# LNCAToolkits 控件(通用版)程序员手册 v2.8.6.0





辽宁省数字证书认证中心
Liaoning Certificate Authority Center

地址：沈阳市和平区和平北大街 28 甲 3（110006）
TEL：024-23210366
FAX：024-23871490
Http://www.lnca.org.cn

# 目 录

# 1 概述

LNCA Toolkits ActiveX 控件，适用于客户端对数字证书操作，只需要注册一次，就可以随时调用，从而简化编程操作。

# 2 功能

# 3 注意事项

- 系统中的错误码均为负数。
- 默认签名操作使用 RSA_MD5 算法。
- 默认对称密钥产生使用 3DES 算法。
- 某些函数说明中有”吉大兼容”，即吉大的控件中有相同的功能。
- 本控件内部维护着一个证书数组，初始时除根证书以外，数组中的其它元素为均空，当成功地初始化证书后，数组中的相应元素保存证书对象。其中的编号顺序为：

  0 = 客户签名证书
  1 = 客户加密证书
  2 = 服务器签名证书
  3 = 服务器加密证书
  4 = 第三方签名证书
  5 = 第三方加密证书
  6 = 返回签名信封中的签名者证书
  7 = 返回加密信封中的接收人证书
  8 = 根证书
  9 = 时间戳包含的证书

控件注册 ID：6FBE853D-0DB0-4C62-B7DC-B49E9D447AF9
控件版本：2,8,6,0

# 4 算法说明

摘要算法：
szOID_OIWSEC_sha1 = "1.3.14.3.2.26" = CALG_SHA1
szOID_RSA_MD5 = "1.2.840.113549.2.5" = CALG_MD5
szOID_RSA_MD4 = "1.2.840.113549.2.4" = CALG_MD4
szOID_RSA_MD2 = "1.2.840.113549.2.2" = CALG_MD2

签名算法：
szOID_OIWSEC_sha1 = "1.3.14.3.2.26" = CALG_SHA1
szOID_RSA_MD5 = "1.2.840.113549.2.5" = CALG_MD5 (默认算法)
szOID_RSA_MD2 = "1.2.840.113549.2.2" = CALG_MD2

加密算法：(包含通用算法、专用算法两种)

1. 通用算法：根据系统软硬件算法支持有关。

szOID_RSA_DES_EDE3_CBC = “1.2.840.113549.3.7” = CALG_3DES (默认算法)
szOID_OIWSEC_desCBC = “1.3.14.3.2.7” = CALG_DES
szOID_RSA_RC4 = "1.2.840.113549.3.4" = CALG_RC4
szOID_RSA_RC2CBC = “1.2.840.113549.3.2”= CALG_RC2

2. 专用算法：由于系统默认不支持专用算法，因此，需要根据具体使用的 Key 硬件提供者支持。

szOID_SSF33_CBC = “1.2.840.113549.3.81” = CALG_SSF33

szOID_SCB2_SPECIAL_ECB = “1.2.840.113549.3.83” = CALG_SCB2_SPECIAL

# 5 使用方法

首先必须先注册 LNCAActiveX 控件组件。方法：开始→运行→regsvr32 “控件或组件的绝对路径”。


## 5.1 网页中调用

声明控件：

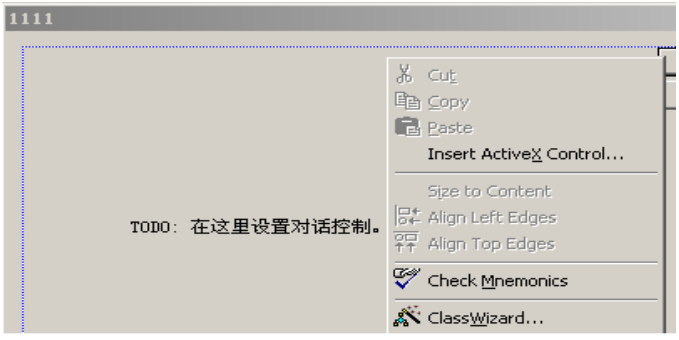
```
<OBJECT id="LNCAToolkits1"
      style="LEFT: 0px; WIDTH: 1px; TOP: 0px; HEIGHT: 1px"
      height="1" width="1"
      classid="clsid: 6FBE853D-0DB0-4C62-B7DC-B49E9D447AF9"
      CODEBASE="http://172.16.1.1/ LNCAToolkits.cab#version=2,8,6,0">
</OBJECT>
```

