| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/RemoteObjectInvocationHandler.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/rmi/server/RemoteObject.html)   [**NEXT CLASS**](http://docs.google.com/java/rmi/server/RemoteRef.html) | [**FRAMES**](http://docs.google.com/index.html?java/rmi/server/RemoteObjectInvocationHandler.html)    [**NO FRAMES**](http://docs.google.com/RemoteObjectInvocationHandler.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#2et92p0) | [CONSTR](#tyjcwt) | [METHOD](#3dy6vkm) | DETAIL: FIELD | [CONSTR](#2s8eyo1) | [METHOD](#3rdcrjn) |

## **java.rmi.server**

Class RemoteObjectInvocationHandler

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 [java.rmi.server.RemoteObject](http://docs.google.com/java/rmi/server/RemoteObject.html)  
 **java.rmi.server.RemoteObjectInvocationHandler**

**All Implemented Interfaces:** [Serializable](http://docs.google.com/java/io/Serializable.html), [InvocationHandler](http://docs.google.com/java/lang/reflect/InvocationHandler.html), [Remote](http://docs.google.com/java/rmi/Remote.html)

public class **RemoteObjectInvocationHandler**extends [RemoteObject](http://docs.google.com/java/rmi/server/RemoteObject.html)implements [InvocationHandler](http://docs.google.com/java/lang/reflect/InvocationHandler.html)

An implementation of the InvocationHandler interface for use with Java Remote Method Invocation (Java RMI). This invocation handler can be used in conjunction with a dynamic proxy instance as a replacement for a pregenerated stub class.

Applications are not expected to use this class directly. A remote object exported to use a dynamic proxy with [UnicastRemoteObject](http://docs.google.com/java/rmi/server/UnicastRemoteObject.html) or [Activatable](http://docs.google.com/java/rmi/activation/Activatable.html) has an instance of this class as that proxy's invocation handler.

**Since:** 1.5 **See Also:**[Serialized Form](http://docs.google.com/serialized-form.html#java.rmi.server.RemoteObjectInvocationHandler)

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

| **Fields inherited from class java.rmi.server.**[**RemoteObject**](http://docs.google.com/java/rmi/server/RemoteObject.html) |
| --- |
| [ref](http://docs.google.com/java/rmi/server/RemoteObject.html#ref) |

| **Constructor Summary** | |
| --- | --- |
| [**RemoteObjectInvocationHandler**](http://docs.google.com/java/rmi/server/RemoteObjectInvocationHandler.html#RemoteObjectInvocationHandler(java.rmi.server.RemoteRef))([RemoteRef](http://docs.google.com/java/rmi/server/RemoteRef.html) ref)            Creates a new RemoteObjectInvocationHandler constructed with the specified RemoteRef. |

| **Method Summary** | |
| --- | --- |
| [Object](http://docs.google.com/java/lang/Object.html) | [**invoke**](http://docs.google.com/java/rmi/server/RemoteObjectInvocationHandler.html#invoke(java.lang.Object,%20java.lang.reflect.Method,%20java.lang.Object%5B%5D))([Object](http://docs.google.com/java/lang/Object.html) proxy, [Method](http://docs.google.com/java/lang/reflect/Method.html) method, [Object](http://docs.google.com/java/lang/Object.html)[] args)            Processes a method invocation made on the encapsulating proxy instance, proxy, and returns the result. |

| **Methods inherited from class java.rmi.server.**[**RemoteObject**](http://docs.google.com/java/rmi/server/RemoteObject.html) |
| --- |
| [equals](http://docs.google.com/java/rmi/server/RemoteObject.html#equals(java.lang.Object)), [getRef](http://docs.google.com/java/rmi/server/RemoteObject.html#getRef()), [hashCode](http://docs.google.com/java/rmi/server/RemoteObject.html#hashCode()), [toString](http://docs.google.com/java/rmi/server/RemoteObject.html#toString()), [toStub](http://docs.google.com/java/rmi/server/RemoteObject.html#toStub(java.rmi.Remote)) |

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

### RemoteObjectInvocationHandler

public **RemoteObjectInvocationHandler**([RemoteRef](http://docs.google.com/java/rmi/server/RemoteRef.html) ref)

Creates a new RemoteObjectInvocationHandler constructed with the specified RemoteRef.

