| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/RMIClassLoader.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/RemoteStub.html)   [**NEXT CLASS**](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html) | [**FRAMES**](http://docs.google.com/index.html?java/rmi/server/RMIClassLoader.html)    [**NO FRAMES**](http://docs.google.com/RMIClassLoader.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#3znysh7) | DETAIL: FIELD | CONSTR | [METHOD](#tyjcwt) |

## **java.rmi.server**

Class RMIClassLoader

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

public class **RMIClassLoader**extends [Object](http://docs.google.com/java/lang/Object.html)

RMIClassLoader comprises static methods to support dynamic class loading with RMI. Included are methods for loading classes from a network location (one or more URLs) and obtaining the location from which an existing class should be loaded by remote parties. These methods are used by the RMI runtime when marshalling and unmarshalling classes contained in the arguments and return values of remote method calls, and they also may be invoked directly by applications in order to mimic RMI's dynamic class loading behavior.

The implementation of the following static methods

* [loadClass(URL,String)](http://docs.google.com/java/rmi/server/RMIClassLoader.html#loadClass(java.net.URL,%20java.lang.String))
* [loadClass(String,String)](http://docs.google.com/java/rmi/server/RMIClassLoader.html#loadClass(java.lang.String,%20java.lang.String))
* [loadClass(String,String,ClassLoader)](http://docs.google.com/java/rmi/server/RMIClassLoader.html#loadClass(java.lang.String,%20java.lang.String,%20java.lang.ClassLoader))
* [loadProxyClass(String,String[],ClassLoader)](http://docs.google.com/java/rmi/server/RMIClassLoader.html#loadProxyClass(java.lang.String,%20java.lang.String%5B%5D,%20java.lang.ClassLoader))
* [getClassLoader(String)](http://docs.google.com/java/rmi/server/RMIClassLoader.html#getClassLoader(java.lang.String))
* [getClassAnnotation(Class)](http://docs.google.com/java/rmi/server/RMIClassLoader.html#getClassAnnotation(java.lang.Class))

is provided by an instance of [RMIClassLoaderSpi](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html), the service provider interface for those methods. When one of the methods is invoked, its behavior is to delegate to a corresponding method on the service provider instance. The details of how each method delegates to the provider instance is described in the documentation for each particular method.

The service provider instance is chosen as follows:

* If the system property java.rmi.server.RMIClassLoaderSpi is defined, then if its value equals the string "default", the provider instance will be the value returned by an invocation of the [getDefaultProviderInstance()](http://docs.google.com/java/rmi/server/RMIClassLoader.html#getDefaultProviderInstance()) method, and for any other value, if a class named with the value of the property can be loaded by the system class loader (see [ClassLoader.getSystemClassLoader()](http://docs.google.com/java/lang/ClassLoader.html#getSystemClassLoader())) and that class is assignable to [RMIClassLoaderSpi](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html) and has a public no-argument constructor, then that constructor will be invoked to create the provider instance. If the property is defined but any other of those conditions are not true, then an unspecified Error will be thrown to code that attempts to use RMIClassLoader, indicating the failure to obtain a provider instance.
* If a resource named META-INF/services/java.rmi.server.RMIClassLoaderSpi is visible to the system class loader, then the contents of that resource are interpreted as a provider-configuration file, and the first class name specified in that file is used as the provider class name. If a class with that name can be loaded by the system class loader and that class is assignable to [RMIClassLoaderSpi](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html) and has a public no-argument constructor, then that constructor will be invoked to create the provider instance. If the resource is found but a provider cannot be instantiated as described, then an unspecified Error will be thrown to code that attempts to use RMIClassLoader, indicating the failure to obtain a provider instance.
* Otherwise, the provider instance will be the value returned by an invocation of the [getDefaultProviderInstance()](http://docs.google.com/java/rmi/server/RMIClassLoader.html#getDefaultProviderInstance()) method.

