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

## **java.security.acl**

Interface Acl

**All Superinterfaces:** [Owner](http://docs.google.com/java/security/acl/Owner.html)

public interface **Acl**extends [Owner](http://docs.google.com/java/security/acl/Owner.html)

Interface representing an Access Control List (ACL). An Access Control List is a data structure used to guard access to resources.

An ACL can be thought of as a data structure with multiple ACL entries. Each ACL entry, of interface type AclEntry, contains a set of permissions associated with a particular principal. (A principal represents an entity such as an individual user or a group). Additionally, each ACL entry is specified as being either positive or negative. If positive, the permissions are to be granted to the associated principal. If negative, the permissions are to be denied.

The ACL Entries in each ACL observe the following rules:

* Each principal can have at most one positive ACL entry and one negative entry; that is, multiple positive or negative ACL entries are not allowed for any principal. Each entry specifies the set of permissions that are to be granted (if positive) or denied (if negative).
* If there is no entry for a particular principal, then the principal is considered to have a null (empty) permission set.
* If there is a positive entry that grants a principal a particular permission, and a negative entry that denies the principal the same permission, the result is as though the permission was never granted or denied.
* Individual permissions always override permissions of the group(s) to which the individual belongs. That is, individual negative permissions (specific denial of permissions) override the groups' positive permissions. And individual positive permissions override the groups' negative permissions.

The java.security.acl package provides the interfaces to the ACL and related data structures (ACL entries, groups, permissions, etc.), and the sun.security.acl classes provide a default implementation of the interfaces. For example, java.security.acl.Acl provides the interface to an ACL and the sun.security.acl.AclImpl class provides the default implementation of the interface.

The java.security.acl.Acl interface extends the java.security.acl.Owner interface. The Owner interface is used to maintain a list of owners for each ACL. Only owners are allowed to modify an ACL. For example, only an owner can call the ACL's addEntry method to add a new ACL entry to the ACL.

**See Also:**[AclEntry](http://docs.google.com/java/security/acl/AclEntry.html), [Owner](http://docs.google.com/java/security/acl/Owner.html), [getPermissions(java.security.Principal)](http://docs.google.com/java/security/acl/Acl.html#getPermissions(java.security.Principal))

| **Method Summary** | |
| --- | --- |
| boolean | [**addEntry**](http://docs.google.com/java/security/acl/Acl.html#addEntry(java.security.Principal,%20java.security.acl.AclEntry))([Principal](http://docs.google.com/java/security/Principal.html) caller, [AclEntry](http://docs.google.com/java/security/acl/AclEntry.html) entry)            Adds an ACL entry to this ACL. |
| boolean | [**checkPermission**](http://docs.google.com/java/security/acl/Acl.html#checkPermission(java.security.Principal,%20java.security.acl.Permission))([Principal](http://docs.google.com/java/security/Principal.html) principal, [Permission](http://docs.google.com/java/security/acl/Permission.html) permission)            Checks whether or not the specified principal has the specified permission. |
| [Enumeration](http://docs.google.com/java/util/Enumeration.html)<[AclEntry](http://docs.google.com/java/security/acl/AclEntry.html)> | [**entries**](http://docs.google.com/java/security/acl/Acl.html#entries())()            Returns an enumeration of the entries in this ACL. |
| [String](http://docs.google.com/java/lang/String.html) | [**getName**](http://docs.google.com/java/security/acl/Acl.html#getName())()            Returns the name of this ACL. |
| [Enumeration](http://docs.google.com/java/util/Enumeration.html)<[Permission](http://docs.google.com/java/security/acl/Permission.html)> | [**getPermissions**](http://docs.google.com/java/security/acl/Acl.html#getPermissions(java.security.Principal))([Principal](http://docs.google.com/java/security/Principal.html) user)            Returns an enumeration for the set of allowed permissions for the specified principal (representing an entity such as an individual or a group). |
| boolean | [**removeEntry**](http://docs.google.com/java/security/acl/Acl.html#removeEntry(java.security.Principal,%20java.security.acl.AclEntry))([Principal](http://docs.google.com/java/security/Principal.html) caller, [AclEntry](http://docs.google.com/java/security/acl/AclEntry.html) entry)            Removes an ACL entry from this ACL. |
| void | [**setName**](http://docs.google.com/java/security/acl/Acl.html#setName(java.security.Principal,%20java.lang.String))([Principal](http://docs.google.com/java/security/Principal.html) caller, [String](http://docs.google.com/java/lang/String.html) name)            Sets the name of this ACL. |
| [String](http://docs.google.com/java/lang/String.html) | [**toString**](http://docs.google.com/java/security/acl/Acl.html#toString())()            Returns a string representation of the ACL contents. |

| **Methods inherited from interface java.security.acl.**[**Owner**](http://docs.google.com/java/security/acl/Owner.html) |
| --- |
| [addOwner](http://docs.google.com/java/security/acl/Owner.html#addOwner(java.security.Principal,%20java.security.Principal)), [deleteOwner](http://docs.google.com/java/security/acl/Owner.html#deleteOwner(java.security.Principal,%20java.security.Principal)), [isOwner](http://docs.google.com/java/security/acl/Owner.html#isOwner(java.security.Principal)) |

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

### setName

void **setName**([Principal](http://docs.google.com/java/security/Principal.html) caller,  
 [String](http://docs.google.com/java/lang/String.html) name)  
 throws [NotOwnerException](http://docs.google.com/java/security/acl/NotOwnerException.html)

Sets the name of this ACL.

