| | [**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/java/nio/charset/spi/package-summary.html)   [**NEXT PACKAGE**](http://docs.google.com/java/rmi/activation/package-summary.html) | [**FRAMES**](http://docs.google.com/index.html?java/rmi/package-summary.html)    [**NO FRAMES**](http://docs.google.com/package-summary.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |

## Package java.rmi

Provides the RMI package.

**See:**

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

| **Interface Summary** | |
| --- | --- |
| [**Remote**](http://docs.google.com/java/rmi/Remote.html) | The Remote interface serves to identify interfaces whose methods may be invoked from a non-local virtual machine. |

| **Class Summary** | |
| --- | --- |
| [**MarshalledObject<T>**](http://docs.google.com/java/rmi/MarshalledObject.html) | A MarshalledObject contains a byte stream with the serialized representation of an object given to its constructor. |
| [**Naming**](http://docs.google.com/java/rmi/Naming.html) | The Naming class provides methods for storing and obtaining references to remote objects in a remote object registry. |
| [**RMISecurityManager**](http://docs.google.com/java/rmi/RMISecurityManager.html) | A subclass of [SecurityManager](http://docs.google.com/java/lang/SecurityManager.html) used by RMI applications that use downloaded code. |

| **Exception Summary** | |
| --- | --- |
| [**AccessException**](http://docs.google.com/java/rmi/AccessException.html) | An AccessException is thrown by certain methods of the java.rmi.Naming class (specifically bind, rebind, and unbind) and methods of the java.rmi.activation.ActivationSystem interface to indicate that the caller does not have permission to perform the action requested by the method call. |
| [**AlreadyBoundException**](http://docs.google.com/java/rmi/AlreadyBoundException.html) | An AlreadyBoundException is thrown if an attempt is made to bind an object in the registry to a name that already has an associated binding. |
| [**ConnectException**](http://docs.google.com/java/rmi/ConnectException.html) | A ConnectException is thrown if a connection is refused to the remote host for a remote method call. |
| [**ConnectIOException**](http://docs.google.com/java/rmi/ConnectIOException.html) | A ConnectIOException is thrown if an IOException occurs while making a connection to the remote host for a remote method call. |
| [**MarshalException**](http://docs.google.com/java/rmi/MarshalException.html) | A MarshalException is thrown if a java.io.IOException occurs while marshalling the remote call header, arguments or return value for a remote method call. |
| [**NoSuchObjectException**](http://docs.google.com/java/rmi/NoSuchObjectException.html) | A NoSuchObjectException is thrown if an attempt is made to invoke a method on an object that no longer exists in the remote virtual machine. |
| [**NotBoundException**](http://docs.google.com/java/rmi/NotBoundException.html) | A NotBoundException is thrown if an attempt is made to lookup or unbind in the registry a name that has no associated binding. |
| [**RemoteException**](http://docs.google.com/java/rmi/RemoteException.html) | A RemoteException is the common superclass for a number of communication-related exceptions that may occur during the execution of a remote method call. |
| [**RMISecurityException**](http://docs.google.com/java/rmi/RMISecurityException.html) | **Deprecated.** *Use* [*SecurityException*](http://docs.google.com/java/lang/SecurityException.html) *instead.* |
| [**ServerError**](http://docs.google.com/java/rmi/ServerError.html) | A ServerError is thrown as a result of a remote method invocation when an Error is thrown while processing the invocation on the server, either while unmarshalling the arguments, executing the remote method itself, or marshalling the return value. |
| [**ServerException**](http://docs.google.com/java/rmi/ServerException.html) | A ServerException is thrown as a result of a remote method invocation when a RemoteException is thrown while processing the invocation on the server, either while unmarshalling the arguments or executing the remote method itself. |
| [**ServerRuntimeException**](http://docs.google.com/java/rmi/ServerRuntimeException.html) | **Deprecated.** *no replacement* |
| [**StubNotFoundException**](http://docs.google.com/java/rmi/StubNotFoundException.html) | A StubNotFoundException is thrown if a valid stub class could not be found for a remote object when it is exported. |
| [**UnexpectedException**](http://docs.google.com/java/rmi/UnexpectedException.html) | An UnexpectedException is thrown if the client of a remote method call receives, as a result of the call, a checked exception that is not among the checked exception types declared in the throws clause of the method in the remote interface. |
| [**UnknownHostException**](http://docs.google.com/java/rmi/UnknownHostException.html) | An UnknownHostException is thrown if a java.net.UnknownHostException occurs while creating a connection to the remote host for a remote method call. |
| [**UnmarshalException**](http://docs.google.com/java/rmi/UnmarshalException.html) | An UnmarshalException can be thrown while unmarshalling the parameters or results of a remote method call if any of the following conditions occur: if an exception occurs while unmarshalling the call header if the protocol for the return value is invalid if a java.io.IOException occurs unmarshalling parameters (on the server side) or the return value (on the client side). |

## Package java.rmi Description

Provides the RMI package. RMI is Remote Method Invocation. It is a mechanism that enables an object on one Java virtual machine to invoke methods on an object in another Java virtual machine. Any object that can be invoked this way must implement the Remote interface. When such an object is invoked, its arguments are ``marshalled'' and sent from the local virtual machine to the remote one, where the arguments are ``unmarshalled.'' When the method terminates, the results are marshalled from the remote machine and sent to the caller's virtual machine. If the method invocation results in an exception being thrown, the exception is indicated to caller.

**Since:** JDK1.1

| | [**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/java/nio/charset/spi/package-summary.html)   [**NEXT PACKAGE**](http://docs.google.com/java/rmi/activation/package-summary.html) | [**FRAMES**](http://docs.google.com/index.html?java/rmi/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).