| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/CipherSpi.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/crypto/CipherOutputStream.html)   [**NEXT CLASS**](http://docs.google.com/javax/crypto/EncryptedPrivateKeyInfo.html) | [**FRAMES**](http://docs.google.com/index.html?javax/crypto/CipherSpi.html)    [**NO FRAMES**](http://docs.google.com/CipherSpi.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#4d34og8) |

## **javax.crypto**

Class CipherSpi

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 **javax.crypto.CipherSpi**

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

This class defines the *Service Provider Interface* (**SPI**) for the Cipher class. All the abstract methods in this class must be implemented by each cryptographic service provider who wishes to supply the implementation of a particular cipher algorithm.

In order to create an instance of Cipher, which encapsulates an instance of this CipherSpi class, an application calls one of the [getInstance](http://docs.google.com/javax/crypto/Cipher.html#getInstance(java.lang.String)) factory methods of the [Cipher](http://docs.google.com/javax/crypto/Cipher.html) engine class and specifies the requested *transformation*. Optionally, the application may also specify the name of a provider.

A *transformation* is a string that describes the operation (or set of operations) to be performed on the given input, to produce some output. A transformation always includes the name of a cryptographic algorithm (e.g., *DES*), and may be followed by a feedback mode and padding scheme.

A transformation is of the form:

* "*algorithm/mode/padding*" or
* "*algorithm*"

(in the latter case, provider-specific default values for the mode and padding scheme are used). For example, the following is a valid transformation:

Cipher c = Cipher.getInstance("*DES/CBC/PKCS5Padding*");

A provider may supply a separate class for each combination of *algorithm/mode/padding*, or may decide to provide more generic classes representing sub-transformations corresponding to *algorithm* or *algorithm/mode* or *algorithm//padding* (note the double slashes), in which case the requested mode and/or padding are set automatically by the getInstance methods of Cipher, which invoke the [engineSetMode](http://docs.google.com/javax/crypto/CipherSpi.html#engineSetMode(java.lang.String)) and [engineSetPadding](http://docs.google.com/javax/crypto/CipherSpi.html#engineSetPadding(java.lang.String)) methods of the provider's subclass of CipherSpi.

A Cipher property in a provider master class may have one of the following formats:

* // provider's subclass of "CipherSpi" implements "algName" with  
   // pluggable mode and padding  
   Cipher.*algName*
* // provider's subclass of "CipherSpi" implements "algName" in the  
   // specified "mode", with pluggable padding  
   Cipher.*algName/mode*
* // provider's subclass of "CipherSpi" implements "algName" with the  
   // specified "padding", with pluggable mode  
   Cipher.*algName//padding*
* // provider's subclass of "CipherSpi" implements "algName" with the  
   // specified "mode" and "padding"  
   Cipher.*algName/mode/padding*

For example, a provider may supply a subclass of CipherSpi that implements *DES/ECB/PKCS5Padding*, one that implements *DES/CBC/PKCS5Padding*, one that implements *DES/CFB/PKCS5Padding*, and yet another one that implements *DES/OFB/PKCS5Padding*. That provider would have the following Cipher properties in its master class:

* Cipher.*DES/ECB/PKCS5Padding*
* Cipher.*DES/CBC/PKCS5Padding*
* Cipher.*DES/CFB/PKCS5Padding*
* Cipher.*DES/OFB/PKCS5Padding*

Another provider may implement a class for each of the above modes (i.e., one class for *ECB*, one for *CBC*, one for *CFB*, and one for *OFB*), one class for *PKCS5Padding*, and a generic *DES* class that subclasses from CipherSpi. That provider would have the following Cipher properties in its master class:

* Cipher.*DES*

The getInstance factory method of the Cipher engine class follows these rules in order to instantiate a provider's implementation of CipherSpi for a transformation of the form "*algorithm*":

1. Check if the provider has registered a subclass of CipherSpi for the specified "*algorithm*".  
   If the answer is YES, instantiate this class, for whose mode and padding scheme default values (as supplied by the provider) are used.  
   If the answer is NO, throw a NoSuchAlgorithmException exception.

