| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/SSLSession.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/net/ssl/SSLServerSocketFactory.html)   [**NEXT CLASS**](http://docs.google.com/javax/net/ssl/SSLSessionBindingEvent.html) | [**FRAMES**](http://docs.google.com/index.html?javax/net/ssl/SSLSession.html)    [**NO FRAMES**](http://docs.google.com/SSLSession.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#3znysh7) | DETAIL: FIELD | CONSTR | [METHOD](#2et92p0) |

## **javax.net.ssl**

Interface SSLSession

public interface **SSLSession**

In SSL, sessions are used to describe an ongoing relationship between two entities. Each SSL connection involves one session at a time, but that session may be used on many connections between those entities, simultaneously or sequentially. The session used on a connection may also be replaced by a different session. Sessions are created, or rejoined, as part of the SSL handshaking protocol. Sessions may be invalidated due to policies affecting security or resource usage, or by an application explicitly calling invalidate. Session management policies are typically used to tune performance.

In addition to the standard session attributes, SSL sessions expose these read-only attributes:

* *Peer Identity.* Sessions are between a particular client and a particular server. The identity of the peer may have been established as part of session setup. Peers are generally identified by X.509 certificate chains.
* *Cipher Suite Name.* Cipher suites describe the kind of cryptographic protection that's used by connections in a particular session.
* *Peer Host.* All connections in a session are between the same two hosts. The address of the host on the other side of the connection is available.

Sessions may be explicitly invalidated. Invalidation may also be done implicitly, when faced with certain kinds of errors.

**Since:** 1.4

| **Method Summary** | |
| --- | --- |
| int | [**getApplicationBufferSize**](http://docs.google.com/javax/net/ssl/SSLSession.html#getApplicationBufferSize())()            Gets the current size of the largest application data that is expected when using this session. |
| [String](http://docs.google.com/java/lang/String.html) | [**getCipherSuite**](http://docs.google.com/javax/net/ssl/SSLSession.html#getCipherSuite())()            Returns the name of the SSL cipher suite which is used for all connections in the session. |
| long | [**getCreationTime**](http://docs.google.com/javax/net/ssl/SSLSession.html#getCreationTime())()            Returns the time at which this Session representation was created, in milliseconds since midnight, January 1, 1970 UTC. |
| byte[] | [**getId**](http://docs.google.com/javax/net/ssl/SSLSession.html#getId())()            Returns the identifier assigned to this Session. |
| long | [**getLastAccessedTime**](http://docs.google.com/javax/net/ssl/SSLSession.html#getLastAccessedTime())()            Returns the last time this Session representation was accessed by the session level infrastructure, in milliseconds since midnight, January 1, 1970 UTC. |
| [Certificate](http://docs.google.com/java/security/cert/Certificate.html)[] | [**getLocalCertificates**](http://docs.google.com/javax/net/ssl/SSLSession.html#getLocalCertificates())()            Returns the certificate(s) that were sent to the peer during handshaking. |
| [Principal](http://docs.google.com/java/security/Principal.html) | [**getLocalPrincipal**](http://docs.google.com/javax/net/ssl/SSLSession.html#getLocalPrincipal())()            Returns the principal that was sent to the peer during handshaking. |
| int | [**getPacketBufferSize**](http://docs.google.com/javax/net/ssl/SSLSession.html#getPacketBufferSize())()            Gets the current size of the largest SSL/TLS packet that is expected when using this session. |
| [X509Certificate](http://docs.google.com/javax/security/cert/X509Certificate.html)[] | [**getPeerCertificateChain**](http://docs.google.com/javax/net/ssl/SSLSession.html#getPeerCertificateChain())()            Returns the identity of the peer which was identified as part of defining the session. |
| [Certificate](http://docs.google.com/java/security/cert/Certificate.html)[] | [**getPeerCertificates**](http://docs.google.com/javax/net/ssl/SSLSession.html#getPeerCertificates())()            Returns the identity of the peer which was established as part of defining the session. |
| [String](http://docs.google.com/java/lang/String.html) | [**getPeerHost**](http://docs.google.com/javax/net/ssl/SSLSession.html#getPeerHost())()            Returns the host name of the peer in this session. |
| int | [**getPeerPort**](http://docs.google.com/javax/net/ssl/SSLSession.html#getPeerPort())()            Returns the port number of the peer in this session. |
| [Principal](http://docs.google.com/java/security/Principal.html) | [**getPeerPrincipal**](http://docs.google.com/javax/net/ssl/SSLSession.html#getPeerPrincipal())()            Returns the identity of the peer which was established as part of defining the session. |
| [String](http://docs.google.com/java/lang/String.html) | [**getProtocol**](http://docs.google.com/javax/net/ssl/SSLSession.html#getProtocol())()            Returns the standard name of the protocol used for all connections in the session. |
| [SSLSessionContext](http://docs.google.com/javax/net/ssl/SSLSessionContext.html) | [**getSessionContext**](http://docs.google.com/javax/net/ssl/SSLSession.html#getSessionContext())()            Returns the context in which this session is bound. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getValue**](http://docs.google.com/javax/net/ssl/SSLSession.html#getValue(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)            Returns the object bound to the given name in the session's application layer data. |
| [String](http://docs.google.com/java/lang/String.html)[] | [**getValueNames**](http://docs.google.com/javax/net/ssl/SSLSession.html#getValueNames())()            Returns an array of the names of all the application layer data objects bound into the Session. |
| void | [**invalidate**](http://docs.google.com/javax/net/ssl/SSLSession.html#invalidate())()            Invalidates the session. |
| boolean | [**isValid**](http://docs.google.com/javax/net/ssl/SSLSession.html#isValid())()            Returns whether this session is valid and available for resuming or joining. |
| void | [**putValue**](http://docs.google.com/javax/net/ssl/SSLSession.html#putValue(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) value)            Binds the specified value object into the session's application layer data with the given name. |
| void | [**removeValue**](http://docs.google.com/javax/net/ssl/SSLSession.html#removeValue(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)            Removes the object bound to the given name in the session's application layer data. |

