| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/SealedObject.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/NullCipher.html)   [**NEXT CLASS**](http://docs.google.com/javax/crypto/SecretKey.html) | [**FRAMES**](http://docs.google.com/index.html?javax/crypto/SealedObject.html)    [**NO FRAMES**](http://docs.google.com/SealedObject.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#2s8eyo1) | [METHOD](#26in1rg) |

## **javax.crypto**

Class SealedObject

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

**All Implemented Interfaces:** [Serializable](http://docs.google.com/java/io/Serializable.html)

public class **SealedObject**extends [Object](http://docs.google.com/java/lang/Object.html)implements [Serializable](http://docs.google.com/java/io/Serializable.html)

This class enables a programmer to create an object and protect its confidentiality with a cryptographic algorithm.

Given any Serializable object, one can create a SealedObject that encapsulates the original object, in serialized format (i.e., a "deep copy"), and seals (encrypts) its serialized contents, using a cryptographic algorithm such as DES, to protect its confidentiality. The encrypted content can later be decrypted (with the corresponding algorithm using the correct decryption key) and de-serialized, yielding the original object.

Note that the Cipher object must be fully initialized with the correct algorithm, key, padding scheme, etc., before being applied to a SealedObject.

The original object that was sealed can be recovered in two different ways:

* by using the [getObject](http://docs.google.com/javax/crypto/SealedObject.html#getObject(javax.crypto.Cipher)) method that takes a Cipher object.  
  This method requires a fully initialized Cipher object, initialized with the exact same algorithm, key, padding scheme, etc., that were used to seal the object.  
  This approach has the advantage that the party who unseals the sealed object does not require knowledge of the decryption key. For example, after one party has initialized the cipher object with the required decryption key, it could hand over the cipher object to another party who then unseals the sealed object.
* by using one of the [getObject](http://docs.google.com/javax/crypto/SealedObject.html#getObject(java.security.Key)) methods that take a Key object.  
  In this approach, the getObject method creates a cipher object for the appropriate decryption algorithm and initializes it with the given decryption key and the algorithm parameters (if any) that were stored in the sealed object.  
  This approach has the advantage that the party who unseals the object does not need to keep track of the parameters (e.g., an IV) that were used to seal the object.

**Since:** 1.4 **See Also:**[Cipher](http://docs.google.com/javax/crypto/Cipher.html), [Serialized Form](http://docs.google.com/serialized-form.html#javax.crypto.SealedObject)

| **Field Summary** | |
| --- | --- |
| protected  byte[] | [**encodedParams**](http://docs.google.com/javax/crypto/SealedObject.html#encodedParams)            The cryptographic parameters used by the sealing Cipher, encoded in the default format. |

| **Constructor Summary** | |
| --- | --- |
| protected | [**SealedObject**](http://docs.google.com/javax/crypto/SealedObject.html#SealedObject(javax.crypto.SealedObject))([SealedObject](http://docs.google.com/javax/crypto/SealedObject.html) so)            Constructs a SealedObject object from the passed-in SealedObject. |
|  | [**SealedObject**](http://docs.google.com/javax/crypto/SealedObject.html#SealedObject(java.io.Serializable,%20javax.crypto.Cipher))([Serializable](http://docs.google.com/java/io/Serializable.html) object, [Cipher](http://docs.google.com/javax/crypto/Cipher.html) c)            Constructs a SealedObject from any Serializable object. |

| **Method Summary** | |
| --- | --- |
| [String](http://docs.google.com/java/lang/String.html) | [**getAlgorithm**](http://docs.google.com/javax/crypto/SealedObject.html#getAlgorithm())()            Returns the algorithm that was used to seal this object. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getObject**](http://docs.google.com/javax/crypto/SealedObject.html#getObject(javax.crypto.Cipher))([Cipher](http://docs.google.com/javax/crypto/Cipher.html) c)            Retrieves the original (encapsulated) object. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getObject**](http://docs.google.com/javax/crypto/SealedObject.html#getObject(java.security.Key))([Key](http://docs.google.com/java/security/Key.html) key)            Retrieves the original (encapsulated) object. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getObject**](http://docs.google.com/javax/crypto/SealedObject.html#getObject(java.security.Key,%20java.lang.String))([Key](http://docs.google.com/java/security/Key.html) key, [String](http://docs.google.com/java/lang/String.html) provider)            Retrieves the original (encapsulated) object. |

| **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)) |

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

### encodedParams

protected byte[] **encodedParams**

The cryptographic parameters used by the sealing Cipher, encoded in the default format.

That is, cipher.getParameters().getEncoded().

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

### SealedObject

public **SealedObject**([Serializable](http://docs.google.com/java/io/Serializable.html) object,  
 [Cipher](http://docs.google.com/javax/crypto/Cipher.html) c)  
 throws [IOException](http://docs.google.com/java/io/IOException.html),  
 [IllegalBlockSizeException](http://docs.google.com/javax/crypto/IllegalBlockSizeException.html)

Constructs a SealedObject from any Serializable object.

The given object is serialized, and its serialized contents are encrypted using the given Cipher, which must be fully initialized.