**Since:** JDK1.1 **See Also:**[RMIClassLoaderSpi](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html)

| **Method Summary** | |
| --- | --- |
| static [String](http://docs.google.com/java/lang/String.html) | [**getClassAnnotation**](http://docs.google.com/java/rmi/server/RMIClassLoader.html#getClassAnnotation(java.lang.Class))([Class](http://docs.google.com/java/lang/Class.html)<?> cl)            Returns the annotation string (representing a location for the class definition) that RMI will use to annotate the class descriptor when marshalling objects of the given class. |
| static [ClassLoader](http://docs.google.com/java/lang/ClassLoader.html) | [**getClassLoader**](http://docs.google.com/java/rmi/server/RMIClassLoader.html#getClassLoader(java.lang.String))([String](http://docs.google.com/java/lang/String.html) codebase)            Returns a class loader that loads classes from the given codebase URL path. |
| static [RMIClassLoaderSpi](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html) | [**getDefaultProviderInstance**](http://docs.google.com/java/rmi/server/RMIClassLoader.html#getDefaultProviderInstance())()            Returns the canonical instance of the default provider for the service provider interface [RMIClassLoaderSpi](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html). |
| static [Object](http://docs.google.com/java/lang/Object.html) | [**getSecurityContext**](http://docs.google.com/java/rmi/server/RMIClassLoader.html#getSecurityContext(java.lang.ClassLoader))([ClassLoader](http://docs.google.com/java/lang/ClassLoader.html) loader)  **Deprecated.** *no replacement. As of the Java 2 platform v1.2, RMI no longer uses this method to obtain a class loader's security context.* |
| static [Class](http://docs.google.com/java/lang/Class.html)<?> | [**loadClass**](http://docs.google.com/java/rmi/server/RMIClassLoader.html#loadClass(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)  **Deprecated.** *replaced by loadClass(String,String) method* |
| static [Class](http://docs.google.com/java/lang/Class.html)<?> | [**loadClass**](http://docs.google.com/java/rmi/server/RMIClassLoader.html#loadClass(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) codebase, [String](http://docs.google.com/java/lang/String.html) name)            Loads a class from a codebase URL path. |
| static [Class](http://docs.google.com/java/lang/Class.html)<?> | [**loadClass**](http://docs.google.com/java/rmi/server/RMIClassLoader.html#loadClass(java.lang.String,%20java.lang.String,%20java.lang.ClassLoader))([String](http://docs.google.com/java/lang/String.html) codebase, [String](http://docs.google.com/java/lang/String.html) name, [ClassLoader](http://docs.google.com/java/lang/ClassLoader.html) defaultLoader)            Loads a class from a codebase URL path, optionally using the supplied loader. |
| static [Class](http://docs.google.com/java/lang/Class.html)<?> | [**loadClass**](http://docs.google.com/java/rmi/server/RMIClassLoader.html#loadClass(java.net.URL,%20java.lang.String))([URL](http://docs.google.com/java/net/URL.html) codebase, [String](http://docs.google.com/java/lang/String.html) name)            Loads a class from a codebase URL. |
| static [Class](http://docs.google.com/java/lang/Class.html)<?> | [**loadProxyClass**](http://docs.google.com/java/rmi/server/RMIClassLoader.html#loadProxyClass(java.lang.String,%20java.lang.String%5B%5D,%20java.lang.ClassLoader))([String](http://docs.google.com/java/lang/String.html) codebase, [String](http://docs.google.com/java/lang/String.html)[] interfaces, [ClassLoader](http://docs.google.com/java/lang/ClassLoader.html) defaultLoader)            Loads a dynamic proxy class (see [Proxy](http://docs.google.com/java/lang/reflect/Proxy.html)) that implements a set of interfaces with the given names from a codebase URL path. |

| **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)) |

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

### loadClass

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public static [Class](http://docs.google.com/java/lang/Class.html)<?> **loadClass**([String](http://docs.google.com/java/lang/String.html) name)  
 throws [MalformedURLException](http://docs.google.com/java/net/MalformedURLException.html),  
 [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html)

**Deprecated.** *replaced by loadClass(String,String) method*

Loads the class with the specified name.

This method delegates to [loadClass(String,String)](http://docs.google.com/java/rmi/server/RMIClassLoader.html#loadClass(java.lang.String,%20java.lang.String)), passing null as the first argument and name as the second argument.

