TimeUnit类介绍

# TimeUnit类介绍

## 包及继承实现体系

public enum TimeUnit **extends Enum**<**TimeUnit**>

**TimeUnit类**存在于**java.util.concurrent包**中，属于并发包中的一个类，直接继承java.lang.Enum<TimeUnit>（枚举）。TimeUnit没有构造方法，只能通过枚举元素获取**TimeUnit**对象。一共有七种TimeUnit时间单位对象：**天、时、分、秒、毫秒、微妙、纳秒**

**java.lang.Object**

**java.lang.Enum<TimeUnit>**

**java.util.concurrent.TimeUnit**

All Implemented Interfaces:

**Serializable, Comparable<TimeUnit>**

**Enum枚举类：**

public abstract class Enum<E extends Enum<E>> extends **Object** implements **Comparable<E>, Serializable**

其中，java.util.concurrent包详见单独文件介绍。

## TimeUnit类的描述

A **TimeUnit** represents **time durations** at a given unit of granularity and provides utility methods to convert across units, and to perform timing and delay operations in these units. **A TimeUnit does not maintain time information, but only helps organize and use time representations that may be maintained separately across various contexts.**

## 使用示例

A TimeUnit is mainly used to **inform time-based methods** how a given timing parameter should be interpreted. For example, the following code will timeout in 50 milliseconds if the lock is not available:

Lock lock = ...;

if (lock.tryLock(50L, **TimeUnit.MILLISECONDS**)) ...

while this code will timeout in 50 seconds:

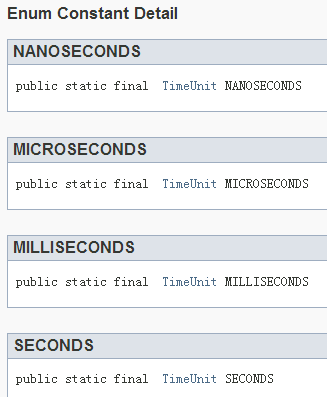
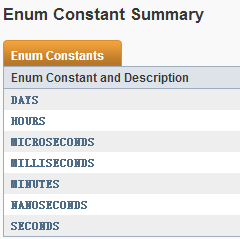
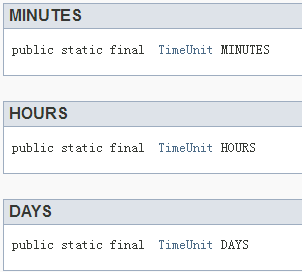
Lock lock = ...;

if (lock.tryLock(50L, **TimeUnit.SECONDS**)) ...

**Note** however, that **there is no guarantee that** a particular timeout implementation will be able to notice the passage of time at the same granularity as the given TimeUnit.

## Enum Constants

一共七个时间单位元素：**天、时、分、秒、毫秒、微妙、纳秒**

## 方法介绍

### **convert(long l,TimeUnit tu)**

**long** convert(long sourceDuration, TimeUnit sourceUnit)

功能：Convert **the given time duration in the given unit** to this unit.

参数:sourceDuration表示给定时间长度，sourceUnit表示给定时间单位，将给定的时间转换为当前时间单位下的时间长度。如：

TimeUnit.SECONDS.convert(8000,TimeUnit.MILLISECONDS);//8

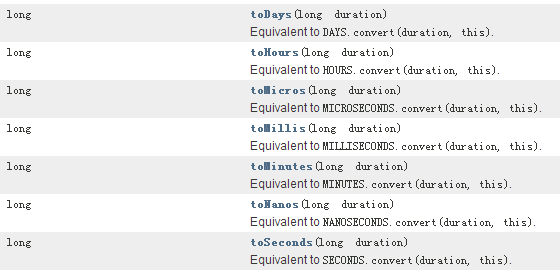
TimeUnit.DAYS.convert(20,TimeUnit.HOURS);//0

TimeUnit.MINUTES.convert(59,TimeUnit.SECONDS);//0

注意：**返回类型为long，只返回结果值的整数部分，并不是四舍五入**。

### toXxx(long duration)

一共7个时间单位，所以有7个toXxx(long duration)方法，其功能相当于**convert(long duration,this)**。



### sleep(long timeout)：睡眠方法，时间单位为当前TimeUnit的类型。

void **sleep(long timeout)**

Performs a Thread.sleep using this time unit.

### Values()、valueOf(String name):两个静态方法。

static TimeUnit **valueOf**(**String name**)

Returns the enum constant of this type with the specified name.

static TimeUnit[] **values()**

Returns an array containing the constants of this enum type, in the order they are declared.

示例：TimeUnit.valueOf("HOURS").toDays(24);//1

TimeUnit.values().length;//7

### **timedJoin(Thread thread, long timeout)**

void **timedJoin**(**Thread thread**, long timeout)

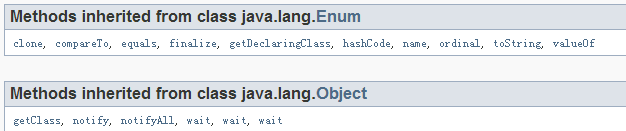
Performs a timed **Thread.join** using this time unit.

### **timedWait(Object obj, long timeout)**

void **timedWait**(Object obj, long timeout)

Performs a timed **Object.wait** using this time unit.

### 其他继承的方法



## 重要的方法介绍

### **timedJoin：等待一定时间，调用线程的join方法**

public void **timedJoin(Thread thread, long timeout)** throws InterruptedException

Performs a timed **Thread.join** using this time unit. This is a convenience method that converts time arguments into the form required by the **Thread.join** method.

Parameters: thread - the thread to wait for

timeout - the maximum time to wait. If less than or equal to zero, do not wait at all.

Throws: InterruptedException - if interrupted while waiting

### **timedWait**:

public void **timedWait(Object obj,long timeout)** throws InterruptedException

Performs a timed **Object.wait** using this time unit. This is a convenience method that converts timeout arguments into the form required by the **Object.wait** method.

For example, you could implement a blocking poll method (see BlockingQueue.poll) using:

public synchronized Object poll(long timeout, TimeUnit unit)

throws InterruptedException {

while (empty) {

unit.timedWait(this, timeout);

...

}

}

**Parameters**: **obj** - the object to wait on

**timeout** - the maximum time to wait. If less than or equal to zero, do not wait at all.

Throws: InterruptedException - if interrupted while waiting