The getInstance factory method of the Cipher engine class follows these rules in order to instantiate a provider's implementation of CipherSpi for a transformation of the form "*algorithm/mode/padding*":

1. Check if the provider has registered a subclass of CipherSpi for the specified "*algorithm/mode/padding*" transformation.  
   If the answer is YES, instantiate it.  
   If the answer is NO, go to the next step.
2. Check if the provider has registered a subclass of CipherSpi for the sub-transformation "*algorithm/mode*".  
   If the answer is YES, instantiate it, and call engineSetPadding(*padding*) on the new instance.  
   If the answer is NO, go to the next step.
3. Check if the provider has registered a subclass of CipherSpi for the sub-transformation "*algorithm//padding*" (note the double slashes).  
   If the answer is YES, instantiate it, and call engineSetMode(*mode*) on the new instance.  
   If the answer is NO, go to the next step.
4. Check if the provider has registered a subclass of CipherSpi for the sub-transformation "*algorithm*".  
   If the answer is YES, instantiate it, and call engineSetMode(*mode*) and engineSetPadding(*padding*) on the new instance.  
   If the answer is NO, throw a NoSuchAlgorithmException exception.

**Since:** 1.4 **See Also:**[KeyGenerator](http://docs.google.com/javax/crypto/KeyGenerator.html), [SecretKey](http://docs.google.com/javax/crypto/SecretKey.html)

| **Constructor Summary** | |
| --- | --- |
| [**CipherSpi**](http://docs.google.com/javax/crypto/CipherSpi.html#CipherSpi())() |

