| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/SecureRandom.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/SecureClassLoader.html)   [**NEXT CLASS**](http://docs.google.com/java/security/SecureRandomSpi.html) | [**FRAMES**](http://docs.google.com/index.html?java/security/SecureRandom.html)    [**NO FRAMES**](http://docs.google.com/SecureRandom.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#1t3h5sf) | [METHOD](#3rdcrjn) |

## **java.security**

Class SecureRandom

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 [java.util.Random](http://docs.google.com/java/util/Random.html)  
 **java.security.SecureRandom**

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

public class **SecureRandom**extends [Random](http://docs.google.com/java/util/Random.html)

This class provides a cryptographically strong random number generator (RNG).

A cryptographically strong random number minimally complies with the statistical random number generator tests specified in  [*FIPS 140-2, Security Requirements for Cryptographic Modules*](http://csrc.nist.gov/cryptval/140-2.htm), section 4.9.1. Additionally, SecureRandom must produce non-deterministic output. Therefore any seed material passed to a SecureRandom object must be unpredictable, and all SecureRandom output sequences must be cryptographically strong, as described in  [*RFC 1750: Randomness Recommendations for Security*](http://www.ietf.org/rfc/rfc1750.txt).

A caller obtains a SecureRandom instance via the no-argument constructor or one of the getInstance methods:

SecureRandom random = new SecureRandom();

Many SecureRandom implementations are in the form of a pseudo-random number generator (PRNG), which means they use a deterministic algorithm to produce a pseudo-random sequence from a true random seed. Other implementations may produce true random numbers, and yet others may use a combination of both techniques.

Typical callers of SecureRandom invoke the following methods to retrieve random bytes:

SecureRandom random = new SecureRandom();  
 byte bytes[] = new byte[20];  
 random.nextBytes(bytes);

Callers may also invoke the generateSeed method to generate a given number of seed bytes (to seed other random number generators, for example):

byte seed[] = random.generateSeed(20);

**See Also:**[SecureRandomSpi](http://docs.google.com/java/security/SecureRandomSpi.html), [Random](http://docs.google.com/java/util/Random.html), [Serialized Form](http://docs.google.com/serialized-form.html#java.security.SecureRandom)

| **Constructor Summary** | |
| --- | --- |
|  | [**SecureRandom**](http://docs.google.com/java/security/SecureRandom.html#SecureRandom())()            Constructs a secure random number generator (RNG) implementing the default random number algorithm. |
|  | [**SecureRandom**](http://docs.google.com/java/security/SecureRandom.html#SecureRandom(byte%5B%5D))(byte[] seed)            Constructs a secure random number generator (RNG) implementing the default random number algorithm. |
| protected | [**SecureRandom**](http://docs.google.com/java/security/SecureRandom.html#SecureRandom(java.security.SecureRandomSpi,%20java.security.Provider))([SecureRandomSpi](http://docs.google.com/java/security/SecureRandomSpi.html) secureRandomSpi, [Provider](http://docs.google.com/java/security/Provider.html) provider)            Creates a SecureRandom object. |

| **Method Summary** | |
| --- | --- |
| byte[] | [**generateSeed**](http://docs.google.com/java/security/SecureRandom.html#generateSeed(int))(int numBytes)            Returns the given number of seed bytes, computed using the seed generation algorithm that this class uses to seed itself. |
| [String](http://docs.google.com/java/lang/String.html) | [**getAlgorithm**](http://docs.google.com/java/security/SecureRandom.html#getAlgorithm())()            Returns the name of the algorithm implemented by this SecureRandom object. |
| static [SecureRandom](http://docs.google.com/java/security/SecureRandom.html) | [**getInstance**](http://docs.google.com/java/security/SecureRandom.html#getInstance(java.lang.String))([String](http://docs.google.com/java/lang/String.html) algorithm)            Returns a SecureRandom object that implements the specified Random Number Generator (RNG) algorithm. |
| static [SecureRandom](http://docs.google.com/java/security/SecureRandom.html) | [**getInstance**](http://docs.google.com/java/security/SecureRandom.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 SecureRandom object that implements the specified Random Number Generator (RNG) algorithm. |
| static [SecureRandom](http://docs.google.com/java/security/SecureRandom.html) | [**getInstance**](http://docs.google.com/java/security/SecureRandom.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 SecureRandom object that implements the specified Random Number Generator (RNG) algorithm. |
| [Provider](http://docs.google.com/java/security/Provider.html) | [**getProvider**](http://docs.google.com/java/security/SecureRandom.html#getProvider())()            Returns the provider of this SecureRandom object. |
| static byte[] | [**getSeed**](http://docs.google.com/java/security/SecureRandom.html#getSeed(int))(int numBytes)            Returns the given number of seed bytes, computed using the seed generation algorithm that this class uses to seed itself. |
| protected  int | [**next**](http://docs.google.com/java/security/SecureRandom.html#next(int))(int numBits)            Generates an integer containing the user-specified number of pseudo-random bits (right justified, with leading zeros). |
| void | [**nextBytes**](http://docs.google.com/java/security/SecureRandom.html#nextBytes(byte%5B%5D))(byte[] bytes)            Generates a user-specified number of random bytes. |
| void | [**setSeed**](http://docs.google.com/java/security/SecureRandom.html#setSeed(byte%5B%5D))(byte[] seed)            Reseeds this random object. |
| void | [**setSeed**](http://docs.google.com/java/security/SecureRandom.html#setSeed(long))(long seed)            Reseeds this random object, using the eight bytes contained in the given long seed. |

