Datetime Class

Contains methods for the Datetime primitive data type.

Namespace

System

Usage

Apex supports both implicit and explicit casting of Date values to Datetime, with the time component being zeroed out in the resulting Datetime value. For more information about the Datetime, see Datetime Data Type.

Datetime Methods

The following are methods for `Datetime`.

- **addDays(additionalDays)**
  Adds the specified number of days to a Datetime.
- **addHours(additionalHours)**
  Adds the specified number of hours to a Datetime.
- **addMinutes(additionalMinutes)**
  Adds the specified number of minutes to a Datetime.
- **addMonths(additionalMonths)**
  Adds the specified number of months to a Datetime.
- **addSeconds(additionalSeconds)**
  Adds the specified number of seconds to a Datetime.
- **addYears(additionalYears)**
  Adds the specified number of years to a Datetime.
- **date()**
  Returns the Date component of a Datetime in the local time zone of the context user.
- **dateGMT()**
  Return the Date component of a Datetime in the GMT time zone.
- **day()**
  Returns the day-of-month component of a Datetime in the local time zone of the context user.
- **dayGmt()**
  Returns the day-of-month component of a Datetime in the GMT time zone.
- **dayOfYear()**
  Returns the day-of-year component of a Datetime in the local time zone of the context user.
- **dayOfYearGmt()**
  Returns the day-of-year component of a Datetime in the GMT time zone.
- **format()**
  Converts the date to the local time zone and returns the converted date as a formatted string using the locale of the context user. If the time zone cannot be determined, GMT is used.

- **format(dateFormatString, timezone)**
  Converts the date to the specified time zone and returns the converted date as a string using the supplied Java simple date format. If the supplied time zone is not in the correct format, GMT is used.
- **formatGmt(dateFormatString)**
  Returns a Datetime as a string using the supplied Java simple date format and the GMT time zone.
- **formatLong()**
  Converts the date to the local time zone and returns the converted date in long date format.
- **getTime()**
  Returns the number of milliseconds since January 1, 1970, 00:00:00 GMT represented by this DateTime object.
- **hour()**
  Returns the hour component of a Datetime in the local time zone of the context user.
- **hourGmt()**
  Returns the hour component of a Datetime in the GMT time zone.
- **isSameDay(dateToCompare)**
  Returns true if the Datetime that called the method is the same as the specified Datetime in the local time zone of the context user.
- **millisecond()**
  Return the millisecond component of a Datetime in the local time zone of the context user.
- **millisecondGmt()**
  Return the millisecond component of a Datetime in the GMT time zone.
- **minute()**
  Returns the minute component of a Datetime in the local time zone of the context user.
- **minuteGmt()**
  Returns the minute component of a Datetime in the GMT time zone.
- **month()**
  Returns the month component of a Datetime in the local time zone of the context user (1=Jan).
- **monthGmt()**
  Returns the month component of a Datetime in the GMT time zone (1=Jan).
- **newInstance(milliseconds)**
  Constructs a Datetime and initializes it to represent the specified number of milliseconds since January 1, 1970, 00:00:00 GMT.
- **newInstance(date, time)**
  Constructs a DateTime from the specified date and time in the local time zone.
- **newInstance(year, month, day)**
  Constructs a Datetime from Integer representations of the specified year, month (1=Jan), and day at midnight in the local time zone.
- **newInstance(year, month, day, hour, minute, second)**
  Constructs a Datetime from Integer representations of the specified year, month (1=Jan), day, hour, minute, and second in the local time zone.
- **newInstanceGmt(date, time)**
  Constructs a DateTime from the specified date and time in the GMT time zone.
- **newInstanceGmt(year, month, date)**
  Constructs a Datetime from Integer representations of the specified year, month (1=Jan), and day at midnight in the GMT time zone
- **newInstanceGmt(year, month, date, hour, minute, second)**
  Constructs a Datetime from Integer representations of the specified year, month (1=Jan), day, hour, minute, and second in the GMT time zone

the user locale.