**Parameters:**name - the name of the class to load **Returns:**the Class object representing the loaded class **Throws:** [MalformedURLException](http://docs.google.com/java/net/MalformedURLException.html) - if a provider-specific URL used to load classes is invalid [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html) - if a definition for the class could not be found at the codebase location**See Also:**[loadClass(String,String)](http://docs.google.com/java/rmi/server/RMIClassLoader.html#loadClass(java.lang.String,%20java.lang.String))

### loadClass

public static [Class](http://docs.google.com/java/lang/Class.html)<?> **loadClass**([URL](http://docs.google.com/java/net/URL.html) codebase,  
 [String](http://docs.google.com/java/lang/String.html) name)  
 throws [MalformedURLException](http://docs.google.com/java/net/MalformedURLException.html),  
 [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html)

Loads a class from a codebase URL. If codebase is null, then this method will behave the same as [loadClass(String,String)](http://docs.google.com/java/rmi/server/RMIClassLoader.html#loadClass(java.lang.String,%20java.lang.String)) with a null codebase and the given class name.

This method delegates to the [RMIClassLoaderSpi.loadClass(String,String,ClassLoader)](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html#loadClass(java.lang.String,%20java.lang.String,%20java.lang.ClassLoader)) method of the provider instance, passing the result of invoking [URL.toString()](http://docs.google.com/java/net/URL.html#toString()) on the given URL (or null if codebase is null) as the first argument, name as the second argument, and null as the third argument.

**Parameters:**codebase - the URL to load the class from, or nullname - the name of the class to load **Returns:**the Class object representing the loaded class **Throws:** [MalformedURLException](http://docs.google.com/java/net/MalformedURLException.html) - if codebase is null and a provider-specific URL used to load classes is invalid [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html) - if a definition for the class could not be found at the specified URL

### loadClass

public static [Class](http://docs.google.com/java/lang/Class.html)<?> **loadClass**([String](http://docs.google.com/java/lang/String.html) codebase,  
 [String](http://docs.google.com/java/lang/String.html) name)  
 throws [MalformedURLException](http://docs.google.com/java/net/MalformedURLException.html),  
 [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html)

Loads a class from a codebase URL path.

This method delegates to the [RMIClassLoaderSpi.loadClass(String,String,ClassLoader)](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html#loadClass(java.lang.String,%20java.lang.String,%20java.lang.ClassLoader)) method of the provider instance, passing codebase as the first argument, name as the second argument, and null as the third argument.

**Parameters:**codebase - the list of URLs (separated by spaces) to load the class from, or nullname - the name of the class to load **Returns:**the Class object representing the loaded class **Throws:** [MalformedURLException](http://docs.google.com/java/net/MalformedURLException.html) - if codebase is non-null and contains an invalid URL, or if codebase is null and a provider-specific URL used to load classes is invalid [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html) - if a definition for the class could not be found at the specified location**Since:** 1.2

### loadClass

public static [Class](http://docs.google.com/java/lang/Class.html)<?> **loadClass**([String](http://docs.google.com/java/lang/String.html) codebase,  
 [String](http://docs.google.com/java/lang/String.html) name,  
 [ClassLoader](http://docs.google.com/java/lang/ClassLoader.html) defaultLoader)  
 throws [MalformedURLException](http://docs.google.com/java/net/MalformedURLException.html),  
 [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html)

Loads a class from a codebase URL path, optionally using the supplied loader. This method should be used when the caller would like to make available to the provider implementation an additional contextual class loader to consider, such as the loader of a caller on the stack. Typically, a provider implementation will attempt to resolve the named class using the given defaultLoader, if specified, before attempting to resolve the class from the codebase URL path.

This method delegates to the [RMIClassLoaderSpi.loadClass(String,String,ClassLoader)](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html#loadClass(java.lang.String,%20java.lang.String,%20java.lang.ClassLoader)) method of the provider instance, passing codebase as the first argument, name as the second argument, and defaultLoader as the third argument.

