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

## **java.util.concurrent**

Class Exchanger<V>

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 **java.util.concurrent.Exchanger<V>**

**Type Parameters:**V - The type of objects that may be exchanged

public class **Exchanger<V>**extends [Object](http://docs.google.com/java/lang/Object.html)

A synchronization point at which threads can pair and swap elements within pairs. Each thread presents some object on entry to the [exchange](http://docs.google.com/java/util/concurrent/Exchanger.html#exchange(V)) method, matches with a partner thread, and receives its partner's object on return. An Exchanger may be viewed as a bidirectional form of a [SynchronousQueue](http://docs.google.com/java/util/concurrent/SynchronousQueue.html). Exchangers may be useful in applications such as genetic algorithms and pipeline designs.

**Sample Usage:** Here are the highlights of a class that uses an Exchanger to swap buffers between threads so that the thread filling the buffer gets a freshly emptied one when it needs it, handing off the filled one to the thread emptying the buffer.

class FillAndEmpty {  
 Exchanger<DataBuffer> exchanger = new Exchanger<DataBuffer>();  
 DataBuffer initialEmptyBuffer = ... a made-up type  
 DataBuffer initialFullBuffer = ...  
  
 class FillingLoop implements Runnable {  
 public void run() {  
 DataBuffer currentBuffer = initialEmptyBuffer;  
 try {  
 while (currentBuffer != null) {  
 addToBuffer(currentBuffer);  
 if (currentBuffer.isFull())  
 currentBuffer = exchanger.exchange(currentBuffer);  
 }  
 } catch (InterruptedException ex) { ... handle ... }  
 }  
 }  
  
 class EmptyingLoop implements Runnable {  
 public void run() {  
 DataBuffer currentBuffer = initialFullBuffer;  
 try {  
 while (currentBuffer != null) {  
 takeFromBuffer(currentBuffer);  
 if (currentBuffer.isEmpty())  
 currentBuffer = exchanger.exchange(currentBuffer);  
 }  
 } catch (InterruptedException ex) { ... handle ...}  
 }  
 }  
  
 void start() {  
 new Thread(new FillingLoop()).start();  
 new Thread(new EmptyingLoop()).start();  
 }  
 }

Memory consistency effects: For each pair of threads that successfully exchange objects via an Exchanger, actions prior to the exchange() in each thread [*happen-before*](http://docs.google.com/package-summary.html#MemoryVisibility) those subsequent to a return from the corresponding exchange() in the other thread.

**Since:** 1.5

| **Constructor Summary** | |
| --- | --- |
| [**Exchanger**](http://docs.google.com/java/util/concurrent/Exchanger.html#Exchanger())()            Creates a new Exchanger. |

| **Method Summary** | |
| --- | --- |
| [V](http://docs.google.com/java/util/concurrent/Exchanger.html) | [**exchange**](http://docs.google.com/java/util/concurrent/Exchanger.html#exchange(V))([V](http://docs.google.com/java/util/concurrent/Exchanger.html) x)            Waits for another thread to arrive at this exchange point (unless the current thread is [interrupted](http://docs.google.com/java/lang/Thread.html#interrupt())), and then transfers the given object to it, receiving its object in return. |
| [V](http://docs.google.com/java/util/concurrent/Exchanger.html) | [**exchange**](http://docs.google.com/java/util/concurrent/Exchanger.html#exchange(V,%20long,%20java.util.concurrent.TimeUnit))([V](http://docs.google.com/java/util/concurrent/Exchanger.html) x, long timeout, [TimeUnit](http://docs.google.com/java/util/concurrent/TimeUnit.html) unit)            Waits for another thread to arrive at this exchange point (unless the current thread is [interrupted](http://docs.google.com/java/lang/Thread.html#interrupt()) or the specified waiting time elapses), and then transfers the given object to it, receiving its object in return. |

| **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()), [equals](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [finalize](http://docs.google.com/java/lang/Object.html#finalize()), [getClass](http://docs.google.com/java/lang/Object.html#getClass()), [hashCode](http://docs.google.com/java/lang/Object.html#hashCode()), [notify](http://docs.google.com/java/lang/Object.html#notify()), [notifyAll](http://docs.google.com/java/lang/Object.html#notifyAll()), [toString](http://docs.google.com/java/lang/Object.html#toString()), [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** |
| --- |

### Exchanger

public **Exchanger**()

Creates a new Exchanger.