| **Method Summary** | |
| --- | --- |
| protected abstract  byte[] | [**engineDoFinal**](http://docs.google.com/javax/crypto/CipherSpi.html#engineDoFinal(byte%5B%5D,%20int,%20int))(byte[] input, int inputOffset, int inputLen)            Encrypts or decrypts data in a single-part operation, or finishes a multiple-part operation. |
| protected abstract  int | [**engineDoFinal**](http://docs.google.com/javax/crypto/CipherSpi.html#engineDoFinal(byte%5B%5D,%20int,%20int,%20byte%5B%5D,%20int))(byte[] input, int inputOffset, int inputLen, byte[] output, int outputOffset)            Encrypts or decrypts data in a single-part operation, or finishes a multiple-part operation. |
| protected  int | [**engineDoFinal**](http://docs.google.com/javax/crypto/CipherSpi.html#engineDoFinal(java.nio.ByteBuffer,%20java.nio.ByteBuffer))([ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html) input, [ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html) output)            Encrypts or decrypts data in a single-part operation, or finishes a multiple-part operation. |
| protected abstract  int | [**engineGetBlockSize**](http://docs.google.com/javax/crypto/CipherSpi.html#engineGetBlockSize())()            Returns the block size (in bytes). |
| protected abstract  byte[] | [**engineGetIV**](http://docs.google.com/javax/crypto/CipherSpi.html#engineGetIV())()            Returns the initialization vector (IV) in a new buffer. |
| protected  int | [**engineGetKeySize**](http://docs.google.com/javax/crypto/CipherSpi.html#engineGetKeySize(java.security.Key))([Key](http://docs.google.com/java/security/Key.html) key)            Returns the key size of the given key object in bits. |
| protected abstract  int | [**engineGetOutputSize**](http://docs.google.com/javax/crypto/CipherSpi.html#engineGetOutputSize(int))(int inputLen)            Returns the length in bytes that an output buffer would need to be in order to hold the result of the next update or doFinal operation, given the input length inputLen (in bytes). |
| protected abstract  [AlgorithmParameters](http://docs.google.com/java/security/AlgorithmParameters.html) | [**engineGetParameters**](http://docs.google.com/javax/crypto/CipherSpi.html#engineGetParameters())()            Returns the parameters used with this cipher. |
| protected abstract  void | [**engineInit**](http://docs.google.com/javax/crypto/CipherSpi.html#engineInit(int,%20java.security.Key,%20java.security.spec.AlgorithmParameterSpec,%20java.security.SecureRandom))(int opmode, [Key](http://docs.google.com/java/security/Key.html) key, [AlgorithmParameterSpec](http://docs.google.com/java/security/spec/AlgorithmParameterSpec.html) params, [SecureRandom](http://docs.google.com/java/security/SecureRandom.html) random)            Initializes this cipher with a key, a set of algorithm parameters, and a source of randomness. |
| protected abstract  void | [**engineInit**](http://docs.google.com/javax/crypto/CipherSpi.html#engineInit(int,%20java.security.Key,%20java.security.AlgorithmParameters,%20java.security.SecureRandom))(int opmode, [Key](http://docs.google.com/java/security/Key.html) key, [AlgorithmParameters](http://docs.google.com/java/security/AlgorithmParameters.html) params, [SecureRandom](http://docs.google.com/java/security/SecureRandom.html) random)            Initializes this cipher with a key, a set of algorithm parameters, and a source of randomness. |
| protected abstract  void | [**engineInit**](http://docs.google.com/javax/crypto/CipherSpi.html#engineInit(int,%20java.security.Key,%20java.security.SecureRandom))(int opmode, [Key](http://docs.google.com/java/security/Key.html) key, [SecureRandom](http://docs.google.com/java/security/SecureRandom.html) random)            Initializes this cipher with a key and a source of randomness. |
| protected abstract  void | [**engineSetMode**](http://docs.google.com/javax/crypto/CipherSpi.html#engineSetMode(java.lang.String))([String](http://docs.google.com/java/lang/String.html) mode)            Sets the mode of this cipher. |
| protected abstract  void | [**engineSetPadding**](http://docs.google.com/javax/crypto/CipherSpi.html#engineSetPadding(java.lang.String))([String](http://docs.google.com/java/lang/String.html) padding)            Sets the padding mechanism of this cipher. |
| protected  [Key](http://docs.google.com/java/security/Key.html) | [**engineUnwrap**](http://docs.google.com/javax/crypto/CipherSpi.html#engineUnwrap(byte%5B%5D,%20java.lang.String,%20int))(byte[] wrappedKey, [String](http://docs.google.com/java/lang/String.html) wrappedKeyAlgorithm, int wrappedKeyType)            Unwrap a previously wrapped key. |
| protected abstract  byte[] | [**engineUpdate**](http://docs.google.com/javax/crypto/CipherSpi.html#engineUpdate(byte%5B%5D,%20int,%20int))(byte[] input, int inputOffset, int inputLen)            Continues a multiple-part encryption or decryption operation (depending on how this cipher was initialized), processing another data part. |
| protected abstract  int | [**engineUpdate**](http://docs.google.com/javax/crypto/CipherSpi.html#engineUpdate(byte%5B%5D,%20int,%20int,%20byte%5B%5D,%20int))(byte[] input, int inputOffset, int inputLen, byte[] output, int outputOffset)            Continues a multiple-part encryption or decryption operation (depending on how this cipher was initialized), processing another data part. |
| protected  int | [**engineUpdate**](http://docs.google.com/javax/crypto/CipherSpi.html#engineUpdate(java.nio.ByteBuffer,%20java.nio.ByteBuffer))([ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html) input, [ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html) output)            Continues a multiple-part encryption or decryption operation (depending on how this cipher was initialized), processing another data part. |
| protected  byte[] | [**engineWrap**](http://docs.google.com/javax/crypto/CipherSpi.html#engineWrap(java.security.Key))([Key](http://docs.google.com/java/security/Key.html) key)            Wrap a key. |

| **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** |
| --- |

### CipherSpi

public **CipherSpi**()

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

### engineSetMode

protected abstract void **engineSetMode**([String](http://docs.google.com/java/lang/String.html) mode)  
 throws [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html)

Sets the mode of this cipher.

