| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/RMIConnector.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/management/remote/rmi/RMIConnectionImpl_Stub.html)   [**NEXT CLASS**](http://docs.google.com/javax/management/remote/rmi/RMIConnectorServer.html) | [**FRAMES**](http://docs.google.com/index.html?javax/management/remote/rmi/RMIConnector.html)    [**NO FRAMES**](http://docs.google.com/RMIConnector.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#tyjcwt) | [METHOD](#3dy6vkm) | DETAIL: FIELD | [CONSTR](#4d34og8) | [METHOD](#3rdcrjn) |

## **javax.management.remote.rmi**

Class RMIConnector

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 **javax.management.remote.rmi.RMIConnector**

**All Implemented Interfaces:** [Closeable](http://docs.google.com/java/io/Closeable.html), [Serializable](http://docs.google.com/java/io/Serializable.html), [JMXAddressable](http://docs.google.com/javax/management/remote/JMXAddressable.html), [JMXConnector](http://docs.google.com/javax/management/remote/JMXConnector.html)

public class **RMIConnector**extends [Object](http://docs.google.com/java/lang/Object.html)implements [JMXConnector](http://docs.google.com/javax/management/remote/JMXConnector.html), [Serializable](http://docs.google.com/java/io/Serializable.html), [JMXAddressable](http://docs.google.com/javax/management/remote/JMXAddressable.html)

A connection to a remote RMI connector. Usually, such connections are made using [JMXConnectorFactory](http://docs.google.com/javax/management/remote/JMXConnectorFactory.html). However, specialized applications can use this class directly, for example with an [RMIServer](http://docs.google.com/javax/management/remote/rmi/RMIServer.html) stub obtained without going through JNDI.

**Since:** 1.5 **See Also:**[Serialized Form](http://docs.google.com/serialized-form.html#javax.management.remote.rmi.RMIConnector)

| **Field Summary** | |
| --- | --- |

| **Fields inherited from interface javax.management.remote.**[**JMXConnector**](http://docs.google.com/javax/management/remote/JMXConnector.html) |
| --- |
| [CREDENTIALS](http://docs.google.com/javax/management/remote/JMXConnector.html#CREDENTIALS) |

| **Constructor Summary** | |
| --- | --- |
| [**RMIConnector**](http://docs.google.com/javax/management/remote/rmi/RMIConnector.html#RMIConnector(javax.management.remote.JMXServiceURL,%20java.util.Map))([JMXServiceURL](http://docs.google.com/javax/management/remote/JMXServiceURL.html) url, [Map](http://docs.google.com/java/util/Map.html)<[String](http://docs.google.com/java/lang/String.html),?> environment)            Constructs an RMIConnector that will connect the RMI connector server with the given address. |
| [**RMIConnector**](http://docs.google.com/javax/management/remote/rmi/RMIConnector.html#RMIConnector(javax.management.remote.rmi.RMIServer,%20java.util.Map))([RMIServer](http://docs.google.com/javax/management/remote/rmi/RMIServer.html) rmiServer, [Map](http://docs.google.com/java/util/Map.html)<[String](http://docs.google.com/java/lang/String.html),?> environment)            Constructs an RMIConnector using the given RMI stub. |

