compact2, compact3

java.sql

**Class Timestamp**

* [java.lang.Object](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html)
  + [java.util.Date](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html)
    - java.sql.Timestamp
* **All Implemented Interfaces:**

[Serializable](https://docs.oracle.com/javase/8/docs/api/java/io/Serializable.html), [Cloneable](https://docs.oracle.com/javase/8/docs/api/java/lang/Cloneable.html" \o "interface in java.lang), [Comparable](https://docs.oracle.com/javase/8/docs/api/java/lang/Comparable.html)<[Date](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html)>

public class **Timestamp**

extends [Date](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html)

A thin wrapper around java.util.Date that allows the JDBC API to identify this as an SQL TIMESTAMP value. It adds the ability to hold the SQL TIMESTAMP fractional seconds value, by allowing the specification of fractional seconds to a precision of nanoseconds. A Timestamp also provides formatting and parsing operations to support the JDBC escape syntax for timestamp values.

The precision of a Timestamp object is calculated to be either:

* + 19 , which is the number of characters in yyyy-mm-dd hh:mm:ss
  + 20 + s , which is the number of characters in the yyyy-mm-dd hh:mm:ss.[fff...] and s represents the scale of the given Timestamp, its fractional seconds precision.

**Note:** This type is a composite of a java.util.Date and a separate nanoseconds value. Only integral seconds are stored in the java.util.Date component. The fractional seconds - the nanos - are separate. The Timestamp.equals(Object) method never returns true when passed an object that isn't an instance of java.sql.Timestamp, because the nanos component of a date is unknown. As a result, the Timestamp.equals(Object) method is not symmetric with respect to thejava.util.Date.equals(Object) method. Also, the hashCode method uses the underlying java.util.Date implementation and therefore does not include nanos in its computation.

Due to the differences between the Timestamp class and the java.util.Date class mentioned above, it is recommended that code not view Timestamp values generically as an instance of java.util.Date. The inheritance relationship between Timestamp and java.util.Date really denotes implementation inheritance, and not type inheritance.

**See Also:**

[Serialized Form](https://docs.oracle.com/javase/8/docs/api/serialized-form.html#java.sql.Timestamp)

* + ***Constructor Summary***

|  |
| --- |
| **Constructors** |
| **Constructor and Description** |
| [**Timestamp**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#Timestamp-int-int-int-int-int-int-int-)(int year, int month, int date, int hour, int minute, int second, int nano)  **Deprecated.**  instead use the constructor Timestamp(long millis) |
| [**Timestamp**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#Timestamp-long-)(long time)  Constructs a Timestamp object using a milliseconds time value. |