**Parameters:**mode - the cipher mode **Throws:** [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if the requested cipher mode does not exist

### engineSetPadding

protected abstract void **engineSetPadding**([String](http://docs.google.com/java/lang/String.html) padding)  
 throws [NoSuchPaddingException](http://docs.google.com/javax/crypto/NoSuchPaddingException.html)

Sets the padding mechanism of this cipher.

**Parameters:**padding - the padding mechanism **Throws:** [NoSuchPaddingException](http://docs.google.com/javax/crypto/NoSuchPaddingException.html) - if the requested padding mechanism does not exist

### engineGetBlockSize

protected abstract int **engineGetBlockSize**()

Returns the block size (in bytes).

**Returns:**the block size (in bytes), or 0 if the underlying algorithm is not a block cipher

### engineGetOutputSize

protected abstract int **engineGetOutputSize**(int inputLen)

Returns the length in bytes that an output buffer would need to be in order to hold the result of the next update or doFinal operation, given the input length inputLen (in bytes).

This call takes into account any unprocessed (buffered) data from a previous update call, and padding.

The actual output length of the next update or doFinal call may be smaller than the length returned by this method.

**Parameters:**inputLen - the input length (in bytes) **Returns:**the required output buffer size (in bytes)

### engineGetIV

protected abstract byte[] **engineGetIV**()

Returns the initialization vector (IV) in a new buffer.

This is useful in the context of password-based encryption or decryption, where the IV is derived from a user-provided passphrase.

**Returns:**the initialization vector in a new buffer, or null if the underlying algorithm does not use an IV, or if the IV has not yet been set.

### engineGetParameters

protected abstract [AlgorithmParameters](http://docs.google.com/java/security/AlgorithmParameters.html) **engineGetParameters**()

Returns the parameters used with this cipher.

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

**Returns:**the parameters used with this cipher, or null if this cipher does not use any parameters.

### engineInit

protected abstract void **engineInit**(int opmode,  
 [Key](http://docs.google.com/java/security/Key.html) key,  
 [SecureRandom](http://docs.google.com/java/security/SecureRandom.html) random)  
 throws [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html)

Initializes this cipher with a key and a source of randomness.

The cipher is initialized for one of the following four operations: encryption, decryption, key wrapping or key unwrapping, depending on the value of opmode.

If this cipher requires any algorithm parameters that cannot be derived from the given key, the underlying cipher implementation is supposed to generate the required parameters itself (using provider-specific default or random values) if it is being initialized for encryption or key wrapping, and raise an InvalidKeyException if it is being initialized for decryption or key unwrapping. The generated parameters can be retrieved using [engineGetParameters](http://docs.google.com/javax/crypto/CipherSpi.html#engineGetParameters()) or [engineGetIV](http://docs.google.com/javax/crypto/CipherSpi.html#engineGetIV()) (if the parameter is an IV).

If this cipher (including its underlying feedback or padding scheme) requires any random bytes (e.g., for parameter generation), it will get them from random.

Note that when a Cipher object is initialized, it loses all previously-acquired state. In other words, initializing a Cipher is equivalent to creating a new instance of that Cipher and initializing it.

**Parameters:**opmode - the operation mode of this cipher (this is one of the following: ENCRYPT\_MODE, DECRYPT\_MODE, WRAP\_MODE or UNWRAP\_MODE)key - the encryption keyrandom - the source of randomness **Throws:** [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html) - if the given key is inappropriate for initializing this cipher, or if this cipher is being initialized for decryption and requires algorithm parameters that cannot be determined from the given key.

