| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Duration.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/javax/xml/datatype/DatatypeFactory.html)   [**NEXT CLASS**](http://docs.google.com/javax/xml/datatype/XMLGregorianCalendar.html) | [**FRAMES**](http://docs.google.com/index.html?javax/xml/datatype/Duration.html)    [**NO FRAMES**](http://docs.google.com/Duration.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#4d34og8) |

## **javax.xml.datatype**

Class Duration

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 **javax.xml.datatype.Duration**

public abstract class **Duration**extends [Object](http://docs.google.com/java/lang/Object.html)

Immutable representation of a time span as defined in the W3C XML Schema 1.0 specification.

A Duration object represents a period of Gregorian time, which consists of six fields (years, months, days, hours, minutes, and seconds) plus a sign (+/-) field.

The first five fields have non-negative (>=0) integers or null (which represents that the field is not set), and the seconds field has a non-negative decimal or null. A negative sign indicates a negative duration.

This class provides a number of methods that make it easy to use for the duration datatype of XML Schema 1.0 with the errata.

## Order relationship

Duration objects only have partial order, where two values A and B maybe either:

1. A<B (A is shorter than B)
2. A>B (A is longer than B)
3. A==B (A and B are of the same duration)
4. A<>B (Comparison between A and B is indeterminate)

For example, 30 days cannot be meaningfully compared to one month. The [compare(Duration duration)](http://docs.google.com/javax/xml/datatype/Duration.html#compare(javax.xml.datatype.Duration)) method implements this relationship.

See the [isLongerThan(Duration)](http://docs.google.com/javax/xml/datatype/Duration.html#isLongerThan(javax.xml.datatype.Duration)) method for details about the order relationship among Duration objects.

## Operations over Duration

This class provides a set of basic arithmetic operations, such as addition, subtraction and multiplication. Because durations don't have total order, an operation could fail for some combinations of operations. For example, you cannot subtract 15 days from 1 month. See the javadoc of those methods for detailed conditions where this could happen.

Also, division of a duration by a number is not provided because the Duration class can only deal with finite precision decimal numbers. For example, one cannot represent 1 sec divided by 3.

However, you could substitute a division by 3 with multiplying by numbers such as 0.3 or 0.333.

## Range of allowed values

Because some operations of Duration rely on [Calendar](http://docs.google.com/java/util/Calendar.html) even though [Duration](http://docs.google.com/javax/xml/datatype/Duration.html) can hold very large or very small values, some of the methods may not work correctly on such Durations. The impacted methods document their dependency on [Calendar](http://docs.google.com/java/util/Calendar.html).

**Since:** 1.5 **See Also:**[XMLGregorianCalendar.add(Duration)](http://docs.google.com/javax/xml/datatype/XMLGregorianCalendar.html#add(javax.xml.datatype.Duration))

| **Constructor Summary** | |
| --- | --- |
| [**Duration**](http://docs.google.com/javax/xml/datatype/Duration.html#Duration())()            Default no-arg constructor. |

