org.bouncycastle.crypto.signers

Class ECDSASigner

```
java.lang.Object
  └org.bouncycastle.crypto.signers.ECDSASigner
```

All Implemented Interfaces:

DSA, org.bouncycastle.math.ec.ECConstants

```
public class ECDSASigner
extends java.lang.Object
implements org.bouncycastle.math.ec.ECConstants, DSA
```

EC-DSA as described in X9.62

Field Summary

Fields inherited from interface org.bouncycastle.math.ec.ECConstants
`FOUR, ONE, THREE, TWO, ZERO`

Constructor Summary

`ECDSASigner()`

Method Summary

`java.math.BigInteger[]`	`generateSignature(byte[] message)` generate a signature for the given message using the key we were initialised with.
`void`	`init(boolean forSigning, CipherParameters param)` initialise the signer for signature generation or signature verification.
`boolean`	`verifySignature(byte[] message, java.math.BigInteger r, java.math.BigInteger s)` return true if the value r and s represent a DSA signature for the passed in message (for standard DSA the message should be a SHA-1 hash of the real message to be verified).

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Detail

ECDSASigner

```
public ECDSASigner()
```

Method Detail

init

```
public void init(boolean forSigning,
                 CipherParameters param)
```

Description copied from interface: DSA
initialise the signer for signature generation or signature verification.

Specified by:
init in interface DSA

Parameters:
forSigning - true if we are generating a signature, false otherwise.
param - key parameters for signature generation.

---

generateSignature

```
public java.math.BigInteger[] generateSignature(byte[] message)
```

generate a signature for the given message using the key we were initialised with. For conventional DSA the message should be a SHA-1 hash of the message of interest.

Specified by:
generateSignature in interface DSA

Parameters:
message - the message that will be verified later.
Returns:
two big integers representing the r and s values respectively.

---

verifySignature

```
public boolean verifySignature(byte[] message,
                               java.math.BigInteger r,
                               java.math.BigInteger s)
```

return true if the value r and s represent a DSA signature for the passed in message (for standard DSA the message should be a SHA-1 hash of the real message to be verified).

Specified by:
verifySignature in interface DSA

Parameters:

`message` - the message that was supposed to have been signed.
`r` - the r signature value.
`s` - the s signature value.