| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/InitialContext.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/naming/ContextNotEmptyException.html)   [**NEXT CLASS**](http://docs.google.com/javax/naming/InsufficientResourcesException.html) | [**FRAMES**](http://docs.google.com/index.html?javax/naming/InitialContext.html)    [**NO FRAMES**](http://docs.google.com/InitialContext.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#2et92p0) | [CONSTR](#3dy6vkm) | [METHOD](#1t3h5sf) | DETAIL: [FIELD](#2s8eyo1) | [CONSTR](#lnxbz9) | [METHOD](#2jxsxqh) |

## **javax.naming**

Class InitialContext

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 **javax.naming.InitialContext**

**All Implemented Interfaces:** [Context](http://docs.google.com/javax/naming/Context.html) **Direct Known Subclasses:** [InitialDirContext](http://docs.google.com/javax/naming/directory/InitialDirContext.html)

public class **InitialContext**extends [Object](http://docs.google.com/java/lang/Object.html)implements [Context](http://docs.google.com/javax/naming/Context.html)

This class is the starting context for performing naming operations.

All naming operations are relative to a context. The initial context implements the Context interface and provides the starting point for resolution of names.

When the initial context is constructed, its environment is initialized with properties defined in the environment parameter passed to the constructor, and in any [application resource files](http://docs.google.com/Context.html#RESOURCEFILES). In addition, a small number of standard JNDI properties may be specified as system properties or as applet parameters (through the use of [Context.APPLET](http://docs.google.com/javax/naming/Context.html#APPLET)). These special properties are listed in the field detail sections of the [Context](http://docs.google.com/Context.html#field_detail) and [LdapContext](http://docs.google.com/ldap/LdapContext.html#field_detail) interface documentation.

JNDI determines each property's value by merging the values from the following two sources, in order:

1. The first occurrence of the property from the constructor's environment parameter and (for appropriate properties) the applet parameters and system properties.
2. The application resource files (jndi.properties).

For each property found in both of these two sources, or in more than one application resource file, the property's value is determined as follows. If the property is one of the standard JNDI properties that specify a list of JNDI factories (see [Context](http://docs.google.com/Context.html#LISTPROPS)), all of the values are concatenated into a single colon-separated list. For other properties, only the first value found is used.

The initial context implementation is determined at runtime. The default policy uses the environment property "[java.naming.factory.initial](http://docs.google.com/javax/naming/Context.html#INITIAL_CONTEXT_FACTORY)", which contains the class name of the initial context factory. An exception to this policy is made when resolving URL strings, as described below.

When a URL string (a String of the form *scheme\_id:rest\_of\_name*) is passed as a name parameter to any method, a URL context factory for handling that scheme is located and used to resolve the URL. If no such factory is found, the initial context specified by "java.naming.factory.initial" is used. Similarly, when a CompositeName object whose first component is a URL string is passed as a name parameter to any method, a URL context factory is located and used to resolve the first name component. See [NamingManager.getURLContext()](http://docs.google.com/javax/naming/spi/NamingManager.html#getURLContext(java.lang.String,%20java.util.Hashtable)) for a description of how URL context factories are located.

This default policy of locating the initial context and URL context factories may be overridden by calling NamingManager.setInitialContextFactoryBuilder().

NoInitialContextException is thrown when an initial context cannot be instantiated. This exception can be thrown during any interaction with the InitialContext, not only when the InitialContext is constructed. For example, the implementation of the initial context might lazily retrieve the context only when actual methods are invoked on it. The application should not have any dependency on when the existence of an initial context is determined.

When the environment property "java.naming.factory.initial" is non-null, the InitialContext constructor will attempt to create the initial context specified therein. At that time, the initial context factory involved might throw an exception if a problem is encountered. However, it is provider implementation-dependent when it verifies and indicates to the users of the initial context any environment property- or connection- related problems. It can do so lazily--delaying until an operation is performed on the context, or eagerly, at the time the context is constructed.

An InitialContext instance is not synchronized against concurrent access by multiple threads. Multiple threads each manipulating a different InitialContext instance need not synchronize. Threads that need to access a single InitialContext instance concurrently should synchronize amongst themselves and provide the necessary locking.

**Since:** JNDI 1.1 / Java 2 Platform, Standard Edition, v 1.3 **See Also:**[Context](http://docs.google.com/javax/naming/Context.html), [NamingManager.setInitialContextFactoryBuilder](http://docs.google.com/javax/naming/spi/NamingManager.html#setInitialContextFactoryBuilder(javax.naming.spi.InitialContextFactoryBuilder))

| **Field Summary** | |
| --- | --- |
| protected  [Context](http://docs.google.com/javax/naming/Context.html) | [**defaultInitCtx**](http://docs.google.com/javax/naming/InitialContext.html#defaultInitCtx)            Field holding the result of calling NamingManager.getInitialContext(). |
| protected  boolean | [**gotDefault**](http://docs.google.com/javax/naming/InitialContext.html#gotDefault)            Field indicating whether the initial context has been obtained by calling NamingManager.getInitialContext(). |
| protected  [Hashtable](http://docs.google.com/java/util/Hashtable.html)<[Object](http://docs.google.com/java/lang/Object.html),[Object](http://docs.google.com/java/lang/Object.html)> | [**myProps**](http://docs.google.com/javax/naming/InitialContext.html#myProps)            The environment associated with this InitialContext. |

| **Fields inherited from interface javax.naming.**[**Context**](http://docs.google.com/javax/naming/Context.html) |
| --- |
| [APPLET](http://docs.google.com/javax/naming/Context.html#APPLET), [AUTHORITATIVE](http://docs.google.com/javax/naming/Context.html#AUTHORITATIVE), [BATCHSIZE](http://docs.google.com/javax/naming/Context.html#BATCHSIZE), [DNS\_URL](http://docs.google.com/javax/naming/Context.html#DNS_URL), [INITIAL\_CONTEXT\_FACTORY](http://docs.google.com/javax/naming/Context.html#INITIAL_CONTEXT_FACTORY), [LANGUAGE](http://docs.google.com/javax/naming/Context.html#LANGUAGE), [OBJECT\_FACTORIES](http://docs.google.com/javax/naming/Context.html#OBJECT_FACTORIES), [PROVIDER\_URL](http://docs.google.com/javax/naming/Context.html#PROVIDER_URL), [REFERRAL](http://docs.google.com/javax/naming/Context.html#REFERRAL), [SECURITY\_AUTHENTICATION](http://docs.google.com/javax/naming/Context.html#SECURITY_AUTHENTICATION), [SECURITY\_CREDENTIALS](http://docs.google.com/javax/naming/Context.html#SECURITY_CREDENTIALS), [SECURITY\_PRINCIPAL](http://docs.google.com/javax/naming/Context.html#SECURITY_PRINCIPAL), [SECURITY\_PROTOCOL](http://docs.google.com/javax/naming/Context.html#SECURITY_PROTOCOL), [STATE\_FACTORIES](http://docs.google.com/javax/naming/Context.html#STATE_FACTORIES), [URL\_PKG\_PREFIXES](http://docs.google.com/javax/naming/Context.html#URL_PKG_PREFIXES) |