| **Method Summary** | |
| --- | --- |
| abstract  [Duration](http://docs.google.com/javax/xml/datatype/Duration.html) | [**add**](http://docs.google.com/javax/xml/datatype/Duration.html#add(javax.xml.datatype.Duration))([Duration](http://docs.google.com/javax/xml/datatype/Duration.html) rhs)            Computes a new duration whose value is this+rhs. |
| abstract  void | [**addTo**](http://docs.google.com/javax/xml/datatype/Duration.html#addTo(java.util.Calendar))([Calendar](http://docs.google.com/java/util/Calendar.html) calendar)            Adds this duration to a [Calendar](http://docs.google.com/java/util/Calendar.html) object. |
| void | [**addTo**](http://docs.google.com/javax/xml/datatype/Duration.html#addTo(java.util.Date))([Date](http://docs.google.com/java/util/Date.html) date)            Adds this duration to a [Date](http://docs.google.com/java/util/Date.html) object. |
| abstract  int | [**compare**](http://docs.google.com/javax/xml/datatype/Duration.html#compare(javax.xml.datatype.Duration))([Duration](http://docs.google.com/javax/xml/datatype/Duration.html) duration)            Partial order relation comparison with this Duration instance. |
| boolean | [**equals**](http://docs.google.com/javax/xml/datatype/Duration.html#equals(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) duration)            Checks if this duration object has the same duration as another Duration object. |
| int | [**getDays**](http://docs.google.com/javax/xml/datatype/Duration.html#getDays())()            Obtains the value of the DAYS field as an integer value, or 0 if not present. |
| abstract  [Number](http://docs.google.com/java/lang/Number.html) | [**getField**](http://docs.google.com/javax/xml/datatype/Duration.html#getField(javax.xml.datatype.DatatypeConstants.Field))([DatatypeConstants.Field](http://docs.google.com/javax/xml/datatype/DatatypeConstants.Field.html) field)            Gets the value of a field. |
| int | [**getHours**](http://docs.google.com/javax/xml/datatype/Duration.html#getHours())()            Obtains the value of the HOURS field as an integer value, or 0 if not present. |
| int | [**getMinutes**](http://docs.google.com/javax/xml/datatype/Duration.html#getMinutes())()            Obtains the value of the MINUTES field as an integer value, or 0 if not present. |
| int | [**getMonths**](http://docs.google.com/javax/xml/datatype/Duration.html#getMonths())()            Obtains the value of the MONTHS field as an integer value, or 0 if not present. |
| int | [**getSeconds**](http://docs.google.com/javax/xml/datatype/Duration.html#getSeconds())()            Obtains the value of the SECONDS field as an integer value, or 0 if not present. |
| abstract  int | [**getSign**](http://docs.google.com/javax/xml/datatype/Duration.html#getSign())()            Returns the sign of this duration in -1,0, or 1. |
| long | [**getTimeInMillis**](http://docs.google.com/javax/xml/datatype/Duration.html#getTimeInMillis(java.util.Calendar))([Calendar](http://docs.google.com/java/util/Calendar.html) startInstant)            Returns the length of the duration in milli-seconds. |
| long | [**getTimeInMillis**](http://docs.google.com/javax/xml/datatype/Duration.html#getTimeInMillis(java.util.Date))([Date](http://docs.google.com/java/util/Date.html) startInstant)            Returns the length of the duration in milli-seconds. |
| [QName](http://docs.google.com/javax/xml/namespace/QName.html) | [**getXMLSchemaType**](http://docs.google.com/javax/xml/datatype/Duration.html#getXMLSchemaType())()            Return the name of the XML Schema date/time type that this instance maps to. |
| int | [**getYears**](http://docs.google.com/javax/xml/datatype/Duration.html#getYears())()            Get the years value of this Duration as an int or 0 if not present. |
| abstract  int | [**hashCode**](http://docs.google.com/javax/xml/datatype/Duration.html#hashCode())()            Returns a hash code consistent with the definition of the equals method. |
| boolean | [**isLongerThan**](http://docs.google.com/javax/xml/datatype/Duration.html#isLongerThan(javax.xml.datatype.Duration))([Duration](http://docs.google.com/javax/xml/datatype/Duration.html) duration)            Checks if this duration object is strictly longer than another Duration object. |
| abstract  boolean | [**isSet**](http://docs.google.com/javax/xml/datatype/Duration.html#isSet(javax.xml.datatype.DatatypeConstants.Field))([DatatypeConstants.Field](http://docs.google.com/javax/xml/datatype/DatatypeConstants.Field.html) field)            Checks if a field is set. |
| boolean | [**isShorterThan**](http://docs.google.com/javax/xml/datatype/Duration.html#isShorterThan(javax.xml.datatype.Duration))([Duration](http://docs.google.com/javax/xml/datatype/Duration.html) duration)            Checks if this duration object is strictly shorter than another Duration object. |
| abstract  [Duration](http://docs.google.com/javax/xml/datatype/Duration.html) | [**multiply**](http://docs.google.com/javax/xml/datatype/Duration.html#multiply(java.math.BigDecimal))([BigDecimal](http://docs.google.com/java/math/BigDecimal.html) factor)            Computes a new duration whose value is factor times longer than the value of this duration. |
| [Duration](http://docs.google.com/javax/xml/datatype/Duration.html) | [**multiply**](http://docs.google.com/javax/xml/datatype/Duration.html#multiply(int))(int factor)            Computes a new duration whose value is factor times longer than the value of this duration. |
| abstract  [Duration](http://docs.google.com/javax/xml/datatype/Duration.html) | [**negate**](http://docs.google.com/javax/xml/datatype/Duration.html#negate())()            Returns a new Duration object whose value is -this. |
| abstract  [Duration](http://docs.google.com/javax/xml/datatype/Duration.html) | [**normalizeWith**](http://docs.google.com/javax/xml/datatype/Duration.html#normalizeWith(java.util.Calendar))([Calendar](http://docs.google.com/java/util/Calendar.html) startTimeInstant)            Converts the years and months fields into the days field by using a specific time instant as the reference point. |
| [Duration](http://docs.google.com/javax/xml/datatype/Duration.html) | [**subtract**](http://docs.google.com/javax/xml/datatype/Duration.html#subtract(javax.xml.datatype.Duration))([Duration](http://docs.google.com/javax/xml/datatype/Duration.html) rhs)            Computes a new duration whose value is this-rhs. |
| [String](http://docs.google.com/java/lang/String.html) | [**toString**](http://docs.google.com/javax/xml/datatype/Duration.html#toString())()            Returns a String representation of this Duration Object. |