| **Methods inherited from class java.util.**[**Random**](http://docs.google.com/java/util/Random.html) |
| --- |
| [nextBoolean](http://docs.google.com/java/util/Random.html#nextBoolean()), [nextDouble](http://docs.google.com/java/util/Random.html#nextDouble()), [nextFloat](http://docs.google.com/java/util/Random.html#nextFloat()), [nextGaussian](http://docs.google.com/java/util/Random.html#nextGaussian()), [nextInt](http://docs.google.com/java/util/Random.html#nextInt()), [nextInt](http://docs.google.com/java/util/Random.html#nextInt(int)), [nextLong](http://docs.google.com/java/util/Random.html#nextLong()) |

| **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** |
| --- |

### SecureRandom

public **SecureRandom**()

Constructs a secure random number generator (RNG) implementing the default random number algorithm.

This constructor traverses the list of registered security Providers, starting with the most preferred Provider. A new SecureRandom object encapsulating the SecureRandomSpi implementation from the first Provider that supports a SecureRandom (RNG) algorithm is returned. If none of the Providers support a RNG algorithm, then an implementation-specific default 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.

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 RNG algorithm names.

The returned SecureRandom object has not been seeded. To seed the returned object, call the setSeed method. If setSeed is not called, the first call to nextBytes will force the SecureRandom object to seed itself. This self-seeding will not occur if setSeed was previously called.

### SecureRandom

public **SecureRandom**(byte[] seed)

Constructs a secure random number generator (RNG) implementing the default random number algorithm. The SecureRandom instance is seeded with the specified seed bytes.

This constructor traverses the list of registered security Providers, starting with the most preferred Provider. A new SecureRandom object encapsulating the SecureRandomSpi implementation from the first Provider that supports a SecureRandom (RNG) algorithm is returned. If none of the Providers support a RNG algorithm, then an implementation-specific default 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.

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 RNG algorithm names.

**Parameters:**seed - the seed.

### SecureRandom

protected **SecureRandom**([SecureRandomSpi](http://docs.google.com/java/security/SecureRandomSpi.html) secureRandomSpi,  
 [Provider](http://docs.google.com/java/security/Provider.html) provider)

Creates a SecureRandom object.

**Parameters:**secureRandomSpi - the SecureRandom implementation.provider - the provider.

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

### getInstance

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

Returns a SecureRandom object that implements the specified Random Number Generator (RNG) algorithm.

This method traverses the list of registered security Providers, starting with the most preferred Provider. A new SecureRandom object encapsulating the SecureRandomSpi 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.

The returned SecureRandom object has not been seeded. To seed the returned object, call the setSeed method. If setSeed is not called, the first call to nextBytes will force the SecureRandom object to seed itself. This self-seeding will not occur if setSeed was previously called.

**Parameters:**algorithm - the name of the RNG 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 RNG algorithm names. **Returns:**the new SecureRandom object. **Throws:** [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if no Provider supports a SecureRandomSpi implementation for the specified algorithm.**Since:** 1.2 **See Also:**[Provider](http://docs.google.com/java/security/Provider.html)

### getInstance

public static [SecureRandom](http://docs.google.com/java/security/SecureRandom.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 SecureRandom object that implements the specified Random Number Generator (RNG) algorithm.

A new SecureRandom object encapsulating the SecureRandomSpi 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.

The returned SecureRandom object has not been seeded. To seed the returned object, call the setSeed method. If setSeed is not called, the first call to nextBytes will force the SecureRandom object to seed itself. This self-seeding will not occur if setSeed was previously called.