* + ***Method Summary***

|  |  |
| --- | --- |
| **All Methods**[**Static Methods**](javascript:show(1);)[**Instance Methods**](javascript:show(2);)[**Concrete Methods**](javascript:show(8);) | |
| **Modifier and Type** | **Method and Description** |
| boolean | [**after**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#after-java.sql.Timestamp-)([**Timestamp**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html) ts)  Indicates whether this Timestamp object is later than the given Timestamp object. |
| boolean | [**before**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#before-java.sql.Timestamp-)([**Timestamp**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html) ts)  Indicates whether this Timestamp object is earlier than the given Timestamp object. |
| int | [**compareTo**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#compareTo-java.util.Date-)([**Date**](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html) o)  Compares this Timestamp object to the given Date object. |
| int | [**compareTo**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#compareTo-java.sql.Timestamp-)([**Timestamp**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html) ts)  Compares this Timestamp object to the given Timestamp object. |
| boolean | [**equals**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#equals-java.lang.Object-)([**Object**](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html) ts)  Tests to see if this Timestamp object is equal to the given object. |
| boolean | [**equals**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#equals-java.sql.Timestamp-)([**Timestamp**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html) ts)  Tests to see if this Timestamp object is equal to the given Timestamp object. |
| static [**Timestamp**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html) | [**from**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#from-java.time.Instant-)([**Instant**](https://docs.oracle.com/javase/8/docs/api/java/time/Instant.html) instant)  Obtains an instance of Timestamp from an [**Instant**](https://docs.oracle.com/javase/8/docs/api/java/time/Instant.html) object. |
| int | [**getNanos**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#getNanos--)()  Gets this Timestamp object's nanos value. |
| long | [**getTime**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#getTime--)()  Returns the number of milliseconds since January 1, 1970, 00:00:00 GMT represented by this Timestamp object. |
| int | [**hashCode**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#hashCode--)()  Returns a hash code value for this object. |
| void | [**setNanos**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#setNanos-int-)(int n)  Sets this Timestamp object's nanos field to the given value. |
| void | [**setTime**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#setTime-long-)(long time)  Sets this Timestamp object to represent a point in time that is time milliseconds after January 1, 1970 00:00:00 GMT. |
| [**Instant**](https://docs.oracle.com/javase/8/docs/api/java/time/Instant.html) | [**toInstant**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#toInstant--)()  Converts this Timestamp object to an Instant. |
| [**LocalDateTime**](https://docs.oracle.com/javase/8/docs/api/java/time/LocalDateTime.html) | [**toLocalDateTime**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#toLocalDateTime--)()  Converts this Timestamp object to a LocalDateTime. |
| [**String**](https://docs.oracle.com/javase/8/docs/api/java/lang/String.html) | [**toString**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#toString--)()  Formats a timestamp in JDBC timestamp escape format. |
| static [**Timestamp**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html) | [**valueOf**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#valueOf-java.time.LocalDateTime-)(**[LocalDateTime](https://docs.oracle.com/javase/8/docs/api/java/time/LocalDateTime.html" \o "class in java.time)** dateTime)  Obtains an instance of Timestamp from a LocalDateTime object, with the same year, month, day of month, hours, minutes, seconds and nanos date-time value as the provided LocalDateTime. |
| static [**Timestamp**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html) | [**valueOf**](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#valueOf-java.lang.String-)([**String**](https://docs.oracle.com/javase/8/docs/api/java/lang/String.html) s)  Converts a String object in JDBC timestamp escape format to a Timestamp value. |

* + - **Methods inherited from class java.util.**[**Date**](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html)

[after](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#after-java.util.Date-), [before](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#before-java.util.Date-), [clone](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#clone--), [getDate](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#getDate--), [getDay](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#getDay--), [getHours](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#getHours--), [getMinutes](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#getMinutes--), [getMonth](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#getMonth--), [getSeconds](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#getSeconds--), [getTimezoneOffset](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#getTimezoneOffset--), [getYear](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#getYear--), [parse](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#parse-java.lang.String-), [setDate](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#setDate-int-), [setHours](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#setHours-int-),[setMinutes](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#setMinutes-int-), [setMonth](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#setMonth-int-), [setSeconds](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#setSeconds-int-), [setYear](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#setYear-int-), [toGMTString](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#toGMTString--), [toLocaleString](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#toLocaleString--), [UTC](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#UTC-int-int-int-int-int-int-)

* + - **Methods inherited from class java.lang.**[**Object**](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html)

[finalize](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html#finalize--), [getClass](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html" \l "getClass--), [notify](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html#notify--), [notifyAll](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html" \l "notifyAll--), [wait](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html#wait--), [wait](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html#wait-long-), [wait](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html#wait-long-int-)

* + ***Constructor Detail***
    - **Timestamp**
    - [@Deprecated](https://docs.oracle.com/javase/8/docs/api/java/lang/Deprecated.html)
    - public Timestamp(int year,
    - int month,
    - int date,
    - int hour,
    - int minute,
    - int second,

int nano)

**Deprecated.** *instead use the constructor Timestamp(long millis)*

Constructs a Timestamp object initialized with the given values.

**Parameters:**

year - the year minus 1900

month - 0 to 11

date - 1 to 31

hour - 0 to 23

minute - 0 to 59

second - 0 to 59

nano - 0 to 999,999,999

**Throws:**

[IllegalArgumentException](https://docs.oracle.com/javase/8/docs/api/java/lang/IllegalArgumentException.html) - if the nano argument is out of bounds

* + - **Timestamp**

public Timestamp(long time)

Constructs a Timestamp object using a milliseconds time value. The integral seconds are stored in the underlying date value; the fractional seconds are stored in the nanos field of the Timestamp object.

**Parameters:**

time - milliseconds since January 1, 1970, 00:00:00 GMT. A negative number is the number of milliseconds before January 1, 1970, 00:00:00 GMT.

**See Also:**

[Calendar](https://docs.oracle.com/javase/8/docs/api/java/util/Calendar.html)

* + ***Method Detail***
    - **setTime**

public void setTime(long time)

Sets this Timestamp object to represent a point in time that is time milliseconds after January 1, 1970 00:00:00 GMT.

