| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Signature.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/SecurityPermission.html)   [**NEXT CLASS**](http://docs.google.com/java/security/SignatureException.html) | [**FRAMES**](http://docs.google.com/index.html?java/security/Signature.html)    [**NO FRAMES**](http://docs.google.com/Signature.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#tyjcwt) | [METHOD](#3dy6vkm) | DETAIL: [FIELD](#2s8eyo1) | [CONSTR](#35nkun2) | [METHOD](#44sinio) |

## **java.security**

Class Signature

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

public abstract class **Signature**extends [SignatureSpi](http://docs.google.com/java/security/SignatureSpi.html)

This Signature class is used to provide applications the functionality of a digital signature algorithm. Digital signatures are used for authentication and integrity assurance of digital data.

The signature algorithm can be, among others, the NIST standard DSA, using DSA and SHA-1. The DSA algorithm using the SHA-1 message digest algorithm can be specified as SHA1withDSA. In the case of RSA, there are multiple choices for the message digest algorithm, so the signing algorithm could be specified as, for example, MD2withRSA, MD5withRSA, or SHA1withRSA. The algorithm name must be specified, as there is no default.

A Signature object can be used to generate and verify digital signatures.

There are three phases to the use of a Signature object for either signing data or verifying a signature:

1. Initialization, with either
   * a public key, which initializes the signature for verification (see [initVerify](http://docs.google.com/java/security/Signature.html#initVerify(java.security.PublicKey))), or
   * a private key (and optionally a Secure Random Number Generator), which initializes the signature for signing (see [initSign(PrivateKey)](http://docs.google.com/java/security/Signature.html#initSign(java.security.PrivateKey)) and [initSign(PrivateKey, SecureRandom)](http://docs.google.com/java/security/Signature.html#initSign(java.security.PrivateKey,%20java.security.SecureRandom))).
2. Updating  
     
   Depending on the type of initialization, this will update the bytes to be signed or verified. See the [update](http://docs.google.com/java/security/Signature.html#update(byte)) methods.
3. Signing or Verifying a signature on all updated bytes. See the [sign](http://docs.google.com/java/security/Signature.html#sign()) methods and the [verify](http://docs.google.com/java/security/Signature.html#verify(byte%5B%5D)) method.

Note that this class is abstract and extends from SignatureSpi for historical reasons. Application developers should only take notice of the methods defined in this Signature class; all the methods in the superclass are intended for cryptographic service providers who wish to supply their own implementations of digital signature algorithms.

| **Field Summary** | |
| --- | --- |
| protected static int | [**SIGN**](http://docs.google.com/java/security/Signature.html#SIGN)            Possible [state](http://docs.google.com/java/security/Signature.html#state) value, signifying that this signature object has been initialized for signing. |
| protected  int | [**state**](http://docs.google.com/java/security/Signature.html#state)            Current state of this signature object. |
| protected static int | [**UNINITIALIZED**](http://docs.google.com/java/security/Signature.html#UNINITIALIZED)            Possible [state](http://docs.google.com/java/security/Signature.html#state) value, signifying that this signature object has not yet been initialized. |
| protected static int | [**VERIFY**](http://docs.google.com/java/security/Signature.html#VERIFY)            Possible [state](http://docs.google.com/java/security/Signature.html#state) value, signifying that this signature object has been initialized for verification. |

| **Fields inherited from class java.security.**[**SignatureSpi**](http://docs.google.com/java/security/SignatureSpi.html) |
| --- |
| [appRandom](http://docs.google.com/java/security/SignatureSpi.html#appRandom) |

| **Constructor Summary** | |
| --- | --- |
| protected | [**Signature**](http://docs.google.com/java/security/Signature.html#Signature(java.lang.String))([String](http://docs.google.com/java/lang/String.html) algorithm)            Creates a Signature object for the specified algorithm. |