**Parameters:**caller - the principal invoking this method. It must be an owner of this ACL.name - the name to be given to this ACL. **Throws:** [NotOwnerException](http://docs.google.com/java/security/acl/NotOwnerException.html) - if the caller principal is not an owner of this ACL.**See Also:**[getName()](http://docs.google.com/java/security/acl/Acl.html#getName())

### getName

[String](http://docs.google.com/java/lang/String.html) **getName**()

Returns the name of this ACL.

**Returns:**the name of this ACL.**See Also:**[setName(java.security.Principal, java.lang.String)](http://docs.google.com/java/security/acl/Acl.html#setName(java.security.Principal,%20java.lang.String))

### addEntry

boolean **addEntry**([Principal](http://docs.google.com/java/security/Principal.html) caller,  
 [AclEntry](http://docs.google.com/java/security/acl/AclEntry.html) entry)  
 throws [NotOwnerException](http://docs.google.com/java/security/acl/NotOwnerException.html)

Adds an ACL entry to this ACL. An entry associates a principal (e.g., an individual or a group) with a set of permissions. Each principal can have at most one positive ACL entry (specifying permissions to be granted to the principal) and one negative ACL entry (specifying permissions to be denied). If there is already an ACL entry of the same type (negative or positive) already in the ACL, false is returned.

**Parameters:**caller - the principal invoking this method. It must be an owner of this ACL.entry - the ACL entry to be added to this ACL. **Returns:**true on success, false if an entry of the same type (positive or negative) for the same principal is already present in this ACL. **Throws:** [NotOwnerException](http://docs.google.com/java/security/acl/NotOwnerException.html) - if the caller principal is not an owner of this ACL.

### removeEntry

boolean **removeEntry**([Principal](http://docs.google.com/java/security/Principal.html) caller,  
 [AclEntry](http://docs.google.com/java/security/acl/AclEntry.html) entry)  
 throws [NotOwnerException](http://docs.google.com/java/security/acl/NotOwnerException.html)

Removes an ACL entry from this ACL.

**Parameters:**caller - the principal invoking this method. It must be an owner of this ACL.entry - the ACL entry to be removed from this ACL. **Returns:**true on success, false if the entry is not part of this ACL. **Throws:** [NotOwnerException](http://docs.google.com/java/security/acl/NotOwnerException.html) - if the caller principal is not an owner of this Acl.

### getPermissions

[Enumeration](http://docs.google.com/java/util/Enumeration.html)<[Permission](http://docs.google.com/java/security/acl/Permission.html)> **getPermissions**([Principal](http://docs.google.com/java/security/Principal.html) user)

Returns an enumeration for the set of allowed permissions for the specified principal (representing an entity such as an individual or a group). This set of allowed permissions is calculated as follows:

* If there is no entry in this Access Control List for the specified principal, an empty permission set is returned.
* Otherwise, the principal's group permission sets are determined. (A principal can belong to one or more groups, where a group is a group of principals, represented by the Group interface.) The group positive permission set is the union of all the positive permissions of each group that the principal belongs to. The group negative permission set is the union of all the negative permissions of each group that the principal belongs to. If there is a specific permission that occurs in both the positive permission set and the negative permission set, it is removed from both.  
  The individual positive and negative permission sets are also determined. The positive permission set contains the permissions specified in the positive ACL entry (if any) for the principal. Similarly, the negative permission set contains the permissions specified in the negative ACL entry (if any) for the principal. The individual positive (or negative) permission set is considered to be null if there is not a positive (negative) ACL entry for the principal in this ACL.  
  The set of permissions granted to the principal is then calculated using the simple rule that individual permissions always override the group permissions. That is, the principal's individual negative permission set (specific denial of permissions) overrides the group positive permission set, and the principal's individual positive permission set overrides the group negative permission set.

**Parameters:**user - the principal whose permission set is to be returned. **Returns:**the permission set specifying the permissions the principal is allowed.

### entries

[Enumeration](http://docs.google.com/java/util/Enumeration.html)<[AclEntry](http://docs.google.com/java/security/acl/AclEntry.html)> **entries**()

Returns an enumeration of the entries in this ACL. Each element in the enumeration is of type AclEntry.

**Returns:**an enumeration of the entries in this ACL.

### checkPermission

boolean **checkPermission**([Principal](http://docs.google.com/java/security/Principal.html) principal,  
 [Permission](http://docs.google.com/java/security/acl/Permission.html) permission)

Checks whether or not the specified principal has the specified permission. If it does, true is returned, otherwise false is returned. More specifically, this method checks whether the passed permission is a member of the allowed permission set of the specified principal. The allowed permission set is determined by the same algorithm as is used by the getPermissions method.

**Parameters:**principal - the principal, assumed to be a valid authenticated Principal.permission - the permission to be checked for. **Returns:**true if the principal has the specified permission, false otherwise.**See Also:**[getPermissions(java.security.Principal)](http://docs.google.com/java/security/acl/Acl.html#getPermissions(java.security.Principal))

### toString

[String](http://docs.google.com/java/lang/String.html) **toString**()

Returns a string representation of the ACL contents.

**Overrides:**[toString](http://docs.google.com/java/lang/Object.html#toString()) in class [Object](http://docs.google.com/java/lang/Object.html) **Returns:**a string representation of the ACL contents.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Acl.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   [**NEXT CLASS**](http://docs.google.com/java/security/acl/AclEntry.html) | [**FRAMES**](http://docs.google.com/index.html?java/security/acl/Acl.html)    [**NO FRAMES**](http://docs.google.com/Acl.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).