| **Method Summary** | |
| --- | --- |
| void | [**addConnectionNotificationListener**](http://docs.google.com/javax/management/remote/rmi/RMIConnector.html#addConnectionNotificationListener(javax.management.NotificationListener,%20javax.management.NotificationFilter,%20java.lang.Object))([NotificationListener](http://docs.google.com/javax/management/NotificationListener.html) listener, [NotificationFilter](http://docs.google.com/javax/management/NotificationFilter.html) filter, [Object](http://docs.google.com/java/lang/Object.html) handback)            Adds a listener to be informed of changes in connection status. |
| void | [**close**](http://docs.google.com/javax/management/remote/rmi/RMIConnector.html#close())()            Closes the client connection to its server. |
| void | [**connect**](http://docs.google.com/javax/management/remote/rmi/RMIConnector.html#connect())()            Establishes the connection to the connector server. |
| void | [**connect**](http://docs.google.com/javax/management/remote/rmi/RMIConnector.html#connect(java.util.Map))([Map](http://docs.google.com/java/util/Map.html)<[String](http://docs.google.com/java/lang/String.html),?> environment)            Establishes the connection to the connector server. |
| [JMXServiceURL](http://docs.google.com/javax/management/remote/JMXServiceURL.html) | [**getAddress**](http://docs.google.com/javax/management/remote/rmi/RMIConnector.html#getAddress())()            The address of this connector. |
| [String](http://docs.google.com/java/lang/String.html) | [**getConnectionId**](http://docs.google.com/javax/management/remote/rmi/RMIConnector.html#getConnectionId())()            Gets this connection's ID from the connector server. |
| [MBeanServerConnection](http://docs.google.com/javax/management/MBeanServerConnection.html) | [**getMBeanServerConnection**](http://docs.google.com/javax/management/remote/rmi/RMIConnector.html#getMBeanServerConnection())()            Returns an MBeanServerConnection object representing a remote MBean server. |
| [MBeanServerConnection](http://docs.google.com/javax/management/MBeanServerConnection.html) | [**getMBeanServerConnection**](http://docs.google.com/javax/management/remote/rmi/RMIConnector.html#getMBeanServerConnection(javax.security.auth.Subject))([Subject](http://docs.google.com/javax/security/auth/Subject.html) delegationSubject)            Returns an MBeanServerConnection object representing a remote MBean server on which operations are performed on behalf of the supplied delegation subject. |
| void | [**removeConnectionNotificationListener**](http://docs.google.com/javax/management/remote/rmi/RMIConnector.html#removeConnectionNotificationListener(javax.management.NotificationListener))([NotificationListener](http://docs.google.com/javax/management/NotificationListener.html) listener)            Removes a listener from the list to be informed of changes in status. |
| void | [**removeConnectionNotificationListener**](http://docs.google.com/javax/management/remote/rmi/RMIConnector.html#removeConnectionNotificationListener(javax.management.NotificationListener,%20javax.management.NotificationFilter,%20java.lang.Object))([NotificationListener](http://docs.google.com/javax/management/NotificationListener.html) listener, [NotificationFilter](http://docs.google.com/javax/management/NotificationFilter.html) filter, [Object](http://docs.google.com/java/lang/Object.html) handback)            Removes a listener from the list to be informed of changes in status. |
| [String](http://docs.google.com/java/lang/String.html) | [**toString**](http://docs.google.com/javax/management/remote/rmi/RMIConnector.html#toString())()            Returns a string representation of this 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()), [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()), [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** |
| --- |

### RMIConnector

public **RMIConnector**([JMXServiceURL](http://docs.google.com/javax/management/remote/JMXServiceURL.html) url,  
 [Map](http://docs.google.com/java/util/Map.html)<[String](http://docs.google.com/java/lang/String.html),?> environment)

Constructs an RMIConnector that will connect the RMI connector server with the given address.

The address can refer directly to the connector server, using one of the following syntaxes:

service:jmx:rmi://*[host[:port]]*/stub/*encoded-stub*  
 service:jmx:iiop://*[host[:port]]*/ior/*encoded-IOR*

(Here, the square brackets [] are not part of the address but indicate that the host and port are optional.)

The address can instead indicate where to find an RMI stub through JNDI, using one of the following syntaxes:

service:jmx:rmi://*[host[:port]]*/jndi/*jndi-name*  
 service:jmx:iiop://*[host[:port]]*/jndi/*jndi-name*

An implementation may also recognize additional address syntaxes, for example:

service:jmx:iiop://*[host[:port]]*/stub/*encoded-stub*

**Parameters:**url - the address of the RMI connector server.environment - additional attributes specifying how to make the connection. For JNDI-based addresses, these attributes can usefully include JNDI attributes recognized by [InitialContext](http://docs.google.com/javax/naming/InitialContext.html#InitialContext(java.util.Hashtable)). This parameter can be null, which is equivalent to an empty Map. **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if url is null.