| **Method Summary** | |
| --- | --- |
| [Object](http://docs.google.com/java/lang/Object.html) | [**clone**](http://docs.google.com/java/security/Signature.html#clone())()            Returns a clone if the implementation is cloneable. |
| [String](http://docs.google.com/java/lang/String.html) | [**getAlgorithm**](http://docs.google.com/java/security/Signature.html#getAlgorithm())()            Returns the name of the algorithm for this signature object. |
| static [Signature](http://docs.google.com/java/security/Signature.html) | [**getInstance**](http://docs.google.com/java/security/Signature.html#getInstance(java.lang.String))([String](http://docs.google.com/java/lang/String.html) algorithm)            Returns a Signature object that implements the specified signature algorithm. |
| static [Signature](http://docs.google.com/java/security/Signature.html) | [**getInstance**](http://docs.google.com/java/security/Signature.html#getInstance(java.lang.String,%20java.security.Provider))([String](http://docs.google.com/java/lang/String.html) algorithm, [Provider](http://docs.google.com/java/security/Provider.html) provider)            Returns a Signature object that implements the specified signature algorithm. |
| static [Signature](http://docs.google.com/java/security/Signature.html) | [**getInstance**](http://docs.google.com/java/security/Signature.html#getInstance(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) algorithm, [String](http://docs.google.com/java/lang/String.html) provider)            Returns a Signature object that implements the specified signature algorithm. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getParameter**](http://docs.google.com/java/security/Signature.html#getParameter(java.lang.String))([String](http://docs.google.com/java/lang/String.html) param)  **Deprecated.** |
| [AlgorithmParameters](http://docs.google.com/java/security/AlgorithmParameters.html) | [**getParameters**](http://docs.google.com/java/security/Signature.html#getParameters())()            Returns the parameters used with this signature object. |
| [Provider](http://docs.google.com/java/security/Provider.html) | [**getProvider**](http://docs.google.com/java/security/Signature.html#getProvider())()            Returns the provider of this signature object. |
| void | [**initSign**](http://docs.google.com/java/security/Signature.html#initSign(java.security.PrivateKey))([PrivateKey](http://docs.google.com/java/security/PrivateKey.html) privateKey)            Initialize this object for signing. |
| void | [**initSign**](http://docs.google.com/java/security/Signature.html#initSign(java.security.PrivateKey,%20java.security.SecureRandom))([PrivateKey](http://docs.google.com/java/security/PrivateKey.html) privateKey, [SecureRandom](http://docs.google.com/java/security/SecureRandom.html) random)            Initialize this object for signing. |
| void | [**initVerify**](http://docs.google.com/java/security/Signature.html#initVerify(java.security.cert.Certificate))([Certificate](http://docs.google.com/java/security/cert/Certificate.html) certificate)            Initializes this object for verification, using the public key from the given certificate. |
| void | [**initVerify**](http://docs.google.com/java/security/Signature.html#initVerify(java.security.PublicKey))([PublicKey](http://docs.google.com/java/security/PublicKey.html) publicKey)            Initializes this object for verification. |
| void | [**setParameter**](http://docs.google.com/java/security/Signature.html#setParameter(java.security.spec.AlgorithmParameterSpec))([AlgorithmParameterSpec](http://docs.google.com/java/security/spec/AlgorithmParameterSpec.html) params)            Initializes this signature engine with the specified parameter set. |
| void | [**setParameter**](http://docs.google.com/java/security/Signature.html#setParameter(java.lang.String,%20java.lang.Object))([String](http://docs.google.com/java/lang/String.html) param, [Object](http://docs.google.com/java/lang/Object.html) value)  **Deprecated.** *Use* [*setParameter*](http://docs.google.com/java/security/Signature.html#setParameter(java.security.spec.AlgorithmParameterSpec))*.* |
| byte[] | [**sign**](http://docs.google.com/java/security/Signature.html#sign())()            Returns the signature bytes of all the data updated. |
| int | [**sign**](http://docs.google.com/java/security/Signature.html#sign(byte%5B%5D,%20int,%20int))(byte[] outbuf, int offset, int len)            Finishes the signature operation and stores the resulting signature bytes in the provided buffer outbuf, starting at offset. |
| [String](http://docs.google.com/java/lang/String.html) | [**toString**](http://docs.google.com/java/security/Signature.html#toString())()            Returns a string representation of this signature object, providing information that includes the state of the object and the name of the algorithm used. |
| void | [**update**](http://docs.google.com/java/security/Signature.html#update(byte))(byte b)            Updates the data to be signed or verified by a byte. |
| void | [**update**](http://docs.google.com/java/security/Signature.html#update(byte%5B%5D))(byte[] data)            Updates the data to be signed or verified, using the specified array of bytes. |
| void | [**update**](http://docs.google.com/java/security/Signature.html#update(byte%5B%5D,%20int,%20int))(byte[] data, int off, int len)            Updates the data to be signed or verified, using the specified array of bytes, starting at the specified offset. |
| void | [**update**](http://docs.google.com/java/security/Signature.html#update(java.nio.ByteBuffer))([ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html) data)            Updates the data to be signed or verified using the specified ByteBuffer. |
| boolean | [**verify**](http://docs.google.com/java/security/Signature.html#verify(byte%5B%5D))(byte[] signature)            Verifies the passed-in signature. |
| boolean | [**verify**](http://docs.google.com/java/security/Signature.html#verify(byte%5B%5D,%20int,%20int))(byte[] signature, int offset, int length)            Verifies the passed-in signature in the specified array of bytes, starting at the specified offset. |