- **second()**
  Returns the second component of a Datetime in the local time zone of the context user.
- **secondGmt()**
  Returns the second component of a Datetime in the GMT time zone.
- **time()**
  Returns the time component of a Datetime in the local time zone of the context user.
- **timeGmt()**
  Returns the time component of a Datetime in the GMT time zone.
- **valueOf(dateTimeString)**
  Returns a Datetime that contains the value of the specified string.
- **valueOf(fieldValue)**
  Converts the specified object to a Datetime. Use this method to convert a history tracking field value or an object that represents a Datetime value.
- **valueOfGmt(dateTimeString)**
  Returns a Datetime that contains the value of the specified String.
- **year()**
  Returns the year component of a Datetime in the local time zone of the context user.
- **yearGmt()**
  Returns the year component of a Datetime in the GMT time zone.

addDays(additionalDays)

Adds the specified number of days to a Datetime.

Signature

```
public Datetime addDays(Integer additionalDays)
```

Parameters

additionalDays

Type: Integer

Return Value

Type: Datetime

Example

```
Datetime myDateTime = Datetime.newInstance(1960, 2, 17);
Datetime newDateTime = myDateTime.addDays(2);
Datetime expected = Datetime.newInstance(1960, 2, 19);
System.assertEquals(expected, newDateTime);
```

addHours(additionalHours)

Adds the specified number of hours to a Datetime.

Signature

```
public Datetime addHours(Integer additionalHours)
```

Parameters

additionalHours

Type: Integer

```
DateTime myDateTime = DateTime.newInstance(1997, 1, 31, 7, 8, 16);
DateTime newDateTime = myDateTime.addHours(3);
DateTime expected = DateTime.newInstance(1997, 1, 31, 10, 8, 16);
System.assertEquals(expected, newDateTime);
```

addMinutes(additionalMinutes)

Adds the specified number of minutes to a Datetime.

Signature

```
public Datetime addMinutes(Integer additionalMinutes)
```

Parameters

additionalMinutes

Type: Integer

Return Value

Type: Datetime

Example

```
DateTime myDateTime = DateTime.newInstance(1999, 2, 11, 8, 6, 16);
DateTime newDateTime = myDateTime.addMinutes(7);
DateTime expected = DateTime.newInstance(1999, 2, 11, 8, 13, 16);
System.assertEquals(expected, newDateTime);
```

addMonths(additionalMonths)

Adds the specified number of months to a Datetime.

Signature

```
public Datetime addMonths(Integer additionalMonths)
```

Parameters

additionalMonths

Type: Integer

Return Value

Type: Datetime

Example

```
DateTime myDateTime = DateTime.newInstance(2000, 7, 7, 7, 8, 12);
DateTime newDateTime = myDateTime.addMonths(1);
DateTime expected = DateTime.newInstance(2000, 8, 7, 7, 8, 12);
System.assertEquals(expected, newDateTime);
```

addSeconds(additionalSeconds)

Parameters

additionalSeconds

Type: Integer

Return Value

Type: Datetime

Example

```
DateTime myDateTime = DateTime.newInstance(2001, 7, 19, 10, 7, 12);
DateTime newDateTime = myDateTime.addSeconds(4);
DateTime expected = DateTime.newInstance(2001, 7, 19, 10, 7, 16);
System.assertEquals(expected, newDateTime);
```

addYears(additionalYears)

Adds the specified number of years to a Datetime.

Signature

```
public Datetime addYears(Integer additionalYears)
```

Parameters

additionalYears

Type: Integer

Return Value

Type: Datetime

Example

```
DateTime myDateTime = DateTime.newInstance(2009, 12, 17, 13, 6, 6);
DateTime newDateTime = myDateTime.addYears(1);
DateTime expected = DateTime.newInstance(2010, 12, 17, 13, 6, 6);
System.assertEquals(expected, newDateTime);
```

date()

Returns the Date component of a Datetime in the local time zone of the context user.

Signature

```
public Date date()
```

Return Value

Type: Date

Example

```
DateTime myDateTime = DateTime.newInstance(2006, 3, 16, 12, 6, 13);
Date myDate = myDateTime.date();
```

Return the Date component of a Datetime in the GMT time zone.

Signature

```
public Date dateGMT()
```

Return Value

Type: Date

Example

```
// California local time, PST
DateTime myDateTime = DateTime.newInstance(2006, 3, 16, 23, 0, 0);
Date myDate = myDateTime.dateGMT();
Date expected = Date.newInstance(2006, 3, 17);
System.assertEquals(expected, myDate);
```

day()

Returns the day-of-month component of a Datetime in the local time zone of the context user.