### engineInit

protected abstract void **engineInit**(int opmode,  
 [Key](http://docs.google.com/java/security/Key.html) key,  
 [AlgorithmParameterSpec](http://docs.google.com/java/security/spec/AlgorithmParameterSpec.html) params,  
 [SecureRandom](http://docs.google.com/java/security/SecureRandom.html) random)  
 throws [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html),  
 [InvalidAlgorithmParameterException](http://docs.google.com/java/security/InvalidAlgorithmParameterException.html)

Initializes this cipher with a key, a set of algorithm parameters, and a source of randomness.

The cipher is initialized for one of the following four operations: encryption, decryption, key wrapping or key unwrapping, depending on the value of opmode.

If this cipher requires any algorithm parameters and params is null, the underlying cipher implementation is supposed to generate the required parameters itself (using provider-specific default or random values) if it is being initialized for encryption or key wrapping, and raise an InvalidAlgorithmParameterException if it is being initialized for decryption or key unwrapping. The generated parameters can be retrieved using [engineGetParameters](http://docs.google.com/javax/crypto/CipherSpi.html#engineGetParameters()) or [engineGetIV](http://docs.google.com/javax/crypto/CipherSpi.html#engineGetIV()) (if the parameter is an IV).

If this cipher (including its underlying feedback or padding scheme) requires any random bytes (e.g., for parameter generation), it will get them from random.

Note that when a Cipher object is initialized, it loses all previously-acquired state. In other words, initializing a Cipher is equivalent to creating a new instance of that Cipher and initializing it.

**Parameters:**opmode - the operation mode of this cipher (this is one of the following: ENCRYPT\_MODE, DECRYPT\_MODE, WRAP\_MODE or UNWRAP\_MODE)key - the encryption keyparams - the algorithm parametersrandom - the source of randomness **Throws:** [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html) - if the given key is inappropriate for initializing this cipher [InvalidAlgorithmParameterException](http://docs.google.com/java/security/InvalidAlgorithmParameterException.html) - if the given algorithm parameters are inappropriate for this cipher, or if this cipher is being initialized for decryption and requires algorithm parameters and params is null.

### engineInit

protected abstract void **engineInit**(int opmode,  
 [Key](http://docs.google.com/java/security/Key.html) key,  
 [AlgorithmParameters](http://docs.google.com/java/security/AlgorithmParameters.html) params,  
 [SecureRandom](http://docs.google.com/java/security/SecureRandom.html) random)  
 throws [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html),  
 [InvalidAlgorithmParameterException](http://docs.google.com/java/security/InvalidAlgorithmParameterException.html)

Initializes this cipher with a key, a set of algorithm parameters, and a source of randomness.

The cipher is initialized for one of the following four operations: encryption, decryption, key wrapping or key unwrapping, depending on the value of opmode.

If this cipher requires any algorithm parameters and params is null, the underlying cipher implementation is supposed to generate the required parameters itself (using provider-specific default or random values) if it is being initialized for encryption or key wrapping, and raise an InvalidAlgorithmParameterException if it is being initialized for decryption or key unwrapping. The generated parameters can be retrieved using [engineGetParameters](http://docs.google.com/javax/crypto/CipherSpi.html#engineGetParameters()) or [engineGetIV](http://docs.google.com/javax/crypto/CipherSpi.html#engineGetIV()) (if the parameter is an IV).

If this cipher (including its underlying feedback or padding scheme) requires any random bytes (e.g., for parameter generation), it will get them from random.

Note that when a Cipher object is initialized, it loses all previously-acquired state. In other words, initializing a Cipher is equivalent to creating a new instance of that Cipher and initializing it.

**Parameters:**opmode - the operation mode of this cipher (this is one of the following: ENCRYPT\_MODE, DECRYPT\_MODE, WRAP\_MODE or UNWRAP\_MODE)key - the encryption keyparams - the algorithm parametersrandom - the source of randomness **Throws:** [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html) - if the given key is inappropriate for initializing this cipher [InvalidAlgorithmParameterException](http://docs.google.com/java/security/InvalidAlgorithmParameterException.html) - if the given algorithm parameters are inappropriate for this cipher, or if this cipher is being initialized for decryption and requires algorithm parameters and params is null.

