corda / net.corda.core.crypto / SignatureScheme

SignatureScheme

`data class SignatureScheme`

This class is used to define a digital signature scheme.

Parameters

`schemeNumberID` - we assign a number ID for more efficient on-wire serialisation. Please ensure uniqueness between schemes. `schemeCodeName` - code name for this signature scheme (e.g. RSA_SHA256, ECDSA_SECP256K1_SHA256, ECDSA_SECP256R1_SHA256, EDDSA_ED25519_SHA512, SPHINCS-256_SHA512). `signatureOID` - ASN.1 algorithm identifier of the signature algorithm (e.g 1.3.101.112 for EdDSA) `alternativeOIDs` - ASN.1 algorithm identifiers for keys of the signature, where we want to map multiple keys to the same signature scheme. `providerName` - the provider's name (e.g. "BC"). `algorithmName` - which signature algorithm is used (e.g. RSA, ECDSA. EdDSA, SPHINCS-256). `signatureName` - a signature-scheme name as required to create Signature objects (e.g. "SHA256withECDSA") `algSpec` - parameter specs for the underlying algorithm. Note that RSA is defined by the key size rather than algSpec. eg. ECGenParameterSpec("secp256k1"). `keySize` - the private key size (currently used for RSA only). `desc` - a human-readable description for this scheme.

Constructors

<init>	`SignatureScheme(schemeNumberID: Int, schemeCodeName: String, signatureOID: AlgorithmIdentifier, alternativeOIDs: List<AlgorithmIdentifier>, providerName: String, algorithmName: String, signatureName: String, algSpec: AlgorithmParameterSpec?, keySize: Int?, desc: String)` This class is used to define a digital signature scheme.

Properties

algSpec	`val algSpec: AlgorithmParameterSpec?`
algorithmName	`val algorithmName: String`
alternativeOIDs	`val alternativeOIDs: List<AlgorithmIdentifier>`
desc	`val desc: String`
keySize	`val keySize: Int?`
providerName	`val providerName: String`
schemeCodeName	`val schemeCodeName: String`
schemeNumberID	`val schemeNumberID: Int`
signatureName	`val signatureName: String`
signatureOID	`val signatureOID: AlgorithmIdentifier`