| **Methods inherited from class java.security.**[**SignatureSpi**](http://docs.google.com/java/security/SignatureSpi.html) |
| --- |
| [engineGetParameter](http://docs.google.com/java/security/SignatureSpi.html#engineGetParameter(java.lang.String)), [engineGetParameters](http://docs.google.com/java/security/SignatureSpi.html#engineGetParameters()), [engineInitSign](http://docs.google.com/java/security/SignatureSpi.html#engineInitSign(java.security.PrivateKey)), [engineInitSign](http://docs.google.com/java/security/SignatureSpi.html#engineInitSign(java.security.PrivateKey,%20java.security.SecureRandom)), [engineInitVerify](http://docs.google.com/java/security/SignatureSpi.html#engineInitVerify(java.security.PublicKey)), [engineSetParameter](http://docs.google.com/java/security/SignatureSpi.html#engineSetParameter(java.security.spec.AlgorithmParameterSpec)), [engineSetParameter](http://docs.google.com/java/security/SignatureSpi.html#engineSetParameter(java.lang.String,%20java.lang.Object)), [engineSign](http://docs.google.com/java/security/SignatureSpi.html#engineSign()), [engineSign](http://docs.google.com/java/security/SignatureSpi.html#engineSign(byte%5B%5D,%20int,%20int)), [engineUpdate](http://docs.google.com/java/security/SignatureSpi.html#engineUpdate(byte)), [engineUpdate](http://docs.google.com/java/security/SignatureSpi.html#engineUpdate(byte%5B%5D,%20int,%20int)), [engineUpdate](http://docs.google.com/java/security/SignatureSpi.html#engineUpdate(java.nio.ByteBuffer)), [engineVerify](http://docs.google.com/java/security/SignatureSpi.html#engineVerify(byte%5B%5D)), [engineVerify](http://docs.google.com/java/security/SignatureSpi.html#engineVerify(byte%5B%5D,%20int,%20int)) |

| **Methods inherited from class java.lang.**[**Object**](http://docs.google.com/java/lang/Object.html) |
| --- |
| [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()), [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)) |

| **Field Detail** |
| --- |

### UNINITIALIZED

protected static final int **UNINITIALIZED**

Possible [state](http://docs.google.com/java/security/Signature.html#state) value, signifying that this signature object has not yet been initialized.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.security.Signature.UNINITIALIZED)

### SIGN

protected static final int **SIGN**

Possible [state](http://docs.google.com/java/security/Signature.html#state) value, signifying that this signature object has been initialized for signing.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.security.Signature.SIGN)

### VERIFY

protected static final int **VERIFY**

Possible [state](http://docs.google.com/java/security/Signature.html#state) value, signifying that this signature object has been initialized for verification.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.security.Signature.VERIFY)

### state

protected int **state**

Current state of this signature object.

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

### Signature

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

Creates a Signature object for the specified algorithm.

**Parameters:**algorithm - the standard string name of the algorithm. 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 algorithm names.

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

### getInstance

public static [Signature](http://docs.google.com/java/security/Signature.html) **getInstance**([String](http://docs.google.com/java/lang/String.html) algorithm)  
 throws [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html)

Returns a Signature object that implements the specified signature algorithm.

This method traverses the list of registered security Providers, starting with the most preferred Provider. A new Signature object encapsulating the SignatureSpi implementation from the first Provider that supports the specified algorithm 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:**algorithm - the standard name of the algorithm requested. 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 algorithm names. **Returns:**the new Signature object. **Throws:** [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if no Provider supports a Signature implementation for the specified algorithm.**See Also:**[Provider](http://docs.google.com/java/security/Provider.html)

### getInstance

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

Returns a Signature object that implements the specified signature algorithm.

