| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/KeyStore.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/security/KeyRep.Type.html)   [**NEXT CLASS**](http://docs.google.com/java/security/KeyStore.Builder.html) | [**FRAMES**](http://docs.google.com/index.html?java/security/KeyStore.html)    [**NO FRAMES**](http://docs.google.com/KeyStore.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | FIELD | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: FIELD | [CONSTR](#1t3h5sf) | [METHOD](#2s8eyo1) |

## **java.security**

Class KeyStore

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

public class **KeyStore**extends [Object](http://docs.google.com/java/lang/Object.html)

This class represents a storage facility for cryptographic keys and certificates.

A KeyStore manages different types of entries. Each type of entry implements the KeyStore.Entry interface. Three basic KeyStore.Entry implementations are provided:

* **KeyStore.PrivateKeyEntry**  
  This type of entry holds a cryptographic PrivateKey, which is optionally stored in a protected format to prevent unauthorized access. It is also accompanied by a certificate chain for the corresponding public key.  
  Private keys and certificate chains are used by a given entity for self-authentication. Applications for this authentication include software distribution organizations which sign JAR files as part of releasing and/or licensing software.
* **KeyStore.SecretKeyEntry**  
  This type of entry holds a cryptographic SecretKey, which is optionally stored in a protected format to prevent unauthorized access.
* **KeyStore.TrustedCertificateEntry**  
  This type of entry contains a single public key Certificate belonging to another party. It is called a *trusted certificate* because the keystore owner trusts that the public key in the certificate indeed belongs to the identity identified by the *subject* (owner) of the certificate.  
  This type of entry can be used to authenticate other parties.

Each entry in a keystore is identified by an "alias" string. In the case of private keys and their associated certificate chains, these strings distinguish among the different ways in which the entity may authenticate itself. For example, the entity may authenticate itself using different certificate authorities, or using different public key algorithms.

Whether aliases are case sensitive is implementation dependent. In order to avoid problems, it is recommended not to use aliases in a KeyStore that only differ in case.

Whether keystores are persistent, and the mechanisms used by the keystore if it is persistent, are not specified here. This allows use of a variety of techniques for protecting sensitive (e.g., private or secret) keys. Smart cards or other integrated cryptographic engines (SafeKeyper) are one option, and simpler mechanisms such as files may also be used (in a variety of formats).

Typical ways to request a KeyStore object include relying on the default type and providing a specific keystore type.

* To rely on the default type:  
   KeyStore ks = KeyStore.getInstance(KeyStore.getDefaultType());  
   The system will return a keystore implementation for the default type.
* To provide a specific keystore type:  
   KeyStore ks = KeyStore.getInstance("JKS");  
   The system will return the most preferred implementation of the specified keystore type available in the environment.

Before a keystore can be accessed, it must be [loaded](http://docs.google.com/java/security/KeyStore.html#load(java.io.InputStream,%20char%5B%5D)).

KeyStore ks = KeyStore.getInstance(KeyStore.getDefaultType());  
  
 // get user password and file input stream  
 char[] password = getPassword();  
  
 java.io.FileInputStream fis = null;  
 try {  
 fis = new java.io.FileInputStream("keyStoreName");  
 ks.load(fis, password);  
 } finally {  
 if (fis != null) {  
 fis.close();  
 }  
 }

To create an empty keystore using the above load method, pass null as the InputStream argument.

Once the keystore has been loaded, it is possible to read existing entries from the keystore, or to write new entries into the keystore:

// get my private key  
 KeyStore.PrivateKeyEntry pkEntry = (KeyStore.PrivateKeyEntry)  
 ks.getEntry("privateKeyAlias", password);  
 PrivateKey myPrivateKey = pkEntry.getPrivateKey();  
  
 // save my secret key  
 javax.crypto.SecretKey mySecretKey;  
 KeyStore.SecretKeyEntry skEntry =  
 new KeyStore.SecretKeyEntry(mySecretKey);  
 ks.setEntry("secretKeyAlias", skEntry,   
 new KeyStore.PasswordProtection(password));  
  
 // store away the keystore  
 java.io.FileOutputStream fos = null;  
 try {  
 fos = new java.io.FileOutputStream("newKeyStoreName");  
 ks.store(fos, password);  
 } finally {  
 if (fos != null) {  
 fos.close();  
 }  
 }

Note that although the same password may be used to load the keystore, to protect the private key entry, to protect the secret key entry, and to store the keystore (as is shown in the sample code above), different passwords or other protection parameters may also be used.