**Overrides:**

[setTime](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#setTime-long-) in class [Date](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html)

**Parameters:**

time - the number of milliseconds.

**See Also:**

[getTime()](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#getTime--), [Timestamp(long time)](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#Timestamp-long-), [Calendar](https://docs.oracle.com/javase/8/docs/api/java/util/Calendar.html)

* + - **getTime**

public long getTime()

Returns the number of milliseconds since January 1, 1970, 00:00:00 GMT represented by this Timestamp object.

**Overrides:**

[getTime](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#getTime--) in class [Date](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html)

**Returns:**

the number of milliseconds since January 1, 1970, 00:00:00 GMT represented by this date.

**See Also:**

[setTime(long)](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#setTime-long-)

* + - **valueOf**

public static [Timestamp](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html) valueOf([String](https://docs.oracle.com/javase/8/docs/api/java/lang/String.html) s)

Converts a String object in JDBC timestamp escape format to a Timestamp value.

**Parameters:**

s - timestamp in format yyyy-[m]m-[d]d hh:mm:ss[.f...]. The fractional seconds may be omitted. The leading zero for mm and dd may also be omitted.

**Returns:**

corresponding Timestamp value

**Throws:**

[IllegalArgumentException](https://docs.oracle.com/javase/8/docs/api/java/lang/IllegalArgumentException.html) - if the given argument does not have the format yyyy-[m]m-[d]d hh:mm:ss[.f...]

* + - **toString**

public [String](https://docs.oracle.com/javase/8/docs/api/java/lang/String.html) toString()

Formats a timestamp in JDBC timestamp escape format. yyyy-mm-dd hh:mm:ss.fffffffff, where ffffffffff indicates nanoseconds.

**Overrides:**

[toString](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#toString--) in class [Date](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html)

**Returns:**

a String object in yyyy-mm-dd hh:mm:ss.fffffffff format

**See Also:**

[Date.toLocaleString()](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#toLocaleString--), [Date.toGMTString()](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html" \l "toGMTString--)

* + - **getNanos**

public int getNanos()

Gets this Timestamp object's nanos value.

**Returns:**

this Timestamp object's fractional seconds component

**See Also:**

[setNanos(int)](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#setNanos-int-)

* + - **setNanos**

public void setNanos(int n)

Sets this Timestamp object's nanos field to the given value.

**Parameters:**

n - the new fractional seconds component

**Throws:**

[IllegalArgumentException](https://docs.oracle.com/javase/8/docs/api/java/lang/IllegalArgumentException.html) - if the given argument is greater than 999999999 or less than 0

**See Also:**

[getNanos()](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html#getNanos--)

* + - **equals**

public boolean equals([Timestamp](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html) ts)

Tests to see if this Timestamp object is equal to the given Timestamp object.

**Parameters:**

ts - the Timestamp value to compare with

**Returns:**

true if the given Timestamp object is equal to this Timestamp object; false otherwise

* + - **equals**

public boolean equals([Object](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html) ts)

Tests to see if this Timestamp object is equal to the given object. This version of the method equals has been added to fix the incorrect signature of Timestamp.equals(Timestamp) and to preserve backward compatibility with existing class files. Note: This method is not symmetric with respect to theequals(Object) method in the base class.

**Overrides:**

[equals](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#equals-java.lang.Object-) in class [Date](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html)

**Parameters:**

ts - the Object value to compare with

**Returns:**

true if the given Object is an instance of a Timestamp that is equal to this Timestamp object; false otherwise

**See Also:**

[Date.getTime()](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#getTime--)

* + - **before**

public boolean before([Timestamp](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html) ts)

Indicates whether this Timestamp object is earlier than the given Timestamp object.

**Parameters:**

ts - the Timestamp value to compare with

**Returns:**

true if this Timestamp object is earlier; false otherwise

* + - **after**

public boolean after([Timestamp](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html) ts)

Indicates whether this Timestamp object is later than the given Timestamp object.

**Parameters:**

ts - the Timestamp value to compare with

**Returns:**

true if this Timestamp object is later; false otherwise

* + - **compareTo**

public int compareTo([Timestamp](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html) ts)

Compares this Timestamp object to the given Timestamp object.