| **Methods inherited from class java.lang.**[**Object**](http://docs.google.com/java/lang/Object.html) |
| --- |
| [clone](http://docs.google.com/java/lang/Object.html#clone()), [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)) |

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

### Duration

public **Duration**()

Default no-arg constructor.

Note: Always use the [DatatypeFactory](http://docs.google.com/javax/xml/datatype/DatatypeFactory.html) to construct an instance of Duration. The constructor on this class cannot be guaranteed to produce an object with a consistent state and may be removed in the future.

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

### getXMLSchemaType

public [QName](http://docs.google.com/javax/xml/namespace/QName.html) **getXMLSchemaType**()

Return the name of the XML Schema date/time type that this instance maps to. Type is computed based on fields that are set, i.e. [isSet(DatatypeConstants.Field field)](http://docs.google.com/javax/xml/datatype/Duration.html#isSet(javax.xml.datatype.DatatypeConstants.Field)) == true.

| Required fields for XML Schema 1.0 Date/Time Datatypes.  *(timezone is optional for all date/time datatypes)* | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| Datatype | year | month | day | hour | minute | second |
| [DatatypeConstants.DURATION](http://docs.google.com/javax/xml/datatype/DatatypeConstants.html#DURATION) | X | X | X | X | X | X |
| [DatatypeConstants.DURATION\_DAYTIME](http://docs.google.com/javax/xml/datatype/DatatypeConstants.html#DURATION_DAYTIME) |  |  | X | X | X | X |
| [DatatypeConstants.DURATION\_YEARMONTH](http://docs.google.com/javax/xml/datatype/DatatypeConstants.html#DURATION_YEARMONTH) | X | X |  |  |  |  |

**Returns:**one of the following constants: [DatatypeConstants.DURATION](http://docs.google.com/javax/xml/datatype/DatatypeConstants.html#DURATION), [DatatypeConstants.DURATION\_DAYTIME](http://docs.google.com/javax/xml/datatype/DatatypeConstants.html#DURATION_DAYTIME) or [DatatypeConstants.DURATION\_YEARMONTH](http://docs.google.com/javax/xml/datatype/DatatypeConstants.html#DURATION_YEARMONTH). **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - If the combination of set fields does not match one of the XML Schema date/time datatypes.

### getSign

public abstract int **getSign**()

Returns the sign of this duration in -1,0, or 1.

**Returns:**-1 if this duration is negative, 0 if the duration is zero, and 1 if the duration is positive.

### getYears

public int **getYears**()

Get the years value of this Duration as an int or 0 if not present.

getYears() is a convenience method for [getField(DatatypeConstants.YEARS)](http://docs.google.com/javax/xml/datatype/Duration.html#getField(javax.xml.datatype.DatatypeConstants.Field)).

As the return value is an int, an incorrect value will be returned for Durations with years that go beyond the range of an int. Use [getField(DatatypeConstants.YEARS)](http://docs.google.com/javax/xml/datatype/Duration.html#getField(javax.xml.datatype.DatatypeConstants.Field)) to avoid possible loss of precision.