**Since:** 1.2 **See Also:**[PrivateKey](http://docs.google.com/java/security/PrivateKey.html), [SecretKey](http://docs.google.com/javax/crypto/SecretKey.html), [Certificate](http://docs.google.com/java/security/cert/Certificate.html)

| **Nested Class Summary** | |
| --- | --- |
| static class | [**KeyStore.Builder**](http://docs.google.com/java/security/KeyStore.Builder.html)            A description of a to-be-instantiated KeyStore object. |
| static class | [**KeyStore.CallbackHandlerProtection**](http://docs.google.com/java/security/KeyStore.CallbackHandlerProtection.html)            A ProtectionParameter encapsulating a CallbackHandler. |
| static interface | [**KeyStore.Entry**](http://docs.google.com/java/security/KeyStore.Entry.html)            A marker interface for KeyStore entry types. |
| static interface | [**KeyStore.LoadStoreParameter**](http://docs.google.com/java/security/KeyStore.LoadStoreParameter.html)            A marker interface for KeyStore [load](http://docs.google.com/java/security/KeyStore.html#load(java.security.KeyStore.LoadStoreParameter)) and [store](http://docs.google.com/java/security/KeyStore.html#store(java.security.KeyStore.LoadStoreParameter)) parameters. |
| static class | [**KeyStore.PasswordProtection**](http://docs.google.com/java/security/KeyStore.PasswordProtection.html)            A password-based implementation of ProtectionParameter. |
| static class | [**KeyStore.PrivateKeyEntry**](http://docs.google.com/java/security/KeyStore.PrivateKeyEntry.html)            A KeyStore entry that holds a PrivateKey and corresponding certificate chain. |
| static interface | [**KeyStore.ProtectionParameter**](http://docs.google.com/java/security/KeyStore.ProtectionParameter.html)            A marker interface for keystore protection parameters. |
| static class | [**KeyStore.SecretKeyEntry**](http://docs.google.com/java/security/KeyStore.SecretKeyEntry.html)            A KeyStore entry that holds a SecretKey. |
| static class | [**KeyStore.TrustedCertificateEntry**](http://docs.google.com/java/security/KeyStore.TrustedCertificateEntry.html)            A KeyStore entry that holds a trusted Certificate. |

| **Constructor Summary** | |
| --- | --- |
| protected | [**KeyStore**](http://docs.google.com/java/security/KeyStore.html#KeyStore(java.security.KeyStoreSpi,%20java.security.Provider,%20java.lang.String))([KeyStoreSpi](http://docs.google.com/java/security/KeyStoreSpi.html) keyStoreSpi, [Provider](http://docs.google.com/java/security/Provider.html) provider, [String](http://docs.google.com/java/lang/String.html) type)            Creates a KeyStore object of the given type, and encapsulates the given provider implementation (SPI object) in it. |