**Parameters:**

ts - the Timestamp object to be compared to this Timestamp object

**Returns:**

the value 0 if the two Timestamp objects are equal; a value less than 0 if this Timestamp object is before the given argument; and a value greater than 0 if this Timestamp object is after the given argument.

**Since:**

1.4

* + - **compareTo**

public int compareTo([Date](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html) o)

Compares this Timestamp object to the given Date object.

**Specified by:**

[compareTo](https://docs.oracle.com/javase/8/docs/api/java/lang/Comparable.html#compareTo-T-) in interface [Comparable](https://docs.oracle.com/javase/8/docs/api/java/lang/Comparable.html)<[Date](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html)>

**Overrides:**

[compareTo](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#compareTo-java.util.Date-) in class [Date](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html)

**Parameters:**

o - the Date to be compared to this Timestamp object

**Returns:**

the value 0 if this Timestamp object and the given object are equal; a value less than 0 if this Timestamp object is before the given argument; and a value greater than 0 if this Timestamp object is after the given argument.

**Since:**

1.5

* + - **hashCode**

public int hashCode()

Returns a hash code value for this object. The result is the exclusive OR of the two halves of the primitive long value returned by the [Date.getTime()](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html" \l "getTime--) method. That is, the hash code is the value of the expression:

(int)(this.getTime()^(this.getTime() >>> 32))

The hashCode method uses the underlying java.util.Date implementation and therefore does not include nanos in its computation.

**Overrides:**

[hashCode](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#hashCode--) in class [Date](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html)

**Returns:**

a hash code value for this object.

**See Also:**

[Object.equals(java.lang.Object)](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html#equals-java.lang.Object-), [System.identityHashCode(java.lang.Object)](https://docs.oracle.com/javase/8/docs/api/java/lang/System.html#identityHashCode-java.lang.Object-)

* + - **valueOf**

public static [Timestamp](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html) valueOf([LocalDateTime](https://docs.oracle.com/javase/8/docs/api/java/time/LocalDateTime.html" \o "class in java.time) dateTime)

Obtains an instance of Timestamp from a LocalDateTime object, with the same year, month, day of month, hours, minutes, seconds and nanos date-time value as the provided LocalDateTime.

The provided LocalDateTime is interpreted as the local date-time in the local time zone.

**Parameters:**

dateTime - a LocalDateTime to convert

**Returns:**

a Timestamp object

**Throws:**

[NullPointerException](https://docs.oracle.com/javase/8/docs/api/java/lang/NullPointerException.html) - if dateTime is null.

**Since:**

1.8

* + - **toLocalDateTime**

public [LocalDateTime](https://docs.oracle.com/javase/8/docs/api/java/time/LocalDateTime.html" \o "class in java.time) toLocalDateTime()

Converts this Timestamp object to a LocalDateTime.

The conversion creates a LocalDateTime that represents the same year, month, day of month, hours, minutes, seconds and nanos date-time value as this Timestampin the local time zone.

**Returns:**

a LocalDateTime object representing the same date-time value

**Since:**

1.8

* + - **from**

public static [Timestamp](https://docs.oracle.com/javase/8/docs/api/java/sql/Timestamp.html) from([Instant](https://docs.oracle.com/javase/8/docs/api/java/time/Instant.html) instant)

Obtains an instance of Timestamp from an [Instant](https://docs.oracle.com/javase/8/docs/api/java/time/Instant.html) object.

Instant can store points on the time-line further in the future and further in the past than Date. In this scenario, this method will throw an exception.

**Parameters:**

instant - the instant to convert

**Returns:**

an Timestamp representing the same point on the time-line as the provided instant

**Throws:**

[NullPointerException](https://docs.oracle.com/javase/8/docs/api/java/lang/NullPointerException.html) - if instant is null.

[IllegalArgumentException](https://docs.oracle.com/javase/8/docs/api/java/lang/IllegalArgumentException.html) - if the instant is too large to represent as a Timesamp

**Since:**

1.8

* + - **toInstant**

public [Instant](https://docs.oracle.com/javase/8/docs/api/java/time/Instant.html) toInstant()

Converts this Timestamp object to an Instant.

The conversion creates an Instant that represents the same point on the time-line as this Timestamp.

**Overrides:**

[toInstant](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html#toInstant--) in class [Date](https://docs.oracle.com/javase/8/docs/api/java/util/Date.html)

**Returns:**

an instant representing the same point on the time-line

**Since:**

1.8