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

## **java.security**

Class KeyPairGenerator

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

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

The KeyPairGenerator class is used to generate pairs of public and private keys. Key pair generators are constructed using the getInstance factory methods (static methods that return instances of a given class).

A Key pair generator for a particular algorithm creates a public/private key pair that can be used with this algorithm. It also associates algorithm-specific parameters with each of the generated keys.

There are two ways to generate a key pair: 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 pair generators share the concepts of a keysize and a source of randomness. The keysize is interpreted differently for different algorithms (e.g., in the case of the *DSA* algorithm, the keysize corresponds to the length of the modulus). There is an [initialize](http://docs.google.com/java/security/KeyPairGenerator.html#initialize(int,%20java.security.SecureRandom)) method in this KeyPairGenerator 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. (If none of the installed providers supply an implementation of SecureRandom, a system-provided source of randomness is used.)  
  Since no other parameters are specified when you call the above algorithm-independent initialize 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.  
  If the algorithm is the *DSA* algorithm, and the keysize (modulus size) is 512, 768, or 1024, then the *Sun* provider uses a set of precomputed values for the p, q, and g parameters. If the modulus size is not one of the above values, the *Sun* provider creates a new set of parameters. Other providers might have precomputed parameter sets for more than just the three modulus sizes mentioned above. Still others might not have a list of precomputed parameters at all and instead always create new parameter sets.
* **Algorithm-Specific Initialization**  
  For situations where a set of algorithm-specific parameters already exists (e.g., so-called *community parameters* in DSA), there are two [initialize](http://docs.google.com/java/security/KeyPairGenerator.html#initialize(java.security.spec.AlgorithmParameterSpec)) methods that have an AlgorithmParameterSpec argument. One also has a SecureRandom argument, while the the other uses the SecureRandom 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 is used.)

In case the client does not explicitly initialize the KeyPairGenerator (via a call to an initialize method), each provider must supply (and document) a default initialization. For example, the *Sun* provider uses a default modulus size (keysize) of 1024 bits.

Note that this class is abstract and extends from KeyPairGeneratorSpi for historical reasons. Application developers should only take notice of the methods defined in this KeyPairGenerator class; all the methods in the superclass are intended for cryptographic service providers who wish to supply their own implementations of key pair generators.

**See Also:**[AlgorithmParameterSpec](http://docs.google.com/java/security/spec/AlgorithmParameterSpec.html)

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

| **Method Summary** | |
| --- | --- |
| [KeyPair](http://docs.google.com/java/security/KeyPair.html) | [**generateKeyPair**](http://docs.google.com/java/security/KeyPairGenerator.html#generateKeyPair())()            Generates a key pair. |
| [KeyPair](http://docs.google.com/java/security/KeyPair.html) | [**genKeyPair**](http://docs.google.com/java/security/KeyPairGenerator.html#genKeyPair())()            Generates a key pair. |
| [String](http://docs.google.com/java/lang/String.html) | [**getAlgorithm**](http://docs.google.com/java/security/KeyPairGenerator.html#getAlgorithm())()            Returns the standard name of the algorithm for this key pair generator. |
| static [KeyPairGenerator](http://docs.google.com/java/security/KeyPairGenerator.html) | [**getInstance**](http://docs.google.com/java/security/KeyPairGenerator.html#getInstance(java.lang.String))([String](http://docs.google.com/java/lang/String.html) algorithm)            Returns a KeyPairGenerator object that generates public/private key pairs for the specified algorithm. |
| static [KeyPairGenerator](http://docs.google.com/java/security/KeyPairGenerator.html) | [**getInstance**](http://docs.google.com/java/security/KeyPairGenerator.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 KeyPairGenerator object that generates public/private key pairs for the specified algorithm. |
| static [KeyPairGenerator](http://docs.google.com/java/security/KeyPairGenerator.html) | [**getInstance**](http://docs.google.com/java/security/KeyPairGenerator.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 KeyPairGenerator object that generates public/private key pairs for the specified algorithm. |
| [Provider](http://docs.google.com/java/security/Provider.html) | [**getProvider**](http://docs.google.com/java/security/KeyPairGenerator.html#getProvider())()            Returns the provider of this key pair generator object. |
| void | [**initialize**](http://docs.google.com/java/security/KeyPairGenerator.html#initialize(java.security.spec.AlgorithmParameterSpec))([AlgorithmParameterSpec](http://docs.google.com/java/security/spec/AlgorithmParameterSpec.html) params)            Initializes the key pair generator using the specified parameter set and the SecureRandom implementation of the highest-priority installed provider as the source of randomness. |
| void | [**initialize**](http://docs.google.com/java/security/KeyPairGenerator.html#initialize(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 the key pair generator with the given parameter set and source of randomness. |
| void | [**initialize**](http://docs.google.com/java/security/KeyPairGenerator.html#initialize(int))(int keysize)            Initializes the key pair generator for a certain keysize using a default parameter set and the SecureRandom implementation of the highest-priority installed provider as the source of randomness. |
| void | [**initialize**](http://docs.google.com/java/security/KeyPairGenerator.html#initialize(int,%20java.security.SecureRandom))(int keysize, [SecureRandom](http://docs.google.com/java/security/SecureRandom.html) random)            Initializes the key pair generator for a certain keysize with the given source of randomness (and a default parameter set). |

| **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** |
| --- |

### KeyPairGenerator

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

Creates a KeyPairGenerator 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** |
| --- |

### getAlgorithm

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

Returns the standard name of the algorithm for this key pair generator. 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 standard string name of the algorithm.