### engineUpdate

protected abstract byte[] **engineUpdate**(byte[] input,  
 int inputOffset,  
 int inputLen)

Continues a multiple-part encryption or decryption operation (depending on how this cipher was initialized), processing another data part.

The first inputLen bytes in the input buffer, starting at inputOffset inclusive, are processed, and the result is stored in a new buffer.

**Parameters:**input - the input bufferinputOffset - the offset in input where the input startsinputLen - the input length **Returns:**the new buffer with the result, or null if the underlying cipher is a block cipher and the input data is too short to result in a new block.

### engineUpdate

protected abstract int **engineUpdate**(byte[] input,  
 int inputOffset,  
 int inputLen,  
 byte[] output,  
 int outputOffset)  
 throws [ShortBufferException](http://docs.google.com/javax/crypto/ShortBufferException.html)

Continues a multiple-part encryption or decryption operation (depending on how this cipher was initialized), processing another data part.

The first inputLen bytes in the input buffer, starting at inputOffset inclusive, are processed, and the result is stored in the output buffer, starting at outputOffset inclusive.

If the output buffer is too small to hold the result, a ShortBufferException is thrown.

**Parameters:**input - the input bufferinputOffset - the offset in input where the input startsinputLen - the input lengthoutput - the buffer for the resultoutputOffset - the offset in output where the result is stored **Returns:**the number of bytes stored in output **Throws:** [ShortBufferException](http://docs.google.com/javax/crypto/ShortBufferException.html) - if the given output buffer is too small to hold the result

### engineUpdate

protected int **engineUpdate**([ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html) input,  
 [ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html) output)  
 throws [ShortBufferException](http://docs.google.com/javax/crypto/ShortBufferException.html)

Continues a multiple-part encryption or decryption operation (depending on how this cipher was initialized), processing another data part.

All input.remaining() bytes starting at input.position() are processed. The result is stored in the output buffer. Upon return, the input buffer's position will be equal to its limit; its limit will not have changed. The output buffer's position will have advanced by n, where n is the value returned by this method; the output buffer's limit will not have changed.

If output.remaining() bytes are insufficient to hold the result, a ShortBufferException is thrown.

Subclasses should consider overriding this method if they can process ByteBuffers more efficiently than byte arrays.

**Parameters:**input - the input ByteBufferoutput - the output ByteByffer **Returns:**the number of bytes stored in output **Throws:** [ShortBufferException](http://docs.google.com/javax/crypto/ShortBufferException.html) - if there is insufficient space in the output buffer [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if either parameter is null**Since:** 1.5

### engineDoFinal

protected abstract byte[] **engineDoFinal**(byte[] input,  
 int inputOffset,  
 int inputLen)  
 throws [IllegalBlockSizeException](http://docs.google.com/javax/crypto/IllegalBlockSizeException.html),  
 [BadPaddingException](http://docs.google.com/javax/crypto/BadPaddingException.html)

Encrypts or decrypts data in a single-part operation, or finishes a multiple-part operation. The data is encrypted or decrypted, depending on how this cipher was initialized.

The first inputLen bytes in the input buffer, starting at inputOffset inclusive, and any input bytes that may have been buffered during a previous update operation, are processed, with padding (if requested) being applied. The result is stored in a new buffer.

Upon finishing, this method resets this cipher object to the state it was in when previously initialized via a call to engineInit. That is, the object is reset and available to encrypt or decrypt (depending on the operation mode that was specified in the call to engineInit) more data.

Note: if any exception is thrown, this cipher object may need to be reset before it can be used again.