| **Method Summary** | |
| --- | --- |
| [Enumeration](http://docs.google.com/java/util/Enumeration.html)<[String](http://docs.google.com/java/lang/String.html)> | [**aliases**](http://docs.google.com/java/security/KeyStore.html#aliases())()            Lists all the alias names of this keystore. |
| boolean | [**containsAlias**](http://docs.google.com/java/security/KeyStore.html#containsAlias(java.lang.String))([String](http://docs.google.com/java/lang/String.html) alias)            Checks if the given alias exists in this keystore. |
| void | [**deleteEntry**](http://docs.google.com/java/security/KeyStore.html#deleteEntry(java.lang.String))([String](http://docs.google.com/java/lang/String.html) alias)            Deletes the entry identified by the given alias from this keystore. |
| boolean | [**entryInstanceOf**](http://docs.google.com/java/security/KeyStore.html#entryInstanceOf(java.lang.String,%20java.lang.Class))([String](http://docs.google.com/java/lang/String.html) alias, [Class](http://docs.google.com/java/lang/Class.html)<? extends [KeyStore.Entry](http://docs.google.com/java/security/KeyStore.Entry.html)> entryClass)            Determines if the keystore Entry for the specified alias is an instance or subclass of the specified entryClass. |
| [Certificate](http://docs.google.com/java/security/cert/Certificate.html) | [**getCertificate**](http://docs.google.com/java/security/KeyStore.html#getCertificate(java.lang.String))([String](http://docs.google.com/java/lang/String.html) alias)            Returns the certificate associated with the given alias. |
| [String](http://docs.google.com/java/lang/String.html) | [**getCertificateAlias**](http://docs.google.com/java/security/KeyStore.html#getCertificateAlias(java.security.cert.Certificate))([Certificate](http://docs.google.com/java/security/cert/Certificate.html) cert)            Returns the (alias) name of the first keystore entry whose certificate matches the given certificate. |
| [Certificate](http://docs.google.com/java/security/cert/Certificate.html)[] | [**getCertificateChain**](http://docs.google.com/java/security/KeyStore.html#getCertificateChain(java.lang.String))([String](http://docs.google.com/java/lang/String.html) alias)            Returns the certificate chain associated with the given alias. |
| [Date](http://docs.google.com/java/util/Date.html) | [**getCreationDate**](http://docs.google.com/java/security/KeyStore.html#getCreationDate(java.lang.String))([String](http://docs.google.com/java/lang/String.html) alias)            Returns the creation date of the entry identified by the given alias. |
| static [String](http://docs.google.com/java/lang/String.html) | [**getDefaultType**](http://docs.google.com/java/security/KeyStore.html#getDefaultType())()            Returns the default keystore type as specified in the Java security properties file, or the string "jks" (acronym for "Java keystore") if no such property exists. |
| [KeyStore.Entry](http://docs.google.com/java/security/KeyStore.Entry.html) | [**getEntry**](http://docs.google.com/java/security/KeyStore.html#getEntry(java.lang.String,%20java.security.KeyStore.ProtectionParameter))([String](http://docs.google.com/java/lang/String.html) alias, [KeyStore.ProtectionParameter](http://docs.google.com/java/security/KeyStore.ProtectionParameter.html) protParam)            Gets a keystore Entry for the specified alias with the specified protection parameter. |
| static [KeyStore](http://docs.google.com/java/security/KeyStore.html) | [**getInstance**](http://docs.google.com/java/security/KeyStore.html#getInstance(java.lang.String))([String](http://docs.google.com/java/lang/String.html) type)            Returns a keystore object of the specified type. |
| static [KeyStore](http://docs.google.com/java/security/KeyStore.html) | [**getInstance**](http://docs.google.com/java/security/KeyStore.html#getInstance(java.lang.String,%20java.security.Provider))([String](http://docs.google.com/java/lang/String.html) type, [Provider](http://docs.google.com/java/security/Provider.html) provider)            Returns a keystore object of the specified type. |
| static [KeyStore](http://docs.google.com/java/security/KeyStore.html) | [**getInstance**](http://docs.google.com/java/security/KeyStore.html#getInstance(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) type, [String](http://docs.google.com/java/lang/String.html) provider)            Returns a keystore object of the specified type. |
| [Key](http://docs.google.com/java/security/Key.html) | [**getKey**](http://docs.google.com/java/security/KeyStore.html#getKey(java.lang.String,%20char%5B%5D))([String](http://docs.google.com/java/lang/String.html) alias, char[] password)            Returns the key associated with the given alias, using the given password to recover it. |
| [Provider](http://docs.google.com/java/security/Provider.html) | [**getProvider**](http://docs.google.com/java/security/KeyStore.html#getProvider())()            Returns the provider of this keystore. |
| [String](http://docs.google.com/java/lang/String.html) | [**getType**](http://docs.google.com/java/security/KeyStore.html#getType())()            Returns the type of this keystore. |
| boolean | [**isCertificateEntry**](http://docs.google.com/java/security/KeyStore.html#isCertificateEntry(java.lang.String))([String](http://docs.google.com/java/lang/String.html) alias)            Returns true if the entry identified by the given alias was created by a call to setCertificateEntry, or created by a call to setEntry with a TrustedCertificateEntry. |
| boolean | [**isKeyEntry**](http://docs.google.com/java/security/KeyStore.html#isKeyEntry(java.lang.String))([String](http://docs.google.com/java/lang/String.html) alias)            Returns true if the entry identified by the given alias was created by a call to setKeyEntry, or created by a call to setEntry with a PrivateKeyEntry or a SecretKeyEntry. |
| void | [**load**](http://docs.google.com/java/security/KeyStore.html#load(java.io.InputStream,%20char%5B%5D))([InputStream](http://docs.google.com/java/io/InputStream.html) stream, char[] password)            Loads this KeyStore from the given input stream. |
| void | [**load**](http://docs.google.com/java/security/KeyStore.html#load(java.security.KeyStore.LoadStoreParameter))([KeyStore.LoadStoreParameter](http://docs.google.com/java/security/KeyStore.LoadStoreParameter.html) param)            Loads this keystore using the given LoadStoreParameter. |
| void | [**setCertificateEntry**](http://docs.google.com/java/security/KeyStore.html#setCertificateEntry(java.lang.String,%20java.security.cert.Certificate))([String](http://docs.google.com/java/lang/String.html) alias, [Certificate](http://docs.google.com/java/security/cert/Certificate.html) cert)            Assigns the given trusted certificate to the given alias. |
| void | [**setEntry**](http://docs.google.com/java/security/KeyStore.html#setEntry(java.lang.String,%20java.security.KeyStore.Entry,%20java.security.KeyStore.ProtectionParameter))([String](http://docs.google.com/java/lang/String.html) alias, [KeyStore.Entry](http://docs.google.com/java/security/KeyStore.Entry.html) entry, [KeyStore.ProtectionParameter](http://docs.google.com/java/security/KeyStore.ProtectionParameter.html) protParam)            Saves a keystore Entry under the specified alias. |
| void | [**setKeyEntry**](http://docs.google.com/java/security/KeyStore.html#setKeyEntry(java.lang.String,%20byte%5B%5D,%20java.security.cert.Certificate%5B%5D))([String](http://docs.google.com/java/lang/String.html) alias, byte[] key, [Certificate](http://docs.google.com/java/security/cert/Certificate.html)[] chain)            Assigns the given key (that has already been protected) to the given alias. |
| void | [**setKeyEntry**](http://docs.google.com/java/security/KeyStore.html#setKeyEntry(java.lang.String,%20java.security.Key,%20char%5B%5D,%20java.security.cert.Certificate%5B%5D))([String](http://docs.google.com/java/lang/String.html) alias, [Key](http://docs.google.com/java/security/Key.html) key, char[] password, [Certificate](http://docs.google.com/java/security/cert/Certificate.html)[] chain)            Assigns the given key to the given alias, protecting it with the given password. |
| int | [**size**](http://docs.google.com/java/security/KeyStore.html#size())()            Retrieves the number of entries in this keystore. |
| void | [**store**](http://docs.google.com/java/security/KeyStore.html#store(java.security.KeyStore.LoadStoreParameter))([KeyStore.LoadStoreParameter](http://docs.google.com/java/security/KeyStore.LoadStoreParameter.html) param)            Stores this keystore using the given LoadStoreParameter. |
| void | [**store**](http://docs.google.com/java/security/KeyStore.html#store(java.io.OutputStream,%20char%5B%5D))([OutputStream](http://docs.google.com/java/io/OutputStream.html) stream, char[] password)            Stores this keystore to the given output stream, and protects its integrity with the given password. |