Signature

```
public Integer day()
```

Return Value

Type: Integer

Example

```
DateTime myDateTime = DateTime.newInstance(1986, 2, 21, 23, 0, 0);
System.assertEquals(21, myDateTime.day());
```

dayGmt()

Returns the day-of-month component of a Datetime in the GMT time zone.

Signature

```
public Integer dayGmt()
```

Return Value

Type: Integer

Example

```
// California local time, PST
DateTime myDateTime = DateTime.newInstance(1987, 1, 14, 23, 0, 3);
System.assertEquals(15, myDateTime.dayGMT());
```

dayOfYear()

Return Value

Type: Integer

Example

For example, February 5, 2008 08:30:12 would be day 36.

```
Datetime myDate = Datetime.newInstance(2008, 2, 5, 8, 30, 12);
system.assertEquals(myDate.dayOfYear(), 36);
```

dayOfYearGmt()

Returns the day-of-year component of a Datetime in the GMT time zone.

Signature

```
public Integer dayOfYearGmt()
```

Return Value

Type: Integer

Example

```
// This sample assumes we are in the PST timezone
DateTime myDateTime = DateTime.newInstance(1999, 2, 5, 23, 0, 3);
// January has 31 days + 5 days in February = 36 days
// dayOfYearGmt() adjusts the time zone from the current time zone to GMT
// by adding 8 hours to the PST time zone, so it's 37 days and not 36 days
System.assertEquals(37, myDateTime.dayOfYearGmt());
```

format()

Converts the date to the local time zone and returns the converted date as a formatted string using the locale of the context user. If the time zone cannot be determined, GMT is used.

Signature

```
public String format()
```

Return Value

Type: String

Example

Note

The sample is executed in an org where the “Enable ICU Locale Formats” crucial update is enabled. See https://releasenotes.docs.salesforce.com/en-us/spring20/release-notes/rn_forcecom_globalization_enable_icu_cruc.htm.

Converts the date to the local time zone and returns the converted date as a string using the supplied Java simple date format. If the time zone cannot be determined, GMT is used.

Signature

```
public String format(String dateFormatString)
```

Parameters

dateFormatString

Type: String

Return Value

Type: String

Usage

For more information on the Java simple date format, see Java SimpleDateFormat.

Example

```
Datetime myDT = DateTime.newInstance(2022, 5, 4, 19, 37, 55);
String myDate = myDT.format('yyyy-MM-dd h:mm a');
String expected = '2022-05-04 7:37 PM';
System.assertEquals(expected, myDate);
```

format(dateFormatString, timezone)

Converts the date to the specified time zone and returns the converted date as a string using the supplied Java simple date format. If the supplied time zone is not in the correct format, GMT is used.

Signature

```
public String format(String dateFormatString, String timezone)
```

Parameters

dateFormatString

Type: String

timezone

Type: String

Valid time zone values for the *timezone* argument are the time zones of the Java TimeZone class that correspond to the time zones returned by the TimeZone.getAvailableIDs method in Java. We recommend you use full time zone names, not the three-letter abbreviations.

Return Value

Type: String

Usage

For more information on the Java simple date format, see Java SimpleDateFormat.

Example

```
Datetime GMTDate =
  Datetime.newInstanceGmt(2011,6,1,12,1,5);
String strConvertedDate =
  GMTDate.format('MM/dd/yyyy HH:mm:ss',
                 'America/New_York');
// Date is converted to
// the new time zone and is adjusted
// for daylight saving time.
System.assertEquals(
  '06/01/2011 08:01:05', strConvertedDate);
```

formatGmt(dateFormatString)

Returns a Datetime as a string using the supplied Java simple date format and the GMT time zone.

Signature

```
public String formatGmt(String dateFormatString)
```

Parameters

dateFormatString

Type: String

Return Value

Type: String

Usage

For more information on the Java simple date format, see Java SimpleDateFormat.

Example

```
DateTime myDateTime = DateTime.newInstance(1993, 6, 6, 3, 3, 3);
String formatted = myDateTime.formatGMT('EEE, MMM d yyyy HH:mm:ss');
String expected = 'Sun, Jun 6 1993 10:03:03';
System.assertEquals(expected, formatted);
```

formatLong()

Converts the date to the local time zone and returns the converted date in long date format.