**Parameters:**input - the input bufferinputOffset - the offset in input where the input startsinputLen - the input length **Returns:**the new buffer with the result **Throws:** [IllegalBlockSizeException](http://docs.google.com/javax/crypto/IllegalBlockSizeException.html) - if this cipher is a block cipher, no padding has been requested (only in encryption mode), and the total input length of the data processed by this cipher is not a multiple of block size; or if this encryption algorithm is unable to process the input data provided. [BadPaddingException](http://docs.google.com/javax/crypto/BadPaddingException.html) - if this cipher is in decryption mode, and (un)padding has been requested, but the decrypted data is not bounded by the appropriate padding bytes

### engineDoFinal

protected abstract int **engineDoFinal**(byte[] input,  
 int inputOffset,  
 int inputLen,  
 byte[] output,  
 int outputOffset)  
 throws [ShortBufferException](http://docs.google.com/javax/crypto/ShortBufferException.html),  
 [IllegalBlockSizeException](http://docs.google.com/javax/crypto/IllegalBlockSizeException.html),  
 [BadPaddingException](http://docs.google.com/javax/crypto/BadPaddingException.html)

Encrypts or decrypts data in a single-part operation, or finishes a multiple-part operation. The data is encrypted or decrypted, depending on how this cipher was initialized.

The first inputLen bytes in the input buffer, starting at inputOffset inclusive, and any input bytes that may have been buffered during a previous update operation, are processed, with padding (if requested) being applied. The result is stored in the output buffer, starting at outputOffset inclusive.

If the output buffer is too small to hold the result, a ShortBufferException is thrown.

Upon finishing, this method resets this cipher object to the state it was in when previously initialized via a call to engineInit. That is, the object is reset and available to encrypt or decrypt (depending on the operation mode that was specified in the call to engineInit) more data.

Note: if any exception is thrown, this cipher object may need to be reset before it can be used again.

**Parameters:**input - the input bufferinputOffset - the offset in input where the input startsinputLen - the input lengthoutput - the buffer for the resultoutputOffset - the offset in output where the result is stored **Returns:**the number of bytes stored in output **Throws:** [IllegalBlockSizeException](http://docs.google.com/javax/crypto/IllegalBlockSizeException.html) - if this cipher is a block cipher, no padding has been requested (only in encryption mode), and the total input length of the data processed by this cipher is not a multiple of block size; or if this encryption algorithm is unable to process the input data provided. [ShortBufferException](http://docs.google.com/javax/crypto/ShortBufferException.html) - if the given output buffer is too small to hold the result [BadPaddingException](http://docs.google.com/javax/crypto/BadPaddingException.html) - if this cipher is in decryption mode, and (un)padding has been requested, but the decrypted data is not bounded by the appropriate padding bytes

### engineDoFinal

protected int **engineDoFinal**([ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html) input,  
 [ByteBuffer](http://docs.google.com/java/nio/ByteBuffer.html) output)  
 throws [ShortBufferException](http://docs.google.com/javax/crypto/ShortBufferException.html),  
 [IllegalBlockSizeException](http://docs.google.com/javax/crypto/IllegalBlockSizeException.html),  
 [BadPaddingException](http://docs.google.com/javax/crypto/BadPaddingException.html)

Encrypts or decrypts data in a single-part operation, or finishes a multiple-part operation. The data is encrypted or decrypted, depending on how this cipher was initialized.

All input.remaining() bytes starting at input.position() are processed. The result is stored in the output buffer. Upon return, the input buffer's position will be equal to its limit; its limit will not have changed. The output buffer's position will have advanced by n, where n is the value returned by this method; the output buffer's limit will not have changed.

If output.remaining() bytes are insufficient to hold the result, a ShortBufferException is thrown.

Upon finishing, this method resets this cipher object to the state it was in when previously initialized via a call to engineInit. That is, the object is reset and available to encrypt or decrypt (depending on the operation mode that was specified in the call to engineInit) more data.

Note: if any exception is thrown, this cipher object may need to be reset before it can be used again.

Subclasses should consider overriding this method if they can process ByteBuffers more efficiently than byte arrays.