### RMIConnector

public **RMIConnector**([RMIServer](http://docs.google.com/javax/management/remote/rmi/RMIServer.html) rmiServer,  
 [Map](http://docs.google.com/java/util/Map.html)<[String](http://docs.google.com/java/lang/String.html),?> environment)

Constructs an RMIConnector using the given RMI stub.

**Parameters:**rmiServer - an RMI stub representing the RMI connector server.environment - additional attributes specifying how to make the connection. This parameter can be null, which is equivalent to an empty Map. **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if rmiServer is null.

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

### toString

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

Returns a string representation of this object. In general, the toString method returns a string that "textually represents" this object. The result should be a concise but informative representation that is easy for a person to read.

**Overrides:**[toString](http://docs.google.com/java/lang/Object.html#toString()) in class [Object](http://docs.google.com/java/lang/Object.html) **Returns:**a String representation of this object.

### getAddress

public [JMXServiceURL](http://docs.google.com/javax/management/remote/JMXServiceURL.html) **getAddress**()

The address of this connector.

**Specified by:**[getAddress](http://docs.google.com/javax/management/remote/JMXAddressable.html#getAddress()) in interface [JMXAddressable](http://docs.google.com/javax/management/remote/JMXAddressable.html) **Returns:**the address of this connector, or null if it does not have one.**Since:** 1.6

### connect

public void **connect**()  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

**Description copied from interface:** [**JMXConnector**](http://docs.google.com/javax/management/remote/JMXConnector.html#connect())

Establishes the connection to the connector server. This method is equivalent to [connect(null)](http://docs.google.com/javax/management/remote/JMXConnector.html#connect(java.util.Map)).

**Specified by:**[connect](http://docs.google.com/javax/management/remote/JMXConnector.html#connect()) in interface [JMXConnector](http://docs.google.com/javax/management/remote/JMXConnector.html) **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if the connection could not be made because of a communication problem.

### connect

public void **connect**([Map](http://docs.google.com/java/util/Map.html)<[String](http://docs.google.com/java/lang/String.html),?> environment)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

**Description copied from interface:** [**JMXConnector**](http://docs.google.com/javax/management/remote/JMXConnector.html#connect(java.util.Map))

Establishes the connection to the connector server.

If connect has already been called successfully on this object, calling it again has no effect. If, however, [JMXConnector.close()](http://docs.google.com/javax/management/remote/JMXConnector.html#close()) was called after connect, the new connect will throw an IOException.

Otherwise, either connect has never been called on this object, or it has been called but produced an exception. Then calling connect will attempt to establish a connection to the connector server.

**Specified by:**[connect](http://docs.google.com/javax/management/remote/JMXConnector.html#connect(java.util.Map)) in interface [JMXConnector](http://docs.google.com/javax/management/remote/JMXConnector.html) **Parameters:**environment - the properties of the connection. Properties in this map override properties in the map specified when the JMXConnector was created, if any. This parameter can be null, which is equivalent to an empty map. **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if the connection could not be made because of a communication problem.

### getConnectionId

public [String](http://docs.google.com/java/lang/String.html) **getConnectionId**()  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

**Description copied from interface:** [**JMXConnector**](http://docs.google.com/javax/management/remote/JMXConnector.html#getConnectionId())

Gets this connection's ID from the connector server. For a given connector server, every connection will have a unique id which does not change during the lifetime of the connection.

**Specified by:**[getConnectionId](http://docs.google.com/javax/management/remote/JMXConnector.html#getConnectionId()) in interface [JMXConnector](http://docs.google.com/javax/management/remote/JMXConnector.html) **Returns:**the unique ID of this connection. This is the same as the ID that the connector server includes in its [JMXConnectionNotification](http://docs.google.com/javax/management/remote/JMXConnectionNotification.html)s. The [package description](http://docs.google.com/javax/management/remote/package-summary.html) describes the conventions for connection IDs. **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if the connection ID cannot be obtained, for instance because the connection is closed or broken.

