| | [**Overview**](http://docs.google.com/overview-summary.html) | **Package** | Class | [**Use**](http://docs.google.com/package-use.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 PACKAGE**](http://docs.google.com/javax/management/relation/package-summary.html)   [**NEXT PACKAGE**](http://docs.google.com/javax/management/remote/rmi/package-summary.html) | [**FRAMES**](http://docs.google.com/index.html?javax/management/remote/package-summary.html)    [**NO FRAMES**](http://docs.google.com/package-summary.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |

## Package javax.management.remote

Interfaces for remote access to JMX MBean servers.

**See:**

[**Description**](#3znysh7)

| **Interface Summary** | |
| --- | --- |
| [**JMXAddressable**](http://docs.google.com/javax/management/remote/JMXAddressable.html) | Implemented by objects that can have a JMXServiceURL address. |
| [**JMXAuthenticator**](http://docs.google.com/javax/management/remote/JMXAuthenticator.html) | Interface to define how remote credentials are converted into a JAAS Subject. |
| [**JMXConnector**](http://docs.google.com/javax/management/remote/JMXConnector.html) | The client end of a JMX API connector. |
| [**JMXConnectorProvider**](http://docs.google.com/javax/management/remote/JMXConnectorProvider.html) | A provider for creating JMX API connector clients using a given protocol. |
| [**JMXConnectorServerMBean**](http://docs.google.com/javax/management/remote/JMXConnectorServerMBean.html) | MBean interface for connector servers. |
| [**JMXConnectorServerProvider**](http://docs.google.com/javax/management/remote/JMXConnectorServerProvider.html) | A provider for creating JMX API connector servers using a given protocol. |
| [**MBeanServerForwarder**](http://docs.google.com/javax/management/remote/MBeanServerForwarder.html) | An object of this class implements the MBeanServer interface and wraps another object that also implements that interface. |

| **Class Summary** | |
| --- | --- |
| [**JMXConnectionNotification**](http://docs.google.com/javax/management/remote/JMXConnectionNotification.html) | Notification emitted when a client connection is opened or closed or when notifications are lost. |
| [**JMXConnectorFactory**](http://docs.google.com/javax/management/remote/JMXConnectorFactory.html) | Factory to create JMX API connector clients. |
| [**JMXConnectorServer**](http://docs.google.com/javax/management/remote/JMXConnectorServer.html) | Superclass of every connector server. |
| [**JMXConnectorServerFactory**](http://docs.google.com/javax/management/remote/JMXConnectorServerFactory.html) | Factory to create JMX API connector servers. |
| [**JMXPrincipal**](http://docs.google.com/javax/management/remote/JMXPrincipal.html) | The identity of a remote client of the JMX Remote API. |
| [**JMXServiceURL**](http://docs.google.com/javax/management/remote/JMXServiceURL.html) | The address of a JMX API connector server. |
| [**NotificationResult**](http://docs.google.com/javax/management/remote/NotificationResult.html) | Result of a query for buffered notifications. |
| [**SubjectDelegationPermission**](http://docs.google.com/javax/management/remote/SubjectDelegationPermission.html) | Permission required by an authentication identity to perform operations on behalf of an authorization identity. |
| [**TargetedNotification**](http://docs.google.com/javax/management/remote/TargetedNotification.html) | A (Notification, Listener ID) pair. |

| **Exception Summary** | |
| --- | --- |
| [**JMXProviderException**](http://docs.google.com/javax/management/remote/JMXProviderException.html) | Exception thrown by [JMXConnectorFactory](http://docs.google.com/javax/management/remote/JMXConnectorFactory.html) and [JMXConnectorServerFactory](http://docs.google.com/javax/management/remote/JMXConnectorServerFactory.html) when a provider exists for the required protocol but cannot be used for some reason. |
| [**JMXServerErrorException**](http://docs.google.com/javax/management/remote/JMXServerErrorException.html) | Exception thrown as the result of a remote [MBeanServer](http://docs.google.com/javax/management/MBeanServer.html) method invocation when an Error is thrown while processing the invocation in the remote MBean server. |

## Package javax.management.remote Description

Interfaces for remote access to JMX MBean servers. This package defines the essential interfaces for making a JMX MBean server manageable remotely. The specification of this functionality is completed by Part III of the  [JMX Specification, version 1.4](http://docs.google.com/technotes/guides/jmx/JMX_1_4_specification.pdf) PDF document.

The JMX specification defines the notion of **connectors**. A connector is attached to a JMX API MBean server and makes it accessible to remote Java clients. The client end of a connector exports essentially the same interface as the MBean server, specifically the [MBeanServerConnection](http://docs.google.com/javax/management/MBeanServerConnection.html) interface.

A connector makes an MBean server remotely accessible through a given protocol. The JMX Remote API allows the use of different type of connectors:

* The JMX Remote API defines a standard connector, the **RMI Connector**, which provides remote access to an MBeanServer through RMI.
* The JMX Remote API also defines an optional connector called **JMXMP Connector** implementing the JMX Message Protocol (JMXMP). As it is optional, it is not part of this bundle (see note below).
* User-defined connector protocols are also possible using the [JMXConnectorFactory](http://docs.google.com/javax/management/remote/JMXConnectorFactory.html) and, optionally, the Generic Connector (not part of this bundle, see note below).