| **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)) |

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

### KeyStore

protected **KeyStore**([KeyStoreSpi](http://docs.google.com/java/security/KeyStoreSpi.html) keyStoreSpi,  
 [Provider](http://docs.google.com/java/security/Provider.html) provider,  
 [String](http://docs.google.com/java/lang/String.html) type)

Creates a KeyStore object of the given type, and encapsulates the given provider implementation (SPI object) in it.

**Parameters:**keyStoreSpi - the provider implementation.provider - the provider.type - the keystore type.

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

### getInstance

public static [KeyStore](http://docs.google.com/java/security/KeyStore.html) **getInstance**([String](http://docs.google.com/java/lang/String.html) type)  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html)

Returns a keystore object of the specified type.

This method traverses the list of registered security Providers, starting with the most preferred Provider. A new KeyStore object encapsulating the KeyStoreSpi implementation from the first Provider that supports the specified type is returned.

Note that the list of registered providers may be retrieved via the [Security.getProviders()](http://docs.google.com/java/security/Security.html#getProviders()) method.

**Parameters:**type - the type of keystore. See Appendix A in the  [Java Cryptography Architecture API Specification & Reference](http://docs.google.com/technotes/guides/security/crypto/CryptoSpec.html#AppA)  for information about standard keystore types. **Returns:**a keystore object of the specified type. **Throws:** [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if no Provider supports a KeyStoreSpi implementation for the specified type.**See Also:**[Provider](http://docs.google.com/java/security/Provider.html)

### getInstance

public static [KeyStore](http://docs.google.com/java/security/KeyStore.html) **getInstance**([String](http://docs.google.com/java/lang/String.html) type,  
 [String](http://docs.google.com/java/lang/String.html) provider)  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html),  
 [NoSuchProviderException](http://docs.google.com/java/security/NoSuchProviderException.html)

Returns a keystore object of the specified type.

A new KeyStore object encapsulating the KeyStoreSpi implementation from the specified provider is returned. The specified provider must be registered in the security provider list.

Note that the list of registered providers may be retrieved via the [Security.getProviders()](http://docs.google.com/java/security/Security.html#getProviders()) method.