| **Constructor Summary** | |
| --- | --- |
|  | [**InitialContext**](http://docs.google.com/javax/naming/InitialContext.html#InitialContext())()            Constructs an initial context. |
| protected | [**InitialContext**](http://docs.google.com/javax/naming/InitialContext.html#InitialContext(boolean))(boolean lazy)            Constructs an initial context with the option of not initializing it. |
|  | [**InitialContext**](http://docs.google.com/javax/naming/InitialContext.html#InitialContext(java.util.Hashtable))([Hashtable](http://docs.google.com/java/util/Hashtable.html)<?,?> environment)            Constructs an initial context using the supplied environment. |

| **Method Summary** | |
| --- | --- |
| [Object](http://docs.google.com/java/lang/Object.html) | [**addToEnvironment**](http://docs.google.com/javax/naming/InitialContext.html#addToEnvironment(java.lang.String,%20java.lang.Object))([String](http://docs.google.com/java/lang/String.html) propName, [Object](http://docs.google.com/java/lang/Object.html) propVal)            Adds a new environment property to the environment of this context. |
| void | [**bind**](http://docs.google.com/javax/naming/InitialContext.html#bind(javax.naming.Name,%20java.lang.Object))([Name](http://docs.google.com/javax/naming/Name.html) name, [Object](http://docs.google.com/java/lang/Object.html) obj)            Binds a name to an object. |
| void | [**bind**](http://docs.google.com/javax/naming/InitialContext.html#bind(java.lang.String,%20java.lang.Object))([String](http://docs.google.com/java/lang/String.html) name, [Object](http://docs.google.com/java/lang/Object.html) obj)            Binds a name to an object. |
| void | [**close**](http://docs.google.com/javax/naming/InitialContext.html#close())()            Closes this context. |
| [Name](http://docs.google.com/javax/naming/Name.html) | [**composeName**](http://docs.google.com/javax/naming/InitialContext.html#composeName(javax.naming.Name,%20javax.naming.Name))([Name](http://docs.google.com/javax/naming/Name.html) name, [Name](http://docs.google.com/javax/naming/Name.html) prefix)            Composes the name of this context with a name relative to this context. |
| [String](http://docs.google.com/java/lang/String.html) | [**composeName**](http://docs.google.com/javax/naming/InitialContext.html#composeName(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) name, [String](http://docs.google.com/java/lang/String.html) prefix)            Composes the name of this context with a name relative to this context. |
| [Context](http://docs.google.com/javax/naming/Context.html) | [**createSubcontext**](http://docs.google.com/javax/naming/InitialContext.html#createSubcontext(javax.naming.Name))([Name](http://docs.google.com/javax/naming/Name.html) name)            Creates and binds a new context. |
| [Context](http://docs.google.com/javax/naming/Context.html) | [**createSubcontext**](http://docs.google.com/javax/naming/InitialContext.html#createSubcontext(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)            Creates and binds a new context. |
| void | [**destroySubcontext**](http://docs.google.com/javax/naming/InitialContext.html#destroySubcontext(javax.naming.Name))([Name](http://docs.google.com/javax/naming/Name.html) name)            Destroys the named context and removes it from the namespace. |
| void | [**destroySubcontext**](http://docs.google.com/javax/naming/InitialContext.html#destroySubcontext(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)            Destroys the named context and removes it from the namespace. |
| static   | <T> T | | --- | | [**doLookup**](http://docs.google.com/javax/naming/InitialContext.html#doLookup(javax.naming.Name))([Name](http://docs.google.com/javax/naming/Name.html) name)            A static method to retrieve the named object. |
| static   | <T> T | | --- | | [**doLookup**](http://docs.google.com/javax/naming/InitialContext.html#doLookup(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)            A static method to retrieve the named object. |
| protected  [Context](http://docs.google.com/javax/naming/Context.html) | [**getDefaultInitCtx**](http://docs.google.com/javax/naming/InitialContext.html#getDefaultInitCtx())()            Retrieves the initial context by calling NamingManager.getInitialContext() and cache it in defaultInitCtx. |
| [Hashtable](http://docs.google.com/java/util/Hashtable.html)<?,?> | [**getEnvironment**](http://docs.google.com/javax/naming/InitialContext.html#getEnvironment())()            Retrieves the environment in effect for this context. |
| [String](http://docs.google.com/java/lang/String.html) | [**getNameInNamespace**](http://docs.google.com/javax/naming/InitialContext.html#getNameInNamespace())()            Retrieves the full name of this context within its own namespace. |
| [NameParser](http://docs.google.com/javax/naming/NameParser.html) | [**getNameParser**](http://docs.google.com/javax/naming/InitialContext.html#getNameParser(javax.naming.Name))([Name](http://docs.google.com/javax/naming/Name.html) name)            Retrieves the parser associated with the named context. |
| [NameParser](http://docs.google.com/javax/naming/NameParser.html) | [**getNameParser**](http://docs.google.com/javax/naming/InitialContext.html#getNameParser(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)            Retrieves the parser associated with the named context. |
| protected  [Context](http://docs.google.com/javax/naming/Context.html) | [**getURLOrDefaultInitCtx**](http://docs.google.com/javax/naming/InitialContext.html#getURLOrDefaultInitCtx(javax.naming.Name))([Name](http://docs.google.com/javax/naming/Name.html) name)            Retrieves a context for resolving name. |
| protected  [Context](http://docs.google.com/javax/naming/Context.html) | [**getURLOrDefaultInitCtx**](http://docs.google.com/javax/naming/InitialContext.html#getURLOrDefaultInitCtx(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)            Retrieves a context for resolving the string name name. |
| protected  void | [**init**](http://docs.google.com/javax/naming/InitialContext.html#init(java.util.Hashtable))([Hashtable](http://docs.google.com/java/util/Hashtable.html)<?,?> environment)            Initializes the initial context using the supplied environment. |
| [NamingEnumeration](http://docs.google.com/javax/naming/NamingEnumeration.html)<[NameClassPair](http://docs.google.com/javax/naming/NameClassPair.html)> | [**list**](http://docs.google.com/javax/naming/InitialContext.html#list(javax.naming.Name))([Name](http://docs.google.com/javax/naming/Name.html) name)            Enumerates the names bound in the named context, along with the class names of objects bound to them. |
| [NamingEnumeration](http://docs.google.com/javax/naming/NamingEnumeration.html)<[NameClassPair](http://docs.google.com/javax/naming/NameClassPair.html)> | [**list**](http://docs.google.com/javax/naming/InitialContext.html#list(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)            Enumerates the names bound in the named context, along with the class names of objects bound to them. |
| [NamingEnumeration](http://docs.google.com/javax/naming/NamingEnumeration.html)<[Binding](http://docs.google.com/javax/naming/Binding.html)> | [**listBindings**](http://docs.google.com/javax/naming/InitialContext.html#listBindings(javax.naming.Name))([Name](http://docs.google.com/javax/naming/Name.html) name)            Enumerates the names bound in the named context, along with the objects bound to them. |
| [NamingEnumeration](http://docs.google.com/javax/naming/NamingEnumeration.html)<[Binding](http://docs.google.com/javax/naming/Binding.html)> | [**listBindings**](http://docs.google.com/javax/naming/InitialContext.html#listBindings(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)            Enumerates the names bound in the named context, along with the objects bound to them. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**lookup**](http://docs.google.com/javax/naming/InitialContext.html#lookup(javax.naming.Name))([Name](http://docs.google.com/javax/naming/Name.html) name)            Retrieves the named object. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**lookup**](http://docs.google.com/javax/naming/InitialContext.html#lookup(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)            Retrieves the named object. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**lookupLink**](http://docs.google.com/javax/naming/InitialContext.html#lookupLink(javax.naming.Name))([Name](http://docs.google.com/javax/naming/Name.html) name)            Retrieves the named object, following links except for the terminal atomic component of the name. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**lookupLink**](http://docs.google.com/javax/naming/InitialContext.html#lookupLink(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)            Retrieves the named object, following links except for the terminal atomic component of the name. |
| void | [**rebind**](http://docs.google.com/javax/naming/InitialContext.html#rebind(javax.naming.Name,%20java.lang.Object))([Name](http://docs.google.com/javax/naming/Name.html) name, [Object](http://docs.google.com/java/lang/Object.html) obj)            Binds a name to an object, overwriting any existing binding. |
| void | [**rebind**](http://docs.google.com/javax/naming/InitialContext.html#rebind(java.lang.String,%20java.lang.Object))([String](http://docs.google.com/java/lang/String.html) name, [Object](http://docs.google.com/java/lang/Object.html) obj)            Binds a name to an object, overwriting any existing binding. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**removeFromEnvironment**](http://docs.google.com/javax/naming/InitialContext.html#removeFromEnvironment(java.lang.String))([String](http://docs.google.com/java/lang/String.html) propName)            Removes an environment property from the environment of this context. |
| void | [**rename**](http://docs.google.com/javax/naming/InitialContext.html#rename(javax.naming.Name,%20javax.naming.Name))([Name](http://docs.google.com/javax/naming/Name.html) oldName, [Name](http://docs.google.com/javax/naming/Name.html) newName)            Binds a new name to the object bound to an old name, and unbinds the old name. |
| void | [**rename**](http://docs.google.com/javax/naming/InitialContext.html#rename(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) oldName, [String](http://docs.google.com/java/lang/String.html) newName)            Binds a new name to the object bound to an old name, and unbinds the old name. |
| void | [**unbind**](http://docs.google.com/javax/naming/InitialContext.html#unbind(javax.naming.Name))([Name](http://docs.google.com/javax/naming/Name.html) name)            Unbinds the named object. |
| void | [**unbind**](http://docs.google.com/javax/naming/InitialContext.html#unbind(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)            Unbinds the named 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()), [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)) |

