System.Security.Cryptography Namespace

Reference

Feedback

In this article

Provides cryptographic services, including secure encoding and decoding of data, as well as many other operations, such as hashing, random number generation, and message authentication. For more information, see Cryptographic Services.

Classes

Expand table

Aes	Represents the abstract base class from which all implementations of the Advanced Encryption Standard (AES) must inherit.
AesCcm	Represents an Advanced Encryption Standard (AES) key to be used with the Counter with CBC-MAC (CCM) mode of operation.
AesCng	Provides a Cryptography Next Generation (CNG) implementation of the Advanced Encryption Standard (AES) algorithm.
AesCryptoServiceProvider	Performs symmetric encryption and decryption using the Cryptographic Application Programming Interfaces (CAPI) implementation of the Advanced Encryption Standard (AES) algorithm.
AesGcm	Represents an Advanced Encryption Standard (AES) key to be used with the Galois/Counter Mode (GCM) mode of operation.
AesManaged	Provides a managed implementation of the Advanced Encryption Standard (AES) symmetric algorithm.
AsnEncodedData	Represents Abstract Syntax Notation One (ASN.1)-encoded data.
AsnEncodedDataCollection	Represents a collection of AsnEncodedData objects. This class cannot be inherited.

AsnEncodedDataEnumerator	Provides the ability to navigate through an AsnEncodedDataCollection object. This class cannot be inherited.
AsymmetricAlgorithm	Represents the abstract base class from which all implementations of asymmetric algorithms must inherit.
AsymmetricKeyExchangeDeformatter	Represents the base class from which all asymmetric key exchange deformatters derive.
AsymmetricKeyExchangeFormatter	Represents the base class from which all asymmetric key exchange formatters derive.
AsymmetricSignatureDeformatter	Represents the abstract base class from which all implementations of asymmetric signature deformatters derive.
AsymmetricSignatureFormatter	Represents the base class from which all implementations of asymmetric signature formatters derive.
AuthenticationTagMismatchException	The exception that is thrown when a decryption operation with an authenticated cipher has an authentication tag mismatch.
ChaCha20Poly1305	Represents a symmetric key to be used with the ChaCha20 stream cipher in the combined mode with the Poly1305 authenticator.
CngAlgorithm	Encapsulates the name of an encryption algorithm.
CngAlgorithmGroup	Encapsulates the name of an encryption algorithm group.
CngKey	Defines the core functionality for keys that are used with Cryptography Next Generation (CNG) objects.
CngKeyBlobFormat	Specifies a key BLOB format for use with Microsoft Cryptography Next Generation (CNG) objects.

CngKeyCreationParameters	Contains advanced properties for key creation.
CngPropertyCollection	Provides a strongly typed collection of Cryptography Next Generation (CNG) properties.
CngProvider	Encapsulates the name of a key storage provider (KSP) for use with Cryptography Next Generation (CNG) objects.
CngUIPolicy	Encapsulates optional configuration parameters for the user interface (UI) that Cryptography Next Generation (CNG) displays when you access a protected key.
CryptoConfig	Accesses the cryptography configuration information.
CryptographicAttributeObject	Contains a type and a collection of values associated with that type.
CryptographicAttributeObjectCollection	Contains a set of CryptographicAttributeObject objects.
CryptographicAttributeObjectEnumerator	Provides enumeration functionality for the CryptographicAttributeObjectCollection collection. This class cannot be inherited.
CryptographicException	The exception that is thrown when an error occurs during a cryptographic operation.
CryptographicOperations	Provides methods for use in working with cryptography to reduce the risk of side-channel information leakage.
CryptographicUnexpectedOperationException	The exception that is thrown when an unexpected operation occurs during a cryptographic operation.
CryptoStream	Defines a stream that links data streams to cryptographic transformations.
CspKeyContainerInfo	Provides additional information about a cryptographic key pair. This class cannot be inherited.

