| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/SocketPermission.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/java/net/SocketOptions.html)   [**NEXT CLASS**](http://docs.google.com/java/net/SocketTimeoutException.html) | [**FRAMES**](http://docs.google.com/index.html?java/net/SocketPermission.html)    [**NO FRAMES**](http://docs.google.com/SocketPermission.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#1t3h5sf) | [METHOD](#2s8eyo1) |

## **java.net**

Class SocketPermission

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 [java.security.Permission](http://docs.google.com/java/security/Permission.html)  
 **java.net.SocketPermission**

**All Implemented Interfaces:** [Serializable](http://docs.google.com/java/io/Serializable.html), [Guard](http://docs.google.com/java/security/Guard.html)

public final class **SocketPermission**extends [Permission](http://docs.google.com/java/security/Permission.html)implements [Serializable](http://docs.google.com/java/io/Serializable.html)

This class represents access to a network via sockets. A SocketPermission consists of a host specification and a set of "actions" specifying ways to connect to that host. The host is specified as

host = (hostname | IPv4address | iPv6reference) [:portrange]  
 portrange = portnumber | -portnumber | portnumber-[portnumber]

The host is expressed as a DNS name, as a numerical IP address, or as "localhost" (for the local machine). The wildcard "\*" may be included once in a DNS name host specification. If it is included, it must be in the leftmost position, as in "\*.sun.com".

The format of the IPv6reference should follow that specified in [*RFC 2732: Format for Literal IPv6 Addresses in URLs*](http://www.ietf.org/rfc/rfc2732.txt):

ipv6reference = "[" IPv6address "]"

For example, you can construct a SocketPermission instance as the following:

String hostAddress = inetaddress.getHostAddress();  
 if (inetaddress instanceof Inet6Address) {  
 sp = new SocketPermission("[" + hostAddress + "]:" + port, action);  
 } else {  
 sp = new SocketPermission(hostAddress + ":" + port, action);  
 }

or

String host = url.getHost();  
 sp = new SocketPermission(host + ":" + port, action);

The [full uncompressed form](http://docs.google.com/Inet6Address.html#lform) of an IPv6 literal address is also valid.

The port or portrange is optional. A port specification of the form "N-", where *N* is a port number, signifies all ports numbered *N* and above, while a specification of the form "-N" indicates all ports numbered *N* and below.

The possible ways to connect to the host are

accept  
 connect  
 listen  
 resolve

The "listen" action is only meaningful when used with "localhost". The "resolve" action is implied when any of the other actions are present. The action "resolve" refers to host/ip name service lookups.

As an example of the creation and meaning of SocketPermissions, note that if the following permission:

p1 = new SocketPermission("puffin.eng.sun.com:7777", "connect,accept");

is granted to some code, it allows that code to connect to port 7777 on puffin.eng.sun.com, and to accept connections on that port.

Similarly, if the following permission:

p1 = new SocketPermission("puffin.eng.sun.com:7777", "connect,accept");  
 p2 = new SocketPermission("localhost:1024-", "accept,connect,listen");

is granted to some code, it allows that code to accept connections on, connect to, or listen on any port between 1024 and 65535 on the local host.

Note: Granting code permission to accept or make connections to remote hosts may be dangerous because malevolent code can then more easily transfer and share confidential data among parties who may not otherwise have access to the data.

**See Also:**[Permissions](http://docs.google.com/java/security/Permissions.html), [SocketPermission](http://docs.google.com/java/net/SocketPermission.html)

| **Constructor Summary** | |
| --- | --- |
| [**SocketPermission**](http://docs.google.com/java/net/SocketPermission.html#SocketPermission(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) host, [String](http://docs.google.com/java/lang/String.html) action)            Creates a new SocketPermission object with the specified actions. |

| **Method Summary** | |
| --- | --- |
| boolean | [**equals**](http://docs.google.com/java/net/SocketPermission.html#equals(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) obj)            Checks two SocketPermission objects for equality. |
| [String](http://docs.google.com/java/lang/String.html) | [**getActions**](http://docs.google.com/java/net/SocketPermission.html#getActions())()            Returns the canonical string representation of the actions. |
| int | [**hashCode**](http://docs.google.com/java/net/SocketPermission.html#hashCode())()            Returns the hash code value for this object. |
| boolean | [**implies**](http://docs.google.com/java/net/SocketPermission.html#implies(java.security.Permission))([Permission](http://docs.google.com/java/security/Permission.html) p)            Checks if this socket permission object "implies" the specified permission. |
| [PermissionCollection](http://docs.google.com/java/security/PermissionCollection.html) | [**newPermissionCollection**](http://docs.google.com/java/net/SocketPermission.html#newPermissionCollection())()            Returns a new PermissionCollection object for storing SocketPermission objects. |

| **Methods inherited from class java.security.**[**Permission**](http://docs.google.com/java/security/Permission.html) |
| --- |
| [checkGuard](http://docs.google.com/java/security/Permission.html#checkGuard(java.lang.Object)), [getName](http://docs.google.com/java/security/Permission.html#getName()), [toString](http://docs.google.com/java/security/Permission.html#toString()) |

| **Methods inherited from class java.lang.**[**Object**](http://docs.google.com/java/lang/Object.html) |
| --- |
| [clone](http://docs.google.com/java/lang/Object.html#clone()), [finalize](http://docs.google.com/java/lang/Object.html#finalize()), [getClass](http://docs.google.com/java/lang/Object.html#getClass()), [notify](http://docs.google.com/java/lang/Object.html#notify()), [notifyAll](http://docs.google.com/java/lang/Object.html#notifyAll()), [wait](http://docs.google.com/java/lang/Object.html#wait()), [wait](http://docs.google.com/java/lang/Object.html#wait(long)), [wait](http://docs.google.com/java/lang/Object.html#wait(long,%20int)) |

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

### SocketPermission

public **SocketPermission**([String](http://docs.google.com/java/lang/String.html) host,  
 [String](http://docs.google.com/java/lang/String.html) action)

Creates a new SocketPermission object with the specified actions. The host is expressed as a DNS name, or as a numerical IP address. Optionally, a port or a portrange may be supplied (separated from the DNS name or IP address by a colon).

