| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/X509Certificate.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/security/cert/CertificateParsingException.html)   NEXT CLASS | [**FRAMES**](http://docs.google.com/index.html?javax/security/cert/X509Certificate.html)    [**NO FRAMES**](http://docs.google.com/X509Certificate.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#1t3h5sf) | [METHOD](#2s8eyo1) |

## **javax.security.cert**

Class X509Certificate

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 [javax.security.cert.Certificate](http://docs.google.com/javax/security/cert/Certificate.html)  
 **javax.security.cert.X509Certificate**

public abstract class **X509Certificate**extends [Certificate](http://docs.google.com/javax/security/cert/Certificate.html)

Abstract class for X.509 v1 certificates. This provides a standard way to access all the version 1 attributes of an X.509 certificate. Attributes that are specific to X.509 v2 or v3 are not available through this interface. Future API evolution will provide full access to complete X.509 v3 attributes.

The basic X.509 format was defined by ISO/IEC and ANSI X9 and is described below in ASN.1:

Certificate ::= SEQUENCE {  
 tbsCertificate TBSCertificate,  
 signatureAlgorithm AlgorithmIdentifier,  
 signature BIT STRING }

These certificates are widely used to support authentication and other functionality in Internet security systems. Common applications include Privacy Enhanced Mail (PEM), Transport Layer Security (SSL), code signing for trusted software distribution, and Secure Electronic Transactions (SET).

These certificates are managed and vouched for by *Certificate Authorities* (CAs). CAs are services which create certificates by placing data in the X.509 standard format and then digitally signing that data. CAs act as trusted third parties, making introductions between principals who have no direct knowledge of each other. CA certificates are either signed by themselves, or by some other CA such as a "root" CA.

The ASN.1 definition of tbsCertificate is:

TBSCertificate ::= SEQUENCE {  
 version [0] EXPLICIT Version DEFAULT v1,  
 serialNumber CertificateSerialNumber,  
 signature AlgorithmIdentifier,  
 issuer Name,  
 validity Validity,  
 subject Name,  
 subjectPublicKeyInfo SubjectPublicKeyInfo,  
 }

Here is sample code to instantiate an X.509 certificate:

InputStream inStream = new FileInputStream("fileName-of-cert");  
 X509Certificate cert = X509Certificate.getInstance(inStream);  
 inStream.close();

OR

byte[] certData = <certificate read from a file, say>  
 X509Certificate cert = X509Certificate.getInstance(certData);

In either case, the code that instantiates an X.509 certificate consults the Java security properties file to locate the actual implementation or instantiates a default implementation.

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. In the Security properties file, a default implementation for X.509 v1 may be given such as:

cert.provider.x509v1=com.sun.security.cert.internal.x509.X509V1CertImpl

The value of this cert.provider.x509v1 property has to be changed to instatiate another implementation. If this security property is not set, a default implementation will be used. Currently, due to possible security restrictions on access to Security properties, this value is looked up and cached at class initialization time and will fallback on a default implementation if the Security property is not accessible.

*Note: The classes in the package javax.security.cert exist for compatibility with earlier versions of the Java Secure Sockets Extension (JSSE). New applications should instead use the standard Java SE certificate classes located in java.security.cert.*

**Since:** 1.4 **See Also:**[Certificate](http://docs.google.com/javax/security/cert/Certificate.html), [X509Extension](http://docs.google.com/java/security/cert/X509Extension.html)

| **Constructor Summary** | |
| --- | --- |
| [**X509Certificate**](http://docs.google.com/javax/security/cert/X509Certificate.html#X509Certificate())() |