调用控件接口：

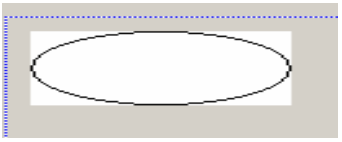
```
parent.TopFrame.document. LNCAToolkits1.InitClientCert
```

## 5.2 VC 中调用

新建 dialog 工程，再设置对话框控制窗口，单击右键选择 Insert ActiveX Control



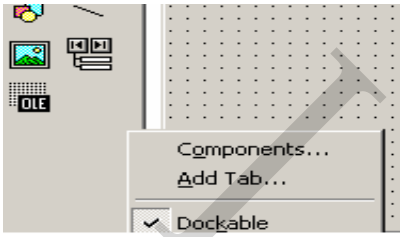
选择相应的控件名称后出现如下控件。设置 ID 后，设置变量值例如 m_LNCATookits。

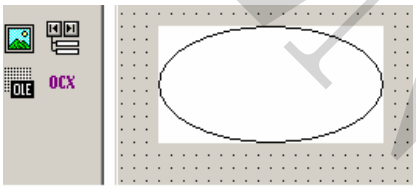


在程序中利用 m_ LNCATookits. InitClientCert();等调用控件接口。

## 5.3 VB 中调用

打开新工程，在工具条上空白处单击右键。选择 Components 再出现的对话框中选择相应控件工具条上出现 ocx 工具，单击此工具在 form 上拖动出现控件。设置控件名称 LNCATookits，在程序中就可以利用 LNCATookits. InitClientCert();等调用控件接口。




# 6 属性

这个控件提供的属性中签名、加密证书属性仅具有只读功能，其它属性同时具有读写功能。用户在使用签名、加密证书属性前，必须先初始化客户端证书。它提供以下功能属性。

## 6.1 属性简要描述

- ClientSignCertSN //吉大兼容
  客户端签名证书的序列号。
- ClientSignCertCN //吉大兼容
  客户端签名证书的公共名称。
- ClientSignCertDN

客户端签名证书的主题。

- ClientSignCertIssuer
  客户端签名证书的颁发者。
- ClientSignCert　　//吉大兼容
  客户端签名证书的 BASE64 编码。

- ClientEncCertSN
  客户端加密证书的序列号。
- ClientEncCertCN
  客户端加密证书的公共名称。
- ClientEncCertDN
  客户端签名证书的主题。
- ClientEncCertIssuer
  客户端加密证书的颁发者。
- ClientEncCert　　//吉大兼容
  客户端加密证书的 BASE64 编码。

- ErrorCode　　//吉大兼容
  错误码，保留程序运行时的最后一个错误代码。
- ErrorMessage　　//吉大兼容
  错误信息，保留程序运行时的最后一个错误信息。
- DNHashType
  获得或设置证书主题的摘要数据格式类型。
- ConfirmWriteFile
  获得或设置写本地磁盘文件时是否弹出确认窗口。
- OnlyOneConfirmWriteFile
  获得或设置写本地磁盘文件时是否只弹出一次确认窗口。
- InputDataType
  获得或设置输入的数据格式类型。
- OutputDataType
  获得或设置输出的数据格式类型。
- FilterCertCN
  获得或设置证书 CN 名称，选择证书时列出指定证书 CN 的证书。
- FilterCertEMail
  获得或设置证书 EMail 名称，选择证书时列出指定证书 EMail 的证书。
- UseDefaultCSP
  获得或设置是否使用缺省 CSP

## 6.2 客户端签名证书属性

### ClientSignCertSN //吉大兼容

属性声明： BSTR ClientSignCertSN;
说明：获取客户端签名证书的序列号

### ClientSignCertCN //吉大兼容

属性声明： BSTR ClientSignCertCN;
说明：获取客户端签名证书的公共名称

### ClientSignCertDN

属性声明： BSTR ClientSignCertDN;
说明：获取客户端签名证书的主题

### ClientSignCertIssuer

属性声明： BSTR ClientSignCertIssuer;
说明：获取客户端签名证书的颁发者

### ClientSignCert //吉大兼容

属性声明： BSTR ClientSignCert;
说明：获取客户端签名证书的内容(BASE64 编码)