### getMBeanServerConnection

public [MBeanServerConnection](http://docs.google.com/javax/management/MBeanServerConnection.html) **getMBeanServerConnection**()  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

**Description copied from interface:** [**JMXConnector**](http://docs.google.com/javax/management/remote/JMXConnector.html#getMBeanServerConnection())

Returns an MBeanServerConnection object representing a remote MBean server. For a given JMXConnector, two successful calls to this method will usually return the same MBeanServerConnection object, though this is not required.

For each method in the returned MBeanServerConnection, calling the method causes the corresponding method to be called in the remote MBean server. The value returned by the MBean server method is the value returned to the client. If the MBean server method produces an Exception, the same Exception is seen by the client. If the MBean server method, or the attempt to call it, produces an Error, the Error is wrapped in a [JMXServerErrorException](http://docs.google.com/javax/management/remote/JMXServerErrorException.html), which is seen by the client.

Calling this method is equivalent to calling [getMBeanServerConnection(null)](http://docs.google.com/javax/management/remote/JMXConnector.html#getMBeanServerConnection(javax.security.auth.Subject)) meaning that no delegation subject is specified and that all the operations called on the MBeanServerConnection must use the authenticated subject, if any.

**Specified by:**[getMBeanServerConnection](http://docs.google.com/javax/management/remote/JMXConnector.html#getMBeanServerConnection()) in interface [JMXConnector](http://docs.google.com/javax/management/remote/JMXConnector.html) **Returns:**an object that implements the MBeanServerConnection interface by forwarding its methods to the remote MBean server. **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if a valid MBeanServerConnection cannot be created, for instance because the connection to the remote MBean server has not yet been established (with the [connect](http://docs.google.com/javax/management/remote/JMXConnector.html#connect(java.util.Map)) method), or it has been closed, or it has broken.

### getMBeanServerConnection

public [MBeanServerConnection](http://docs.google.com/javax/management/MBeanServerConnection.html) **getMBeanServerConnection**([Subject](http://docs.google.com/javax/security/auth/Subject.html) delegationSubject)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

**Description copied from interface:** [**JMXConnector**](http://docs.google.com/javax/management/remote/JMXConnector.html#getMBeanServerConnection(javax.security.auth.Subject))

Returns an MBeanServerConnection object representing a remote MBean server on which operations are performed on behalf of the supplied delegation subject. For a given JMXConnector and Subject, two successful calls to this method will usually return the same MBeanServerConnection object, though this is not required.

For each method in the returned MBeanServerConnection, calling the method causes the corresponding method to be called in the remote MBean server on behalf of the given delegation subject instead of the authenticated subject. The value returned by the MBean server method is the value returned to the client. If the MBean server method produces an Exception, the same Exception is seen by the client. If the MBean server method, or the attempt to call it, produces an Error, the Error is wrapped in a [JMXServerErrorException](http://docs.google.com/javax/management/remote/JMXServerErrorException.html), which is seen by the client.

**Specified by:**[getMBeanServerConnection](http://docs.google.com/javax/management/remote/JMXConnector.html#getMBeanServerConnection(javax.security.auth.Subject)) in interface [JMXConnector](http://docs.google.com/javax/management/remote/JMXConnector.html) **Parameters:**delegationSubject - the Subject on behalf of which requests will be performed. Can be null, in which case requests will be performed on behalf of the authenticated Subject, if any. **Returns:**an object that implements the MBeanServerConnection interface by forwarding its methods to the remote MBean server on behalf of a given delegation subject. **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if a valid MBeanServerConnection cannot be created, for instance because the connection to the remote MBean server has not yet been established (with the [connect](http://docs.google.com/javax/management/remote/JMXConnector.html#connect(java.util.Map)) method), or it has been closed, or it has broken.