CspParameters	Contains parameters that are passed to the cryptographic service provider (CSP) that performs cryptographic computations. This class cannot be inherited.
DeriveBytes	Represents the abstract base class from which all classes that derive byte sequences of a specified length inherit.
DES	Represents the base class for the Data Encryption Standard (DES) algorithm from which all DES implementations must derive.
DESCryptoServiceProvider	Defines a wrapper object to access the cryptographic service provider (CSP) version of the Data Encryption Standard (DES) algorithm. This class cannot be inherited.
DSA	Represents the abstract base class from which all implementations of the Digital Signature Algorithm (DSA) must inherit.
DSACng	Provides a Cryptography Next Generation (CNG) implementation of the Digital Signature Algorithm (DSA).
DSACryptoServiceProvider	Defines a wrapper object to access the cryptographic service provider (CSP) implementation of the DSA algorithm. This class cannot be inherited.
DSAOpenSsl	Provides an implementation of the Digital Signature Algorithm (DSA) backed by OpenSSL.
DSASignatureDeformatter	Verifies a Digital Signature Algorithm (DSA) PKCS#1 v1.5 signature.
DSASignatureFormatter	Creates a Digital Signature Algorithm (DSA) signature.
ECAlgorithm	Represents the abstract class from which elliptic-curve asymmetric algorithms can inherit.

ECCurve.NamedCurves	Represents a factory class for creating named curves.
ECDiffieHellman	Provides an abstract base class that Elliptic Curve Diffie-Hellman (ECDH) algorithm implementations can derive from. This class provides the basic set of operations that all ECDH implementations must support.
ECDiffieHellmanCng	Provides a Cryptography Next Generation (CNG) implementation of the Elliptic Curve Diffie-Hellman (ECDH) algorithm. This class is used to perform cryptographic operations.
ECDiffieHellmanCngPublicKey	Specifies an Elliptic Curve Diffie-Hellman (ECDH) public key for use with the ECDiffieHellmanCng class.
ECDiffieHellmanOpenSsl	Provides an implementation of the Elliptic Curve Diffie-Hellman (ECDH) algorithm backed by OpenSSL.
ECDiffieHellmanPublicKey	Provides an abstract base class from which all ECDiffieHellmanCngPublicKey implementations must inherit.
ECDsa	Provides an abstract base class that encapsulates the Elliptic Curve Digital Signature Algorithm (ECDSA).
ECDsaCng	Provides a Cryptography Next Generation (CNG) implementation of the Elliptic Curve Digital Signature Algorithm (ECDSA).
ECDsaOpenSsl	Provides an implementation of the Elliptic Curve Digital Signature Algorithm (ECDSA) backed by OpenSSL.
FromBase64Transform	Converts a CryptoStream from base 64.
HashAlgorithm	Represents the base class from which all implementations of cryptographic hash algorithms must derive.

HKDF	RFC5869 HMAC-based Extract-and-Expand Key Derivation (HKDF)
HMAC	Represents the abstract class from which all implementations of Hash-based Message Authentication Code (HMAC) must derive.
HMACMD5	Computes a Hash-based Message Authentication Code (HMAC) by using the MD5 hash function.
HMACSHA1	Computes a Hash-based Message Authentication Code (HMAC) using the SHA1 hash function.
HMACSHA256	Computes a Hash-based Message Authentication Code (HMAC) by using the SHA256 hash function.
HMACSHA3_256	Computes a Hash-based Message Authentication Code (HMAC) by using the SHA3-256 hash function.
HMACSHA3_384	Computes a Hash-based Message Authentication Code (HMAC) by using the SHA3-384 hash function.
HMACSHA3_512	Computes a Hash-based Message Authentication Code (HMAC) by using the SHA3-512 hash function.
HMACSHA384	Computes a Hash-based Message Authentication Code (HMAC) using the SHA384 hash function.
HMACSHA512	Computes a Hash-based Message Authentication Code (HMAC) using the SHA512 hash function.
IncrementalHash	Provides support for computing a hash or HMAC value incrementally across several segments.
KeyedHashAlgorithm	Represents the abstract class from which all implementations of keyed hash algorithms must derive.
KeySizes	Determines the set of valid key sizes for the symmetric cryptographic algorithms.