### getInstance

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

Returns a KeyPairGenerator object that generates public/private key pairs for the specified algorithm.

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

### getInstance

public static [KeyPairGenerator](http://docs.google.com/java/security/KeyPairGenerator.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 KeyPairGenerator object that generates public/private key pairs for the specified algorithm.

A new KeyPairGenerator object encapsulating the KeyPairGeneratorSpi 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 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.provider - the string name of the provider. **Returns:**the new KeyPairGenerator object. **Throws:** [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if a KeyPairGeneratorSpi 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 [KeyPairGenerator](http://docs.google.com/java/security/KeyPairGenerator.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 KeyPairGenerator object that generates public/private key pairs for the specified algorithm.

A new KeyPairGenerator object encapsulating the KeyPairGeneratorSpi 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 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.provider - the provider. **Returns:**the new KeyPairGenerator object. **Throws:** [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if a KeyPairGeneratorSpi implementation for the specified algorithm is not available from the specified Provider object. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the specified 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 key pair generator object.

**Returns:**the provider of this key pair generator object

### initialize

public void **initialize**(int keysize)

Initializes the key pair generator for a certain keysize using a default parameter set and the SecureRandom 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 is used.)

**Parameters:**keysize - the keysize. This is an algorithm-specific metric, such as modulus length, specified in number of bits. **Throws:** [InvalidParameterException](http://docs.google.com/java/security/InvalidParameterException.html) - if the keysize is not supported by this KeyPairGenerator object.

### initialize

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

Initializes the key pair generator for a certain keysize with the given source of randomness (and a default parameter set).

**Specified by:**[initialize](http://docs.google.com/java/security/KeyPairGeneratorSpi.html#initialize(int,%20java.security.SecureRandom)) in class [KeyPairGeneratorSpi](http://docs.google.com/java/security/KeyPairGeneratorSpi.html) **Parameters:**keysize - the keysize. This is an algorithm-specific metric, such as modulus length, specified in number of bits.random - the source of randomness. **Throws:** [InvalidParameterException](http://docs.google.com/java/security/InvalidParameterException.html) - if the keysize is not supported by this KeyPairGenerator object.**Since:** 1.2

### initialize

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

Initializes the key pair generator using the specified parameter set and the SecureRandom 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 is used.).

This concrete method has been added to this previously-defined abstract class. This method calls the KeyPairGeneratorSpi [initialize](http://docs.google.com/java/security/KeyPairGeneratorSpi.html#initialize(java.security.spec.AlgorithmParameterSpec,%20java.security.SecureRandom)) method, passing it params and a source of randomness (obtained from the highest-priority installed provider or system-provided if none of the installed providers supply one). That initialize method always throws an UnsupportedOperationException if it is not overridden by the provider.

**Parameters:**params - the parameter set used to generate the keys. **Throws:** [InvalidAlgorithmParameterException](http://docs.google.com/java/security/InvalidAlgorithmParameterException.html) - if the given parameters are inappropriate for this key pair generator.**Since:** 1.2

### initialize

public void **initialize**([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 the key pair generator with the given parameter set and source of randomness.

This concrete method has been added to this previously-defined abstract class. This method calls the KeyPairGeneratorSpi [initialize](http://docs.google.com/java/security/KeyPairGeneratorSpi.html#initialize(java.security.spec.AlgorithmParameterSpec,%20java.security.SecureRandom)) method, passing it params and random. That initialize method always throws an UnsupportedOperationException if it is not overridden by the provider.

**Overrides:**[initialize](http://docs.google.com/java/security/KeyPairGeneratorSpi.html#initialize(java.security.spec.AlgorithmParameterSpec,%20java.security.SecureRandom)) in class [KeyPairGeneratorSpi](http://docs.google.com/java/security/KeyPairGeneratorSpi.html) **Parameters:**params - the parameter set used to generate the keys.random - the source of randomness. **Throws:** [InvalidAlgorithmParameterException](http://docs.google.com/java/security/InvalidAlgorithmParameterException.html) - if the given parameters are inappropriate for this key pair generator.**Since:** 1.2

### genKeyPair

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

Generates a key pair.

If this KeyPairGenerator has not been initialized explicitly, provider-specific defaults will be used for the size and other (algorithm-specific) values of the generated keys.

This will generate a new key pair every time it is called.

This method is functionally equivalent to [generateKeyPair](http://docs.google.com/java/security/KeyPairGenerator.html#generateKeyPair()).

**Returns:**the generated key pair**Since:** 1.2

### generateKeyPair

public [KeyPair](http://docs.google.com/java/security/KeyPair.html) **generateKeyPair**()

Generates a key pair.

If this KeyPairGenerator has not been initialized explicitly, provider-specific defaults will be used for the size and other (algorithm-specific) values of the generated keys.

This will generate a new key pair every time it is called.

This method is functionally equivalent to [genKeyPair](http://docs.google.com/java/security/KeyPairGenerator.html#genKeyPair()).

**Specified by:**[generateKeyPair](http://docs.google.com/java/security/KeyPairGeneratorSpi.html#generateKeyPair()) in class [KeyPairGeneratorSpi](http://docs.google.com/java/security/KeyPairGeneratorSpi.html) **Returns:**the generated key pair

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/KeyPairGenerator.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/KeyPair.html)   [**NEXT CLASS**](http://docs.google.com/java/security/KeyPairGeneratorSpi.html) | [**FRAMES**](http://docs.google.com/index.html?java/security/KeyPairGenerator.html)    [**NO FRAMES**](http://docs.google.com/KeyPairGenerator.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).