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

### getId

byte[] **getId**()

Returns the identifier assigned to this Session.

**Returns:**the Session identifier

### getSessionContext

[SSLSessionContext](http://docs.google.com/javax/net/ssl/SSLSessionContext.html) **getSessionContext**()

Returns the context in which this session is bound.

This context may be unavailable in some environments, in which case this method returns null.

If the context is available and there is a security manager installed, the caller may require permission to access it or a security exception may be thrown. In a Java environment, the security manager's checkPermission method is called with a SSLPermission("getSSLSessionContext") permission.

**Returns:**the session context used for this session, or null if the context is unavailable. **Throws:** [SecurityException](http://docs.google.com/java/lang/SecurityException.html) - if the calling thread does not have permission to get SSL session context.

### getCreationTime

long **getCreationTime**()

Returns the time at which this Session representation was created, in milliseconds since midnight, January 1, 1970 UTC.

**Returns:**the time this Session was created

### getLastAccessedTime

long **getLastAccessedTime**()

Returns the last time this Session representation was accessed by the session level infrastructure, in milliseconds since midnight, January 1, 1970 UTC.

Access indicates a new connection being established using session data. Application level operations, such as getting or setting a value associated with the session, are not reflected in this access time.

This information is particularly useful in session management policies. For example, a session manager thread could leave all sessions in a given context which haven't been used in a long time; or, the sessions might be sorted according to age to optimize some task.

**Returns:**the last time this Session was accessed

### invalidate

void **invalidate**()

Invalidates the session.

Future connections will not be able to resume or join this session. However, any existing connection using this session can continue to use the session until the connection is closed.

**See Also:**[isValid()](http://docs.google.com/javax/net/ssl/SSLSession.html#isValid())

### isValid

boolean **isValid**()

Returns whether this session is valid and available for resuming or joining.

**Returns:**true if this session may be rejoined.**Since:** 1.5 **See Also:**[invalidate()](http://docs.google.com/javax/net/ssl/SSLSession.html#invalidate())

### putValue

void **putValue**([String](http://docs.google.com/java/lang/String.html) name,  
 [Object](http://docs.google.com/java/lang/Object.html) value)

Binds the specified value object into the session's application layer data with the given name.

Any existing binding using the same name is replaced. If the new (or existing) value implements the SSLSessionBindingListener interface, the object represented by value is notified appropriately.

For security reasons, the same named values may not be visible across different access control contexts.

**Parameters:**name - the name to which the data object will be bound. This may not be null.value - the data object to be bound. This may not be null. **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if either argument is null.