## 6.3 客户端加密证书属性

### ClientEncCertSN

属性声明： BSTR ClientEncCertSN;
说明：获取客户端加密证书的序列号

### ClientEncCertCN

属性声明： BSTR ClientEncCertCN;
说明：获取客户端加密证书的公共名称

## ClientEncCertDN

属性声明： BSTR ClientEncCertDN;
说明：获取客户端加密证书的主题

## ClientEncCertIssuer

属性声明： BSTR ClientEncCertIssuer;
说明：获取客户端加密证书的颁发者

## ClientEncCert　　//吉大兼容

属性声明： BSTR ClientEncCert;
说明：获取客户端加密证书的内容(BASE64 编码)

# 6.4 其它属性

## ErrorCode　　//吉大兼容

属性声明： long ErrorCode;
说明：获取错误号，在程序出错的时候，调用本属性，0 表示成功；其他值为最后发生的错误号码

## ErrorMessage　　//吉大兼容

属性声明： BSTR ErrorMessage;
说明：获取错误消息，在程序出错的时候，调用本属性

## DNHashType

属性声明：
　　设置属性：void DNHashType (long newVal);
　　获得属性：long DNHashType;
说明：获得或设置证书主题的摘要数据格式类型，该接口属性是在调用 ReadCertDNHash() 方法时用于控制返回的摘要数据格式。
　　控件初始时，此属性值为 1。
　　1 = Base64 编码
　　2 = 十六进制编码

## ConfirmWriteFile

属性声明：

设置属性：void ConfirmWriteFile (long newVal);

获得属性：long ConfirmWriteFile;

说明：获得或设置写本地磁盘文件时是否弹出确认窗口，该接口属性是在调用所有写文件操作的方法时起作用。

控件初始时，此属性值为 1。

0 = 不弹出确认窗口

1 = 弹出确认窗口

## OnlyOneConfirmWriteFile

属性声明：

设置属性：void OnlyOneConfirmWriteFile (long newVal);

获得属性：long OnlyOneConfirmWriteFile;

说明：获得或设置写本地磁盘文件时是否只弹出一次确认窗口，该接口属性与上面一个属性配合使用，是在调用所有写文件操作的方法时起作用。

控件初始时，此属性值为 0。

0 = 当 ConfirmWriteFile 属性为 1 时，每次都弹出确认窗口

1 = 当 ConfirmWriteFile 属性为 1 时，只弹出一次确认窗口

## InputDataType

属性声明：

设置属性：void InputDataType (long newVal);

获得属性：long InputDataType;

说明：获得或设置输入的数据格式类型，在其他函数中用来配合控制输入文件数据的格式。

控件初始时，此属性值为 0。

0 = 输入二进制格式，不进行转码；

1 = 输入 BASE64 编码，进行转码。

## OutputDataType

属性声明：

设置属性：void OutputDataType (long newVal);

获得属性：long OutputDataType;

说明：获得或设置输出的数据格式类型，在其他函数中用来配合控制输出文件数据的格式。

控件初始时，此属性值为 0。

0 = 输出二进制格式，不进行转码；

1 = 输出 BASE64 编码，进行转码。

# FilterCertCN

属性声明：

设置属性：void FilterCertCN (BSTR newVal);

获得属性：BSTR FilterCertCN;

说明：获得或设置过滤证书的 CN 名称，选择证书时列出指定证书 CN 的证书。

在下列函数中用来配合控制过滤证书。

InitClientCert()，

InitClientCertEx()，

InitServerCertFromStore()，

InitServerCertFromStoreEx()，

InitThirdCertFromStore()，

InitThirdCertFromStoreEx()

控件初始时，此属性值为 NULL，不进行证书过滤。

# FilterCertEMail

属性声明：

设置属性：void FilterCertEMail (BSTR newVal);

获得属性：BSTR FilterCertEMail;

说明：获得或设置过滤证书的 EMail 名称，选择证书时列出指定证书 EMail 的证书。

在下列函数中用来配合控制过滤证书。

InitClientCert()，

InitClientCertEx()，

InitServerCertFromStore()，

InitServerCertFromStoreEx()，

InitThirdCertFromStore()，

InitThirdCertFromStoreEx()

