| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/KeyGenerator.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/KeyAgreementSpi.html)   [**NEXT CLASS**](http://docs.google.com/javax/crypto/KeyGeneratorSpi.html) | [**FRAMES**](http://docs.google.com/index.html?javax/crypto/KeyGenerator.html)    [**NO FRAMES**](http://docs.google.com/KeyGenerator.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 KeyGenerator

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

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

This class provides the functionality of a secret (symmetric) key generator.

Key generators are constructed using one of the getInstance class methods of this class.

KeyGenerator objects are reusable, i.e., after a key has been generated, the same KeyGenerator object can be re-used to generate further keys.

There are two ways to generate a key: in an algorithm-independent manner, and in an algorithm-specific manner. The only difference between the two is the initialization of the object:

* **Algorithm-Independent Initialization**  
  All key generators share the concepts of a *keysize* and a *source of randomness*. There is an [init](http://docs.google.com/javax/crypto/KeyGenerator.html#init(int,%20java.security.SecureRandom)) method in this KeyGenerator class that takes these two universally shared types of arguments. There is also one that takes just a keysize argument, and uses the SecureRandom implementation of the highest-priority installed provider as the source of randomness (or a system-provided source of randomness if none of the installed providers supply a SecureRandom implementation), and one that takes just a source of randomness.  
  Since no other parameters are specified when you call the above algorithm-independent init methods, it is up to the provider what to do about the algorithm-specific parameters (if any) to be associated with each of the keys.
* **Algorithm-Specific Initialization**  
  For situations where a set of algorithm-specific parameters already exists, there are two [init](http://docs.google.com/javax/crypto/KeyGenerator.html#init(java.security.spec.AlgorithmParameterSpec)) methods that have an AlgorithmParameterSpec argument. One also has a SecureRandom argument, while the other uses the SecureRandom implementation of the highest-priority installed provider as the source of randomness (or a system-provided source of randomness if none of the installed providers supply a SecureRandom implementation).

In case the client does not explicitly initialize the KeyGenerator (via a call to an init method), each provider must supply (and document) a default initialization.

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

| **Constructor Summary** | |
| --- | --- |
| protected | [**KeyGenerator**](http://docs.google.com/javax/crypto/KeyGenerator.html#KeyGenerator(javax.crypto.KeyGeneratorSpi,%20java.security.Provider,%20java.lang.String))([KeyGeneratorSpi](http://docs.google.com/javax/crypto/KeyGeneratorSpi.html) keyGenSpi, [Provider](http://docs.google.com/java/security/Provider.html) provider, [String](http://docs.google.com/java/lang/String.html) algorithm)            Creates a KeyGenerator object. |

| **Method Summary** | |
| --- | --- |
| [SecretKey](http://docs.google.com/javax/crypto/SecretKey.html) | [**generateKey**](http://docs.google.com/javax/crypto/KeyGenerator.html#generateKey())()            Generates a secret key. |
| [String](http://docs.google.com/java/lang/String.html) | [**getAlgorithm**](http://docs.google.com/javax/crypto/KeyGenerator.html#getAlgorithm())()            Returns the algorithm name of this KeyGenerator object. |
| static [KeyGenerator](http://docs.google.com/javax/crypto/KeyGenerator.html) | [**getInstance**](http://docs.google.com/javax/crypto/KeyGenerator.html#getInstance(java.lang.String))([String](http://docs.google.com/java/lang/String.html) algorithm)            Returns a KeyGenerator object that generates secret keys for the specified algorithm. |
| static [KeyGenerator](http://docs.google.com/javax/crypto/KeyGenerator.html) | [**getInstance**](http://docs.google.com/javax/crypto/KeyGenerator.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 KeyGenerator object that generates secret keys for the specified algorithm. |
| static [KeyGenerator](http://docs.google.com/javax/crypto/KeyGenerator.html) | [**getInstance**](http://docs.google.com/javax/crypto/KeyGenerator.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 KeyGenerator object that generates secret keys for the specified algorithm. |
| [Provider](http://docs.google.com/java/security/Provider.html) | [**getProvider**](http://docs.google.com/javax/crypto/KeyGenerator.html#getProvider())()            Returns the provider of this KeyGenerator object. |
| void | [**init**](http://docs.google.com/javax/crypto/KeyGenerator.html#init(java.security.spec.AlgorithmParameterSpec))([AlgorithmParameterSpec](http://docs.google.com/java/security/spec/AlgorithmParameterSpec.html) params)            Initializes this key generator with the specified parameter set. |
| void | [**init**](http://docs.google.com/javax/crypto/KeyGenerator.html#init(java.security.spec.AlgorithmParameterSpec,%20java.security.SecureRandom))([AlgorithmParameterSpec](http://docs.google.com/java/security/spec/AlgorithmParameterSpec.html) params, [SecureRandom](http://docs.google.com/java/security/SecureRandom.html) random)            Initializes this key generator with the specified parameter set and a user-provided source of randomness. |
| void | [**init**](http://docs.google.com/javax/crypto/KeyGenerator.html#init(int))(int keysize)            Initializes this key generator for a certain keysize. |
| void | [**init**](http://docs.google.com/javax/crypto/KeyGenerator.html#init(int,%20java.security.SecureRandom))(int keysize, [SecureRandom](http://docs.google.com/java/security/SecureRandom.html) random)            Initializes this key generator for a certain keysize, using a user-provided source of randomness. |
| void | [**init**](http://docs.google.com/javax/crypto/KeyGenerator.html#init(java.security.SecureRandom))([SecureRandom](http://docs.google.com/java/security/SecureRandom.html) random)            Initializes this key generator. |

| **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** |
| --- |

### KeyGenerator

protected **KeyGenerator**([KeyGeneratorSpi](http://docs.google.com/javax/crypto/KeyGeneratorSpi.html) keyGenSpi,  
 [Provider](http://docs.google.com/java/security/Provider.html) provider,  
 [String](http://docs.google.com/java/lang/String.html) algorithm)

Creates a KeyGenerator object.