To specify the local machine, use "localhost" as the *host*. Also note: An empty *host* String ("") is equivalent to "localhost".

The *actions* parameter contains a comma-separated list of the actions granted for the specified host (and port(s)). Possible actions are "connect", "listen", "accept", "resolve", or any combination of those. "resolve" is automatically added when any of the other three are specified.

Examples of SocketPermission instantiation are the following:

nr = new SocketPermission("www.catalog.com", "connect");  
 nr = new SocketPermission("www.sun.com:80", "connect");  
 nr = new SocketPermission("\*.sun.com", "connect");  
 nr = new SocketPermission("\*.edu", "resolve");  
 nr = new SocketPermission("204.160.241.0", "connect");  
 nr = new SocketPermission("localhost:1024-65535", "listen");  
 nr = new SocketPermission("204.160.241.0:1024-65535", "connect");

**Parameters:**host - the hostname or IPaddress of the computer, optionally including a colon followed by a port or port range.action - the action string.

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

### implies

public boolean **implies**([Permission](http://docs.google.com/java/security/Permission.html) p)

Checks if this socket permission object "implies" the specified permission.

More specifically, this method first ensures that all of the following are true (and returns false if any of them are not):

* *p* is an instanceof SocketPermission,
* *p*'s actions are a proper subset of this object's actions, and
* *p*'s port range is included in this port range. Note: port range is ignored when p only contains the action, 'resolve'.

Then implies checks each of the following, in order, and for each returns true if the stated condition is true:

* If this object was initialized with a single IP address and one of *p*'s IP addresses is equal to this object's IP address.
* If this object is a wildcard domain (such as \*.sun.com), and *p*'s canonical name (the name without any preceding \*) ends with this object's canonical host name. For example, \*.sun.com implies \*.eng.sun.com..
* If this object was not initialized with a single IP address, and one of this object's IP addresses equals one of *p*'s IP addresses.
* If this canonical name equals *p*'s canonical name.

If none of the above are true, implies returns false.

**Specified by:**[implies](http://docs.google.com/java/security/Permission.html#implies(java.security.Permission)) in class [Permission](http://docs.google.com/java/security/Permission.html) **Parameters:**p - the permission to check against. **Returns:**true if the specified permission is implied by this object, false if not.

### equals

public boolean **equals**([Object](http://docs.google.com/java/lang/Object.html) obj)

Checks two SocketPermission objects for equality.

**Specified by:**[equals](http://docs.google.com/java/security/Permission.html#equals(java.lang.Object)) in class [Permission](http://docs.google.com/java/security/Permission.html) **Parameters:**obj - the object to test for equality with this object. **Returns:**true if *obj* is a SocketPermission, and has the same hostname, port range, and actions as this SocketPermission object. However, port range will be ignored in the comparison if *obj* only contains the action, 'resolve'.**See Also:**[Object.hashCode()](http://docs.google.com/java/lang/Object.html#hashCode()), [Hashtable](http://docs.google.com/java/util/Hashtable.html)

### hashCode

public int **hashCode**()

Returns the hash code value for this object.

**Specified by:**[hashCode](http://docs.google.com/java/security/Permission.html#hashCode()) in class [Permission](http://docs.google.com/java/security/Permission.html) **Returns:**a hash code value for this object.**See Also:**[Object.equals(java.lang.Object)](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [Hashtable](http://docs.google.com/java/util/Hashtable.html)

### getActions

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

Returns the canonical string representation of the actions. Always returns present actions in the following order: connect, listen, accept, resolve.

**Specified by:**[getActions](http://docs.google.com/java/security/Permission.html#getActions()) in class [Permission](http://docs.google.com/java/security/Permission.html) **Returns:**the canonical string representation of the actions.

### newPermissionCollection

public [PermissionCollection](http://docs.google.com/java/security/PermissionCollection.html) **newPermissionCollection**()

Returns a new PermissionCollection object for storing SocketPermission objects.

SocketPermission objects must be stored in a manner that allows them to be inserted into the collection in any order, but that also enables the PermissionCollection implies method to be implemented in an efficient (and consistent) manner.

**Overrides:**[newPermissionCollection](http://docs.google.com/java/security/Permission.html#newPermissionCollection()) in class [Permission](http://docs.google.com/java/security/Permission.html) **Returns:**a new PermissionCollection object suitable for storing SocketPermissions.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/SocketPermission.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/java/net/SocketOptions.html)   [**NEXT CLASS**](http://docs.google.com/java/net/SocketTimeoutException.html) | [**FRAMES**](http://docs.google.com/index.html?java/net/SocketPermission.html)    [**NO FRAMES**](http://docs.google.com/SocketPermission.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#1t3h5sf) | [METHOD](#2s8eyo1) |

[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).