**Parameters:**codebase - the list of URLs (separated by spaces) to load the class from, or nullname - the name of the class to loaddefaultLoader - additional contextual class loader to use, or null **Returns:**the Class object representing the loaded class **Throws:** [MalformedURLException](http://docs.google.com/java/net/MalformedURLException.html) - if codebase is non-null and contains an invalid URL, or if codebase is null and a provider-specific URL used to load classes is invalid [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html) - if a definition for the class could not be found at the specified location**Since:** 1.4

### loadProxyClass

public static [Class](http://docs.google.com/java/lang/Class.html)<?> **loadProxyClass**([String](http://docs.google.com/java/lang/String.html) codebase,  
 [String](http://docs.google.com/java/lang/String.html)[] interfaces,  
 [ClassLoader](http://docs.google.com/java/lang/ClassLoader.html) defaultLoader)  
 throws [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html),  
 [MalformedURLException](http://docs.google.com/java/net/MalformedURLException.html)

Loads a dynamic proxy class (see [Proxy](http://docs.google.com/java/lang/reflect/Proxy.html)) that implements a set of interfaces with the given names from a codebase URL path.

The interfaces will be resolved similar to classes loaded via the [loadClass(String,String)](http://docs.google.com/java/rmi/server/RMIClassLoader.html#loadClass(java.lang.String,%20java.lang.String)) method using the given codebase.

This method delegates to the [RMIClassLoaderSpi.loadProxyClass(String,String[],ClassLoader)](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html#loadProxyClass(java.lang.String,%20java.lang.String%5B%5D,%20java.lang.ClassLoader)) method of the provider instance, passing codebase as the first argument, interfaces as the second argument, and defaultLoader as the third argument.

**Parameters:**codebase - the list of URLs (space-separated) to load classes from, or nullinterfaces - the names of the interfaces for the proxy class to implementdefaultLoader - additional contextual class loader to use, or null **Returns:**a dynamic proxy class that implements the named interfaces **Throws:** [MalformedURLException](http://docs.google.com/java/net/MalformedURLException.html) - if codebase is non-null and contains an invalid URL, or if codebase is null and a provider-specific URL used to load classes is invalid [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html) - if a definition for one of the named interfaces could not be found at the specified location, or if creation of the dynamic proxy class failed (such as if [Proxy.getProxyClass(ClassLoader,Class[])](http://docs.google.com/java/lang/reflect/Proxy.html#getProxyClass(java.lang.ClassLoader,%20java.lang.Class...)) would throw an IllegalArgumentException for the given interface list)**Since:** 1.4

### getClassLoader

public static [ClassLoader](http://docs.google.com/java/lang/ClassLoader.html) **getClassLoader**([String](http://docs.google.com/java/lang/String.html) codebase)  
 throws [MalformedURLException](http://docs.google.com/java/net/MalformedURLException.html),  
 [SecurityException](http://docs.google.com/java/lang/SecurityException.html)

Returns a class loader that loads classes from the given codebase URL path.

The class loader returned is the class loader that the [loadClass(String,String)](http://docs.google.com/java/rmi/server/RMIClassLoader.html#loadClass(java.lang.String,%20java.lang.String)) method would use to load classes for the same codebase argument.

This method delegates to the [RMIClassLoaderSpi.getClassLoader(String)](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html#getClassLoader(java.lang.String)) method of the provider instance, passing codebase as the argument.

If there is a security manger, its checkPermission method will be invoked with a RuntimePermission("getClassLoader") permission; this could result in a SecurityException. The provider implementation of this method may also perform further security checks to verify that the calling context has permission to connect to all of the URLs in the codebase URL path.

**Parameters:**codebase - the list of URLs (space-separated) from which the returned class loader will load classes from, or null **Returns:**a class loader that loads classes from the given codebase URL path **Throws:** [MalformedURLException](http://docs.google.com/java/net/MalformedURLException.html) - if codebase is non-null and contains an invalid URL, or if codebase is null and a provider-specific URL used to identify the class loader is invalid [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if there is a security manager and the invocation of its checkPermission method fails, or if the caller does not have permission to connect to all of the URLs in the codebase URL path**Since:** 1.3

### getClassAnnotation

public static [String](http://docs.google.com/java/lang/String.html) **getClassAnnotation**([Class](http://docs.google.com/java/lang/Class.html)<?> cl)

Returns the annotation string (representing a location for the class definition) that RMI will use to annotate the class descriptor when marshalling objects of the given class.

This method delegates to the [RMIClassLoaderSpi.getClassAnnotation(Class)](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html#getClassAnnotation(java.lang.Class)) method of the provider instance, passing cl as the argument.