### addConnectionNotificationListener

public void **addConnectionNotificationListener**([NotificationListener](http://docs.google.com/javax/management/NotificationListener.html) listener,  
 [NotificationFilter](http://docs.google.com/javax/management/NotificationFilter.html) filter,  
 [Object](http://docs.google.com/java/lang/Object.html) handback)

**Description copied from interface:** [**JMXConnector**](http://docs.google.com/javax/management/remote/JMXConnector.html#addConnectionNotificationListener(javax.management.NotificationListener,%20javax.management.NotificationFilter,%20java.lang.Object))

Adds a listener to be informed of changes in connection status. The listener will receive notifications of type [JMXConnectionNotification](http://docs.google.com/javax/management/remote/JMXConnectionNotification.html). An implementation can send other types of notifications too.

Any number of listeners can be added with this method. The same listener can be added more than once with the same or different values for the filter and handback. There is no special treatment of a duplicate entry. For example, if a listener is registered twice with no filter, then its handleNotification method will be called twice for each notification.

**Specified by:**[addConnectionNotificationListener](http://docs.google.com/javax/management/remote/JMXConnector.html#addConnectionNotificationListener(javax.management.NotificationListener,%20javax.management.NotificationFilter,%20java.lang.Object)) in interface [JMXConnector](http://docs.google.com/javax/management/remote/JMXConnector.html) **Parameters:**listener - a listener to receive connection status notifications.filter - a filter to select which notifications are to be delivered to the listener, or null if all notifications are to be delivered.handback - an object to be given to the listener along with each notification. Can be null.**See Also:**[JMXConnector.removeConnectionNotificationListener(javax.management.NotificationListener)](http://docs.google.com/javax/management/remote/JMXConnector.html#removeConnectionNotificationListener(javax.management.NotificationListener)), [NotificationBroadcaster.addNotificationListener(javax.management.NotificationListener, javax.management.NotificationFilter, java.lang.Object)](http://docs.google.com/javax/management/NotificationBroadcaster.html#addNotificationListener(javax.management.NotificationListener,%20javax.management.NotificationFilter,%20java.lang.Object))

### removeConnectionNotificationListener

public void **removeConnectionNotificationListener**([NotificationListener](http://docs.google.com/javax/management/NotificationListener.html) listener)  
 throws [ListenerNotFoundException](http://docs.google.com/javax/management/ListenerNotFoundException.html)

**Description copied from interface:** [**JMXConnector**](http://docs.google.com/javax/management/remote/JMXConnector.html#removeConnectionNotificationListener(javax.management.NotificationListener))

Removes a listener from the list to be informed of changes in status. The listener must previously have been added. If there is more than one matching listener, all are removed.

**Specified by:**[removeConnectionNotificationListener](http://docs.google.com/javax/management/remote/JMXConnector.html#removeConnectionNotificationListener(javax.management.NotificationListener)) in interface [JMXConnector](http://docs.google.com/javax/management/remote/JMXConnector.html) **Parameters:**listener - a listener to receive connection status notifications. **Throws:** [ListenerNotFoundException](http://docs.google.com/javax/management/ListenerNotFoundException.html) - if the listener is not registered with this JMXConnector.**See Also:**[JMXConnector.removeConnectionNotificationListener(NotificationListener, NotificationFilter, Object)](http://docs.google.com/javax/management/remote/JMXConnector.html#removeConnectionNotificationListener(javax.management.NotificationListener,%20javax.management.NotificationFilter,%20java.lang.Object)), [JMXConnector.addConnectionNotificationListener(javax.management.NotificationListener, javax.management.NotificationFilter, java.lang.Object)](http://docs.google.com/javax/management/remote/JMXConnector.html#addConnectionNotificationListener(javax.management.NotificationListener,%20javax.management.NotificationFilter,%20java.lang.Object)), [NotificationEmitter.removeNotificationListener(javax.management.NotificationListener, javax.management.NotificationFilter, java.lang.Object)](http://docs.google.com/javax/management/NotificationEmitter.html#removeNotificationListener(javax.management.NotificationListener,%20javax.management.NotificationFilter,%20java.lang.Object))