MaskGenerationMethod	Represents the abstract class from which all mask generator algorithms must derive.
MD5	Represents the abstract class from which all implementations of the MD5 hash algorithm inherit.
MD5CryptoServiceProvider	Computes the MD5 hash value for the input data using the implementation provided by the cryptographic service provider (CSP). This class cannot be inherited.
Oid	Represents a cryptographic object identifier. This class cannot be inherited.
OidCollection	Represents a collection of Oid objects. This class cannot be inherited.
OidEnumerator	Provides the ability to navigate through an OidCollection object. This class cannot be inherited.
PasswordDeriveBytes	Derives a key from a password using an extension of the PBKDF1 algorithm.
PbeParameters	Represents parameters to be used for Password-Based Encryption (PBE).
PemEncoding	Provides methods for reading and writing the IETF RFC 7468 subset of PEM (Privacy-Enhanced Mail) textual encodings. This class cannot be inherited.
PKCS1MaskGenerationMethod	Computes masks according to PKCS #1 for use by key exchange algorithms.
ProtectedData	Provides methods for encrypting and decrypting data. This class cannot be inherited.
RandomNumberGenerator	Provides functionality for generating random values.
RC2	Represents the base class from which all implementations of the RC2 algorithm must derive.

RC2CryptoServiceProvider	Defines a wrapper object to access the cryptographic service provider (CSP) implementation of the RC2 algorithm. This class cannot be inherited.
Rfc2898DeriveBytes	Implements password-based key derivation functionality, PBKDF2, by using a pseudo-random number generator based on HMACSHA1.
Rijndael	Represents the base class from which all implementations of the Rijndael symmetric encryption algorithm must inherit.
RijndaelManaged	Accesses the managed version of the Rijndael algorithm. This class cannot be inherited.
RNGCryptoServiceProvider	Implements a cryptographic Random Number Generator (RNG) using the implementation provided by the cryptographic service provider (CSP). This class cannot be inherited.
RSA	Represents the base class from which all implementations of the RSA algorithm inherit.
RSACng	Provides a Cryptography Next Generation (CNG) implementation of the RSA algorithm.
RSACryptoServiceProvider	Performs asymmetric encryption and decryption using the implementation of the RSA algorithm provided by the cryptographic service provider (CSP). This class cannot be inherited.
RSAEncryptionPadding	Specifies the padding mode and parameters to use with RSA encryption or decryption operations.
RSAOAEPKeyExchangeDeformatter	Decrypts Optimal Asymmetric Encryption Padding (OAEP) key exchange data.
RSAOAEPKeyExchangeFormatter	Creates Optimal Asymmetric Encryption Padding (OAEP) key exchange data using RSA.
RSAOpenSsl	Provides an implementation of the RSA algorithm backed by OpenSSL.

RSAPKCS1KeyExchangeDeformatter	Decrypts the PKCS #1 key exchange data.
RSAPKCS1KeyExchangeFormatter	Creates the PKCS#1 key exchange data using RSA.
RSAPKCS1SignatureDeformatter	Verifies an RSA PKCS #1 version 1.5 signature.
RSAPKCS1SignatureFormatter	Creates an RSA PKCS #1 version 1.5 signature.
RSASignaturePadding	Specifies the padding mode and parameters to use with RSA signature creation or verification operations.
SafeEvpPKeyHandle	Represents the `EVP_PKEY*` pointer type from OpenSSL.
SHA1	Computes the SHA1 hash for the input data.
SHA1CryptoServiceProvider	Computes the SHA1 hash value for the input data using the implementation provided by the cryptographic service provider (CSP). This class cannot be inherited.
SHA1Managed	Computes the SHA1 hash for the input data using the managed library.
SHA256	Computes the SHA256 hash for the input data.
SHA256CryptoServiceProvider	Defines a wrapper object to access the cryptographic service provider (CSP) implementation of the SHA256 algorithm.
SHA256Managed	Computes the SHA256 hash for the input data using the managed library.
SHA3_256	Computes the SHA3-256 hash for the input data.
SHA3_384	Computes the SHA3-384 hash for the input data.
SHA3_512	Computes the SHA3-512 hash for the input data.
SHA384	Computes the SHA384 hash for the input data.