| **Field Detail** |
| --- |

### myProps

protected [Hashtable](http://docs.google.com/java/util/Hashtable.html)<[Object](http://docs.google.com/java/lang/Object.html),[Object](http://docs.google.com/java/lang/Object.html)> **myProps**

The environment associated with this InitialContext. It is initialized to null and is updated by the constructor that accepts an environment or by the init() method.

**See Also:**[addToEnvironment(java.lang.String, java.lang.Object)](http://docs.google.com/javax/naming/InitialContext.html#addToEnvironment(java.lang.String,%20java.lang.Object)), [removeFromEnvironment(java.lang.String)](http://docs.google.com/javax/naming/InitialContext.html#removeFromEnvironment(java.lang.String)), [getEnvironment()](http://docs.google.com/javax/naming/InitialContext.html#getEnvironment())

### defaultInitCtx

protected [Context](http://docs.google.com/javax/naming/Context.html) **defaultInitCtx**

Field holding the result of calling NamingManager.getInitialContext(). It is set by getDefaultInitCtx() the first time getDefaultInitCtx() is called. Subsequent invocations of getDefaultInitCtx() return the value of defaultInitCtx.

**See Also:**[getDefaultInitCtx()](http://docs.google.com/javax/naming/InitialContext.html#getDefaultInitCtx())

### gotDefault

protected boolean **gotDefault**

Field indicating whether the initial context has been obtained by calling NamingManager.getInitialContext(). If true, its result is in defaultInitCtx.

| **Constructor Detail** |
| --- |

### InitialContext

protected **InitialContext**(boolean lazy)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

Constructs an initial context with the option of not initializing it. This may be used by a constructor in a subclass when the value of the environment parameter is not yet known at the time the InitialContext constructor is called. The subclass's constructor will call this constructor, compute the value of the environment, and then call init() before returning.

**Parameters:**lazy - true means do not initialize the initial context; false is equivalent to calling new InitialContext() **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered**Since:** 1.3 **See Also:**[init(Hashtable)](http://docs.google.com/javax/naming/InitialContext.html#init(java.util.Hashtable))

### InitialContext

public **InitialContext**()  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

Constructs an initial context. No environment properties are supplied. Equivalent to new InitialContext(null).

**Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered**See Also:**[InitialContext(Hashtable)](http://docs.google.com/javax/naming/InitialContext.html#InitialContext(java.util.Hashtable))

### InitialContext

public **InitialContext**([Hashtable](http://docs.google.com/java/util/Hashtable.html)<?,?> environment)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

Constructs an initial context using the supplied environment. Environment properties are discussed in the class description.

This constructor will not modify environment or save a reference to it, but may save a clone.

**Parameters:**environment - environment used to create the initial context. Null indicates an empty environment. **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered

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

### init

protected void **init**([Hashtable](http://docs.google.com/java/util/Hashtable.html)<?,?> environment)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

Initializes the initial context using the supplied environment. Environment properties are discussed in the class description.

This method will modify environment and save a reference to it. The caller may no longer modify it.

**Parameters:**environment - environment used to create the initial context. Null indicates an empty environment. **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered**Since:** 1.3 **See Also:**[InitialContext(boolean)](http://docs.google.com/javax/naming/InitialContext.html#InitialContext(boolean))

### doLookup

public static <T> T **doLookup**([Name](http://docs.google.com/javax/naming/Name.html) name)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

A static method to retrieve the named object. This is a shortcut method equivalent to invoking:

InitialContext ic = new InitialContext(); Object obj = ic.lookup();

If name is empty, returns a new instance of this context (which represents the same naming context as this context, but its environment may be modified independently and it may be accessed concurrently).

**Parameters:**name - the name of the object to look up **Returns:**the object bound to name **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered**Since:** 1.6 **See Also:**[doLookup(String)](http://docs.google.com/javax/naming/InitialContext.html#doLookup(java.lang.String)), [lookup(Name)](http://docs.google.com/javax/naming/InitialContext.html#lookup(javax.naming.Name))

### doLookup

public static <T> T **doLookup**([String](http://docs.google.com/java/lang/String.html) name)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

A static method to retrieve the named object. See [doLookup(Name)](http://docs.google.com/javax/naming/InitialContext.html#doLookup(javax.naming.Name)) for details.

**Parameters:**name - the name of the object to look up **Returns:**the object bound to name **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered**Since:** 1.6

### getDefaultInitCtx

protected [Context](http://docs.google.com/javax/naming/Context.html) **getDefaultInitCtx**()  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

Retrieves the initial context by calling NamingManager.getInitialContext() and cache it in defaultInitCtx. Set gotDefault so that we know we've tried this before.

**Returns:**The non-null cached initial context. **Throws:** [NoInitialContextException](http://docs.google.com/javax/naming/NoInitialContextException.html) - If cannot find an initial context. [NamingException](http://docs.google.com/javax/naming/NamingException.html) - If a naming exception was encountered.

### getURLOrDefaultInitCtx

protected [Context](http://docs.google.com/javax/naming/Context.html) **getURLOrDefaultInitCtx**([String](http://docs.google.com/java/lang/String.html) name)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

Retrieves a context for resolving the string name name. If name name is a URL string, then attempt to find a URL context for it. If none is found, or if name is not a URL string, then return getDefaultInitCtx().

See getURLOrDefaultInitCtx(Name) for description of how a subclass should use this method.

**Parameters:**name - The non-null name for which to get the context. **Returns:**A URL context for name or the cached initial context. The result cannot be null. **Throws:** [NoInitialContextException](http://docs.google.com/javax/naming/NoInitialContextException.html) - If cannot find an initial context. [NamingException](http://docs.google.com/javax/naming/NamingException.html) - In a naming exception is encountered.**See Also:**[NamingManager.getURLContext(java.lang.String, java.util.Hashtable)](http://docs.google.com/javax/naming/spi/NamingManager.html#getURLContext(java.lang.String,%20java.util.Hashtable))

### getURLOrDefaultInitCtx

protected [Context](http://docs.google.com/javax/naming/Context.html) **getURLOrDefaultInitCtx**([Name](http://docs.google.com/javax/naming/Name.html) name)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

Retrieves a context for resolving name. If the first component of name name is a URL string, then attempt to find a URL context for it. If none is found, or if the first component of name is not a URL string, then return getDefaultInitCtx().

When creating a subclass of InitialContext, use this method as follows. Define a new method that uses this method to get an initial context of the desired subclass.

protected XXXContext getURLOrDefaultInitXXXCtx(Name name)   
 throws NamingException {  
 Context answer = getURLOrDefaultInitCtx(name);  
 if (!(answer instanceof XXXContext)) {  
 if (answer == null) {  
 throw new NoInitialContextException();  
 } else {  
 throw new NotContextException("Not an XXXContext");  
 }  
 }  
 return (XXXContext)answer;  
 }

When providing implementations for the new methods in the subclass, use this newly defined method to get the initial context.

public Object XXXMethod1(Name name, ...) {  
 throws NamingException {  
 return getURLOrDefaultInitXXXCtx(name).XXXMethod1(name, ...);  
 }

**Parameters:**name - The non-null name for which to get the context. **Returns:**A URL context for name or the cached initial context. The result cannot be null. **Throws:** [NoInitialContextException](http://docs.google.com/javax/naming/NoInitialContextException.html) - If cannot find an initial context. [NamingException](http://docs.google.com/javax/naming/NamingException.html) - In a naming exception is encountered.**See Also:**[NamingManager.getURLContext(java.lang.String, java.util.Hashtable)](http://docs.google.com/javax/naming/spi/NamingManager.html#getURLContext(java.lang.String,%20java.util.Hashtable))

### lookup

public [Object](http://docs.google.com/java/lang/Object.html) **lookup**([String](http://docs.google.com/java/lang/String.html) name)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#lookup(java.lang.String)) Retrieves the named object. See [Context.lookup(Name)](http://docs.google.com/javax/naming/Context.html#lookup(javax.naming.Name)) for details.

**Specified by:**[lookup](http://docs.google.com/javax/naming/Context.html#lookup(java.lang.String)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - the name of the object to look up **Returns:**the object bound to name **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered

### lookup

public [Object](http://docs.google.com/java/lang/Object.html) **lookup**([Name](http://docs.google.com/javax/naming/Name.html) name)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#lookup(javax.naming.Name)) Retrieves the named object. If name is empty, returns a new instance of this context (which represents the same naming context as this context, but its environment may be modified independently and it may be accessed concurrently).