**Parameters:**keyGenSpi - the delegateprovider - the provideralgorithm - the algorithm

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

### getAlgorithm

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

Returns the algorithm name of this KeyGenerator object.

This is the same name that was specified in one of the getInstance calls that created this KeyGenerator object.

**Returns:**the algorithm name of this KeyGenerator object.

### getInstance

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

Returns a KeyGenerator object that generates secret keys for the specified algorithm.

This method traverses the list of registered security Providers, starting with the most preferred Provider. A new KeyGenerator object encapsulating the KeyGeneratorSpi 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 requested key algorithm. See Appendix A in the  [Java Cryptography Architecture Reference Guide](http://docs.google.com/technotes/guides/security/crypto/CryptoSpec.html#AppA) for information about standard algorithm names. **Returns:**the new KeyGenerator object. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified algorithm is null. [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if no Provider supports a KeyGeneratorSpi implementation for the specified algorithm.**See Also:**[Provider](http://docs.google.com/java/security/Provider.html)

### getInstance

public static final [KeyGenerator](http://docs.google.com/javax/crypto/KeyGenerator.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 KeyGenerator object that generates secret keys for the specified algorithm.

A new KeyGenerator object encapsulating the KeyGeneratorSpi 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 standard name of the requested key algorithm. See Appendix A in the  [Java Cryptography Architecture Reference Guide](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 KeyGenerator object. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified algorithm is null. [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if a KeyGeneratorSpi 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 is null or empty.**See Also:**[Provider](http://docs.google.com/java/security/Provider.html)

### getInstance

public static final [KeyGenerator](http://docs.google.com/javax/crypto/KeyGenerator.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 KeyGenerator object that generates secret keys for the specified algorithm.

A new KeyGenerator object encapsulating the KeyGeneratorSpi 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 standard name of the requested key algorithm. See Appendix A in the  [Java Cryptography Architecture Reference Guide](http://docs.google.com/technotes/guides/security/crypto/CryptoSpec.html#AppA) for information about standard algorithm names.provider - the provider. **Returns:**the new KeyGenerator object. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified algorithm is null. [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if a KeyGeneratorSpi 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.**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 KeyGenerator object.

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

### init

public final void **init**([SecureRandom](http://docs.google.com/java/security/SecureRandom.html) random)

Initializes this key generator.

**Parameters:**random - the source of randomness for this generator

### init

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

Initializes this key generator with the specified parameter set.

If this key generator requires any random bytes, it will get them using the [SecureRandom](http://docs.google.com/java/security/SecureRandom.html) implementation of the highest-priority installed provider as the source of randomness. (If none of the installed providers supply an implementation of SecureRandom, a system-provided source of randomness will be used.)

**Parameters:**params - the key generation parameters **Throws:** [InvalidAlgorithmParameterException](http://docs.google.com/java/security/InvalidAlgorithmParameterException.html) - if the given parameters are inappropriate for this key generator

### init

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

Initializes this key generator with the specified parameter set and a user-provided source of randomness.

**Parameters:**params - the key generation parametersrandom - the source of randomness for this key generator **Throws:** [InvalidAlgorithmParameterException](http://docs.google.com/java/security/InvalidAlgorithmParameterException.html) - if params is inappropriate for this key generator

### init

public final void **init**(int keysize)

Initializes this key generator for a certain keysize.

If this key generator requires any random bytes, it will get them using the [SecureRandom](http://docs.google.com/java/security/SecureRandom.html) implementation of the highest-priority installed provider as the source of randomness. (If none of the installed providers supply an implementation of SecureRandom, a system-provided source of randomness will be used.)

**Parameters:**keysize - the keysize. This is an algorithm-specific metric, specified in number of bits. **Throws:** [InvalidParameterException](http://docs.google.com/java/security/InvalidParameterException.html) - if the keysize is wrong or not supported.

### init

public final void **init**(int keysize,  
 [SecureRandom](http://docs.google.com/java/security/SecureRandom.html) random)

Initializes this key generator for a certain keysize, using a user-provided source of randomness.

**Parameters:**keysize - the keysize. This is an algorithm-specific metric, specified in number of bits.random - the source of randomness for this key generator **Throws:** [InvalidParameterException](http://docs.google.com/java/security/InvalidParameterException.html) - if the keysize is wrong or not supported.

### generateKey

public final [SecretKey](http://docs.google.com/javax/crypto/SecretKey.html) **generateKey**()

Generates a secret key.

**Returns:**the new key

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/KeyGenerator.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/KeyAgreementSpi.html)   [**NEXT CLASS**](http://docs.google.com/javax/crypto/KeyGeneratorSpi.html) | [**FRAMES**](http://docs.google.com/index.html?javax/crypto/KeyGenerator.html)    [**NO FRAMES**](http://docs.google.com/KeyGenerator.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).