**Parameters:**type - the type of keystore. See Appendix A in the  [Java Cryptography Architecture API Specification & Reference](http://docs.google.com/technotes/guides/security/crypto/CryptoSpec.html#AppA)  for information about standard keystore types.provider - the name of the provider. **Returns:**a keystore object of the specified type. **Throws:** [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if a KeyStoreSpi implementation for the specified type is not available from the specified provider. [NoSuchProviderException](http://docs.google.com/java/security/NoSuchProviderException.html) - if the specified provider is not registered in the security provider list. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the provider name is null or empty.**See Also:**[Provider](http://docs.google.com/java/security/Provider.html)

### getInstance

public static [KeyStore](http://docs.google.com/java/security/KeyStore.html) **getInstance**([String](http://docs.google.com/java/lang/String.html) type,  
 [Provider](http://docs.google.com/java/security/Provider.html) provider)  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html)

Returns a keystore object of the specified type.

A new KeyStore object encapsulating the KeyStoreSpi implementation from the specified Provider object is returned. Note that the specified Provider object does not have to be registered in the provider list.

**Parameters:**type - the type of keystore. See Appendix A in the  [Java Cryptography Architecture API Specification & Reference](http://docs.google.com/technotes/guides/security/crypto/CryptoSpec.html#AppA)  for information about standard keystore types.provider - the provider. **Returns:**a keystore object of the specified type. **Throws:** [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if KeyStoreSpi implementation for the specified type is not available from the specified Provider object. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the specified provider is null.**Since:** 1.4 **See Also:**[Provider](http://docs.google.com/java/security/Provider.html)

### getDefaultType

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

Returns the default keystore type as specified in the Java security properties file, or the string "jks" (acronym for "Java keystore") if no such property exists. The Java security properties file is located in the file named <JAVA\_HOME>/lib/security/java.security. <JAVA\_HOME> refers to the value of the java.home system property, and specifies the directory where the JRE is installed.

The default keystore type can be used by applications that do not want to use a hard-coded keystore type when calling one of the getInstance methods, and want to provide a default keystore type in case a user does not specify its own.

The default keystore type can be changed by setting the value of the "keystore.type" security property (in the Java security properties file) to the desired keystore type.

**Returns:**the default keystore type as specified in the Java security properties file, or the string "jks" if no such property exists.

### getProvider

public final [Provider](http://docs.google.com/java/security/Provider.html) **getProvider**()

Returns the provider of this keystore.

**Returns:**the provider of this keystore.

### getType

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

Returns the type of this keystore.

**Returns:**the type of this keystore.

### getKey

public final [Key](http://docs.google.com/java/security/Key.html) **getKey**([String](http://docs.google.com/java/lang/String.html) alias,  
 char[] password)  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html),  
 [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html),  
 [UnrecoverableKeyException](http://docs.google.com/java/security/UnrecoverableKeyException.html)

Returns the key associated with the given alias, using the given password to recover it. The key must have been associated with the alias by a call to setKeyEntry, or by a call to setEntry with a PrivateKeyEntry or SecretKeyEntry.

**Parameters:**alias - the alias namepassword - the password for recovering the key **Returns:**the requested key, or null if the given alias does not exist or does not identify a key-related entry. **Throws:** [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if the keystore has not been initialized (loaded). [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if the algorithm for recovering the key cannot be found [UnrecoverableKeyException](http://docs.google.com/java/security/UnrecoverableKeyException.html) - if the key cannot be recovered (e.g., the given password is wrong).

### getCertificateChain

public final [Certificate](http://docs.google.com/java/security/cert/Certificate.html)[] **getCertificateChain**([String](http://docs.google.com/java/lang/String.html) alias)  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html)

Returns the certificate chain associated with the given alias. The certificate chain must have been associated with the alias by a call to setKeyEntry, or by a call to setEntry with a PrivateKeyEntry.

**Parameters:**alias - the alias name **Returns:**the certificate chain (ordered with the user's certificate first and the root certificate authority last), or null if the given alias does not exist or does not contain a certificate chain **Throws:** [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if the keystore has not been initialized (loaded).

### getCertificate

public final [Certificate](http://docs.google.com/java/security/cert/Certificate.html) **getCertificate**([String](http://docs.google.com/java/lang/String.html) alias)  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html)

Returns the certificate associated with the given alias.

If the given alias name identifies an entry created by a call to setCertificateEntry, or created by a call to setEntry with a TrustedCertificateEntry, then the trusted certificate contained in that entry is returned.

If the given alias name identifies an entry created by a call to setKeyEntry, or created by a call to setEntry with a PrivateKeyEntry, then the first element of the certificate chain in that entry is returned.