控件初始时，此属性值为 NULL，不进行证书过滤。

# UseDefaultCSP

属性声明：

设置属性：void UseDefaultCSP (long newVal);

获得属性：long UseDefaultCSP;

说明：获得或设置是否使用操作系统缺省 CSP。

在下列函数中用来配合控制加密函数时的加密源方式。

EncryptDataEx ()，

EncryptData ()，

EncryptFileEx ()，

SignEnvDataEx ()，

SignEncFileEx ()，

SymCipherInit()

控件初始时，此属性值为 1，使用操作系统缺省 CSP。

0 = 使用本人的证书 CSP；

1 = 使用操作系统缺省 CSP。

# 7 方法

## 7.1 证书初始化方法

### EnumClientCert

函数声明：long EnumClientCert();

说明：从微软证书库枚举客户端双证书，如果证书库中有不是一对双证书，则显示证书列表窗口，通过人工选择来初始化证书，并且此证书必须包含私钥。在选择证书时，所显示的证书默认为加密证书。

参数：

返回值：成功时返回 0，证书对象保存在证书数组[0]和[1]。

失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

### EnumClientCertEx

函数声明：long EnumClientCertEx(long FilterCertType);

说明：从微软证书库枚举客户端双证书，如果证书库中有不是一对双证书，则显示证书列表窗口，通过人工选择来初始化证书，并且此证书必须包含私钥。

参数：

- FilterCertType：当人工选择证书时，在证书选择窗体中显示的证书类型

  可选值：0 = 显示所有证书

  1 = 只显示签名证书

  2 = 只显示加密证书

返回值：成功时返回 0，证书对象保存在证书数组[0]和[1]。

失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## InitClientCert　　//吉大兼容

函数声明：long InitClientCert();
说明：从微软证书库选择初始化客户端双证书，并且此证书必须包含私钥。在选择证书时，所显示的证书默认为加密证书。
参数：
返回值：成功时返回 0，证书对象保存在证书数组[0]和[1]。
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## InitClientCertEx

函数声明：long InitClientCertEx(long FilterCertType);
说明：从微软证书库选择初始化客户端双证书，并且此证书必须包含私钥。
参数：

- FilterCertType：当人工选择证书时，在证书选择窗体中显示的证书类型
  可选值：0 = 显示所有证书
  1 = 只显示签名证书
  2 = 只显示加密证书

返回值：成功时返回 0，证书对象保存在证书数组[0]和[1]。
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## InitClientCertByCertSN　　//吉大兼容

函数声明：long InitClientCertByCertSN(BSTR szSN);
说明：通过证书序列号初始化客户端双证书，并且此证书必须包含私钥。
参数：

- szSN：客户端的证书序列号

返回值：成功时返回 0，证书对象保存在证书数组[0]和[1]。
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## InitClientCertByCertDN

函数声明：long InitClientCertByCertDN(BSTR szDN);

说明：通过证书主题初始化客户端双证书，并且此证书必须包含私钥。

参数：

- szDN：客户端的主题

返回值：成功时返回 0，证书对象保存在证书数组[0]和[1]。

失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## InitClientCertByCertCode

函数声明：long InitClientCertByCertCode(BSTR szCertCode);

说明：通过公钥证书 BASE64 编码初始化客户端双证书，并且此证书必须包含私钥。

参数：

- szCertCode：公钥证书 BASE64 编码

返回值：成功时返回 0，证书对象保存在证书数组[0]和[1]。

失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## InitServerCertFromStore

函数声明：long InitServerCertFromStore();

说明：从系统证书库中选择服务器（对方）双证书，在选择证书时，所显示的证书默认为加密证书。

参数：

返回值：成功时返回 0，证书对象保存在证书数组[2]和[3]。

失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## InitServerCertFromStoreEx

函数声明：long InitServerCertFromStoreEx(long FilterCertType);

说明：从系统证书库中选择服务器（对方）双证书

参数：

- FilterCertType：当人工选择证书时，在证书选择窗体中显示的证书类型

  可选值：0 = 显示所有证书

  1 = 只显示签名证书

  2 = 只显示加密证书

返回值：成功时返回 0，证书对象保存在证书数组[2]和[3]。

失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# InitThirdCertFromStore

函数声明：long InitThirdCertFromStore();
说明：从系统证书库中选择第三方双证书，在选择证书时，所显示的证书默认为加密证书。
参数：
返回值：成功时返回 0，证书对象保存在证书数组[4]和[5]。
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# InitThirdCertFromStoreEx