Note: the optional packages implementing the optional part of the *JMX Remote API* are not included in the *Java SE Platform* but are available from the *JMX Remote API*  [*Reference Implementation*](http://java.sun.com/products/JavaManagement/download.html).

### Connector addresses

Typically, a connector server has an address, represented by the class [JMXServiceURL](http://docs.google.com/javax/management/remote/JMXServiceURL.html). An address for the RMI Connector can look like this:

service:jmx:rmi:///jndi/rmi://myhost:1099/myname

In this JMXServiceURL, the first rmi: specifies the RMI connector, while the second rmi: specifies the RMI registry into which the RMI connector server has stored its stub.

The example above shows only one form of address. An address for the RMI Connector can take several forms, as detailed in the documentation for the package [javax.management.remote.rmi](http://docs.google.com/javax/management/remote/rmi/package-summary.html).

### Creating a connector server

A connector server is created by constructing an instance of a subclass of [JMXConnectorServer](http://docs.google.com/javax/management/remote/JMXConnectorServer.html). Usually, this instance is created using the method [JMXConnectorServerFactory.newJMXConnectorServer](http://docs.google.com/javax/management/remote/JMXConnectorServerFactory.html#newJMXConnectorServer(javax.management.remote.JMXServiceURL,%20java.util.Map,%20javax.management.MBeanServer)).

Typically, a connector server is associated with an MBean server either by registering it in that MBean server, or by supplying the MBean server as a parameter when creating the connector server.

### Creating a connector client

A connector client is usually created by supplying the JMXServiceURL of the connector server to connect to to the [JMXConnectorFactory.connect](http://docs.google.com/javax/management/remote/JMXConnectorFactory.html#connect(javax.management.remote.JMXServiceURL)) method.

For more specialized uses, a connector client can be created by directly instantiating a class that implements the [JMXConnector](http://docs.google.com/javax/management/remote/JMXConnector.html) interface, for example the class [RMIConnector](http://docs.google.com/javax/management/remote/rmi/RMIConnector.html).

### Additional client or server parameters

When creating a connector client or server, it is possible to supply an object of type [Map](http://docs.google.com/java/util/Map.html) that defines additional parameters. Each entry in this Map has a key that is a string and an associated value whose type is appropriate for that key. The standard keys defined by the JMX Remote API all begin with the string "jmx.remote.". The document *JMX Remote API* lists these standard keys.

### Connection identifiers

Every connection opened by a connector server has a string identifier, called its **connection id**. This identifier appears in the [JMXConnectionNotification](http://docs.google.com/javax/management/remote/JMXConnectionNotification.html) events emitted by the connector server, in the list returned by [getConnectionIds()](http://docs.google.com/javax/management/remote/JMXConnectorServerMBean.html#getConnectionIds()), and in the value returned by the client's [getConnectionId()](http://docs.google.com/javax/management/remote/JMXConnector.html#getConnectionId()) method.

As an example, a connection ID can look something like this:

rmi://192.18.1.9 username 1

The formal grammar for connection ids that follow this convention is as follows (using the [grammar notation](http://java.sun.com/docs/books/jls/second_edition/html/grammars.doc.html#90767) from *The Java Language Specification, Second Edition*):

*ConnectionId:*  
 *Protocol* : *ClientAddressopt* Space *ClientIdopt* Space *ArbitraryText*  
  
*ClientAddress:*  
 // *HostAddress* *ClientPortopt*  
  
*ClientPort*  
 : *HostPort*

The *Protocol* is a protocol that would be recognized by [JMXConnectorFactory](http://docs.google.com/javax/management/remote/JMXConnectorFactory.html).

The *ClientAddress* is the address and port of the connecting client, if these can be determined, otherwise nothing. The *HostAddress* is the Internet address of the host that the client is connecting from, in numeric or DNS form. Numeric IPv6 addresses are enclosed in square brackets []. The *HostPort* is the decimal port number that the client is connecting from.

The *ClientId* is the identity of the client entity, typically a string returned by [JMXPrincipal.getName()](http://docs.google.com/javax/management/remote/JMXPrincipal.html#getName()). This string must not contain spaces.

The *ArbitraryText* is any additional text that the connector server adds when creating the client id. At a minimum, it must be enough to distinguish this connection ID from the ID of any other connection currently opened by this connector server.

**Since:** 1.5 **See Also:** [Java SE 6 Platform documentation on JMX technology](http://docs.google.com/technotes/guides/jmx/), in particular the  [JMX Specification, version 1.4](http://docs.google.com/technotes/guides/jmx/JMX_1_4_specification.pdf)

| | [**Overview**](http://docs.google.com/overview-summary.html) | **Package** | Class | [**Use**](http://docs.google.com/package-use.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 PACKAGE**](http://docs.google.com/javax/management/relation/package-summary.html)   [**NEXT PACKAGE**](http://docs.google.com/javax/management/remote/rmi/package-summary.html) | [**FRAMES**](http://docs.google.com/index.html?javax/management/remote/package-summary.html)    [**NO FRAMES**](http://docs.google.com/package-summary.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |

[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).