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

### exchange

public [V](http://docs.google.com/java/util/concurrent/Exchanger.html) **exchange**([V](http://docs.google.com/java/util/concurrent/Exchanger.html) x)  
 throws [InterruptedException](http://docs.google.com/java/lang/InterruptedException.html)

Waits for another thread to arrive at this exchange point (unless the current thread is [interrupted](http://docs.google.com/java/lang/Thread.html#interrupt())), and then transfers the given object to it, receiving its object in return.

If another thread is already waiting at the exchange point then it is resumed for thread scheduling purposes and receives the object passed in by the current thread. The current thread returns immediately, receiving the object passed to the exchange by that other thread.

If no other thread is already waiting at the exchange then the current thread is disabled for thread scheduling purposes and lies dormant until one of two things happens:

* Some other thread enters the exchange; or
* Some other thread [interrupts](http://docs.google.com/java/lang/Thread.html#interrupt()) the current thread.

If the current thread:

* has its interrupted status set on entry to this method; or
* is [interrupted](http://docs.google.com/java/lang/Thread.html#interrupt()) while waiting for the exchange,

then [InterruptedException](http://docs.google.com/java/lang/InterruptedException.html) is thrown and the current thread's interrupted status is cleared.

**Parameters:**x - the object to exchange **Returns:**the object provided by the other thread **Throws:** [InterruptedException](http://docs.google.com/java/lang/InterruptedException.html) - if the current thread was interrupted while waiting

### exchange

public [V](http://docs.google.com/java/util/concurrent/Exchanger.html) **exchange**([V](http://docs.google.com/java/util/concurrent/Exchanger.html) x,  
 long timeout,  
 [TimeUnit](http://docs.google.com/java/util/concurrent/TimeUnit.html) unit)  
 throws [InterruptedException](http://docs.google.com/java/lang/InterruptedException.html),  
 [TimeoutException](http://docs.google.com/java/util/concurrent/TimeoutException.html)

Waits for another thread to arrive at this exchange point (unless the current thread is [interrupted](http://docs.google.com/java/lang/Thread.html#interrupt()) or the specified waiting time elapses), and then transfers the given object to it, receiving its object in return.

If another thread is already waiting at the exchange point then it is resumed for thread scheduling purposes and receives the object passed in by the current thread. The current thread returns immediately, receiving the object passed to the exchange by that other thread.

If no other thread is already waiting at the exchange then the current thread is disabled for thread scheduling purposes and lies dormant until one of three things happens:

* Some other thread enters the exchange; or
* Some other thread [interrupts](http://docs.google.com/java/lang/Thread.html#interrupt()) the current thread; or
* The specified waiting time elapses.

If the current thread:

* has its interrupted status set on entry to this method; or
* is [interrupted](http://docs.google.com/java/lang/Thread.html#interrupt()) while waiting for the exchange,

then [InterruptedException](http://docs.google.com/java/lang/InterruptedException.html) is thrown and the current thread's interrupted status is cleared.

If the specified waiting time elapses then [TimeoutException](http://docs.google.com/java/util/concurrent/TimeoutException.html) is thrown. If the time is less than or equal to zero, the method will not wait at all.

**Parameters:**x - the object to exchangetimeout - the maximum time to waitunit - the time unit of the timeout argument **Returns:**the object provided by the other thread **Throws:** [InterruptedException](http://docs.google.com/java/lang/InterruptedException.html) - if the current thread was interrupted while waiting [TimeoutException](http://docs.google.com/java/util/concurrent/TimeoutException.html) - if the specified waiting time elapses before another thread enters the exchange

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Exchanger.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/java/util/concurrent/DelayQueue.html)   [**NEXT CLASS**](http://docs.google.com/java/util/concurrent/ExecutionException.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/concurrent/Exchanger.html)    [**NO FRAMES**](http://docs.google.com/Exchanger.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).