**Parameters:**ref - the remote ref **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if ref is null

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

### invoke

public [Object](http://docs.google.com/java/lang/Object.html) **invoke**([Object](http://docs.google.com/java/lang/Object.html) proxy,  
 [Method](http://docs.google.com/java/lang/reflect/Method.html) method,  
 [Object](http://docs.google.com/java/lang/Object.html)[] args)  
 throws [Throwable](http://docs.google.com/java/lang/Throwable.html)

Processes a method invocation made on the encapsulating proxy instance, proxy, and returns the result.

RemoteObjectInvocationHandler implements this method as follows:

If method is one of the following methods, it is processed as described below:

* [Object.hashCode](http://docs.google.com/java/lang/Object.html#hashCode()): Returns the hash code value for the proxy.
* [Object.equals](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)): Returns true if the argument (args[0]) is an instance of a dynamic proxy class and this invocation handler is equal to the invocation handler of that argument, and returns false otherwise.
* [Object.toString](http://docs.google.com/java/lang/Object.html#toString()): Returns a string representation of the proxy.

Otherwise, a remote call is made as follows:

* If proxy is not an instance of the interface [Remote](http://docs.google.com/java/rmi/Remote.html), then an [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) is thrown.
* Otherwise, the [invoke](http://docs.google.com/java/rmi/server/RemoteRef.html#invoke(java.rmi.Remote,%20java.lang.reflect.Method,%20java.lang.Object%5B%5D,%20long)) method is invoked on this invocation handler's RemoteRef, passing proxy, method, args, and the method hash (defined in section 8.3 of the "Java Remote Method Invocation (RMI) Specification") for method, and the result is returned.
* If an exception is thrown by RemoteRef.invoke and that exception is a checked exception that is not assignable to any exception in the throws clause of the method implemented by the proxy's class, then that exception is wrapped in an [UnexpectedException](http://docs.google.com/java/rmi/UnexpectedException.html) and the wrapped exception is thrown. Otherwise, the exception thrown by invoke is thrown by this method.

The semantics of this method are unspecified if the arguments could not have been produced by an instance of some valid dynamic proxy class containing this invocation handler.

**Specified by:**[invoke](http://docs.google.com/java/lang/reflect/InvocationHandler.html#invoke(java.lang.Object,%20java.lang.reflect.Method,%20java.lang.Object%5B%5D)) in interface [InvocationHandler](http://docs.google.com/java/lang/reflect/InvocationHandler.html) **Parameters:**proxy - the proxy instance that the method was invoked onmethod - the Method instance corresponding to the interface method invoked on the proxy instanceargs - an array of objects containing the values of the arguments passed in the method invocation on the proxy instance, or null if the method takes no arguments **Returns:**the value to return from the method invocation on the proxy instance **Throws:** [Throwable](http://docs.google.com/java/lang/Throwable.html) - the exception to throw from the method invocation on the proxy instance**See Also:**

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/RemoteObjectInvocationHandler.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/rmi/server/RemoteObject.html)   [**NEXT CLASS**](http://docs.google.com/java/rmi/server/RemoteRef.html) | [**FRAMES**](http://docs.google.com/index.html?java/rmi/server/RemoteObjectInvocationHandler.html)    [**NO FRAMES**](http://docs.google.com/RemoteObjectInvocationHandler.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#2et92p0) | [CONSTR](#tyjcwt) | [METHOD](#3dy6vkm) | DETAIL: FIELD | [CONSTR](#2s8eyo1) | [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).