java.util

Class Calendar

java.lang.Object
    java.util.Calendar

All Implemented Interfaces:

Serializable, Cloneable, Comparable<Calendar>

Direct Known Subclasses:

GregorianCalendar

```
public abstract class Calendar
extends Object
implements Serializable, Cloneable, Comparable<Calendar>
```

The `Calendar` class is an abstract class that provides methods for converting between a specific instant in time and a set of `calendar 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();
```

getInstance

```
public static Calendar getInstance(TimeZone 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.

https://docs.oracle.com/javase/7/docs/api/java/util/Calendar.html

Overview Package Class Use Tree Deprecated Index Help Java™ Platform Standard Ed. 7

java.util

Class TimeZone

java.lang.Object
    java.util.TimeZone

All Implemented Interfaces:

Serializable, Cloneable

Direct Known Subclasses:

SimpleTimeZone

```
public abstract class TimeZone
extends Object
implements Serializable, Cloneable
```

`TimeZone` represents a time zone offset, and also figures out daylight savings.

Typically, you get a `TimeZone` using `getDefault` which creates a `TimeZone` based on the time zone where the program is running. For example, for a program running in Japan, `getDefault` creates a `TimeZone` object based on Japanese Standard Time.

You can also get a `TimeZone` using `getTimeZone` along with a time zone ID. For instance, the time zone ID for the U.S. Pacific Time zone is "America/Los_Angeles". So, you can get a U.S. Pacific Time `TimeZone` object with:

```
TimeZone tz = TimeZone.getTimeZone("America/Los_Angeles");
```

getTimeZone

```
public static TimeZone getTimeZone(String ID)
```

Gets the `TimeZone` for the given ID.

Parameters:

`ID` - the ID for a `TimeZone`, either an abbreviation such as "PST", a full name such as "America/Los_Angeles", or a custom ID such as "GMT-8:00". Note that the support of abbreviations is for JDK 1.1.x compatibility only and full names should be used.

Returns:

the specified `TimeZone`, or the GMT zone if the given ID cannot be understood.

https://docs.oracle.com/javase/7/docs/api/java/util/TimeZone.html