**Returns:**If the years field is present, return its value as an int, else return 0.

### getMonths

public int **getMonths**()

Obtains the value of the MONTHS field as an integer value, or 0 if not present. This method works just like [getYears()](http://docs.google.com/javax/xml/datatype/Duration.html#getYears()) except that this method works on the MONTHS field.

**Returns:**Months of this Duration.

### getDays

public int **getDays**()

Obtains the value of the DAYS field as an integer value, or 0 if not present. This method works just like [getYears()](http://docs.google.com/javax/xml/datatype/Duration.html#getYears()) except that this method works on the DAYS field.

**Returns:**Days of this Duration.

### getHours

public int **getHours**()

Obtains the value of the HOURS field as an integer value, or 0 if not present. This method works just like [getYears()](http://docs.google.com/javax/xml/datatype/Duration.html#getYears()) except that this method works on the HOURS field.

**Returns:**Hours of this Duration.

### getMinutes

public int **getMinutes**()

Obtains the value of the MINUTES field as an integer value, or 0 if not present. This method works just like [getYears()](http://docs.google.com/javax/xml/datatype/Duration.html#getYears()) except that this method works on the MINUTES field.

**Returns:**Minutes of this Duration.

### getSeconds

public int **getSeconds**()

Obtains the value of the SECONDS field as an integer value, or 0 if not present. This method works just like [getYears()](http://docs.google.com/javax/xml/datatype/Duration.html#getYears()) except that this method works on the SECONDS field.

**Returns:**seconds in the integer value. The fraction of seconds will be discarded (for example, if the actual value is 2.5, this method returns 2)

### getTimeInMillis

public long **getTimeInMillis**([Calendar](http://docs.google.com/java/util/Calendar.html) startInstant)

Returns the length of the duration in milli-seconds.

If the seconds field carries more digits than milli-second order, those will be simply discarded (or in other words, rounded to zero.) For example, for any Calendar value x,

new Duration("PT10.00099S").getTimeInMills(x) == 10000.  
 new Duration("-PT10.00099S").getTimeInMills(x) == -10000.

Note that this method uses the [addTo(Calendar)](http://docs.google.com/javax/xml/datatype/Duration.html#addTo(java.util.Calendar)) method, which may work incorrectly with Duration objects with very large values in its fields. See the [addTo(Calendar)](http://docs.google.com/javax/xml/datatype/Duration.html#addTo(java.util.Calendar)) method for details.

**Parameters:**startInstant - The length of a month/year varies. The startInstant is used to disambiguate this variance. Specifically, this method returns the difference between startInstant and startInstant+duration **Returns:**milliseconds between startInstant and startInstant plus this Duration **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if startInstant parameter is null.

### getTimeInMillis

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

Returns the length of the duration in milli-seconds.

If the seconds field carries more digits than milli-second order, those will be simply discarded (or in other words, rounded to zero.) For example, for any Date value x,

new Duration("PT10.00099S").getTimeInMills(x) == 10000.  
 new Duration("-PT10.00099S").getTimeInMills(x) == -10000.

Note that this method uses the [addTo(Date)](http://docs.google.com/javax/xml/datatype/Duration.html#addTo(java.util.Date)) method, which may work incorrectly with Duration objects with very large values in its fields. See the [addTo(Date)](http://docs.google.com/javax/xml/datatype/Duration.html#addTo(java.util.Date)) method for details.

**Parameters:**startInstant - The length of a month/year varies. The startInstant is used to disambiguate this variance. Specifically, this method returns the difference between startInstant and startInstant+duration. **Returns:**milliseconds between startInstant and startInstant plus this Duration **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - If the startInstant parameter is null.**See Also:**[getTimeInMillis(Calendar)](http://docs.google.com/javax/xml/datatype/Duration.html#getTimeInMillis(java.util.Calendar))

### getField

public abstract [Number](http://docs.google.com/java/lang/Number.html) **getField**([DatatypeConstants.Field](http://docs.google.com/javax/xml/datatype/DatatypeConstants.Field.html) field)

Gets the value of a field. Fields of a duration object may contain arbitrary large value. Therefore this method is designed to return a [Number](http://docs.google.com/java/lang/Number.html) object. In case of YEARS, MONTHS, DAYS, HOURS, and MINUTES, the returned number will be a non-negative integer. In case of seconds, the returned number may be a non-negative decimal value.