**Parameters:**input - the input ByteBufferoutput - the output ByteByffer **Returns:**the number of bytes stored in output **Throws:** [IllegalBlockSizeException](http://docs.google.com/javax/crypto/IllegalBlockSizeException.html) - if this cipher is a block cipher, no padding has been requested (only in encryption mode), and the total input length of the data processed by this cipher is not a multiple of block size; or if this encryption algorithm is unable to process the input data provided. [ShortBufferException](http://docs.google.com/javax/crypto/ShortBufferException.html) - if there is insufficient space in the output buffer [BadPaddingException](http://docs.google.com/javax/crypto/BadPaddingException.html) - if this cipher is in decryption mode, and (un)padding has been requested, but the decrypted data is not bounded by the appropriate padding bytes [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if either parameter is null**Since:** 1.5

### engineWrap

protected byte[] **engineWrap**([Key](http://docs.google.com/java/security/Key.html) key)  
 throws [IllegalBlockSizeException](http://docs.google.com/javax/crypto/IllegalBlockSizeException.html),  
 [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html)

Wrap a key.

This concrete method has been added to this previously-defined abstract class. (For backwards compatibility, it cannot be abstract.) It may be overridden by a provider to wrap a key. Such an override is expected to throw an IllegalBlockSizeException or InvalidKeyException (under the specified circumstances), if the given key cannot be wrapped. If this method is not overridden, it always throws an UnsupportedOperationException.

**Parameters:**key - the key to be wrapped. **Returns:**the wrapped key. **Throws:** [IllegalBlockSizeException](http://docs.google.com/javax/crypto/IllegalBlockSizeException.html) - if this cipher is a block cipher, no padding has been requested, and the length of the encoding of the key to be wrapped is not a multiple of the block size. [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html) - if it is impossible or unsafe to wrap the key with this cipher (e.g., a hardware protected key is being passed to a software-only cipher).

### engineUnwrap

protected [Key](http://docs.google.com/java/security/Key.html) **engineUnwrap**(byte[] wrappedKey,  
 [String](http://docs.google.com/java/lang/String.html) wrappedKeyAlgorithm,  
 int wrappedKeyType)  
 throws [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html),  
 [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html)

Unwrap a previously wrapped key.

This concrete method has been added to this previously-defined abstract class. (For backwards compatibility, it cannot be abstract.) It may be overridden by a provider to unwrap a previously wrapped key. Such an override is expected to throw an InvalidKeyException if the given wrapped key cannot be unwrapped. If this method is not overridden, it always throws an UnsupportedOperationException.

**Parameters:**wrappedKey - the key to be unwrapped.wrappedKeyAlgorithm - the algorithm associated with the wrapped key.wrappedKeyType - the type of the wrapped key. This is one of SECRET\_KEY, PRIVATE\_KEY, or PUBLIC\_KEY. **Returns:**the unwrapped key. **Throws:** [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if no installed providers can create keys of type wrappedKeyType for the wrappedKeyAlgorithm. [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html) - if wrappedKey does not represent a wrapped key of type wrappedKeyType for the wrappedKeyAlgorithm.

### engineGetKeySize

protected int **engineGetKeySize**([Key](http://docs.google.com/java/security/Key.html) key)  
 throws [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html)

Returns the key size of the given key object in bits.

This concrete method has been added to this previously-defined abstract class. It throws an UnsupportedOperationException if it is not overridden by the provider.

**Parameters:**key - the key object. **Returns:**the key size of the given key object. **Throws:** [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html) - if key is invalid.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/CipherSpi.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/crypto/CipherOutputStream.html)   [**NEXT CLASS**](http://docs.google.com/javax/crypto/EncryptedPrivateKeyInfo.html) | [**FRAMES**](http://docs.google.com/index.html?javax/crypto/CipherSpi.html)    [**NO FRAMES**](http://docs.google.com/CipherSpi.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#4d34og8) |

[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).