### getValue

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

Returns the object bound to the given name in the session's application layer data. Returns null if there is no such binding.

For security reasons, the same named values may not be visible across different access control contexts.

**Parameters:**name - the name of the binding to find. **Returns:**the value bound to that name, or null if the binding does not exist. **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the argument is null.

### removeValue

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

Removes the object bound to the given name in the session's application layer data. Does nothing if there is no object bound to the given name. If the bound existing object implements the SessionBindingListener interface, it is notified appropriately.

For security reasons, the same named values may not be visible across different access control contexts.

**Parameters:**name - the name of the object to remove visible across different access control contexts **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the argument is null.

### getValueNames

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

Returns an array of the names of all the application layer data objects bound into the Session.

For security reasons, the same named values may not be visible across different access control contexts.

**Returns:**a non-null (possibly empty) array of names of the objects bound to this Session.

### getPeerCertificates

[Certificate](http://docs.google.com/java/security/cert/Certificate.html)[] **getPeerCertificates**()  
 throws [SSLPeerUnverifiedException](http://docs.google.com/javax/net/ssl/SSLPeerUnverifiedException.html)

Returns the identity of the peer which was established as part of defining the session.

Note: This method can be used only when using certificate-based cipher suites; using it with non-certificate-based cipher suites, such as Kerberos, will throw an SSLPeerUnverifiedException.

**Returns:**an ordered array of peer certificates, with the peer's own certificate first followed by any certificate authorities. **Throws:** [SSLPeerUnverifiedException](http://docs.google.com/javax/net/ssl/SSLPeerUnverifiedException.html) - if the peer's identity has not been verified**See Also:**[getPeerPrincipal()](http://docs.google.com/javax/net/ssl/SSLSession.html#getPeerPrincipal())

### getLocalCertificates

[Certificate](http://docs.google.com/java/security/cert/Certificate.html)[] **getLocalCertificates**()

Returns the certificate(s) that were sent to the peer during handshaking.

Note: This method is useful only when using certificate-based cipher suites.

When multiple certificates are available for use in a handshake, the implementation chooses what it considers the "best" certificate chain available, and transmits that to the other side. This method allows the caller to know which certificate chain was actually used.

**Returns:**an ordered array of certificates, with the local certificate first followed by any certificate authorities. If no certificates were sent, then null is returned.**See Also:**[getLocalPrincipal()](http://docs.google.com/javax/net/ssl/SSLSession.html#getLocalPrincipal())

### getPeerCertificateChain

[X509Certificate](http://docs.google.com/javax/security/cert/X509Certificate.html)[] **getPeerCertificateChain**()  
 throws [SSLPeerUnverifiedException](http://docs.google.com/javax/net/ssl/SSLPeerUnverifiedException.html)

Returns the identity of the peer which was identified as part of defining the session.

Note: This method can be used only when using certificate-based cipher suites; using it with non-certificate-based cipher suites, such as Kerberos, will throw an SSLPeerUnverifiedException.

*Note: this method exists for compatibility with previous releases. New applications should use* [*getPeerCertificates()*](http://docs.google.com/javax/net/ssl/SSLSession.html#getPeerCertificates()) *instead.*

**Returns:**an ordered array of peer X.509 certificates, with the peer's own certificate first followed by any certificate authorities. (The certificates are in the original JSSE certificate [X509Certificate](http://docs.google.com/javax/security/cert/X509Certificate.html) format.) **Throws:** [SSLPeerUnverifiedException](http://docs.google.com/javax/net/ssl/SSLPeerUnverifiedException.html) - if the peer's identity has not been verified**See Also:**[getPeerPrincipal()](http://docs.google.com/javax/net/ssl/SSLSession.html#getPeerPrincipal())

### getPeerPrincipal

[Principal](http://docs.google.com/java/security/Principal.html) **getPeerPrincipal**()  
 throws [SSLPeerUnverifiedException](http://docs.google.com/javax/net/ssl/SSLPeerUnverifiedException.html)

Returns the identity of the peer which was established as part of defining the session.