**Parameters:**field - one of the six Field constants (YEARS,MONTHS,DAYS,HOURS, MINUTES, or SECONDS.) **Returns:**If the specified field is present, this method returns a non-null non-negative [Number](http://docs.google.com/java/lang/Number.html) object that represents its value. If it is not present, return null. For YEARS, MONTHS, DAYS, HOURS, and MINUTES, this method returns a [BigInteger](http://docs.google.com/java/math/BigInteger.html) object. For SECONDS, this method returns a [BigDecimal](http://docs.google.com/java/math/BigDecimal.html). **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - If the field is null.

### isSet

public abstract boolean **isSet**([DatatypeConstants.Field](http://docs.google.com/javax/xml/datatype/DatatypeConstants.Field.html) field)

Checks if a field is set. A field of a duration object may or may not be present. This method can be used to test if a field is present.

**Parameters:**field - one of the six Field constants (YEARS,MONTHS,DAYS,HOURS, MINUTES, or SECONDS.) **Returns:**true if the field is present. false if not. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - If the field parameter is null.

### add

public abstract [Duration](http://docs.google.com/javax/xml/datatype/Duration.html) **add**([Duration](http://docs.google.com/javax/xml/datatype/Duration.html) rhs)

Computes a new duration whose value is this+rhs.

For example,

"1 day" + "-3 days" = "-2 days"  
 "1 year" + "1 day" = "1 year and 1 day"  
 "-(1 hour,50 minutes)" + "-20 minutes" = "-(1 hours,70 minutes)"  
 "15 hours" + "-3 days" = "-(2 days,9 hours)"  
 "1 year" + "-1 day" = IllegalStateException

Since there's no way to meaningfully subtract 1 day from 1 month, there are cases where the operation fails in [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html).

Formally, the computation is defined as follows.

Firstly, we can assume that two Durations to be added are both positive without losing generality (i.e., (-X)+Y=Y-X, X+(-Y)=X-Y, (-X)+(-Y)=-(X+Y))

Addition of two positive Durations are simply defined as field by field addition where missing fields are treated as 0.

A field of the resulting Duration will be unset if and only if respective fields of two input Durations are unset.

Note that lhs.add(rhs) will be always successful if lhs.signum()\*rhs.signum()!=-1 or both of them are normalized.

**Parameters:**rhs - Duration to add to this Duration **Returns:**non-null valid Duration object. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - If the rhs parameter is null. [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - If two durations cannot be meaningfully added. For example, adding negative one day to one month causes this exception.**See Also:**[subtract(Duration)](http://docs.google.com/javax/xml/datatype/Duration.html#subtract(javax.xml.datatype.Duration))

### addTo

public abstract void **addTo**([Calendar](http://docs.google.com/java/util/Calendar.html) calendar)

Adds this duration to a [Calendar](http://docs.google.com/java/util/Calendar.html) object.

Calls [Calendar.add(int,int)](http://docs.google.com/java/util/Calendar.html#add(int,%20int)) in the order of YEARS, MONTHS, DAYS, HOURS, MINUTES, SECONDS, and MILLISECONDS if those fields are present. Because the [Calendar](http://docs.google.com/java/util/Calendar.html) class uses int to hold values, there are cases where this method won't work correctly (for example if values of fields exceed the range of int.)

Also, since this duration class is a Gregorian duration, this method will not work correctly if the given [Calendar](http://docs.google.com/java/util/Calendar.html) object is based on some other calendar systems.

Any fractional parts of this Duration object beyond milliseconds will be simply ignored. For example, if this duration is "P1.23456S", then 1 is added to SECONDS, 234 is added to MILLISECONDS, and the rest will be unused.

Note that because [Calendar.add(int, int)](http://docs.google.com/java/util/Calendar.html#add(int,%20int)) is using int, Duration with values beyond the range of int in its fields will cause overflow/underflow to the given [Calendar](http://docs.google.com/java/util/Calendar.html). [XMLGregorianCalendar.add(Duration)](http://docs.google.com/javax/xml/datatype/XMLGregorianCalendar.html#add(javax.xml.datatype.Duration)) provides the same basic operation as this method while avoiding the overflow/underflow issues.