SHA384CryptoServiceProvider	Defines a wrapper object to access the cryptographic service provider (CSP) implementation of the SHA384 algorithm.
SHA384Managed	Computes the SHA384 hash for the input data using the managed library.
SHA512	Computes the SHA512 hash for the input data.
SHA512CryptoServiceProvider	Defines a wrapper object to access the cryptographic service provider (CSP) implementation of the SHA512 algorithm.
SHA512Managed	Computes the SHA512 hash algorithm for the input data using the managed library.
Shake128	Computes the SHAKE128 hash for the input data.
Shake256	Computes the SHAKE256 hash for the input data.
SignatureDescription	Contains information about the properties of a digital signature.
SP800108HmacCounterKdf	NIST SP 800-108 HMAC CTR Key-Based Key Derivation (KBKDF)
SymmetricAlgorithm	Represents the abstract base class from which all implementations of symmetric algorithms must inherit.
ToBase64Transform	Converts a CryptoStream to base 64.
TripleDES	Represents the base class for Triple Data Encryption Standard algorithms from which all TripleDES implementations must derive.
TripleDESCng	Provides a Cryptography Next Generation (CNG) implementation of the Triple Data Encryption Standard (3DES) algorithm.
TripleDESCryptoServiceProvider	Defines a wrapper object to access the cryptographic service provider (CSP) version of the TripleDES algorithm. This class cannot be inherited.

Structs

Expand table

CngProperty	Encapsulates a property of a Cryptography Next Generation (CNG) key or provider.
DSAParameters	Contains the typical parameters for the DSA algorithm.
ECCurve	Represents an elliptic curve.
ECParameters	Represents the standard parameters for the elliptic curve cryptography (ECC) algorithm.
ECPoint	Represents a (X,Y) coordinate pair for elliptic curve cryptography (ECC) structures.
HashAlgorithmName	Specifies the name of a cryptographic hash algorithm.
PemFields	Contains information about the location of PEM data.
RSAParameters	Represents the standard parameters for the RSA algorithm.

Interfaces

Expand table

ICryptoTransform	Defines the basic operations of cryptographic transformations.
ICspAsymmetricAlgorithm	Defines methods that allow an AsymmetricAlgorithm class to enumerate key container information, and import and export Microsoft Cryptographic API (CAPI)-compatible key blobs.

Enums

Expand table

CipherMode	Specifies the block cipher mode to use for encryption.
CngExportPolicies	Specifies the key export policies for a key.
CngKeyCreationOptions	Specifies options used for key creation.
CngKeyHandleOpenOptions	Specifies options for opening key handles.
CngKeyOpenOptions	Specifies options for opening a key.
CngKeyUsages	Specifies the cryptographic operations that a Cryptography Next Generation (CNG) key may be used with.
CngPropertyOptions	Specifies Cryptography Next Generation (CNG) key property options.
CngUIProtectionLevels	Specifies the protection level for the key in user interface (UI) prompting scenarios.
CryptoStreamMode	Specifies the mode of a cryptographic stream.
CspProviderFlags	Specifies flags that modify the behavior of the cryptographic service providers (CSP).
DataProtectionScope	Specifies the scope of the data protection to be applied by the Protect(Byte[], Byte[], DataProtectionScope) method.
DSASignatureFormat	Specifies the data format for signatures with the DSA family of algorithms.
ECCurve.ECCurveType	Indicates how to interpret the data contained in an ECCurve object.
ECDiffieHellmanKeyDerivationFunction	Specifies the key derivation function that the ECDiffieHellmanCng class will use to convert secret agreements into key material.
ECKeyXmlFormat	Defines XML serialization formats for elliptic curve keys.

FromBase64TransformMode	Specifies whether white space should be ignored in the base 64 transformation.
KeyNumber	Specifies whether to create an asymmetric signature key or an asymmetric exchange key.
OidGroup	Identifies Windows cryptographic object identifier (OID) groups.
PaddingMode	Specifies the type of padding to apply when the message data block is shorter than the full number of bytes needed for a cryptographic operation.
PbeEncryptionAlgorithm	Specifies encryption algorithms to be used with Password-Based Encryption (PBE).
RSAEncryptionPaddingMode	Specifies the padding mode to use with RSA encryption or decryption operations.
RSASignaturePaddingMode	Specifies the padding mode to use with RSA signature creation or verification operations.