A new Signature object encapsulating the SignatureSpi 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:**algorithm - the name of the algorithm requested. 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 algorithm names.provider - the name of the provider. **Returns:**the new Signature object. **Throws:** [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if a SignatureSpi implementation for the specified algorithm 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 [Signature](http://docs.google.com/java/security/Signature.html) **getInstance**([String](http://docs.google.com/java/lang/String.html) algorithm,  
 [Provider](http://docs.google.com/java/security/Provider.html) provider)  
 throws [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html)

Returns a Signature object that implements the specified signature algorithm.

A new Signature object encapsulating the SignatureSpi 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:**algorithm - the name of the algorithm requested. 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 algorithm names.provider - the provider. **Returns:**the new Signature object. **Throws:** [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if a SignatureSpi implementation for the specified algorithm is not available from the specified Provider object. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the provider is null.**Since:** 1.4 **See Also:**[Provider](http://docs.google.com/java/security/Provider.html)

### getProvider

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

Returns the provider of this signature object.

**Returns:**the provider of this signature object

### initVerify

public final void **initVerify**([PublicKey](http://docs.google.com/java/security/PublicKey.html) publicKey)  
 throws [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html)

Initializes this object for verification. If this method is called again with a different argument, it negates the effect of this call.

**Parameters:**publicKey - the public key of the identity whose signature is going to be verified. **Throws:** [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html) - if the key is invalid.

### initVerify

public final void **initVerify**([Certificate](http://docs.google.com/java/security/cert/Certificate.html) certificate)  
 throws [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html)

Initializes this object for verification, using the public key from the given certificate.

If the certificate is of type X.509 and has a *key usage* extension field marked as critical, and the value of the *key usage* extension field implies that the public key in the certificate and its corresponding private key are not supposed to be used for digital signatures, an InvalidKeyException is thrown.

**Parameters:**certificate - the certificate of the identity whose signature is going to be verified. **Throws:** [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html) - if the public key in the certificate is not encoded properly or does not include required parameter information or cannot be used for digital signature purposes.**Since:** 1.3

### initSign

public final void **initSign**([PrivateKey](http://docs.google.com/java/security/PrivateKey.html) privateKey)  
 throws [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html)

Initialize this object for signing. If this method is called again with a different argument, it negates the effect of this call.

**Parameters:**privateKey - the private key of the identity whose signature is going to be generated. **Throws:** [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html) - if the key is invalid.

### initSign

public final void **initSign**([PrivateKey](http://docs.google.com/java/security/PrivateKey.html) privateKey,  
 [SecureRandom](http://docs.google.com/java/security/SecureRandom.html) random)  
 throws [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html)

Initialize this object for signing. If this method is called again with a different argument, it negates the effect of this call.

**Parameters:**privateKey - the private key of the identity whose signature is going to be generated.random - the source of randomness for this signature. **Throws:** [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html) - if the key is invalid.

### sign

public final byte[] **sign**()  
 throws [SignatureException](http://docs.google.com/java/security/SignatureException.html)

Returns the signature bytes of all the data updated. The format of the signature depends on the underlying signature scheme.

A call to this method resets this signature object to the state it was in when previously initialized for signing via a call to initSign(PrivateKey). That is, the object is reset and available to generate another signature from the same signer, if desired, via new calls to update and sign.

**Returns:**the signature bytes of the signing operation's result. **Throws:** [SignatureException](http://docs.google.com/java/security/SignatureException.html) - if this signature object is not initialized properly or if this signature algorithm is unable to process the input data provided.

### sign

public final int **sign**(byte[] outbuf,  
 int offset,  
 int len)  
 throws [SignatureException](http://docs.google.com/java/security/SignatureException.html)

Finishes the signature operation and stores the resulting signature bytes in the provided buffer outbuf, starting at offset. The format of the signature depends on the underlying signature scheme.

This signature object is reset to its initial state (the state it was in after a call to one of the initSign methods) and can be reused to generate further signatures with the same private key.