**Parameters:**calendar - A calendar object whose value will be modified. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the calendar parameter is null.

### addTo

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

Adds this duration to a [Date](http://docs.google.com/java/util/Date.html) object.

The given date is first converted into a [GregorianCalendar](http://docs.google.com/java/util/GregorianCalendar.html), then the duration is added exactly like the [addTo(Calendar)](http://docs.google.com/javax/xml/datatype/Duration.html#addTo(java.util.Calendar)) method.

The updated time instant is then converted back into a [Date](http://docs.google.com/java/util/Date.html) object and used to update the given [Date](http://docs.google.com/java/util/Date.html) object.

This somewhat redundant computation is necessary to unambiguously determine the duration of months and years.

**Parameters:**date - A date object whose value will be modified. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the date parameter is null.

### subtract

public [Duration](http://docs.google.com/javax/xml/datatype/Duration.html) **subtract**([Duration](http://docs.google.com/javax/xml/datatype/Duration.html) rhs)

Computes a new duration whose value is this-rhs.

For example:

"1 day" - "-3 days" = "4 days"  
 "1 year" - "1 day" = IllegalStateException  
 "-(1 hour,50 minutes)" - "-20 minutes" = "-(1hours,30 minutes)"  
 "15 hours" - "-3 days" = "3 days and 15 hours"  
 "1 year" - "-1 day" = "1 year and 1 day"

Since there's no way to meaningfully subtract 1 day from 1 month, there are cases where the operation fails in [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html).

Formally the computation is defined as follows. First, we can assume that two Durations are both positive without losing generality. (i.e., (-X)-Y=-(X+Y), X-(-Y)=X+Y, (-X)-(-Y)=-(X-Y))

Then two durations are subtracted field by field. If the sign of any non-zero field F is different from the sign of the most significant field, 1 (if F is negative) or -1 (otherwise) will be borrowed from the next bigger unit of F.

This process is repeated until all the non-zero fields have the same sign.

If a borrow occurs in the days field (in other words, if the computation needs to borrow 1 or -1 month to compensate days), then the computation fails by throwing an [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html).

**Parameters:**rhs - Duration to subtract from this Duration. **Returns:**New Duration created from subtracting rhs from this Duration. **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - If two durations cannot be meaningfully subtracted. For example, subtracting one day from one month causes this exception. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - If the rhs parameter is null.**See Also:**[add(Duration)](http://docs.google.com/javax/xml/datatype/Duration.html#add(javax.xml.datatype.Duration))

### multiply

public [Duration](http://docs.google.com/javax/xml/datatype/Duration.html) **multiply**(int factor)

Computes a new duration whose value is factor times longer than the value of this duration.

This method is provided for the convenience. It is functionally equivalent to the following code:

multiply(new BigDecimal(String.valueOf(factor)))

**Parameters:**factor - Factor times longer of new Duration to create. **Returns:**New Duration that is factortimes longer than this Duration.**See Also:**[multiply(BigDecimal)](http://docs.google.com/javax/xml/datatype/Duration.html#multiply(java.math.BigDecimal))

### multiply

public abstract [Duration](http://docs.google.com/javax/xml/datatype/Duration.html) **multiply**([BigDecimal](http://docs.google.com/java/math/BigDecimal.html) factor)

Computes a new duration whose value is factor times longer than the value of this duration.

For example,

"P1M" (1 month) \* "12" = "P12M" (12 months)  
 "PT1M" (1 min) \* "0.3" = "PT18S" (18 seconds)  
 "P1M" (1 month) \* "1.5" = IllegalStateException

Since the Duration class is immutable, this method doesn't change the value of this object. It simply computes a new Duration object and returns it.

The operation will be performed field by field with the precision of [BigDecimal](http://docs.google.com/java/math/BigDecimal.html). Since all the fields except seconds are restricted to hold integers, any fraction produced by the computation will be carried down toward the next lower unit. For example, if you multiply "P1D" (1 day) with "0.5", then it will be 0.5 day, which will be carried down to "PT12H" (12 hours). When fractions of month cannot be meaningfully carried down to days, or year to months, this will cause an [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) to be thrown. For example if you multiple one month by 0.5.

