolean)), [add(int,int)](http://docs.google.com/java/util/Calendar.html#add(int,%20int)), [set(int,int)](http://docs.google.com/java/util/Calendar.html#set(int,%20int))

### setTimeZone

public void **setTimeZone**([TimeZone](http://docs.google.com/java/util/TimeZone.html) value)

Sets the time zone with the given time zone value.

**Parameters:**value - the given time zone.

### getTimeZone

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

Gets the time zone.

**Returns:**the time zone object associated with this calendar.

### setLenient

public void **setLenient**(boolean lenient)

Specifies whether or not date/time interpretation is to be lenient. With lenient interpretation, a date such as "February 942, 1996" will be treated as being equivalent to the 941st day after February 1, 1996. With strict (non-lenient) interpretation, such dates will cause an exception to be thrown. The default is lenient.

**Parameters:**lenient - true if the lenient mode is to be turned on; false if it is to be turned off.**See Also:**[isLenient()](http://docs.google.com/java/util/Calendar.html#isLenient()), [DateFormat.setLenient(boolean)](http://docs.google.com/java/text/DateFormat.html#setLenient(boolean))

### isLenient

public boolean **isLenient**()

Tells whether date/time interpretation is to be lenient.

**Returns:**true if the interpretation mode of this calendar is lenient; false otherwise.**See Also:**[setLenient(boolean)](http://docs.google.com/java/util/Calendar.html#setLenient(boolean))

### setFirstDayOfWeek

public void **setFirstDayOfWeek**(int value)

Sets what the first day of the week is; e.g., SUNDAY in the U.S., MONDAY in France.

**Parameters:**value - the given first day of the week.**See Also:**[getFirstDayOfWeek()](http://docs.google.com/java/util/Calendar.html#getFirstDayOfWeek()), [getMinimalDaysInFirstWeek()](http://docs.google.com/java/util/Calendar.html#getMinimalDaysInFirstWeek())

### getFirstDayOfWeek

public int **getFirstDayOfWeek**()

Gets what the first day of the week is; e.g., SUNDAY in the U.S., MONDAY in France.

**Returns:**the first day of the week.**See Also:**[setFirstDayOfWeek(int)](http://docs.google.com/java/util/Calendar.html#setFirstDayOfWeek(int)), [getMinimalDaysInFirstWeek()](http://docs.google.com/java/util/Calendar.html#getMinimalDaysInFirstWeek())

### setMinimalDaysInFirstWeek

public void **setMinimalDaysInFirstWeek**(int value)

Sets what the minimal days required in the first week of the year are; For example, if the first week is defined as one that contains the first day of the first month of a year, call this method with value 1. If it must be a full week, use value 7.

**Parameters:**value - the given minimal days required in the first week of the year.**See Also:**[getMinimalDaysInFirstWeek()](http://docs.google.com/java/util/Calendar.html#getMinimalDaysInFirstWeek())

### getMinimalDaysInFirstWeek

public int **getMinimalDaysInFirstWeek**()

Gets what the minimal days required in the first week of the year are; e.g., if the first week is defined as one that contains the first day of the first month of a year, this method returns 1. If the minimal days required must be a full week, this method returns 7.

**Returns:**the minimal days required in the first week of the year.**See Also:**[setMinimalDaysInFirstWeek(int)](http://docs.google.com/java/util/Calendar.html#setMinimalDaysInFirstWeek(int))

### getMinimum

public abstract int **getMinimum**(int field)

Returns the minimum value for the given calendar field of this Calendar instance. The minimum value is defined as the smallest value returned by the [get](http://docs.google.com/java/util/Calendar.html#get(int)) method for any possible time value. The minimum value depends on calendar system specific parameters of the instance.

**Parameters:**field - the calendar field. **Returns:**the minimum value for the given calendar field.**See Also:**[getMaximum(int)](http://docs.google.com/java/util/Calendar.html#getMaximum(int)), [getGreatestMinimum(int)](http://docs.google.com/java/util/Calendar.html#getGreatestMinimum(int)), [getLeastMaximum(int)](http://docs.google.com/java/util/Calendar.html#getLeastMaximum(int)), [getActualMinimum(int)](http://docs.google.com/java/util/Calendar.html#getActualMinimum(int)), [getActualMaximum(int)](http://docs.google.com/java/util/Calendar.html#getActualMaximum(int))

### getMaximum

public abstract int **getMaximum**(int field)

Returns the maximum value for the given calendar field of this Calendar instance. The maximum value is defined as the largest value returned by the [get](http://docs.google.com/java/util/Calendar.html#get(int)) method for any possible time value. The maximum value depends on calendar system specific parameters of the instance.