**Specified by:**[lookup](http://docs.google.com/javax/naming/Context.html#lookup(javax.naming.Name)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - the name of the object to look up **Returns:**the object bound to name **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered**See Also:**[Context.lookup(String)](http://docs.google.com/javax/naming/Context.html#lookup(java.lang.String)), [Context.lookupLink(Name)](http://docs.google.com/javax/naming/Context.html#lookupLink(javax.naming.Name))

### bind

public void **bind**([String](http://docs.google.com/java/lang/String.html) name,  
 [Object](http://docs.google.com/java/lang/Object.html) obj)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#bind(java.lang.String,%20java.lang.Object)) Binds a name to an object. See [Context.bind(Name, Object)](http://docs.google.com/javax/naming/Context.html#bind(javax.naming.Name,%20java.lang.Object)) for details.

**Specified by:**[bind](http://docs.google.com/javax/naming/Context.html#bind(java.lang.String,%20java.lang.Object)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - the name to bind; may not be emptyobj - the object to bind; possibly null **Throws:** [NameAlreadyBoundException](http://docs.google.com/javax/naming/NameAlreadyBoundException.html) - if name is already bound [InvalidAttributesException](http://docs.google.com/javax/naming/directory/InvalidAttributesException.html) - if object did not supply all mandatory attributes [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered

### bind

public void **bind**([Name](http://docs.google.com/javax/naming/Name.html) name,  
 [Object](http://docs.google.com/java/lang/Object.html) obj)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#bind(javax.naming.Name,%20java.lang.Object)) Binds a name to an object. All intermediate contexts and the target context (that named by all but terminal atomic component of the name) must already exist.

**Specified by:**[bind](http://docs.google.com/javax/naming/Context.html#bind(javax.naming.Name,%20java.lang.Object)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - the name to bind; may not be emptyobj - the object to bind; possibly null **Throws:** [NameAlreadyBoundException](http://docs.google.com/javax/naming/NameAlreadyBoundException.html) - if name is already bound [InvalidAttributesException](http://docs.google.com/javax/naming/directory/InvalidAttributesException.html) - if object did not supply all mandatory attributes [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered**See Also:**[Context.bind(String, Object)](http://docs.google.com/javax/naming/Context.html#bind(java.lang.String,%20java.lang.Object)), [Context.rebind(Name, Object)](http://docs.google.com/javax/naming/Context.html#rebind(javax.naming.Name,%20java.lang.Object)), [DirContext.bind(Name, Object, javax.naming.directory.Attributes)](http://docs.google.com/javax/naming/directory/DirContext.html#bind(javax.naming.Name,%20java.lang.Object,%20javax.naming.directory.Attributes))

### rebind

public void **rebind**([String](http://docs.google.com/java/lang/String.html) name,  
 [Object](http://docs.google.com/java/lang/Object.html) obj)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#rebind(java.lang.String,%20java.lang.Object)) Binds a name to an object, overwriting any existing binding. See [Context.rebind(Name, Object)](http://docs.google.com/javax/naming/Context.html#rebind(javax.naming.Name,%20java.lang.Object)) for details.

**Specified by:**[rebind](http://docs.google.com/javax/naming/Context.html#rebind(java.lang.String,%20java.lang.Object)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - the name to bind; may not be emptyobj - the object to bind; possibly null **Throws:** [InvalidAttributesException](http://docs.google.com/javax/naming/directory/InvalidAttributesException.html) - if object did not supply all mandatory attributes [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered

### rebind

public void **rebind**([Name](http://docs.google.com/javax/naming/Name.html) name,  
 [Object](http://docs.google.com/java/lang/Object.html) obj)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#rebind(javax.naming.Name,%20java.lang.Object)) Binds a name to an object, overwriting any existing binding. All intermediate contexts and the target context (that named by all but terminal atomic component of the name) must already exist.

If the object is a DirContext, any existing attributes associated with the name are replaced with those of the object. Otherwise, any existing attributes associated with the name remain unchanged.

**Specified by:**[rebind](http://docs.google.com/javax/naming/Context.html#rebind(javax.naming.Name,%20java.lang.Object)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - the name to bind; may not be emptyobj - the object to bind; possibly null **Throws:** [InvalidAttributesException](http://docs.google.com/javax/naming/directory/InvalidAttributesException.html) - if object did not supply all mandatory attributes [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered**See Also:**[Context.rebind(String, Object)](http://docs.google.com/javax/naming/Context.html#rebind(java.lang.String,%20java.lang.Object)), [Context.bind(Name, Object)](http://docs.google.com/javax/naming/Context.html#bind(javax.naming.Name,%20java.lang.Object)), [DirContext.rebind(Name, Object, javax.naming.directory.Attributes)](http://docs.google.com/javax/naming/directory/DirContext.html#rebind(javax.naming.Name,%20java.lang.Object,%20javax.naming.directory.Attributes)), [DirContext](http://docs.google.com/javax/naming/directory/DirContext.html)

### unbind

public void **unbind**([String](http://docs.google.com/java/lang/String.html) name)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#unbind(java.lang.String)) Unbinds the named object. See [Context.unbind(Name)](http://docs.google.com/javax/naming/Context.html#unbind(javax.naming.Name)) for details.

**Specified by:**[unbind](http://docs.google.com/javax/naming/Context.html#unbind(java.lang.String)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - the name to unbind; may not be empty **Throws:** [NameNotFoundException](http://docs.google.com/javax/naming/NameNotFoundException.html) - if an intermediate context does not exist [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered

### unbind

public void **unbind**([Name](http://docs.google.com/javax/naming/Name.html) name)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#unbind(javax.naming.Name)) Unbinds the named object. Removes the terminal atomic name in name from the target context--that named by all but the terminal atomic part of name.

This method is idempotent. It succeeds even if the terminal atomic name is not bound in the target context, but throws NameNotFoundException if any of the intermediate contexts do not exist.

Any attributes associated with the name are removed. Intermediate contexts are not changed.