Signature

```
public String formatLong()
```

Return Value

Type: String

Example

```
// Passing local date based on the PST time zone
Datetime dt = DateTime.newInstance(2012,12,28,10,0,0);
// Writes 12/28/2012 10:00:00 AM PST
System.debug('dt.formatLong()=' + dt.formatLong());
```

getTime()

```
public Long getTime()
```

Return Value

Type: Long

Example

```
DateTime dt = DateTime.newInstanceGMT(2007, 6, 23, 3, 3, 3);
Long gettime = dt.getTime();
Long expected = 1182567783000L;
System.assertEquals(expected, gettime);
```

hour()

Returns the hour component of a Datetime in the local time zone of the context user.

Signature

```
public Integer hour()
```

Return Value

Type: Integer

Example

```
DateTime myDateTime = DateTime.newInstance(1998, 11, 21, 3, 3, 3);
System.assertEquals(3 , myDateTime.hour());
```

hourGmt()

Returns the hour component of a Datetime in the GMT time zone.

Signature

```
public Integer hourGmt()
```

Return Value

Type: Integer

Example

```
// California local time
DateTime myDateTime = DateTime.newInstance(2000, 4, 27, 3, 3, 3);
System.assertEquals(10 , myDateTime.hourGMT());
```

isSameDay(dateToCompare)

Returns true if the Datetime that called the method is the same as the specified Datetime in the local time zone of the context user.

Signature

```
public Boolean isSameDay(Datetime dateToCompare)
```

Return Value

Type: Boolean

Example

```
datetime myDate = datetime.now();
datetime dueDate =
     datetime.newInstance(2008, 1, 30);
boolean dueNow = myDate.isSameDay(dueDate);
```

millisecond()

Return the millisecond component of a Datetime in the local time zone of the context user.

Signature

```
public Integer millisecond()
```

Return Value

Type: Integer

Example

```
DateTime myDateTime = DateTime.now();
system.debug(myDateTime.millisecond());
```

millisecondGmt()

Return the millisecond component of a Datetime in the GMT time zone.

Signature

```
public Integer millisecondGmt()
```

Return Value

Type: Integer

Example

```
DateTime myDateTime = DateTime.now();
system.debug(myDateTime.millisecondGMT());
```

minute()

Returns the minute component of a Datetime in the local time zone of the context user.

Signature

```
public Integer minute()
```

Return Value

Type: Integer

```
system.assertEquals(3, myDateTime.minute());
```

minuteGmt()

Returns the minute component of a Datetime in the GMT time zone.

Signature

```
public Integer minuteGmt()
```

Return Value

Type: Integer

Example

```
DateTime myDateTime = DateTime.newInstance(2002, 12, 3, 3, 3, 3);
system.assertEquals(3, myDateTime.minuteGMT());
```

month()

Returns the month component of a Datetime in the local time zone of the context user (1=Jan).

Signature

```
public Integer month()
```

Return Value

Type: Integer

Example

```
DateTime myDateTime = DateTime.newInstance(2004, 11, 4, 3, 3, 3);
system.assertEquals(11, myDateTime.month());
```

monthGmt()

Returns the month component of a Datetime in the GMT time zone (1=Jan).

Signature

```
public Integer monthGmt()
```

Return Value

Type: Integer

Example

```
DateTime myDateTime = DateTime.newInstance(2006, 11, 19, 3, 3, 3);
system.assertEquals(11, myDateTime.monthGMT());
```

newInstance(milliseconds)

```
public static Datetime newInstance(Long milliseconds)
```

Parameters

milliseconds

Type: Long

Return Value

Type: Datetime

The returned date is in the GMT time zone.

Example

```
Long longtime = 1341828183000L;
DateTime dt = DateTime.newInstance(longtime);
DateTime expected = DateTime.newInstance(2012, 7, 09, 3, 3, 3);
System.assertEquals(expected, dt);
```

newInstance(date, time)

Constructs a DateTime from the specified date and time in the local time zone.

Signature

```
public static Datetime newInstance(Date date, Time time)
```

Parameters

date

Type: Date

time

Type: Time

Return Value

Type: Datetime

The returned date is in the GMT time zone.