**Parameters:**field - the calendar field. **Returns:**the maximum value for the given calendar field.**See Also:**[getMinimum(int)](http://docs.google.com/java/util/Calendar.html#getMinimum(int)), [getGreatestMinimum(int)](http://docs.google.com/java/util/Calendar.html#getGreatestMinimum(int)), [getLeastMaximum(int)](http://docs.google.com/java/util/Calendar.html#getLeastMaximum(int)), [getActualMinimum(int)](http://docs.google.com/java/util/Calendar.html#getActualMinimum(int)), [getActualMaximum(int)](http://docs.google.com/java/util/Calendar.html#getActualMaximum(int))

### getGreatestMinimum

public abstract int **getGreatestMinimum**(int field)

Returns the highest minimum value for the given calendar field of this Calendar instance. The highest minimum value is defined as the largest value returned by [getActualMinimum(int)](http://docs.google.com/java/util/Calendar.html#getActualMinimum(int)) for any possible time value. The greatest minimum value depends on calendar system specific parameters of the instance.

**Parameters:**field - the calendar field. **Returns:**the highest minimum value for the given calendar field.**See Also:**[getMinimum(int)](http://docs.google.com/java/util/Calendar.html#getMinimum(int)), [getMaximum(int)](http://docs.google.com/java/util/Calendar.html#getMaximum(int)), [getLeastMaximum(int)](http://docs.google.com/java/util/Calendar.html#getLeastMaximum(int)), [getActualMinimum(int)](http://docs.google.com/java/util/Calendar.html#getActualMinimum(int)), [getActualMaximum(int)](http://docs.google.com/java/util/Calendar.html#getActualMaximum(int))

### getLeastMaximum

public abstract int **getLeastMaximum**(int field)

Returns the lowest maximum value for the given calendar field of this Calendar instance. The lowest maximum value is defined as the smallest value returned by [getActualMaximum(int)](http://docs.google.com/java/util/Calendar.html#getActualMaximum(int)) for any possible time value. The least maximum value depends on calendar system specific parameters of the instance. For example, a Calendar for the Gregorian calendar system returns 28 for the DAY\_OF\_MONTH field, because the 28th is the last day of the shortest month of this calendar, February in a common year.

**Parameters:**field - the calendar field. **Returns:**the lowest maximum value for the given calendar field.**See Also:**[getMinimum(int)](http://docs.google.com/java/util/Calendar.html#getMinimum(int)), [getMaximum(int)](http://docs.google.com/java/util/Calendar.html#getMaximum(int)), [getGreatestMinimum(int)](http://docs.google.com/java/util/Calendar.html#getGreatestMinimum(int)), [getActualMinimum(int)](http://docs.google.com/java/util/Calendar.html#getActualMinimum(int)), [getActualMaximum(int)](http://docs.google.com/java/util/Calendar.html#getActualMaximum(int))

### getActualMinimum

public int **getActualMinimum**(int field)

Returns the minimum value that the specified calendar field could have, given the time value of this Calendar.

The default implementation of this method uses an iterative algorithm to determine the actual minimum value for the calendar field. Subclasses should, if possible, override this with a more efficient implementation - in many cases, they can simply return getMinimum().

**Parameters:**field - the calendar field **Returns:**the minimum of the given calendar field for the time value of this Calendar**Since:** 1.2 **See Also:**[getMinimum(int)](http://docs.google.com/java/util/Calendar.html#getMinimum(int)), [getMaximum(int)](http://docs.google.com/java/util/Calendar.html#getMaximum(int)), [getGreatestMinimum(int)](http://docs.google.com/java/util/Calendar.html#getGreatestMinimum(int)), [getLeastMaximum(int)](http://docs.google.com/java/util/Calendar.html#getLeastMaximum(int)), [getActualMaximum(int)](http://docs.google.com/java/util/Calendar.html#getActualMaximum(int))

### getActualMaximum

public int **getActualMaximum**(int field)

Returns the maximum value that the specified calendar field could have, given the time value of this Calendar. For example, the actual maximum value of the MONTH field is 12 in some years, and 13 in other years in the Hebrew calendar system.

The default implementation of this method uses an iterative algorithm to determine the actual maximum value for the calendar field. Subclasses should, if possible, override this with a more efficient implementation.

**Parameters:**field - the calendar field **Returns:**the maximum of the given calendar field for the time value of this Calendar**Since:** 1.2 **See Also:**[getMinimum(int)](http://docs.google.com/java/util/Calendar.html#getMinimum(int)), [getMaximum(int)](http://docs.google.com/java/util/Calendar.html#getMaximum(int)), [getGreatestMinimum(int)](http://docs.google.com/java/util/Calendar.html#getGreatestMinimum(int)), [getLeastMaximum(int)](http://docs.google.com/java/util/Calendar.html#getLeastMaximum(int)), [getActualMinimum(int)](http://docs.google.com/java/util/Calendar.html#getActualMinimum(int))

### clone

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

Creates and returns a copy of this object.

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

### toString

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

Return a string representation of this calendar. This method is intended to be used only for debugging purposes, and the format of the returned string may vary between implementations. The returned string may be empty but may not be null.

**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 this calendar.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Calendar.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/util/BitSet.html)   [**NEXT CLASS**](http://docs.google.com/java/util/Collection.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/Calendar.html)    [**NO FRAMES**](http://docs.google.com/Calendar.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3dy6vkm) | [CONSTR](#1t3h5sf) | [METHOD](#4d34og8) | DETAIL: [FIELD](#17dp8vu) | [CONSTR](#sqyw64) | [METHOD](#4bvk7pj) |

[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).