**Returns:**the peer's principal. Returns an X500Principal of the end-entity certiticate for X509-based cipher suites, and KerberosPrincipal for Kerberos cipher suites. **Throws:** [SSLPeerUnverifiedException](http://docs.google.com/javax/net/ssl/SSLPeerUnverifiedException.html) - if the peer's identity has not been verified**Since:** 1.5 **See Also:**[getPeerCertificates()](http://docs.google.com/javax/net/ssl/SSLSession.html#getPeerCertificates()), [getLocalPrincipal()](http://docs.google.com/javax/net/ssl/SSLSession.html#getLocalPrincipal())

### getLocalPrincipal

[Principal](http://docs.google.com/java/security/Principal.html) **getLocalPrincipal**()

Returns the principal that was sent to the peer during handshaking.

**Returns:**the principal sent to the peer. Returns an X500Principal of the end-entity certificate for X509-based cipher suites, and KerberosPrincipal for Kerberos cipher suites. If no principal was sent, then null is returned.**Since:** 1.5 **See Also:**[getLocalCertificates()](http://docs.google.com/javax/net/ssl/SSLSession.html#getLocalCertificates()), [getPeerPrincipal()](http://docs.google.com/javax/net/ssl/SSLSession.html#getPeerPrincipal())

### getCipherSuite

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

Returns the name of the SSL cipher suite which is used for all connections in the session.

This defines the level of protection provided to the data sent on the connection, including the kind of encryption used and most aspects of how authentication is done.

**Returns:**the name of the session's cipher suite

### getProtocol

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

Returns the standard name of the protocol used for all connections in the session.

This defines the protocol used in the connection.

**Returns:**the standard name of the protocol used for all connections in the session.

### getPeerHost

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

Returns the host name of the peer in this session.

For the server, this is the client's host; and for the client, it is the server's host. The name may not be a fully qualified host name or even a host name at all as it may represent a string encoding of the peer's network address. If such a name is desired, it might be resolved through a name service based on the value returned by this method.

This value is not authenticated and should not be relied upon. It is mainly used as a hint for SSLSession caching strategies.

**Returns:**the host name of the peer host, or null if no information is available.

### getPeerPort

int **getPeerPort**()

Returns the port number of the peer in this session.

For the server, this is the client's port number; and for the client, it is the server's port number.

This value is not authenticated and should not be relied upon. It is mainly used as a hint for SSLSession caching strategies.

**Returns:**the port number of the peer host, or -1 if no information is available.**Since:** 1.5

### getPacketBufferSize

int **getPacketBufferSize**()

Gets the current size of the largest SSL/TLS packet that is expected when using this session.

A SSLEngine using this session may generate SSL/TLS packets of any size up to and including the value returned by this method. All SSLEngine network buffers should be sized at least this large to avoid insufficient space problems when performing wrap and unwrap calls.

**Returns:**the current maximum expected network packet size**Since:** 1.5 **See Also:**[SSLEngine.wrap(ByteBuffer, ByteBuffer)](http://docs.google.com/javax/net/ssl/SSLEngine.html#wrap(java.nio.ByteBuffer,%20java.nio.ByteBuffer)), [SSLEngine.unwrap(ByteBuffer, ByteBuffer)](http://docs.google.com/javax/net/ssl/SSLEngine.html#unwrap(java.nio.ByteBuffer,%20java.nio.ByteBuffer))

### getApplicationBufferSize

int **getApplicationBufferSize**()

Gets the current size of the largest application data that is expected when using this session.

SSLEngine application data buffers must be large enough to hold the application data from any inbound network application data packet received. Typically, outbound application data buffers can be of any size.

**Returns:**the current maximum expected application packet size**Since:** 1.5 **See Also:**[SSLEngine.wrap(ByteBuffer, ByteBuffer)](http://docs.google.com/javax/net/ssl/SSLEngine.html#wrap(java.nio.ByteBuffer,%20java.nio.ByteBuffer)), [SSLEngine.unwrap(ByteBuffer, ByteBuffer)](http://docs.google.com/javax/net/ssl/SSLEngine.html#unwrap(java.nio.ByteBuffer,%20java.nio.ByteBuffer))

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/SSLSession.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/net/ssl/SSLServerSocketFactory.html)   [**NEXT CLASS**](http://docs.google.com/javax/net/ssl/SSLSessionBindingEvent.html) | [**FRAMES**](http://docs.google.com/index.html?javax/net/ssl/SSLSession.html)    [**NO FRAMES**](http://docs.google.com/SSLSession.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#3znysh7) | DETAIL: FIELD | CONSTR | [METHOD](#2et92p0) |

[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).