Any algorithm parameters that may be used in the encryption operation are stored inside of the new SealedObject.

**Parameters:**object - the object to be sealed; can be null.c - the cipher used to seal the object. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the given cipher is null. [IOException](http://docs.google.com/java/io/IOException.html) - if an error occurs during serialization [IllegalBlockSizeException](http://docs.google.com/javax/crypto/IllegalBlockSizeException.html) - if the given cipher is a block cipher, no padding has been requested, and the total input length (i.e., the length of the serialized object contents) is not a multiple of the cipher's block size

### SealedObject

protected **SealedObject**([SealedObject](http://docs.google.com/javax/crypto/SealedObject.html) so)

Constructs a SealedObject object from the passed-in SealedObject.

**Parameters:**so - a SealedObject object **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the given sealed object is null.

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

### getAlgorithm

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

Returns the algorithm that was used to seal this object.

**Returns:**the algorithm that was used to seal this object.

### getObject

public final [Object](http://docs.google.com/java/lang/Object.html) **getObject**([Key](http://docs.google.com/java/security/Key.html) key)  
 throws [IOException](http://docs.google.com/java/io/IOException.html),  
 [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html),  
 [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html),  
 [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html)

Retrieves the original (encapsulated) object.

This method creates a cipher for the algorithm that had been used in the sealing operation. If the default provider package provides an implementation of that algorithm, an instance of Cipher containing that implementation is used. If the algorithm is not available in the default package, other packages are searched. The Cipher object is initialized for decryption, using the given key and the parameters (if any) that had been used in the sealing operation.

The encapsulated object is unsealed and de-serialized, before it is returned.

**Parameters:**key - the key used to unseal the object. **Returns:**the original object. **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if an error occurs during de-serialiazation. [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html) - if an error occurs during de-serialiazation. [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if the algorithm to unseal the object is not available. [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html) - if the given key cannot be used to unseal the object (e.g., it has the wrong algorithm). [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if key is null.

### getObject

public final [Object](http://docs.google.com/java/lang/Object.html) **getObject**([Cipher](http://docs.google.com/javax/crypto/Cipher.html) c)  
 throws [IOException](http://docs.google.com/java/io/IOException.html),  
 [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html),  
 [IllegalBlockSizeException](http://docs.google.com/javax/crypto/IllegalBlockSizeException.html),  
 [BadPaddingException](http://docs.google.com/javax/crypto/BadPaddingException.html)

Retrieves the original (encapsulated) object.

The encapsulated object is unsealed (using the given Cipher, assuming that the Cipher is already properly initialized) and de-serialized, before it is returned.

**Parameters:**c - the cipher used to unseal the object **Returns:**the original object. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the given cipher is null. [IOException](http://docs.google.com/java/io/IOException.html) - if an error occurs during de-serialiazation [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html) - if an error occurs during de-serialiazation [IllegalBlockSizeException](http://docs.google.com/javax/crypto/IllegalBlockSizeException.html) - if the given cipher is a block cipher, no padding has been requested, and the total input length is not a multiple of the cipher's block size [BadPaddingException](http://docs.google.com/javax/crypto/BadPaddingException.html) - if the given cipher has been initialized for decryption, and padding has been specified, but the input data does not have proper expected padding bytes

### getObject

public final [Object](http://docs.google.com/java/lang/Object.html) **getObject**([Key](http://docs.google.com/java/security/Key.html) key,  
 [String](http://docs.google.com/java/lang/String.html) provider)  
 throws [IOException](http://docs.google.com/java/io/IOException.html),  
 [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html),  
 [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html),  
 [NoSuchProviderException](http://docs.google.com/java/security/NoSuchProviderException.html),  
 [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html)

Retrieves the original (encapsulated) object.

This method creates a cipher for the algorithm that had been used in the sealing operation, using an implementation of that algorithm from the given provider. The Cipher object is initialized for decryption, using the given key and the parameters (if any) that had been used in the sealing operation.

The encapsulated object is unsealed and de-serialized, before it is returned.

**Parameters:**key - the key used to unseal the object.provider - the name of the provider of the algorithm to unseal the object. **Returns:**the original object. **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the given provider is null or empty. [IOException](http://docs.google.com/java/io/IOException.html) - if an error occurs during de-serialiazation. [ClassNotFoundException](http://docs.google.com/java/lang/ClassNotFoundException.html) - if an error occurs during de-serialiazation. [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if the algorithm to unseal the object is not available. [NoSuchProviderException](http://docs.google.com/java/security/NoSuchProviderException.html) - if the given provider is not configured. [InvalidKeyException](http://docs.google.com/java/security/InvalidKeyException.html) - if the given key cannot be used to unseal the object (e.g., it has the wrong algorithm). [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if key is null.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/SealedObject.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/NullCipher.html)   [**NEXT CLASS**](http://docs.google.com/javax/crypto/SecretKey.html) | [**FRAMES**](http://docs.google.com/index.html?javax/crypto/SealedObject.html)    [**NO FRAMES**](http://docs.google.com/SealedObject.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#2s8eyo1) | [METHOD](#26in1rg) |

[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).