**Parameters:**outbuf - buffer for the signature result.offset - offset into outbuf where the signature is stored.len - number of bytes within outbuf allotted for the signature. **Returns:**the number of bytes placed into outbuf. **Throws:** [SignatureException](http://docs.google.com/java/security/SignatureException.html) - if this signature object is not initialized properly, if this signature algorithm is unable to process the input data provided, or if len is less than the actual signature length.**Since:** 1.2

### verify

public final boolean **verify**(byte[] signature)  
 throws [SignatureException](http://docs.google.com/java/security/SignatureException.html)

Verifies the passed-in signature.

A call to this method resets this signature object to the state it was in when previously initialized for verification via a call to initVerify(PublicKey). That is, the object is reset and available to verify another signature from the identity whose public key was specified in the call to initVerify.

**Parameters:**signature - the signature bytes to be verified. **Returns:**true if the signature was verified, false if not. **Throws:** [SignatureException](http://docs.google.com/java/security/SignatureException.html) - if this signature object is not initialized properly, the passed-in signature is improperly encoded or of the wrong type, if this signature algorithm is unable to process the input data provided, etc.

### verify

public final boolean **verify**(byte[] signature,  
 int offset,  
 int length)  
 throws [SignatureException](http://docs.google.com/java/security/SignatureException.html)

Verifies the passed-in signature in the specified array of bytes, starting at the specified offset.

A call to this method resets this signature object to the state it was in when previously initialized for verification via a call to initVerify(PublicKey). That is, the object is reset and available to verify another signature from the identity whose public key was specified in the call to initVerify.

**Parameters:**signature - the signature bytes to be verified.offset - the offset to start from in the array of bytes.length - the number of bytes to use, starting at offset. **Returns:**true if the signature was verified, false if not. **Throws:** [SignatureException](http://docs.google.com/java/security/SignatureException.html) - if this signature object is not initialized properly, the passed-in signature is improperly encoded or of the wrong type, if this signature algorithm is unable to process the input data provided, etc. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the signature byte array is null, or the offset or length is less than 0, or the sum of the offset and length is greater than the length of the signature byte array.**Since:** 1.4

### update

public final void **update**(byte b)  
 throws [SignatureException](http://docs.google.com/java/security/SignatureException.html)

Updates the data to be signed or verified by a byte.

**Parameters:**b - the byte to use for the update. **Throws:** [SignatureException](http://docs.google.com/java/security/SignatureException.html) - if this signature object is not initialized properly.

### update

public final void **update**(byte[] data)  
 throws [SignatureException](http://docs.google.com/java/security/SignatureException.html)

Updates the data to be signed or verified, using the specified array of bytes.

**Parameters:**data - the byte array to use for the update. **Throws:** [SignatureException](http://docs.google.com/java/security/SignatureException.html) - if this signature object is not initialized properly.

### update

public final void **update**(byte[] data,  
 int off,  
 int len)  
 throws [SignatureException](http://docs.google.com/java/security/SignatureException.html)

Updates the data to be signed or verified, using the specified array of bytes, starting at the specified offset.

**Parameters:**data - the array of bytes.off - the offset to start from in the array of bytes.len - the number of bytes to use, starting at offset. **Throws:** [SignatureException](http://docs.google.com/java/security/SignatureException.html) - if this signature object is not initialized properly.

### update

public final void **update**([ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html) data)  
 throws [SignatureException](http://docs.google.com/java/security/SignatureException.html)

Updates the data to be signed or verified using the specified ByteBuffer. Processes the data.remaining() bytes starting at at data.position(). Upon return, the buffer's position will be equal to its limit; its limit will not have changed.

**Parameters:**data - the ByteBuffer **Throws:** [SignatureException](http://docs.google.com/java/security/SignatureException.html) - if this signature object is not initialized properly.**Since:** 1.5

### getAlgorithm

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

Returns the name of the algorithm for this signature object.

**Returns:**the name of the algorithm for this signature object.

### toString

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

Returns a string representation of this signature object, providing information that includes the state of the object and the name of the algorithm used.

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

### setParameter

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public final void **setParameter**([String](http://docs.google.com/java/lang/String.html) param,  
 [Object](http://docs.google.com/java/lang/Object.html) value)  
 throws [InvalidParameterException](http://docs.google.com/java/security/InvalidParameterException.html)

**Deprecated.** *Use* [*setParameter*](http://docs.google.com/java/security/Signature.html#setParameter(java.security.spec.AlgorithmParameterSpec))*.*

Sets the specified algorithm parameter to the specified value. This method supplies a general-purpose mechanism through which it is possible to set the various parameters of this object. A parameter may be any settable parameter for the algorithm, such as a parameter size, or a source of random bits for signature generation (if appropriate), or an indication of whether or not to perform a specific but optional computation. A uniform algorithm-specific naming scheme for each parameter is desirable but left unspecified at this time.