| **Method Summary** | |
| --- | --- |
| abstract  void | [**checkValidity**](http://docs.google.com/javax/security/cert/X509Certificate.html#checkValidity())()            Checks that the certificate is currently valid. |
| abstract  void | [**checkValidity**](http://docs.google.com/javax/security/cert/X509Certificate.html#checkValidity(java.util.Date))([Date](http://docs.google.com/java/util/Date.html) date)            Checks that the specified date is within the certificate's validity period. |
| static [X509Certificate](http://docs.google.com/javax/security/cert/X509Certificate.html) | [**getInstance**](http://docs.google.com/javax/security/cert/X509Certificate.html#getInstance(byte%5B%5D))(byte[] certData)            Instantiates an X509Certificate object, and initializes it with the specified byte array. |
| static [X509Certificate](http://docs.google.com/javax/security/cert/X509Certificate.html) | [**getInstance**](http://docs.google.com/javax/security/cert/X509Certificate.html#getInstance(java.io.InputStream))([InputStream](http://docs.google.com/java/io/InputStream.html) inStream)            Instantiates an X509Certificate object, and initializes it with the data read from the input stream inStream. |
| abstract  [Principal](http://docs.google.com/java/security/Principal.html) | [**getIssuerDN**](http://docs.google.com/javax/security/cert/X509Certificate.html#getIssuerDN())()            Gets the issuer (issuer distinguished name) value from the certificate. |
| abstract  [Date](http://docs.google.com/java/util/Date.html) | [**getNotAfter**](http://docs.google.com/javax/security/cert/X509Certificate.html#getNotAfter())()            Gets the notAfter date from the validity period of the certificate. |
| abstract  [Date](http://docs.google.com/java/util/Date.html) | [**getNotBefore**](http://docs.google.com/javax/security/cert/X509Certificate.html#getNotBefore())()            Gets the notBefore date from the validity period of the certificate. |
| abstract  [BigInteger](http://docs.google.com/java/math/BigInteger.html) | [**getSerialNumber**](http://docs.google.com/javax/security/cert/X509Certificate.html#getSerialNumber())()            Gets the serialNumber value from the certificate. |
| abstract  [String](http://docs.google.com/java/lang/String.html) | [**getSigAlgName**](http://docs.google.com/javax/security/cert/X509Certificate.html#getSigAlgName())()            Gets the signature algorithm name for the certificate signature algorithm. |
| abstract  [String](http://docs.google.com/java/lang/String.html) | [**getSigAlgOID**](http://docs.google.com/javax/security/cert/X509Certificate.html#getSigAlgOID())()            Gets the signature algorithm OID string from the certificate. |
| abstract  byte[] | [**getSigAlgParams**](http://docs.google.com/javax/security/cert/X509Certificate.html#getSigAlgParams())()            Gets the DER-encoded signature algorithm parameters from this certificate's signature algorithm. |
| abstract  [Principal](http://docs.google.com/java/security/Principal.html) | [**getSubjectDN**](http://docs.google.com/javax/security/cert/X509Certificate.html#getSubjectDN())()            Gets the subject (subject distinguished name) value from the certificate. |
| abstract  int | [**getVersion**](http://docs.google.com/javax/security/cert/X509Certificate.html#getVersion())()            Gets the version (version number) value from the certificate. |

| **Methods inherited from class javax.security.cert.**[**Certificate**](http://docs.google.com/javax/security/cert/Certificate.html) |
| --- |
| [equals](http://docs.google.com/javax/security/cert/Certificate.html#equals(java.lang.Object)), [getEncoded](http://docs.google.com/javax/security/cert/Certificate.html#getEncoded()), [getPublicKey](http://docs.google.com/javax/security/cert/Certificate.html#getPublicKey()), [hashCode](http://docs.google.com/javax/security/cert/Certificate.html#hashCode()), [toString](http://docs.google.com/javax/security/cert/Certificate.html#toString()), [verify](http://docs.google.com/javax/security/cert/Certificate.html#verify(java.security.PublicKey)), [verify](http://docs.google.com/javax/security/cert/Certificate.html#verify(java.security.PublicKey,%20java.lang.String)) |

| **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** |
| --- |

### X509Certificate

public **X509Certificate**()

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

### getInstance

public static final [X509Certificate](http://docs.google.com/javax/security/cert/X509Certificate.html) **getInstance**([InputStream](http://docs.google.com/java/io/InputStream.html) inStream)  
 throws [CertificateException](http://docs.google.com/javax/security/cert/CertificateException.html)

Instantiates an X509Certificate object, and initializes it with the data read from the input stream inStream. The implementation (X509Certificate is an abstract class) is provided by the class specified as the value of the cert.provider.x509v1 property in the security properties file.