**Specified by:**[unbind](http://docs.google.com/javax/naming/Context.html#unbind(javax.naming.Name)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - the name to unbind; may not be empty **Throws:** [NameNotFoundException](http://docs.google.com/javax/naming/NameNotFoundException.html) - if an intermediate context does not exist [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered**See Also:**[Context.unbind(String)](http://docs.google.com/javax/naming/Context.html#unbind(java.lang.String))

### rename

public void **rename**([String](http://docs.google.com/java/lang/String.html) oldName,  
 [String](http://docs.google.com/java/lang/String.html) newName)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#rename(java.lang.String,%20java.lang.String)) Binds a new name to the object bound to an old name, and unbinds the old name. See [Context.rename(Name, Name)](http://docs.google.com/javax/naming/Context.html#rename(javax.naming.Name,%20javax.naming.Name)) for details.

**Specified by:**[rename](http://docs.google.com/javax/naming/Context.html#rename(java.lang.String,%20java.lang.String)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**oldName - the name of the existing binding; may not be emptynewName - the name of the new binding; may not be empty **Throws:** [NameAlreadyBoundException](http://docs.google.com/javax/naming/NameAlreadyBoundException.html) - if newName is already bound [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered

### rename

public void **rename**([Name](http://docs.google.com/javax/naming/Name.html) oldName,  
 [Name](http://docs.google.com/javax/naming/Name.html) newName)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#rename(javax.naming.Name,%20javax.naming.Name)) Binds a new name to the object bound to an old name, and unbinds the old name. Both names are relative to this context. Any attributes associated with the old name become associated with the new name. Intermediate contexts of the old name are not changed.

**Specified by:**[rename](http://docs.google.com/javax/naming/Context.html#rename(javax.naming.Name,%20javax.naming.Name)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**oldName - the name of the existing binding; may not be emptynewName - the name of the new binding; may not be empty **Throws:** [NameAlreadyBoundException](http://docs.google.com/javax/naming/NameAlreadyBoundException.html) - if newName is already bound [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered**See Also:**[Context.rename(String, String)](http://docs.google.com/javax/naming/Context.html#rename(java.lang.String,%20java.lang.String)), [Context.bind(Name, Object)](http://docs.google.com/javax/naming/Context.html#bind(javax.naming.Name,%20java.lang.Object)), [Context.rebind(Name, Object)](http://docs.google.com/javax/naming/Context.html#rebind(javax.naming.Name,%20java.lang.Object))

### list

public [NamingEnumeration](http://docs.google.com/javax/naming/NamingEnumeration.html)<[NameClassPair](http://docs.google.com/javax/naming/NameClassPair.html)> **list**([String](http://docs.google.com/java/lang/String.html) name)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#list(java.lang.String)) Enumerates the names bound in the named context, along with the class names of objects bound to them. See [Context.list(Name)](http://docs.google.com/javax/naming/Context.html#list(javax.naming.Name)) for details.

**Specified by:**[list](http://docs.google.com/javax/naming/Context.html#list(java.lang.String)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - the name of the context to list **Returns:**an enumeration of the names and class names of the bindings in this context. Each element of the enumeration is of type NameClassPair. **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered

### list

public [NamingEnumeration](http://docs.google.com/javax/naming/NamingEnumeration.html)<[NameClassPair](http://docs.google.com/javax/naming/NameClassPair.html)> **list**([Name](http://docs.google.com/javax/naming/Name.html) name)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#list(javax.naming.Name)) Enumerates the names bound in the named context, along with the class names of objects bound to them. The contents of any subcontexts are not included.

If a binding is added to or removed from this context, its effect on an enumeration previously returned is undefined.

**Specified by:**[list](http://docs.google.com/javax/naming/Context.html#list(javax.naming.Name)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - the name of the context to list **Returns:**an enumeration of the names and class names of the bindings in this context. Each element of the enumeration is of type NameClassPair. **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered**See Also:**[Context.list(String)](http://docs.google.com/javax/naming/Context.html#list(java.lang.String)), [Context.listBindings(Name)](http://docs.google.com/javax/naming/Context.html#listBindings(javax.naming.Name)), [NameClassPair](http://docs.google.com/javax/naming/NameClassPair.html)

### listBindings

public [NamingEnumeration](http://docs.google.com/javax/naming/NamingEnumeration.html)<[Binding](http://docs.google.com/javax/naming/Binding.html)> **listBindings**([String](http://docs.google.com/java/lang/String.html) name)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#listBindings(java.lang.String)) Enumerates the names bound in the named context, along with the objects bound to them. See [Context.listBindings(Name)](http://docs.google.com/javax/naming/Context.html#listBindings(javax.naming.Name)) for details.

**Specified by:**[listBindings](http://docs.google.com/javax/naming/Context.html#listBindings(java.lang.String)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - the name of the context to list **Returns:**an enumeration of the bindings in this context. Each element of the enumeration is of type Binding. **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered

### listBindings

public [NamingEnumeration](http://docs.google.com/javax/naming/NamingEnumeration.html)<[Binding](http://docs.google.com/javax/naming/Binding.html)> **listBindings**([Name](http://docs.google.com/javax/naming/Name.html) name)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#listBindings(javax.naming.Name)) Enumerates the names bound in the named context, along with the objects bound to them. The contents of any subcontexts are not included.

If a binding is added to or removed from this context, its effect on an enumeration previously returned is undefined.

**Specified by:**[listBindings](http://docs.google.com/javax/naming/Context.html#listBindings(javax.naming.Name)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - the name of the context to list **Returns:**an enumeration of the bindings in this context. Each element of the enumeration is of type Binding. **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered**See Also:**[Context.listBindings(String)](http://docs.google.com/javax/naming/Context.html#listBindings(java.lang.String)), [Context.list(Name)](http://docs.google.com/javax/naming/Context.html#list(javax.naming.Name)), [Binding](http://docs.google.com/javax/naming/Binding.html)

### destroySubcontext

public void **destroySubcontext**([String](http://docs.google.com/java/lang/String.html) name)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#destroySubcontext(java.lang.String)) Destroys the named context and removes it from the namespace. See [Context.destroySubcontext(Name)](http://docs.google.com/javax/naming/Context.html#destroySubcontext(javax.naming.Name)) for details.

**Specified by:**[destroySubcontext](http://docs.google.com/javax/naming/Context.html#destroySubcontext(java.lang.String)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - the name of the context to be destroyed; may not be empty **Throws:** [NameNotFoundException](http://docs.google.com/javax/naming/NameNotFoundException.html) - if an intermediate context does not exist [NotContextException](http://docs.google.com/javax/naming/NotContextException.html) - if the name is bound but does not name a context, or does not name a context of the appropriate type [ContextNotEmptyException](http://docs.google.com/javax/naming/ContextNotEmptyException.html) - if the named context is not empty [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered

### destroySubcontext

public void **destroySubcontext**([Name](http://docs.google.com/javax/naming/Name.html) name)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#destroySubcontext(javax.naming.Name)) Destroys the named context and removes it from the namespace. Any attributes associated with the name are also removed. Intermediate contexts are not destroyed.