**Parameters:**alias - the alias name **Returns:**the certificate, or null if the given alias does not exist or does not contain a certificate. **Throws:** [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if the keystore has not been initialized (loaded).

### getCreationDate

public final [Date](http://docs.google.com/java/util/Date.html) **getCreationDate**([String](http://docs.google.com/java/lang/String.html) alias)  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html)

Returns the creation date of the entry identified by the given alias.

**Parameters:**alias - the alias name **Returns:**the creation date of this entry, or null if the given alias does not exist **Throws:** [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if the keystore has not been initialized (loaded).

### setKeyEntry

public final void **setKeyEntry**([String](http://docs.google.com/java/lang/String.html) alias,  
 [Key](http://docs.google.com/java/security/Key.html) key,  
 char[] password,  
 [Certificate](http://docs.google.com/java/security/cert/Certificate.html)[] chain)  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html)

Assigns the given key to the given alias, protecting it with the given password.

If the given key is of type java.security.PrivateKey, it must be accompanied by a certificate chain certifying the corresponding public key.

If the given alias already exists, the keystore information associated with it is overridden by the given key (and possibly certificate chain).

**Parameters:**alias - the alias namekey - the key to be associated with the aliaspassword - the password to protect the keychain - the certificate chain for the corresponding public key (only required if the given key is of type java.security.PrivateKey). **Throws:** [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if the keystore has not been initialized (loaded), the given key cannot be protected, or this operation fails for some other reason

### setKeyEntry

public final void **setKeyEntry**([String](http://docs.google.com/java/lang/String.html) alias,  
 byte[] key,  
 [Certificate](http://docs.google.com/java/security/cert/Certificate.html)[] chain)  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html)

Assigns the given key (that has already been protected) to the given alias.

If the protected key is of type java.security.PrivateKey, it must be accompanied by a certificate chain certifying the corresponding public key. If the underlying keystore implementation is of type jks, key must be encoded as an EncryptedPrivateKeyInfo as defined in the PKCS #8 standard.

If the given alias already exists, the keystore information associated with it is overridden by the given key (and possibly certificate chain).

**Parameters:**alias - the alias namekey - the key (in protected format) to be associated with the aliaschain - the certificate chain for the corresponding public key (only useful if the protected key is of type java.security.PrivateKey). **Throws:** [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if the keystore has not been initialized (loaded), or if this operation fails for some other reason.

### setCertificateEntry

public final void **setCertificateEntry**([String](http://docs.google.com/java/lang/String.html) alias,  
 [Certificate](http://docs.google.com/java/security/cert/Certificate.html) cert)  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html)

Assigns the given trusted certificate to the given alias.

If the given alias identifies an existing entry created by a call to setCertificateEntry, or created by a call to setEntry with a TrustedCertificateEntry, the trusted certificate in the existing entry is overridden by the given certificate.

**Parameters:**alias - the alias namecert - the certificate **Throws:** [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if the keystore has not been initialized, or the given alias already exists and does not identify an entry containing a trusted certificate, or this operation fails for some other reason.

### deleteEntry

public final void **deleteEntry**([String](http://docs.google.com/java/lang/String.html) alias)  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html)

Deletes the entry identified by the given alias from this keystore.

**Parameters:**alias - the alias name **Throws:** [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if the keystore has not been initialized, or if the entry cannot be removed.

### aliases

public final [Enumeration](http://docs.google.com/java/util/Enumeration.html)<[String](http://docs.google.com/java/lang/String.html)> **aliases**()  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html)

Lists all the alias names of this keystore.

**Returns:**enumeration of the alias names **Throws:** [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if the keystore has not been initialized (loaded).

### containsAlias

public final boolean **containsAlias**([String](http://docs.google.com/java/lang/String.html) alias)  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html)

Checks if the given alias exists in this keystore.

**Parameters:**alias - the alias name **Returns:**true if the alias exists, false otherwise **Throws:** [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if the keystore has not been initialized (loaded).

### size

public final int **size**()  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html)

Retrieves the number of entries in this keystore.

**Returns:**the number of entries in this keystore **Throws:** [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if the keystore has not been initialized (loaded).

### isKeyEntry

public final boolean **isKeyEntry**([String](http://docs.google.com/java/lang/String.html) alias)  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html)

Returns true if the entry identified by the given alias was created by a call to setKeyEntry, or created by a call to setEntry with a PrivateKeyEntry or a SecretKeyEntry.

**Parameters:**alias - the alias for the keystore entry to be checked **Returns:**true if the entry identified by the given alias is a key-related entry, false otherwise. **Throws:** [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if the keystore has not been initialized (loaded).