**Parameters:**cl - the class to obtain the annotation for **Returns:**a string to be used to annotate the given class when it gets marshalled, or null **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if cl is null**Since:** 1.2

### getDefaultProviderInstance

public static [RMIClassLoaderSpi](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html) **getDefaultProviderInstance**()

Returns the canonical instance of the default provider for the service provider interface [RMIClassLoaderSpi](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html). If the system property java.rmi.server.RMIClassLoaderSpi is not defined, then the RMIClassLoader static methods

* [loadClass(URL,String)](http://docs.google.com/java/rmi/server/RMIClassLoader.html#loadClass(java.net.URL,%20java.lang.String))
* [loadClass(String,String)](http://docs.google.com/java/rmi/server/RMIClassLoader.html#loadClass(java.lang.String,%20java.lang.String))
* [loadClass(String,String,ClassLoader)](http://docs.google.com/java/rmi/server/RMIClassLoader.html#loadClass(java.lang.String,%20java.lang.String,%20java.lang.ClassLoader))
* [loadProxyClass(String,String[],ClassLoader)](http://docs.google.com/java/rmi/server/RMIClassLoader.html#loadProxyClass(java.lang.String,%20java.lang.String%5B%5D,%20java.lang.ClassLoader))
* [getClassLoader(String)](http://docs.google.com/java/rmi/server/RMIClassLoader.html#getClassLoader(java.lang.String))
* [getClassAnnotation(Class)](http://docs.google.com/java/rmi/server/RMIClassLoader.html#getClassAnnotation(java.lang.Class))

will use the canonical instance of the default provider as the service provider instance.

If there is a security manager, its checkPermission method will be invoked with a RuntimePermission("setFactory") permission; this could result in a SecurityException.

The default service provider instance implements [RMIClassLoaderSpi](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html) as follows:

The [**getClassAnnotation**](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html#getClassAnnotation(java.lang.Class)) method returns a String representing the codebase URL path that a remote party should use to download the definition for the specified class. The format of the returned string is a path of URLs separated by spaces. The codebase string returned depends on the defining class loader of the specified class:

* If the class loader is the system class loader (see [ClassLoader.getSystemClassLoader()](http://docs.google.com/java/lang/ClassLoader.html#getSystemClassLoader())), a parent of the system class loader such as the loader used for installed extensions, or the bootstrap class loader (which may be represented by null), then the value of the java.rmi.server.codebase property (or possibly an earlier cached value) is returned, or null is returned if that property is not set.
* Otherwise, if the class loader is an instance of URLClassLoader, then the returned string is a space-separated list of the external forms of the URLs returned by invoking the getURLs methods of the loader. If the URLClassLoader was created by this provider to service an invocation of its loadClass or loadProxyClass methods, then no permissions are required to get the associated codebase string. If it is an arbitrary other URLClassLoader instance, then if there is a security manager, its checkPermission method will be invoked once for each URL returned by the getURLs method, with the permission returned by invoking openConnection().getPermission() on each URL; if any of those invocations throws a SecurityException or an IOException, then the value of the java.rmi.server.codebase property (or possibly an earlier cached value) is returned, or null is returned if that property is not set.
* Finally, if the class loader is not an instance of URLClassLoader, then the value of the java.rmi.server.codebase property (or possibly an earlier cached value) is returned, or null is returned if that property is not set.

For the implementations of the methods described below, which all take a String parameter named codebase that is a space-separated list of URLs, each invocation has an associated *codebase loader* that is identified using the codebase argument in conjunction with the current thread's context class loader (see [Thread.getContextClassLoader()](http://docs.google.com/java/lang/Thread.html#getContextClassLoader())). When there is a security manager, this provider maintains an internal table of class loader instances (which are at least instances of [URLClassLoader](http://docs.google.com/java/net/URLClassLoader.html)) keyed by the pair of their parent class loader and their codebase URL path (an ordered list of URLs). If the codebase argument is null, the codebase URL path is the value of the system property java.rmi.server.codebase or possibly an earlier cached value. For a given codebase URL path passed as the codebase argument to an invocation of one of the below methods in a given context, the codebase loader is the loader in the table with the specified codebase URL path and the current thread's context class loader as its parent. If no such loader exists, then one is created and added to the table. The table does not maintain strong references to its contained loaders, in order to allow them and their defined classes to be garbage collected when not otherwise reachable. In order to prevent arbitrary untrusted code from being implicitly loaded into a virtual machine with no security manager, if there is no security manager set, the codebase loader is just the current thread's context class loader (the supplied codebase URL path is ignored, so remote class loading is disabled).