**Parameters:**algorithm - the name of the RNG 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 RNG algorithm names.provider - the name of the provider. **Returns:**the new SecureRandom object. **Throws:** [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if a SecureRandomSpi 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.**Since:** 1.2 **See Also:**[Provider](http://docs.google.com/java/security/Provider.html)

### getInstance

public static [SecureRandom](http://docs.google.com/java/security/SecureRandom.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 SecureRandom object that implements the specified Random Number Generator (RNG) algorithm.

A new SecureRandom object encapsulating the SecureRandomSpi implementation from the specified Provider object is returned. Note that the specified Provider object does not have to be registered in the provider list.

The returned SecureRandom object has not been seeded. To seed the returned object, call the setSeed method. If setSeed is not called, the first call to nextBytes will force the SecureRandom object to seed itself. This self-seeding will not occur if setSeed was previously called.

**Parameters:**algorithm - the name of the RNG 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 RNG algorithm names.provider - the provider. **Returns:**the new SecureRandom object. **Throws:** [NoSuchAlgorithmException](http://docs.google.com/java/security/NoSuchAlgorithmException.html) - if a SecureRandomSpi 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 SecureRandom object.

**Returns:**the provider of this SecureRandom object.

### getAlgorithm

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

Returns the name of the algorithm implemented by this SecureRandom object.

**Returns:**the name of the algorithm or unknown if the algorithm name cannot be determined.**Since:** 1.5

### setSeed

public void **setSeed**(byte[] seed)

Reseeds this random object. The given seed supplements, rather than replaces, the existing seed. Thus, repeated calls are guaranteed never to reduce randomness.

**Parameters:**seed - the seed.**See Also:**[getSeed(int)](http://docs.google.com/java/security/SecureRandom.html#getSeed(int))

### setSeed

public void **setSeed**(long seed)

Reseeds this random object, using the eight bytes contained in the given long seed. The given seed supplements, rather than replaces, the existing seed. Thus, repeated calls are guaranteed never to reduce randomness.

This method is defined for compatibility with java.util.Random.

**Overrides:**[setSeed](http://docs.google.com/java/util/Random.html#setSeed(long)) in class [Random](http://docs.google.com/java/util/Random.html) **Parameters:**seed - the seed.**See Also:**[getSeed(int)](http://docs.google.com/java/security/SecureRandom.html#getSeed(int))

### nextBytes

public void **nextBytes**(byte[] bytes)

Generates a user-specified number of random bytes.

If a call to setSeed had not occurred previously, the first call to this method forces this SecureRandom object to seed itself. This self-seeding will not occur if setSeed was previously called.

**Overrides:**[nextBytes](http://docs.google.com/java/util/Random.html#nextBytes(byte%5B%5D)) in class [Random](http://docs.google.com/java/util/Random.html) **Parameters:**bytes - the array to be filled in with random bytes.

### next

protected final int **next**(int numBits)

Generates an integer containing the user-specified number of pseudo-random bits (right justified, with leading zeros). This method overrides a java.util.Random method, and serves to provide a source of random bits to all of the methods inherited from that class (for example, nextInt, nextLong, and nextFloat).

**Overrides:**[next](http://docs.google.com/java/util/Random.html#next(int)) in class [Random](http://docs.google.com/java/util/Random.html) **Parameters:**numBits - number of pseudo-random bits to be generated, where 0 <= numBits <= 32. **Returns:**an int containing the user-specified number of pseudo-random bits (right justified, with leading zeros).

### getSeed

public static byte[] **getSeed**(int numBytes)

Returns the given number of seed bytes, computed using the seed generation algorithm that this class uses to seed itself. This call may be used to seed other random number generators.

This method is only included for backwards compatibility. The caller is encouraged to use one of the alternative getInstance methods to obtain a SecureRandom object, and then call the generateSeed method to obtain seed bytes from that object.

**Parameters:**numBytes - the number of seed bytes to generate. **Returns:**the seed bytes.**See Also:**[setSeed(byte[])](http://docs.google.com/java/security/SecureRandom.html#setSeed(byte%5B%5D))

### generateSeed

public byte[] **generateSeed**(int numBytes)

Returns the given number of seed bytes, computed using the seed generation algorithm that this class uses to seed itself. This call may be used to seed other random number generators.

**Parameters:**numBytes - the number of seed bytes to generate. **Returns:**the seed bytes.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/SecureRandom.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/SecureClassLoader.html)   [**NEXT CLASS**](http://docs.google.com/java/security/SecureRandomSpi.html) | [**FRAMES**](http://docs.google.com/index.html?java/security/SecureRandom.html)    [**NO FRAMES**](http://docs.google.com/SecureRandom.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#1t3h5sf) | [METHOD](#3rdcrjn) |

[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).