### isCertificateEntry

public final boolean **isCertificateEntry**([String](http://docs.google.com/java/lang/String.html) alias)  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html)

Returns true if the entry identified by the given alias was created by a call to setCertificateEntry, or created by a call to setEntry with a TrustedCertificateEntry.

**Parameters:**alias - the alias for the keystore entry to be checked **Returns:**true if the entry identified by the given alias contains a trusted certificate, false otherwise. **Throws:** [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if the keystore has not been initialized (loaded).

### getCertificateAlias

public final [String](http://docs.google.com/java/lang/String.html) **getCertificateAlias**([Certificate](http://docs.google.com/java/security/cert/Certificate.html) cert)  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html)

Returns the (alias) name of the first keystore entry whose certificate matches the given certificate.

This method attempts to match the given certificate with each keystore entry. If the entry being considered was created by a call to setCertificateEntry, or created by a call to setEntry with a TrustedCertificateEntry, then the given certificate is compared to that entry's certificate.

If the entry being considered was created by a call to setKeyEntry, or created by a call to setEntry with a PrivateKeyEntry, then the given certificate is compared to the first element of that entry's certificate chain.

**Parameters:**cert - the certificate to match with. **Returns:**the alias name of the first entry with a matching certificate, or null if no such entry exists in this keystore. **Throws:** [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if the keystore has not been initialized (loaded).

### store

public final void **store**([OutputStream](http://docs.google.com/java/io/OutputStream.html) stream,  
 char[] password)  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html),  
 [IOException](http://docs.google.com/java/io/IOException.html),  
 [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html),  
 [CertificateException](http://docs.google.com/java/security/cert/CertificateException.html)

Stores this keystore to the given output stream, and protects its integrity with the given password.

**Parameters:**stream - the output stream to which this keystore is written.password - the password to generate the keystore integrity check **Throws:** [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if the keystore has not been initialized (loaded). [IOException](http://docs.google.com/java/io/IOException.html) - if there was an I/O problem with data [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if the appropriate data integrity algorithm could not be found [CertificateException](http://docs.google.com/java/security/cert/CertificateException.html) - if any of the certificates included in the keystore data could not be stored

### store

public final void **store**([KeyStore.LoadStoreParameter](http://docs.google.com/java/security/KeyStore.LoadStoreParameter.html) param)  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html),  
 [IOException](http://docs.google.com/java/io/IOException.html),  
 [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html),  
 [CertificateException](http://docs.google.com/java/security/cert/CertificateException.html)

Stores this keystore using the given LoadStoreParameter.

**Parameters:**param - the LoadStoreParameter that specifies how to store the keystore, which may be null **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the given LoadStoreParameter input is not recognized [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if the keystore has not been initialized (loaded) [IOException](http://docs.google.com/java/io/IOException.html) - if there was an I/O problem with data [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if the appropriate data integrity algorithm could not be found [CertificateException](http://docs.google.com/java/security/cert/CertificateException.html) - if any of the certificates included in the keystore data could not be stored**Since:** 1.5

### load

public final void **load**([InputStream](http://docs.google.com/java/io/InputStream.html) stream,  
 char[] password)  
 throws [IOException](http://docs.google.com/java/io/IOException.html),  
 [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html),  
 [CertificateException](http://docs.google.com/java/security/cert/CertificateException.html)

Loads this KeyStore from the given input stream.

A password may be given to unlock the keystore (e.g. the keystore resides on a hardware token device), or to check the integrity of the keystore data. If a password is not given for integrity checking, then integrity checking is not performed.

In order to create an empty keystore, or if the keystore cannot be initialized from a stream, pass null as the stream argument.

Note that if this keystore has already been loaded, it is reinitialized and loaded again from the given input stream.

**Parameters:**stream - the input stream from which the keystore is loaded, or nullpassword - the password used to check the integrity of the keystore, the password used to unlock the keystore, or null **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if there is an I/O or format problem with the keystore data, if a password is required but not given, or if the given password was incorrect. If the error is due to a wrong password, the [cause](http://docs.google.com/java/lang/Throwable.html#getCause()) of the IOException should be an UnrecoverableKeyException [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if the algorithm used to check the integrity of the keystore cannot be found [CertificateException](http://docs.google.com/java/security/cert/CertificateException.html) - if any of the certificates in the keystore could not be loaded

### load

public final void **load**([KeyStore.LoadStoreParameter](http://docs.google.com/java/security/KeyStore.LoadStoreParameter.html) param)  
 throws [IOException](http://docs.google.com/java/io/IOException.html),  
 [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html),  
 [CertificateException](http://docs.google.com/java/security/cert/CertificateException.html)

Loads this keystore using the given LoadStoreParameter.

