| | [**Overview**](http://docs.google.com/overview-summary.html) | **Package** | Class | [**Use**](http://docs.google.com/package-use.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV PACKAGE**](http://docs.google.com/javax/naming/directory/package-summary.html)   [**NEXT PACKAGE**](http://docs.google.com/javax/naming/ldap/package-summary.html) | [**FRAMES**](http://docs.google.com/index.html?javax/naming/event/package-summary.html)    [**NO FRAMES**](http://docs.google.com/package-summary.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |

## Package javax.naming.event

Provides support for event notification when accessing naming and directory services.

**See:**

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

| **Interface Summary** | |
| --- | --- |
| [**EventContext**](http://docs.google.com/javax/naming/event/EventContext.html) | Contains methods for registering/deregistering listeners to be notified of events fired when objects named in a context changes. |
| [**EventDirContext**](http://docs.google.com/javax/naming/event/EventDirContext.html) | Contains methods for registering listeners to be notified of events fired when objects named in a directory context changes. |
| [**NamespaceChangeListener**](http://docs.google.com/javax/naming/event/NamespaceChangeListener.html) | Specifies the methods that a listener interested in namespace changes must implement. |
| [**NamingListener**](http://docs.google.com/javax/naming/event/NamingListener.html) | This interface is the root of listener interfaces that handle NamingEvents. |
| [**ObjectChangeListener**](http://docs.google.com/javax/naming/event/ObjectChangeListener.html) | Specifies the method that a listener of a NamingEvent with event type of OBJECT\_CHANGED must implement. |

| **Class Summary** | |
| --- | --- |
| [**NamingEvent**](http://docs.google.com/javax/naming/event/NamingEvent.html) | This class represents an event fired by a naming/directory service. |
| [**NamingExceptionEvent**](http://docs.google.com/javax/naming/event/NamingExceptionEvent.html) | This class represents an event fired when the procedures/processes used to collect information for notifying listeners of NamingEvents threw a NamingException. |

## Package javax.naming.event Description

Provides support for event notification when accessing naming and directory services.

This package defines the event notification operations of the Java Naming and Directory InterfaceTM (JNDI).   JNDI provides naming and directory functionality to applications written in the Java programming language. It is designed to be independent of any specific naming or directory service implementation. Thus a variety of services--new, emerging, and already deployed ones--can be accessed in a common way.

#### Naming Events

This package defines a NamingEvent class to represent an event that is generated by a naming/directory service. It also defines subinterfaces of Context and DirContext, called EventContext and EventDirContext, through which applications can register their interest in events fired by the context.

NamingEvent represents an event that occurs in a naming or directory service. There are two categories of naming events:

* Those that affect the namespace (add/remove/rename an object)
* Those that affect the objects' contents.

Each category of events is handled by a corresponding listener: NamespaceChangeListener, ObjectChangeListener.

An application, for example, can register its interest in changes to objects in a context as follows:

EventContext src =   
 (EventContext)(new InitialContext()).lookup("o=wiz,c=us");  
src.addNamingListener("ou=users", EventContext.ONELEVEL\_SCOPE,  
 new ChangeHandler());  
...  
class ChangeHandler implements ObjectChangeListener {  
 public void objectChanged(NamingEvent evt) {  
 System.out.println(evt.getNewBinding());  
 }  
 public void namingExceptionThrown(NamingExceptionEvent evt) {  
 System.out.println(evt.getException());  
 }  
}

#### Threading Issues

When an event is dispatched to a listener, the listener method (such as objectChanged()) may be executed in a thread other than the one in which the call to addNamingListener() was executed. The choice of which thread to use is made by the service provider. When an event is dispatched to multiple listeners, the service provider may choose (and is generally encouraged) to execute the listener methods concurrently in separate threads.

When a listener instance invokes NamingEvent.getEventContext(), it must take into account the possibility that other threads will be working with that context concurrently. Likewise, when a listener is registered via addNamingListener(), the registering thread must take into account the likely possibility that the service provider will later invoke the listeners in newly-created threads. As Context instances are not guaranteed to be thread-safe in general, all context operations must be synchronized as needed.

#### Exception Handling

When a listener registers for events with a context, the context might need to do some internal processing in order to collect information required to generate the events. The context, for example, might need to make a request to the server to register interest in changes on the server that will eventually be translated into events. If an exception occurs that prevents information about the events from being collected, the listener will never be notified of the events. When such an exception occurs, a NamingExceptionEvent is fired to notify the listener. The listener's namingExceptionThrown() method is invoked, as shown in the sample code above, and the listener is automatically deregistered.

## Package Specification

The JNDI API Specification and related documents can be found in the [JNDI documentation](http://docs.google.com/technotes/guides/jndi/index.html).

**Since:** 1.3

| | [**Overview**](http://docs.google.com/overview-summary.html) | **Package** | Class | [**Use**](http://docs.google.com/package-use.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV PACKAGE**](http://docs.google.com/javax/naming/directory/package-summary.html)   [**NEXT PACKAGE**](http://docs.google.com/javax/naming/ldap/package-summary.html) | [**FRAMES**](http://docs.google.com/index.html?javax/naming/event/package-summary.html)    [**NO FRAMES**](http://docs.google.com/package-summary.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |

[Submit a bug or feature](http://bugs.sun.com/services/bugreport/index.jsp)

For further API reference and developer documentation, see [Java SE Developer Documentation](http://docs.google.com/webnotes/devdocs-vs-specs.html). That documentation contains more detailed, developer-targeted descriptions, with conceptual overviews, definitions of terms, workarounds, and working code examples.

Copyright 2006 Sun Microsystems, Inc. All rights reserved. Use is subject to [license terms](http://docs.google.com/legal/license.html). Also see the [documentation redistribution policy](http://java.sun.com/docs/redist.html).