Note: Only one DER-encoded certificate is expected to be in the input stream. Also, all X509Certificate subclasses must provide a constructor of the form:

public <subClass>(InputStream inStream) ...

**Parameters:**inStream - an input stream with the data to be read to initialize the certificate. **Returns:**an X509Certificate object initialized with the data from the input stream. **Throws:** [CertificateException](http://docs.google.com/javax/security/cert/CertificateException.html) - if a class initialization or certificate parsing error occurs.

### getInstance

public static final [X509Certificate](http://docs.google.com/javax/security/cert/X509Certificate.html) **getInstance**(byte[] certData)  
 throws [CertificateException](http://docs.google.com/javax/security/cert/CertificateException.html)

Instantiates an X509Certificate object, and initializes it with the specified byte array. The implementation (X509Certificate is an abstract class) is provided by the class specified as the value of the cert.provider.x509v1 property in the security properties file.

Note: All X509Certificate subclasses must provide a constructor of the form:

public <subClass>(InputStream inStream) ...

**Parameters:**certData - a byte array containing the DER-encoded certificate. **Returns:**an X509Certificate object initialized with the data from certData. **Throws:** [CertificateException](http://docs.google.com/javax/security/cert/CertificateException.html) - if a class initialization or certificate parsing error occurs.

### checkValidity

public abstract void **checkValidity**()  
 throws [CertificateExpiredException](http://docs.google.com/javax/security/cert/CertificateExpiredException.html),  
 [CertificateNotYetValidException](http://docs.google.com/javax/security/cert/CertificateNotYetValidException.html)

Checks that the certificate is currently valid. It is if the current date and time are within the validity period given in the certificate.

The validity period consists of two date/time values: the first and last dates (and times) on which the certificate is valid. It is defined in ASN.1 as:

validity Validity

Validity ::= SEQUENCE {  
 notBefore CertificateValidityDate,  
 notAfter CertificateValidityDate }

CertificateValidityDate ::= CHOICE {  
 utcTime UTCTime,  
 generalTime GeneralizedTime }

**Throws:** [CertificateExpiredException](http://docs.google.com/javax/security/cert/CertificateExpiredException.html) - if the certificate has expired. [CertificateNotYetValidException](http://docs.google.com/javax/security/cert/CertificateNotYetValidException.html) - if the certificate is not yet valid.

### checkValidity

public abstract void **checkValidity**([Date](http://docs.google.com/java/util/Date.html) date)  
 throws [CertificateExpiredException](http://docs.google.com/javax/security/cert/CertificateExpiredException.html),  
 [CertificateNotYetValidException](http://docs.google.com/javax/security/cert/CertificateNotYetValidException.html)

Checks that the specified date is within the certificate's validity period. In other words, this determines whether the certificate would be valid at the specified date/time.

**Parameters:**date - the Date to check against to see if this certificate is valid at that date/time. **Throws:** [CertificateExpiredException](http://docs.google.com/javax/security/cert/CertificateExpiredException.html) - if the certificate has expired with respect to the date supplied. [CertificateNotYetValidException](http://docs.google.com/javax/security/cert/CertificateNotYetValidException.html) - if the certificate is not yet valid with respect to the date supplied.**See Also:**[checkValidity()](http://docs.google.com/javax/security/cert/X509Certificate.html#checkValidity())

### getVersion

public abstract int **getVersion**()

Gets the version (version number) value from the certificate. The ASN.1 definition for this is:

version [0] EXPLICIT Version DEFAULT v1

Version ::= INTEGER { v1(0), v2(1), v3(2) }

**Returns:**the version number from the ASN.1 encoding, i.e. 0, 1 or 2.

### getSerialNumber

public abstract [BigInteger](http://docs.google.com/java/math/BigInteger.html) **getSerialNumber**()

Gets the serialNumber value from the certificate. The serial number is an integer assigned by the certification authority to each certificate. It must be unique for each certificate issued by a given CA (i.e., the issuer name and serial number identify a unique certificate). The ASN.1 definition for this is:

serialNumber CertificateSerialNumber

CertificateSerialNumber ::= INTEGER

**Returns:**the serial number.

### getIssuerDN

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

Gets the issuer (issuer distinguished name) value from the certificate. The issuer name identifies the entity that signed (and issued) the certificate.