This method is idempotent. It succeeds even if the terminal atomic name is not bound in the target context, but throws NameNotFoundException if any of the intermediate contexts do not exist.

In a federated naming system, a context from one naming system may be bound to a name in another. One can subsequently look up and perform operations on the foreign context using a composite name. However, an attempt destroy the context using this composite name will fail with NotContextException, because the foreign context is not a "subcontext" of the context in which it is bound. Instead, use unbind() to remove the binding of the foreign context. Destroying the foreign context requires that the destroySubcontext() be performed on a context from the foreign context's "native" naming system.

**Specified by:**[destroySubcontext](http://docs.google.com/javax/naming/Context.html#destroySubcontext(javax.naming.Name)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - the name of the context to be destroyed; may not be empty **Throws:** [NameNotFoundException](http://docs.google.com/javax/naming/NameNotFoundException.html) - if an intermediate context does not exist [NotContextException](http://docs.google.com/javax/naming/NotContextException.html) - if the name is bound but does not name a context, or does not name a context of the appropriate type [ContextNotEmptyException](http://docs.google.com/javax/naming/ContextNotEmptyException.html) - if the named context is not empty [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered**See Also:**[Context.destroySubcontext(String)](http://docs.google.com/javax/naming/Context.html#destroySubcontext(java.lang.String))

### createSubcontext

public [Context](http://docs.google.com/javax/naming/Context.html) **createSubcontext**([String](http://docs.google.com/java/lang/String.html) name)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#createSubcontext(java.lang.String)) Creates and binds a new context. See [Context.createSubcontext(Name)](http://docs.google.com/javax/naming/Context.html#createSubcontext(javax.naming.Name)) for details.

**Specified by:**[createSubcontext](http://docs.google.com/javax/naming/Context.html#createSubcontext(java.lang.String)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - the name of the context to create; may not be empty **Returns:**the newly created context **Throws:** [NameAlreadyBoundException](http://docs.google.com/javax/naming/NameAlreadyBoundException.html) - if name is already bound [InvalidAttributesException](http://docs.google.com/javax/naming/directory/InvalidAttributesException.html) - if creation of the subcontext requires specification of mandatory attributes [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered

### createSubcontext

public [Context](http://docs.google.com/javax/naming/Context.html) **createSubcontext**([Name](http://docs.google.com/javax/naming/Name.html) name)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#createSubcontext(javax.naming.Name)) Creates and binds a new context. Creates a new context with the given name and binds it in the target context (that named by all but terminal atomic component of the name). All intermediate contexts and the target context must already exist.

**Specified by:**[createSubcontext](http://docs.google.com/javax/naming/Context.html#createSubcontext(javax.naming.Name)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - the name of the context to create; may not be empty **Returns:**the newly created context **Throws:** [NameAlreadyBoundException](http://docs.google.com/javax/naming/NameAlreadyBoundException.html) - if name is already bound [InvalidAttributesException](http://docs.google.com/javax/naming/directory/InvalidAttributesException.html) - if creation of the subcontext requires specification of mandatory attributes [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered**See Also:**[Context.createSubcontext(String)](http://docs.google.com/javax/naming/Context.html#createSubcontext(java.lang.String)), [DirContext.createSubcontext(javax.naming.Name, javax.naming.directory.Attributes)](http://docs.google.com/javax/naming/directory/DirContext.html#createSubcontext(javax.naming.Name,%20javax.naming.directory.Attributes))

### lookupLink

public [Object](http://docs.google.com/java/lang/Object.html) **lookupLink**([String](http://docs.google.com/java/lang/String.html) name)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#lookupLink(java.lang.String)) Retrieves the named object, following links except for the terminal atomic component of the name. See [Context.lookupLink(Name)](http://docs.google.com/javax/naming/Context.html#lookupLink(javax.naming.Name)) for details.

**Specified by:**[lookupLink](http://docs.google.com/javax/naming/Context.html#lookupLink(java.lang.String)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - the name of the object to look up **Returns:**the object bound to name, not following the terminal link (if any) **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered

### lookupLink

public [Object](http://docs.google.com/java/lang/Object.html) **lookupLink**([Name](http://docs.google.com/javax/naming/Name.html) name)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#lookupLink(javax.naming.Name)) Retrieves the named object, following links except for the terminal atomic component of the name. If the object bound to name is not a link, returns the object itself.

**Specified by:**[lookupLink](http://docs.google.com/javax/naming/Context.html#lookupLink(javax.naming.Name)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - the name of the object to look up **Returns:**the object bound to name, not following the terminal link (if any). **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered**See Also:**[Context.lookupLink(String)](http://docs.google.com/javax/naming/Context.html#lookupLink(java.lang.String))

### getNameParser

public [NameParser](http://docs.google.com/javax/naming/NameParser.html) **getNameParser**([String](http://docs.google.com/java/lang/String.html) name)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#getNameParser(java.lang.String)) Retrieves the parser associated with the named context. See [Context.getNameParser(Name)](http://docs.google.com/javax/naming/Context.html#getNameParser(javax.naming.Name)) for details.

**Specified by:**[getNameParser](http://docs.google.com/javax/naming/Context.html#getNameParser(java.lang.String)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - the name of the context from which to get the parser **Returns:**a name parser that can parse compound names into their atomic components **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered

### getNameParser

public [NameParser](http://docs.google.com/javax/naming/NameParser.html) **getNameParser**([Name](http://docs.google.com/javax/naming/Name.html) name)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#getNameParser(javax.naming.Name)) Retrieves the parser associated with the named context. In a federation of namespaces, different naming systems will parse names differently. This method allows an application to get a parser for parsing names into their atomic components using the naming convention of a particular naming system. Within any single naming system, NameParser objects returned by this method must be equal (using the equals() test).