Example

```
Date myDate = Date.newInstance(2011, 11, 18);
Time myTime = Time.newInstance(3, 3, 3, 0);
DateTime dt = DateTime.newInstance(myDate, myTime);
DateTime expected = DateTime.newInstance(2011, 11, 18, 3, 3, 3);
System.assertEquals(expected, dt);
```

newInstance(year, month, day)

Constructs a Datetime from Integer representations of the specified year, month (1=Jan), and day at midnight in the local time zone.

Signature

```
public static Datetime newInstance(Integer year, Integer month, Integer day)
```

month

Type: Integer

day

Type: Integer

Return Value

Type: Datetime

The returned date is in the GMT time zone.

Example

```
datetime myDate = datetime.newInstance(2008, 12, 1);
```

newInstance(year, month, day, hour, minute, second)

Constructs a Datetime from Integer representations of the specified year, month (1=Jan), day, hour, minute, and second in the local time zone.

Signature

```
public static Datetime newInstance(Integer year, Integer month, Integer day, Integer hour, Integer minute, Integer second)
```

Parameters

year

Type: Integer

month

Type: Integer

day

Type: Integer

hour

Type: Integer

minute

Type: Integer

second

Type: Integer

Return Value

Type: Datetime

The returned date is in the GMT time zone.

Example

```
Datetime myDate = Datetime.newInstance(2008, 12, 1, 12, 30, 2);
```

```
public static Datetime newInstanceGmt(Date date, Time time)
```

Parameters

date

Type: Date

time

Type: Time

Return Value

Type: Datetime

Example

```
Date myDate = Date.newInstance(2013, 11, 12);
Time myTime = Time.newInstance(3, 3, 3, 0);
DateTime dt = DateTime.newInstanceGMT(myDate, myTime);
DateTime expected = DateTime.newInstanceGMT(2013, 11, 12, 3, 3, 3);
System.assertEquals(expected, dt);
```

newInstanceGmt(year, month, date)

Constructs a Datetime from Integer representations of the specified year, month (1=Jan), and day at midnight in the GMT time zone

Signature

```
public static Datetime newInstanceGmt(Integer year, Integer month, Integer date)
```

Parameters

year

Type: Integer

month

Type: Integer

date

Type: Integer

Return Value

Type: Datetime

Example

```
DateTime dt = DateTime.newInstanceGMT(1996, 3, 22);
```

newInstanceGmt(year, month, date, hour, minute, second)

Constructs a Datetime from Integer representations of the specified year, month (1=Jan), day, hour, minute, and second in the GMT time zone

Signature

year

Type: Integer

month

Type: Integer

date

Type: Integer

hour

Type: Integer

minute

Type: Integer

second

Type: Integer

Return Value

Type: Datetime

Example

```
//California local time
DateTime dt = DateTime.newInstanceGMT(1998, 1, 29, 2, 2, 3);
DateTime expected = DateTime.newInstance(1998, 1, 28, 18, 2, 3);
System.assertEquals(expected, dt);
```

now()

Returns the current Datetime based on a GMT calendar.

Signature

```
public static Datetime now()
```

Return Value

Type: Datetime

The format of the returned datetime is: `'MM/DD/YYYY HH:MM PERIOD'`

Example

```
datetime myDateTime = datetime.now();
```

parse(datetimeString)

Constructs a Datetime from the given String in the local time zone and in the format of the user locale.

Signature

```
public static Datetime parse(String datetimeString)
```

Parameters

Type: Datetime

The returned date is in the GMT time zone.

Example

This example uses `parse` to create a Datetime from a date passed in as a string and that is formatted for the English (United States) locale. You may need to change the format of the date string if you have a different locale.

> **Note**
>
> This sample is executed in an org where the “Enable ICU Locale Formats” crucial update is enabled. See https://releasenotes.docs.salesforce.com/en-us/spring20/release-notes/rn_forcecom_globalization_enable_icu_cruc.htm.

```
Datetime dt = DateTime.parse('10/14/2011, 11:46 AM');
String myDtString = dt.format();
system.assertEquals(myDtString, '10/14/2011, 11:46 AM');
```

second()

Returns the second component of a Datetime in the local time zone of the context user.