Note that if this KeyStore has already been loaded, it is reinitialized and loaded again from the given parameter.

**Parameters:**param - the LoadStoreParameter that specifies how to load the keystore, which may be null **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the given LoadStoreParameter input is not recognized [IOException](http://docs.google.com/java/io/IOException.html) - if there is an I/O or format problem with the keystore data. If the error is due to an incorrect ProtectionParameter (e.g. wrong password) the [cause](http://docs.google.com/java/lang/Throwable.html#getCause()) of the IOException should be an UnrecoverableKeyException [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if the algorithm used to check the integrity of the keystore cannot be found [CertificateException](http://docs.google.com/java/security/cert/CertificateException.html) - if any of the certificates in the keystore could not be loaded**Since:** 1.5

### getEntry

public final [KeyStore.Entry](http://docs.google.com/java/security/KeyStore.Entry.html) **getEntry**([String](http://docs.google.com/java/lang/String.html) alias,  
 [KeyStore.ProtectionParameter](http://docs.google.com/java/security/KeyStore.ProtectionParameter.html) protParam)  
 throws [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html),  
 [UnrecoverableEntryException](http://docs.google.com/java/security/UnrecoverableEntryException.html),  
 [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html)

Gets a keystore Entry for the specified alias with the specified protection parameter.

**Parameters:**alias - get the keystore Entry for this aliasprotParam - the ProtectionParameter used to protect the Entry, which may be null **Returns:**the keystore Entry for the specified alias, or null if there is no such entry **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if alias is null [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if the algorithm for recovering the entry cannot be found [UnrecoverableEntryException](http://docs.google.com/java/security/UnrecoverableEntryException.html) - if the specified protParam were insufficient or invalid [UnrecoverableKeyException](http://docs.google.com/java/security/UnrecoverableKeyException.html) - if the entry is a PrivateKeyEntry or SecretKeyEntry and the specified protParam does not contain the information needed to recover the key (e.g. wrong password) [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if the keystore has not been initialized (loaded).**Since:** 1.5 **See Also:**[setEntry(String, KeyStore.Entry, KeyStore.ProtectionParameter)](http://docs.google.com/java/security/KeyStore.html#setEntry(java.lang.String,%20java.security.KeyStore.Entry,%20java.security.KeyStore.ProtectionParameter))

### setEntry

public final void **setEntry**([String](http://docs.google.com/java/lang/String.html) alias,  
 [KeyStore.Entry](http://docs.google.com/java/security/KeyStore.Entry.html) entry,  
 [KeyStore.ProtectionParameter](http://docs.google.com/java/security/KeyStore.ProtectionParameter.html) protParam)  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html)

Saves a keystore Entry under the specified alias. The protection parameter is used to protect the Entry.

If an entry already exists for the specified alias, it is overridden.

**Parameters:**alias - save the keystore Entry under this aliasentry - the Entry to saveprotParam - the ProtectionParameter used to protect the Entry, which may be null **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if alias or entry is null [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if the keystore has not been initialized (loaded), or if this operation fails for some other reason**Since:** 1.5 **See Also:**[getEntry(String, KeyStore.ProtectionParameter)](http://docs.google.com/java/security/KeyStore.html#getEntry(java.lang.String,%20java.security.KeyStore.ProtectionParameter))

### entryInstanceOf

public final boolean **entryInstanceOf**([String](http://docs.google.com/java/lang/String.html) alias,  
 [Class](http://docs.google.com/java/lang/Class.html)<? extends [KeyStore.Entry](http://docs.google.com/java/security/KeyStore.Entry.html)> entryClass)  
 throws [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html)

Determines if the keystore Entry for the specified alias is an instance or subclass of the specified entryClass.

**Parameters:**alias - the alias nameentryClass - the entry class **Returns:**true if the keystore Entry for the specified alias is an instance or subclass of the specified entryClass, false otherwise **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if alias or entryClass is null [KeyStoreException](http://docs.google.com/java/security/KeyStoreException.html) - if the keystore has not been initialized (loaded)**Since:** 1.5

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/KeyStore.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/security/KeyRep.Type.html)   [**NEXT CLASS**](http://docs.google.com/java/security/KeyStore.Builder.html) | [**FRAMES**](http://docs.google.com/index.html?java/security/KeyStore.html)    [**NO FRAMES**](http://docs.google.com/KeyStore.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | FIELD | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | 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).