To avoid [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html), use the [normalizeWith(Calendar)](http://docs.google.com/javax/xml/datatype/Duration.html#normalizeWith(java.util.Calendar)) method to remove the years and months fields.

**Parameters:**factor - to multiply by **Returns:**returns a non-null valid Duration object **Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if operation produces fraction in the months field. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the factor parameter is null.

### negate

public abstract [Duration](http://docs.google.com/javax/xml/datatype/Duration.html) **negate**()

Returns a new Duration object whose value is -this.

Since the Duration class is immutable, this method doesn't change the value of this object. It simply computes a new Duration object and returns it.

**Returns:**always return a non-null valid Duration object.

### normalizeWith

public abstract [Duration](http://docs.google.com/javax/xml/datatype/Duration.html) **normalizeWith**([Calendar](http://docs.google.com/java/util/Calendar.html) startTimeInstant)

Converts the years and months fields into the days field by using a specific time instant as the reference point.

For example, duration of one month normalizes to 31 days given the start time instance "July 8th 2003, 17:40:32".

Formally, the computation is done as follows:

1. the given Calendar object is cloned
2. the years, months and days fields will be added to the [Calendar](http://docs.google.com/java/util/Calendar.html) object by using the [Calendar.add(int,int)](http://docs.google.com/java/util/Calendar.html#add(int,%20int)) method
3. the difference between the two Calendars in computed in milliseconds and converted to days, if a remainder occurs due to Daylight Savings Time, it is discarded
4. the computed days, along with the hours, minutes and seconds fields of this duration object is used to construct a new Duration object.

Note that since the Calendar class uses int to hold the value of year and month, this method may produce an unexpected result if this duration object holds a very large value in the years or months fields.

**Parameters:**startTimeInstant - Calendar reference point. **Returns:**Duration of years and months of this Duration as days. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - If the startTimeInstant parameter is null.

### compare

public abstract int **compare**([Duration](http://docs.google.com/javax/xml/datatype/Duration.html) duration)

Partial order relation comparison with this Duration instance.

Comparison result must be in accordance with [W3C XML Schema 1.0 Part 2, Section 3.2.7.6.2, *Order relation on duration*](http://www.w3.org/TR/xmlschema-2/#duration-order).

Return:

* [DatatypeConstants.LESSER](http://docs.google.com/javax/xml/datatype/DatatypeConstants.html#LESSER) if this Duration is shorter than duration parameter
* [DatatypeConstants.EQUAL](http://docs.google.com/javax/xml/datatype/DatatypeConstants.html#EQUAL) if this Duration is equal to duration parameter
* [DatatypeConstants.GREATER](http://docs.google.com/javax/xml/datatype/DatatypeConstants.html#GREATER) if this Duration is longer than duration parameter
* [DatatypeConstants.INDETERMINATE](http://docs.google.com/javax/xml/datatype/DatatypeConstants.html#INDETERMINATE) if a conclusive partial order relation cannot be determined

**Parameters:**duration - to compare **Returns:**the relationship between this Durationand duration parameter as [DatatypeConstants.LESSER](http://docs.google.com/javax/xml/datatype/DatatypeConstants.html#LESSER), [DatatypeConstants.EQUAL](http://docs.google.com/javax/xml/datatype/DatatypeConstants.html#EQUAL), [DatatypeConstants.GREATER](http://docs.google.com/javax/xml/datatype/DatatypeConstants.html#GREATER) or [DatatypeConstants.INDETERMINATE](http://docs.google.com/javax/xml/datatype/DatatypeConstants.html#INDETERMINATE). **Throws:** [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - If the underlying implementation cannot reasonably process the request, e.g. W3C XML Schema allows for arbitrarily large/small/precise values, the request may be beyond the implementations capability. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if duration is null.**See Also:**[isShorterThan(Duration)](http://docs.google.com/javax/xml/datatype/Duration.html#isShorterThan(javax.xml.datatype.Duration)), [isLongerThan(Duration)](http://docs.google.com/javax/xml/datatype/Duration.html#isLongerThan(javax.xml.datatype.Duration))

### isLongerThan

public boolean **isLongerThan**([Duration](http://docs.google.com/javax/xml/datatype/Duration.html) duration)

Checks if this duration object is strictly longer than another Duration object.