Signature

```
public Integer second()
```

Return Value

Type: Integer

Example

```
DateTime dt = DateTime.newInstanceGMT(1999, 9, 22, 3, 1, 2);
System.assertEquals(2, dt.second());
```

secondGmt()

Returns the second component of a Datetime in the GMT time zone.

Signature

```
public Integer secondGmt()
```

Return Value

Type: Integer

Example

```
DateTime dt = DateTime.newInstance(2000, 2, 3, 3, 1, 5);
System.assertEquals(5, dt.secondGMT());
```

```
public Time time()
```

Return Value

Type: Time

Example

```
DateTime dt = DateTime.newInstance(2002, 11, 21, 0, 2, 2);
Time expected = Time.newInstance(0, 2, 2, 0);
System.assertEquals(expected, dt.time());
```

timeGmt()

Returns the time component of a Datetime in the GMT time zone.

Signature

```
public Time timeGmt()
```

Return Value

Type: Time

Example

```
// This sample is based on the PST time zone
DateTime dt = DateTime.newInstance(2004, 1, 27, 4, 1, 2);
Time expected = Time.newInstance(12, 1, 2, 0);
// 8 hours are added to the time to convert it from
// PST to GMT
System.assertEquals(expected, dt.timeGMT());
```

valueOf(dateTimeString)

Returns a Datetime that contains the value of the specified string.

Signature

```
public static Datetime valueOf(String dateTimeString)
```

Parameters

dateTimeString

Type: String

Return Value

Type: Datetime

The returned date is in the GMT time zone.

Usage

The specified string should use the standard date format “yyyy-MM-dd HH:mm:ss” in the local time zone.

Example

```
string hour = '12';
string minute = '20';
string second = '20';
string stringDate = year + '-' + month + '-' + day + ' ' + hour + ':'
    + minute +  ':' + second;

Datetime myDate = Datetime.valueOf(stringDate);
```

valueOf(fieldValue)

Converts the specified object to a Datetime. Use this method to convert a history tracking field value or an object that represents a Datetime value.

Signature

```
public static Datetime valueOf(Object fieldValue)
```

Parameters

fieldValue

Type: Object

Return Value

Type: Datetime

Usage

Use this method with the `OldValue` or `NewValue` fields of history sObjects, such as `AccountHistory`, when the field is a Date/Time field.

Example

```
List<AccountHistory> ahlist = [SELECT Field,OldValue,NewValue FROM AccountHistory];
for(AccountHistory ah : ahlist) {
  System.debug('Field: ' + ah.Field);
  if (ah.field == 'MyDatetime__c') {
    Datetime oldValue = Datetime.valueOf(ah.OldValue);
    Datetime newValue = Datetime.valueOf(ah.NewValue);
  }
}
```

valueOfGmt(dateTimeString)

Returns a Datetime that contains the value of the specified String.

Signature

```
public static Datetime valueOfGmt(String dateTimeString)
```

Parameters

dateTimeString

Type: String

Return Value

Type: Datetime

Usage

```
// California locale time
string year = '2009';
string month = '3';
string day = '5';
string hour = '5';
string minute = '2';
string second = '2';
string stringDate = year + '-' + month + '-' + day + ' ' + hour + ':'
    + minute +  ':' + second;

Datetime myDate = Datetime.valueOfGMT(stringDate);

DateTime expected = DateTime.newInstance(2009, 3, 4, 21, 2, 2);
System.assertEquals(expected, myDate);
```

year()

Returns the year component of a Datetime in the local time zone of the context user.

Signature

```
public Integer year()
```

Return Value

Type: Integer

Example

```
DateTime dt = DateTime.newInstance(2012, 1, 26, 5, 2, 4);
System.assertEquals(2012, dt.year());
```

yearGmt()

Returns the year component of a Datetime in the GMT time zone.

Signature

```
public Integer yearGmt()
```

Return Value

Type: Integer

Example

```
DateTime dt = DateTime.newInstance(2012, 10, 4, 6, 4, 6);
System.assertEquals(2012, dt.yearGMT());
```

MuleSoft
Tableau
Commerce Cloud
Lightning Design System
Einstein
Quip

Component Library
APIs
Trailhead
Sample Apps
Podcasts
AppExchange

Events and Calendar
Partner Community
Blog
Salesforce Admins
Salesforce Architects