函数声明：long InitThirdCertFromStoreEx(long FilterCertType);
说明：从系统证书库中选择第三方双证书
参数：

- FilterCertType：当人工选择证书时，在证书选择窗体中显示的证书类型
  可选值：0 = 显示所有证书
  1 = 只显示签名证书
  2 = 只显示加密证书

返回值：成功时返回 0，证书对象保存在证书数组[4]和[5]。
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# InitServerCertFromFile

函数声明：long InitServerCertFromFile(BSTR szSignCertFile,
BSTR szEncCertFile,
long IsBase64);
说明：从指定公钥证书文件初始化服务器（对方）双证书。
参数：

- szSignCertFile：服务器（对方）签名证书文件
- szEncCertFile：服务器（对方）加密证书文件
- IsBase64：文件编码格式
  0 = 二进制编码，1 = BASE64 编码

返回值：成功时返回 0，证书对象保存在证书数组[2]和[3]。
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# InitThirdCertFromFile

函数声明：long InitThirdCertFromFile(BSTR szSignCertFile,
BSTR szEncCertFile,
long IsBase64);
说明：从指定公钥证书文件初始化第三方（对方）双证书。
参数：

- szSignCertFile：第三方（对方）签名证书文件
- szEncCertFile：第三方（对方）加密证书文件
- IsBase64：文件编码格式
  0 = 二进制编码，1 = BASE64 编码

返回值：成功时返回 0，证书对象保存在证书数组[4]和[5]。
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# InitServerCert　　//吉大兼容

函数声明： long InitServerCert(BSTR szSignCert, BSTR szEncCert);
说明：根据公钥证书数据(BASE64 编码)初始化服务器（对方）证书
参数：

- szSignCert：服务器（对方）签名证书内容(BASE64 编码)
- szEncCert：服务器（对方）加密证书内容(BASE64 编码)

返回值：成功时返回 0=初始化双证书，1=初始化签名证书，2=初始化加密证书。证书对象保存在证书数组[2]和[3]。
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# InitThirdCert

函数声明： long InitThirdCert(BSTR szSignCert, BSTR szEncCert);
说明：根据公钥证书数据(BASE64 编码)初始化第三方证书
参数：

- szSignCert：第三方签名公钥证书内容(BASE64 编码)
- szEncCert：第三方加密公钥证书内容(BASE64 编码)

返回值：成功时返回 0=初始化双证书，1=初始化签名证书，2=初始化加密证书。证书对象保存在证书数组[4]和[5]。
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## FreeEncCertArray

函数声明： long FreeEncCertArray ();
说明：清空多人加密公钥证书数组
参数：
返回值：成功时返回 0。
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## AddEncCert

函数声明： long AddEncCert(BSTR sEncCertBase64Data);
说明：通过数据(BASE64 编码)添加多人加密公钥证书
参数：

- sEncCertBase64Data：加密公钥证书数据(BASE64 编码)

返回值：成功时返回加密证书数组元素总数。
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## AddEncCertFromFile

函数声明： long AddEncCertFromFile(BSTR sEncCertFile, long IsBase64);
说明：通过文件名添加多人加密公钥证书
参数：

- sEncCertFile：加密公钥证书文件名(全路径)
- IsBase64：证书文件编码格式
  0 = 二进制编码， 1 = BASE64 编码

返回值：成功时返回加密证书数组元素总数。
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# 7.2 证书解析方法

## ReadCertSN

函数声明： BSTR ReadCertSN(long nCertIndex);
说明：获取指定证书的序列号
参数：

- nCertIndex：输入证书数组序号

返回值：成功时返回证书序列号的十六进制编码，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReadCertCN

函数声明： BSTR ReadCertCN(long nCertIndex);
说明：获取指定证书的公共名称
参数：

- nCertIndex：输入证书数组序号

返回值：成功时返回证书的公共名称，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReadCertDN

函数声明： BSTR ReadCertDN(long nCertIndex);
说明：获取指定证书的主题
参数：

- nCertIndex：输入证书数组序号

返回值：成功时返回证书的主题，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReadCertDNHash

函数声明： BSTR ReadCertDNHash(BSTR Base64CertCode, BSTR HashAlgo);
说明：获取指定证书的主题的哈希值，可用此哈希值做用户证书的标识符。
参数：

- Base64CertCode： 输入公钥证书 BASE64 编码数据
- HashAlgo： 哈希算法 OID

| | | |
|---|---|---|
| SHA-1: | "1.3.14.3.2.26" | 产生哈希值长度：40字节 |
| MD5: | "1.2.840.113549.2.5" | 产生哈希值长度：32字节 |
| MD4: | "1.2.840.113549.2.4" | 产生哈希值长度：32字节 |
| MD2: | "1.2.840.113549.2.2" | 产生哈希值长度：32字节 |

返回值：成功时返回证书的主题的哈希值十六进制编码，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# ReadCertIssuer

函数声明： BSTR ReadCertIssuer(long nCertIndex);
说明：获取指定证书的颁发者
参数：

- nCertIndex：输入证书数组序号

返回值：成功时返回证书的颁发者，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# ReadCert

函数声明： BSTR ReadCert(long nCertIndex);
说明：获取指定证书的内容(BASE64 编码)
参数：

- nCertIndex：输入证书数组序号

返回值：成功时返回证书内容的 BASE64 编码，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# ReadCertUsage

函数声明： long ReadCertUsage (long nCertIndex);
说明：获取证书密钥用法
签名证书：返回 128
加密证书：返回 48
无密钥用法：返回 255
参数：

- nCertIndex：输入证书数组序号

返回值：成功时返回证书密钥用法
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# ReadCertUsageStr

函数声明： BSTR ReadCertUsageStr(long nCertIndex);
说明：获取证书密钥用法字符串
签名证书：返回"Digital Signature(80)"

加密证书：返回"Key Encipherment, Data Encipherment(30)"
无密钥用法：返回""

参数：

- nCertIndex：输入证书数组序号

返回值：成功时返回证书密钥用法字符串，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReadCertBeginValidityTime

函数声明： BSTR ReadCertBeginValidityTime (long nCertIndex);
说明：获取证书的起始有效期字符串(格式 YYYY-MM-DD hh:mm:ss)
参数：

- nCertIndex：输入证书数组序号

返回值：成功时返回证书起始有效期字符串，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReadCertEndValidityTime

函数声明： BSTR ReadCertEndValidityTime (long nCertIndex);
说明：获取证书的终止有效期字符串(格式 YYYY-MM-DD hh:mm:ss)
参数：

- nCertIndex：输入证书数组序号

返回值：成功时返回证书终止有效期字符串，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReadCertExtension

函数声明： BSTR ReadCertExtension(BSTR szOID, long nCertIndex);
说明：从选择的证书获得证书扩展信息
参数：

- szOID：证书扩展域

可选值: 1.2.86.11.7.1 个人身份识别码
1.2.86.11.7.2 个人社会保险号
1.2.86.11.7.3 企业机构代码证号
1.2.86.11.7.4 企业工商注册号

1.2.86.11.7.5 企业(国税)税号
1.2.86.77.1 企业(地税)税号

- nCertIndex：输入证书数组序号

返回值：成功时返回证书扩展域信息，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReadCertExtensionCount

函数声明： long ReadCertExtensionCount( long nCertIndex);
说明：从选择的证书获得证书扩展域总数
参数：
- nCertIndex：输入证书数组序号

返回值：成功时返回证书扩展域总数>=0，
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReadCertExtensionOID

函数声明： BSTR ReadCertExtensionOID( long nIdx);
说明：从选择的证书获得证书扩展域包含的 OID，
前提条件：已经成功地调用 ReadCertExtensionCount()函数
参数：
- nIdx：输入证书扩展域序号(0--证书扩展域总数-1)

返回值：成功时返回证书扩展域 OID，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReadCertExtensionValue

函数声明： BSTR ReadCertExtensionValue( BSTR szOID);
说明：从选择的证书获得证书扩展域包含的值，
前提条件：已经成功地调用 ReadCertExtensionCount()函数
参数：
- szOID：输入证书扩展域 OID