**Specified by:**[getNameParser](http://docs.google.com/javax/naming/Context.html#getNameParser(javax.naming.Name)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - the name of the context from which to get the parser **Returns:**a name parser that can parse compound names into their atomic components **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered**See Also:**[Context.getNameParser(String)](http://docs.google.com/javax/naming/Context.html#getNameParser(java.lang.String)), [CompoundName](http://docs.google.com/javax/naming/CompoundName.html)

### composeName

public [String](http://docs.google.com/java/lang/String.html) **composeName**([String](http://docs.google.com/java/lang/String.html) name,  
 [String](http://docs.google.com/java/lang/String.html) prefix)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

Composes the name of this context with a name relative to this context. Since an initial context may never be named relative to any context other than itself, the value of the prefix parameter must be an empty name ("").

**Specified by:**[composeName](http://docs.google.com/javax/naming/Context.html#composeName(java.lang.String,%20java.lang.String)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - a name relative to this contextprefix - the name of this context relative to one of its ancestors **Returns:**the composition of prefix and name **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered

### composeName

public [Name](http://docs.google.com/javax/naming/Name.html) **composeName**([Name](http://docs.google.com/javax/naming/Name.html) name,  
 [Name](http://docs.google.com/javax/naming/Name.html) prefix)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

Composes the name of this context with a name relative to this context. Since an initial context may never be named relative to any context other than itself, the value of the prefix parameter must be an empty name.

**Specified by:**[composeName](http://docs.google.com/javax/naming/Context.html#composeName(javax.naming.Name,%20javax.naming.Name)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**name - a name relative to this contextprefix - the name of this context relative to one of its ancestors **Returns:**the composition of prefix and name **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered**See Also:**[Context.composeName(String, String)](http://docs.google.com/javax/naming/Context.html#composeName(java.lang.String,%20java.lang.String))

### addToEnvironment

public [Object](http://docs.google.com/java/lang/Object.html) **addToEnvironment**([String](http://docs.google.com/java/lang/String.html) propName,  
 [Object](http://docs.google.com/java/lang/Object.html) propVal)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#addToEnvironment(java.lang.String,%20java.lang.Object)) Adds a new environment property to the environment of this context. If the property already exists, its value is overwritten. See class description for more details on environment properties.

**Specified by:**[addToEnvironment](http://docs.google.com/javax/naming/Context.html#addToEnvironment(java.lang.String,%20java.lang.Object)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**propName - the name of the environment property to add; may not be nullpropVal - the value of the property to add; may not be null **Returns:**the previous value of the property, or null if the property was not in the environment before **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered**See Also:**[Context.getEnvironment()](http://docs.google.com/javax/naming/Context.html#getEnvironment()), [Context.removeFromEnvironment(String)](http://docs.google.com/javax/naming/Context.html#removeFromEnvironment(java.lang.String))

### removeFromEnvironment

public [Object](http://docs.google.com/java/lang/Object.html) **removeFromEnvironment**([String](http://docs.google.com/java/lang/String.html) propName)  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#removeFromEnvironment(java.lang.String)) Removes an environment property from the environment of this context. See class description for more details on environment properties.

**Specified by:**[removeFromEnvironment](http://docs.google.com/javax/naming/Context.html#removeFromEnvironment(java.lang.String)) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Parameters:**propName - the name of the environment property to remove; may not be null **Returns:**the previous value of the property, or null if the property was not in the environment **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered**See Also:**[Context.getEnvironment()](http://docs.google.com/javax/naming/Context.html#getEnvironment()), [Context.addToEnvironment(String, Object)](http://docs.google.com/javax/naming/Context.html#addToEnvironment(java.lang.String,%20java.lang.Object))

### getEnvironment

public [Hashtable](http://docs.google.com/java/util/Hashtable.html)<?,?> **getEnvironment**()  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#getEnvironment()) Retrieves the environment in effect for this context. See class description for more details on environment properties.

The caller should not make any changes to the object returned: their effect on the context is undefined. The environment of this context may be changed using addToEnvironment() and removeFromEnvironment().

**Specified by:**[getEnvironment](http://docs.google.com/javax/naming/Context.html#getEnvironment()) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Returns:**the environment of this context; never null **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered**See Also:**[Context.addToEnvironment(String, Object)](http://docs.google.com/javax/naming/Context.html#addToEnvironment(java.lang.String,%20java.lang.Object)), [Context.removeFromEnvironment(String)](http://docs.google.com/javax/naming/Context.html#removeFromEnvironment(java.lang.String))

### close

public void **close**()  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#close()) Closes this context. This method releases this context's resources immediately, instead of waiting for them to be released automatically by the garbage collector.

This method is idempotent: invoking it on a context that has already been closed has no effect. Invoking any other method on a closed context is not allowed, and results in undefined behaviour.

**Specified by:**[close](http://docs.google.com/javax/naming/Context.html#close()) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Throws:** [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered

### getNameInNamespace

public [String](http://docs.google.com/java/lang/String.html) **getNameInNamespace**()  
 throws [NamingException](http://docs.google.com/javax/naming/NamingException.html)

**Description copied from interface:** [**Context**](http://docs.google.com/javax/naming/Context.html#getNameInNamespace()) Retrieves the full name of this context within its own namespace.

Many naming services have a notion of a "full name" for objects in their respective namespaces. For example, an LDAP entry has a distinguished name, and a DNS record has a fully qualified name. This method allows the client application to retrieve this name. The string returned by this method is not a JNDI composite name and should not be passed directly to context methods. In naming systems for which the notion of full name does not make sense, OperationNotSupportedException is thrown.

**Specified by:**[getNameInNamespace](http://docs.google.com/javax/naming/Context.html#getNameInNamespace()) in interface [Context](http://docs.google.com/javax/naming/Context.html) **Returns:**this context's name in its own namespace; never null **Throws:** [OperationNotSupportedException](http://docs.google.com/javax/naming/OperationNotSupportedException.html) - if the naming system does not have the notion of a full name [NamingException](http://docs.google.com/javax/naming/NamingException.html) - if a naming exception is encountered

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/InitialContext.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/naming/ContextNotEmptyException.html)   [**NEXT CLASS**](http://docs.google.com/javax/naming/InsufficientResourcesException.html) | [**FRAMES**](http://docs.google.com/index.html?javax/naming/InitialContext.html)    [**NO FRAMES**](http://docs.google.com/InitialContext.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#2et92p0) | [CONSTR](#3dy6vkm) | [METHOD](#1t3h5sf) | DETAIL: [FIELD](#2s8eyo1) | [CONSTR](#lnxbz9) | [METHOD](#2jxsxqh) |

[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).