**Parameters:**param - the string identifier of the parameter.value - the parameter value. **Throws:** [InvalidParameterException](http://docs.google.com/java/security/InvalidParameterException.html) - if param is an invalid parameter for this signature algorithm engine, the parameter is already set and cannot be set again, a security exception occurs, and so on.**See Also:**[getParameter(java.lang.String)](http://docs.google.com/java/security/Signature.html#getParameter(java.lang.String))

### setParameter

public final void **setParameter**([AlgorithmParameterSpec](http://docs.google.com/java/security/spec/AlgorithmParameterSpec.html) params)  
 throws [InvalidAlgorithmParameterException](http://docs.google.com/java/security/InvalidAlgorithmParameterException.html)

Initializes this signature engine with the specified parameter set.

**Parameters:**params - the parameters **Throws:** [InvalidAlgorithmParameterException](http://docs.google.com/java/security/InvalidAlgorithmParameterException.html) - if the given parameters are inappropriate for this signature engine**See Also:**[getParameters()](http://docs.google.com/java/security/Signature.html#getParameters())

### getParameters

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

Returns the parameters used with this signature object.

The returned parameters may be the same that were used to initialize this signature, or may contain a combination of default and randomly generated parameter values used by the underlying signature implementation if this signature requires algorithm parameters but was not initialized with any.

**Returns:**the parameters used with this signature, or null if this signature does not use any parameters.**Since:** 1.4 **See Also:**[setParameter(AlgorithmParameterSpec)](http://docs.google.com/java/security/Signature.html#setParameter(java.security.spec.AlgorithmParameterSpec))

### getParameter

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public final [Object](http://docs.google.com/java/lang/Object.html) **getParameter**([String](http://docs.google.com/java/lang/String.html) param)  
 throws [InvalidParameterException](http://docs.google.com/java/security/InvalidParameterException.html)

**Deprecated.**

Gets the value of the specified algorithm parameter. This method supplies a general-purpose mechanism through which it is possible to get the various parameters of this object. A parameter may be any settable parameter for the algorithm, such as a parameter size, or a source of random bits for signature generation (if appropriate), or an indication of whether or not to perform a specific but optional computation. A uniform algorithm-specific naming scheme for each parameter is desirable but left unspecified at this time.

**Parameters:**param - the string name of the parameter. **Returns:**the object that represents the parameter value, or null if there is none. **Throws:** [InvalidParameterException](http://docs.google.com/java/security/InvalidParameterException.html) - if param is an invalid parameter for this engine, or another exception occurs while trying to get this parameter.**See Also:**[setParameter(String, Object)](http://docs.google.com/java/security/Signature.html#setParameter(java.lang.String,%20java.lang.Object))

### clone

public [Object](http://docs.google.com/java/lang/Object.html) **clone**()  
 throws [CloneNotSupportedException](http://docs.google.com/java/lang/CloneNotSupportedException.html)

Returns a clone if the implementation is cloneable.

**Overrides:**[clone](http://docs.google.com/java/security/SignatureSpi.html#clone()) in class [SignatureSpi](http://docs.google.com/java/security/SignatureSpi.html) **Returns:**a clone if the implementation is cloneable. **Throws:** [CloneNotSupportedException](http://docs.google.com/java/lang/CloneNotSupportedException.html) - if this is called on an implementation that does not support Cloneable.**See Also:**[Cloneable](http://docs.google.com/java/lang/Cloneable.html)

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Signature.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/SecurityPermission.html)   [**NEXT CLASS**](http://docs.google.com/java/security/SignatureException.html) | [**FRAMES**](http://docs.google.com/index.html?java/security/Signature.html)    [**NO FRAMES**](http://docs.google.com/Signature.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#tyjcwt) | [METHOD](#3dy6vkm) | DETAIL: [FIELD](#2s8eyo1) | [CONSTR](#35nkun2) | [METHOD](#44sinio) |

[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).