可选值: 1.2.86.11.7.1 个人身份识别码
1.2.86.11.7.2 个人社会保险号
1.2.86.11.7.3 企业机构代码证号

1.2.86.11.7.4 企业工商注册号
1.2.86.11.7.5 企业(国税)税号
1.2.86.77.1 企业(地税)税号

返回值：成功时返回返回证书扩展域值，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReadCertProviderName

函数声明： BSTR ReadCertProviderName (long nCertIndex);
说明：获取指定证书提供者名称
参数：

- nCertIndex：输入证书数组序号

返回值：成功时返回证书提供者名称，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## CompareCertProviderName

函数声明： long CompareCertProviderName (long nCertIndex, BSTR CompareStr);
说明：比较指定证书提供者名称
参数：

- nCertIndex：输入证书数组序号
- CompareStr：输入要比较的证书提供者名称

返回值：一致时返回 0，不一致时返回>0，
调用失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# 7.3 证书主题解析方法

## ReadCertDN_OU

函数声明： BSTR ReadCertDN_OU(long nCertIndex);
说明：获取指定证书的主题 OU(组织单位)
参数：

- nCertIndex：输入证书数组序号

返回值：成功时返回证书的主题 OU(组织单位)，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReadCertDN_OUa

函数声明： BSTR ReadCertDN_OUa(long nCertIndex);
说明：获取指定证书的主题 OU@(企业代码证号)
参数：

- nCertIndex：输入证书数组序号

返回值：成功时返回指定证书的主题 OU@(企业代码证号)，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReadCertDN_OUi

函数声明： BSTR ReadCertDN_OUi(long nCertIndex);
说明：获取指定证书的主题 OU!(备注项)
参数：

- nCertIndex：输入证书数组序号

返回值：成功时返回指定证书的主题 OU!(备注项)，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReadCertDN_O

函数声明： BSTR ReadCertDN_O(long nCertIndex);
说明：获取指定证书的主题 O(组织)
参数：

- nCertIndex：输入证书数组序号

返回值：成功时返回指定证书的主题 O(组织)，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReadCertDN_L

函数声明： BSTR ReadCertDN_L(long nCertIndex);
说明：获取指定证书的主题 L(市)

参数：

- nCertIndex：输入证书数组序号

返回值：成功时返回指定证书的主题 L(市)，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReadCertDN_S

函数声明： BSTR ReadCertDN_S(long nCertIndex);
说明：获取指定证书的主题 S(省)
参数：

- nCertIndex：输入证书数组序号

返回值：成功时返回指定证书的主题 S(省)，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReadCertDN_C

函数声明： BSTR ReadCertDN_C(long nCertIndex);
说明：获取指定证书的主题 C(国家)
参数：

- nCertIndex：输入证书数组序号

返回值：成功时返回指定证书的主题 C(国家)，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReadCertDN_E

函数声明： BSTR ReadCertDN_E(long nCertIndex);
说明：获取指定证书的主题 E(Email)
参数：

- nCertIndex：输入证书数组序号

返回值：成功时返回指定证书的主题 E(Email)，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## 7.4 证书操作

### CertInitStatus

函数声明：long CertInitStatus (long Index);
说明：判断控件内指定证书是否已初始化。
参数：

- Index：证书索引号

返回值：成功返回 0，
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

### ViewCertWnd

函数声明：long ViewCertWnd(long Index);
说明：显示指定的证书窗口，前提条件：已经成功初始化证书。
参数：

- Index：证书索引号

返回值：成功返回 0，
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

### RemoveCert

函数声明：long RemoveCert (long Index);
说明：清除控件内证书数组中指定的证书，证书数组中根证书:下标[8]除外。前提条件：已经成功初始化证书。
参数：

- Index：证书索引号

返回值：成功返回 0，
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## 7.5 CRL 操作

### InitCRL

函数声明：long InitCRL (BSTR CRLBase64Data);
说明：通过数据(BASE64 编码)初始化 CRL。
参数：

- CRLBase64Data： CRL 数据(BASE64 编码)

返回值：成功时返回 0，
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。
提示：还可以通过从辽宁 CA 的 LDAP 服务器来初始化 CRL。参见 ReadCRLFromLdap。