Duration X is "longer" than Y if and only if X>Y as defined in the section 3.2.6.2 of the XML Schema 1.0 specification.

For example, "P1D" (one day) > "PT12H" (12 hours) and "P2Y" (two years) > "P23M" (23 months).

**Parameters:**duration - Duration to test this Duration against. **Returns:**true if the duration represented by this object is longer than the given duration. false otherwise. **Throws:** [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - If the underlying implementation cannot reasonably process the request, e.g. W3C XML Schema allows for arbitrarily large/small/precise values, the request may be beyond the implementations capability. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - If duration is null.**See Also:**[isShorterThan(Duration)](http://docs.google.com/javax/xml/datatype/Duration.html#isShorterThan(javax.xml.datatype.Duration)), [compare(Duration duration)](http://docs.google.com/javax/xml/datatype/Duration.html#compare(javax.xml.datatype.Duration))

### isShorterThan

public boolean **isShorterThan**([Duration](http://docs.google.com/javax/xml/datatype/Duration.html) duration)

Checks if this duration object is strictly shorter than another Duration object.

**Parameters:**duration - Duration to test this Duration against. **Returns:**true if duration parameter is shorter than this Duration, else false. **Throws:** [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - If the underlying implementation cannot reasonably process the request, e.g. W3C XML Schema allows for arbitrarily large/small/precise values, the request may be beyond the implementations capability. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if duration is null.**See Also:**[isLongerThan(Duration duration)](http://docs.google.com/javax/xml/datatype/Duration.html#isLongerThan(javax.xml.datatype.Duration)), [compare(Duration duration)](http://docs.google.com/javax/xml/datatype/Duration.html#compare(javax.xml.datatype.Duration))

### equals

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

Checks if this duration object has the same duration as another Duration object.

For example, "P1D" (1 day) is equal to "PT24H" (24 hours).

Duration X is equal to Y if and only if time instant t+X and t+Y are the same for all the test time instants specified in the section 3.2.6.2 of the XML Schema 1.0 specification.

Note that there are cases where two Durations are "incomparable" to each other, like one month and 30 days. For example,

!new Duration("P1M").isShorterThan(new Duration("P30D"))  
 !new Duration("P1M").isLongerThan(new Duration("P30D"))  
 !new Duration("P1M").equals(new Duration("P30D"))

**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:**duration - The object to compare this Duration against. **Returns:**true if this duration is the same length as duration. false if duration is null, is not a Duration object, or its length is different from this duration. **Throws:** [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - If the underlying implementation cannot reasonably process the request, e.g. W3C XML Schema allows for arbitrarily large/small/precise values, the request may be beyond the implementations capability.**See Also:**[compare(Duration duration)](http://docs.google.com/javax/xml/datatype/Duration.html#compare(javax.xml.datatype.Duration))

### hashCode

public abstract int **hashCode**()

Returns a hash code consistent with the definition of the equals method.

**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.**See Also:**[Object.hashCode()](http://docs.google.com/java/lang/Object.html#hashCode())

### toString

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

Returns a String representation of this Duration Object.

The result is formatted according to the XML Schema 1.0 spec and can be always parsed back later into the equivalent Duration Object by [DatatypeFactory.newDuration(String lexicalRepresentation)](http://docs.google.com/javax/xml/datatype/DatatypeFactory.html#newDuration(java.lang.String)).

Formally, the following holds for any Duration Object x:

new Duration(x.toString()).equals(x)

**Overrides:**[toString](http://docs.google.com/java/lang/Object.html#toString()) in class [Object](http://docs.google.com/java/lang/Object.html) **Returns:**A non-null valid String representation of this Duration.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Duration.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/javax/xml/datatype/DatatypeFactory.html)   [**NEXT CLASS**](http://docs.google.com/javax/xml/datatype/XMLGregorianCalendar.html) | [**FRAMES**](http://docs.google.com/index.html?javax/xml/datatype/Duration.html)    [**NO FRAMES**](http://docs.google.com/Duration.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#4d34og8) |

[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).