### removeConnectionNotificationListener

public void **removeConnectionNotificationListener**([NotificationListener](http://docs.google.com/javax/management/NotificationListener.html) listener,  
 [NotificationFilter](http://docs.google.com/javax/management/NotificationFilter.html) filter,  
 [Object](http://docs.google.com/java/lang/Object.html) handback)  
 throws [ListenerNotFoundException](http://docs.google.com/javax/management/ListenerNotFoundException.html)

**Description copied from interface:** [**JMXConnector**](http://docs.google.com/javax/management/remote/JMXConnector.html#removeConnectionNotificationListener(javax.management.NotificationListener,%20javax.management.NotificationFilter,%20java.lang.Object))

Removes a listener from the list to be informed of changes in status. The listener must previously have been added with the same three parameters. If there is more than one matching listener, only one is removed.

**Specified by:**[removeConnectionNotificationListener](http://docs.google.com/javax/management/remote/JMXConnector.html#removeConnectionNotificationListener(javax.management.NotificationListener,%20javax.management.NotificationFilter,%20java.lang.Object)) in interface [JMXConnector](http://docs.google.com/javax/management/remote/JMXConnector.html) **Parameters:**listener - a listener to receive connection status notifications.filter - a filter to select which notifications are to be delivered to the listener. Can be null.handback - an object to be given to the listener along with each notification. Can be null. **Throws:** [ListenerNotFoundException](http://docs.google.com/javax/management/ListenerNotFoundException.html) - if the listener is not registered with this JMXConnector, or is not registered with the given filter and handback.**See Also:**[JMXConnector.removeConnectionNotificationListener(NotificationListener)](http://docs.google.com/javax/management/remote/JMXConnector.html#removeConnectionNotificationListener(javax.management.NotificationListener)), [JMXConnector.addConnectionNotificationListener(javax.management.NotificationListener, javax.management.NotificationFilter, java.lang.Object)](http://docs.google.com/javax/management/remote/JMXConnector.html#addConnectionNotificationListener(javax.management.NotificationListener,%20javax.management.NotificationFilter,%20java.lang.Object)), [NotificationEmitter.removeNotificationListener(javax.management.NotificationListener, javax.management.NotificationFilter, java.lang.Object)](http://docs.google.com/javax/management/NotificationEmitter.html#removeNotificationListener(javax.management.NotificationListener,%20javax.management.NotificationFilter,%20java.lang.Object))

### close

public void **close**()  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

**Description copied from interface:** [**JMXConnector**](http://docs.google.com/javax/management/remote/JMXConnector.html#close())

Closes the client connection to its server. Any ongoing or new request using the MBeanServerConnection returned by [JMXConnector.getMBeanServerConnection()](http://docs.google.com/javax/management/remote/JMXConnector.html#getMBeanServerConnection()) will get an IOException.

If close has already been called successfully on this object, calling it again has no effect. If close has never been called, or if it was called but produced an exception, an attempt will be made to close the connection. This attempt can succeed, in which case close will return normally, or it can generate an exception.

Closing a connection is a potentially slow operation. For example, if the server has crashed, the close operation might have to wait for a network protocol timeout. Callers that do not want to block in a close operation should do it in a separate thread.

**Specified by:**[close](http://docs.google.com/java/io/Closeable.html#close()) in interface [Closeable](http://docs.google.com/java/io/Closeable.html)**Specified by:**[close](http://docs.google.com/javax/management/remote/JMXConnector.html#close()) in interface [JMXConnector](http://docs.google.com/javax/management/remote/JMXConnector.html) **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if the connection cannot be closed cleanly. If this exception is thrown, it is not known whether the server end of the connection has been cleanly closed.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/RMIConnector.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/management/remote/rmi/RMIConnectionImpl_Stub.html)   [**NEXT CLASS**](http://docs.google.com/javax/management/remote/rmi/RMIConnectorServer.html) | [**FRAMES**](http://docs.google.com/index.html?javax/management/remote/rmi/RMIConnector.html)    [**NO FRAMES**](http://docs.google.com/RMIConnector.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#tyjcwt) | [METHOD](#3dy6vkm) | DETAIL: FIELD | [CONSTR](#4d34og8) | [METHOD](#3rdcrjn) |

[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).