The issuer name field contains an X.500 distinguished name (DN). The ASN.1 definition for this is:

issuer Name

Name ::= CHOICE { RDNSequence }  
 RDNSequence ::= SEQUENCE OF RelativeDistinguishedName  
 RelativeDistinguishedName ::=  
 SET OF AttributeValueAssertion  
  
 AttributeValueAssertion ::= SEQUENCE {  
 AttributeType,  
 AttributeValue }  
 AttributeType ::= OBJECT IDENTIFIER  
 AttributeValue ::= ANY

The Name describes a hierarchical name composed of attributes, such as country name, and corresponding values, such as US. The type of the AttributeValue component is determined by the AttributeType; in general it will be a directoryString. A directoryString is usually one of PrintableString, TeletexString or UniversalString.

**Returns:**a Principal whose name is the issuer distinguished name.

### getSubjectDN

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

Gets the subject (subject distinguished name) value from the certificate. The ASN.1 definition for this is:

subject Name

See [getIssuerDN](#qsh70q) for Name and other relevant definitions.

**Returns:**a Principal whose name is the subject name.**See Also:**[getIssuerDN()](http://docs.google.com/javax/security/cert/X509Certificate.html#getIssuerDN())

### getNotBefore

public abstract [Date](http://docs.google.com/java/util/Date.html) **getNotBefore**()

Gets the notBefore date from the validity period of the certificate. The relevant ASN.1 definitions are:

validity Validity

Validity ::= SEQUENCE {  
 notBefore CertificateValidityDate,  
 notAfter CertificateValidityDate }

CertificateValidityDate ::= CHOICE {  
 utcTime UTCTime,  
 generalTime GeneralizedTime }

**Returns:**the start date of the validity period.**See Also:**[checkValidity()](http://docs.google.com/javax/security/cert/X509Certificate.html#checkValidity())

### getNotAfter

public abstract [Date](http://docs.google.com/java/util/Date.html) **getNotAfter**()

Gets the notAfter date from the validity period of the certificate. See [getNotBefore](#3as4poj) for relevant ASN.1 definitions.

**Returns:**the end date of the validity period.**See Also:**[checkValidity()](http://docs.google.com/javax/security/cert/X509Certificate.html#checkValidity())

### getSigAlgName

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

Gets the signature algorithm name for the certificate signature algorithm. An example is the string "SHA-1/DSA". The ASN.1 definition for this is:

signatureAlgorithm AlgorithmIdentifier

AlgorithmIdentifier ::= SEQUENCE {  
 algorithm OBJECT IDENTIFIER,  
 parameters ANY DEFINED BY algorithm OPTIONAL }  
 -- contains a value of the type  
 -- registered for use with the  
 -- algorithm object identifier value

The algorithm name is determined from the algorithm OID string.

**Returns:**the signature algorithm name.

### getSigAlgOID

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

Gets the signature algorithm OID string from the certificate. An OID is represented by a set of positive whole numbers separated by periods. For example, the string "1.2.840.10040.4.3" identifies the SHA-1 with DSA signature algorithm, as per the PKIX part I.

See [getSigAlgName](#1pxezwc) for relevant ASN.1 definitions.

**Returns:**the signature algorithm OID string.

### getSigAlgParams

public abstract byte[] **getSigAlgParams**()

Gets the DER-encoded signature algorithm parameters from this certificate's signature algorithm. In most cases, the signature algorithm parameters are null; the parameters are usually supplied with the certificate's public key.

See [getSigAlgName](#1pxezwc) for relevant ASN.1 definitions.

**Returns:**the DER-encoded signature algorithm parameters, or null if no parameters are present.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/X509Certificate.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/security/cert/CertificateParsingException.html)   NEXT CLASS | [**FRAMES**](http://docs.google.com/index.html?javax/security/cert/X509Certificate.html)    [**NO FRAMES**](http://docs.google.com/X509Certificate.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).