The [**getClassLoader**](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html#getClassLoader(java.lang.String)) method returns the codebase loader for the specified codebase URL path. If there is a security manager, then if the calling context does not have permission to connect to all of the URLs in the codebase URL path, a SecurityException will be thrown.

The [**loadClass**](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html#loadClass(java.lang.String,%20java.lang.String,%20java.lang.ClassLoader)) method attempts to load the class with the specified name as follows:

If the defaultLoader argument is non-null, it first attempts to load the class with the specified name using the defaultLoader, such as by evaluating

Class.forName(name, false, defaultLoader)

If the class is successfully loaded from the defaultLoader, that class is returned. If an exception other than ClassNotFoundException is thrown, that exception is thrown to the caller.

Next, the loadClass method attempts to load the class with the specified name using the codebase loader for the specified codebase URL path. If there is a security manager, then the calling context must have permission to connect to all of the URLs in the codebase URL path; otherwise, the current thread's context class loader will be used instead of the codebase loader.

The [**loadProxyClass**](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html#loadProxyClass(java.lang.String,%20java.lang.String%5B%5D,%20java.lang.ClassLoader)) method attempts to return a dynamic proxy class with the named interface as follows:

If the defaultLoader argument is non-null and all of the named interfaces can be resolved through that loader, then,

* if all of the resolved interfaces are public, then it first attempts to obtain a dynamic proxy class (using [Proxy.getProxyClass](http://docs.google.com/java/lang/reflect/Proxy.html#getProxyClass(java.lang.ClassLoader,%20java.lang.Class...))) for the resolved interfaces defined in the codebase loader; if that attempt throws an IllegalArgumentException, it then attempts to obtain a dynamic proxy class for the resolved interfaces defined in the defaultLoader. If both attempts throw IllegalArgumentException, then this method throws a ClassNotFoundException. If any other exception is thrown, that exception is thrown to the caller.
* if all of the non-public resolved interfaces are defined in the same class loader, then it attempts to obtain a dynamic proxy class for the resolved interfaces defined in that loader.
* otherwise, a LinkageError is thrown (because a class that implements all of the specified interfaces cannot be defined in any loader).

Otherwise, if all of the named interfaces can be resolved through the codebase loader, then,

* if all of the resolved interfaces are public, then it attempts to obtain a dynamic proxy class for the resolved interfaces in the codebase loader. If the attempt throws an IllegalArgumentException, then this method throws a ClassNotFoundException.
* if all of the non-public resolved interfaces are defined in the same class loader, then it attempts to obtain a dynamic proxy class for the resolved interfaces defined in that loader.
* otherwise, a LinkageError is thrown (because a class that implements all of the specified interfaces cannot be defined in any loader).

Otherwise, a ClassNotFoundException is thrown for one of the named interfaces that could not be resolved.

**Returns:**the canonical instance of the default service provider **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if there is a security manager and the invocation of its checkPermission method fails**Since:** 1.4

### getSecurityContext

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public static [Object](http://docs.google.com/java/lang/Object.html) **getSecurityContext**([ClassLoader](http://docs.google.com/java/lang/ClassLoader.html) loader)

**Deprecated.** *no replacement. As of the Java 2 platform v1.2, RMI no longer uses this method to obtain a class loader's security context.*

Returns the security context of the given class loader.

**Parameters:**loader - a class loader from which to get the security context **Returns:**the security context**See Also:**[SecurityManager.getSecurityContext()](http://docs.google.com/java/lang/SecurityManager.html#getSecurityContext())

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/RMIClassLoader.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/RemoteStub.html)   [**NEXT CLASS**](http://docs.google.com/java/rmi/server/RMIClassLoaderSpi.html) | [**FRAMES**](http://docs.google.com/index.html?java/rmi/server/RMIClassLoader.html)    [**NO FRAMES**](http://docs.google.com/RMIClassLoader.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#3znysh7) | DETAIL: FIELD | CONSTR | [METHOD](#tyjcwt) |

[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).