### VerifyCRL

函数声明：long VerifyCRL (BSTR CRLBase64Data, long IsVerifyTime, BSTR sSystemTime);
说明：验证 CRL 有效性。如果验证成功，同时初始化 CRL 对象，CRL 数据可以从 LDAP 服务器下载。参见 ReadCRLFromLdap。
参数：

- CRLBase64Data：CRL 数据(BASE64 编码)
- IsVerifyTime：是否校验 CRL 时间有效性
  0 = 不校验 CRL 时间
  1 = 校验 CRL 时间，并且下一个参数有效
- sSystemTime：外部输入要验证的日期和时间
  输入格式必须为: yyyy.mm.dd hh:mm:ss 或 yyyy.m.d. h:m:s，其中日期与时间之间用空格隔开，如果为"" = 取本机系统时间

返回值：有效返回 0，否则返回-1：未启用无效，1：超期无效
失败时返回其它值，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

### CRLVerifyCert

函数声明：long CRLVerifyCert (long nCertIndex);
说明：使用 CRL 校验证书，前提条件：已经成功地初始化 CRL 对象。
参数：

- Index：证书索引号

返回值：证书有效返回 0，
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信

息。

## 7.6 哈希操作方法

### HashData

函数声明：BSTR HashData(BSTR sSourceData, BSTR HashAlgo);
说明：哈希数据操作
参数：

- sSourceData：原文数据
- HashAlgo： 哈希算法

szOID_OIWSEC_sha1 = “1.3.14.3.2.26” 产生 28 字节(BASE64 编码)
szOID_RSA_MD5 = “1.2.840.113549.2.5” 产生 24 字节(BASE64 编码)
szOID_RSA_MD4 = “1.2.840.113549.2.4” 产生 24 字节(BASE64 编码)
szOID_RSA_MD2 = “1.2.840.113549.2.2” 产生 24 字节(BASE64 编码)

返回值：成功时返回 BASE64 编码哈希值，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

### VerifyHashData

函数声明：long VerifyHashData (BSTR sSourceData, BSTR sHashData, BSTR HashAlgo);
说明：验证哈希数据操作
参数：

- sSourceData：原文数据
- sHashData： 原 BASE64 编码哈希值
- HashAlgo： 哈希算法

: szOID_OIWSEC_sha1 = “1.3.14.3.2.26”
szOID_RSA_MD5 = “1.2.840.113549.2.5”
szOID_RSA_MD4 = “1.2.840.113549.2.4”
szOID_RSA_MD2 = “1.2.840.113549.2.2”

返回值：成功时返回 0，
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

### HashFile

函数声明：BSTR HashFile(BSTR sSourceFile, BSTR HashAlgo);
说明：哈希文件操作
参数：

- sSourceFile： 原文文件
- HashAlgo： 哈希算法

  szOID_OIWSEC_sha1 = “1.3.14.3.2.26” 产生 28 字节(BASE64 编码)
  szOID_RSA_MD5 = “1.2.840.113549.2.5” 产生 24 字节(BASE64 编码)
  szOID_RSA_MD4 = “1.2.840.113549.2.4” 产生 24 字节(BASE64 编码)
  szOID_RSA_MD2 = “1.2.840.113549.2.2” 产生 24 字节(BASE64 编码)

其中：通过 InputDataType 属性来指定原文文件格式
0 = 输入原文文件为二进制编码，此函数内部不进行转码
1 = 输入原文文件为 BASE64 编码，此函数内部进行转码

返回值：成功时返回 BASE64 编码哈希值，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## VerifyHashFile

函数声明：long VerifyHashFile (BSTR sSourceFile, BSTR sHashData, BSTR HashAlgo);
说明：验证哈希文件操作
参数：

- sSourceFile： 原文文件
- sHashData： 原 BASE64 编码哈希值
- HashAlgo： 哈希算法

:
szOID_OIWSEC_sha1 = “1.3.14.3.2.26”
szOID_RSA_MD5 = “1.2.840.113549.2.5”
szOID_RSA_MD4 = “1.2.840.113549.2.4”
szOID_RSA_MD2 = “1.2.840.113549.2.2”

其中：通过 InputDataType 属性来指定原文文件格式
0 = 输入原文文件为二进制编码，此函数内部不进行转码
1 = 输入原文文件为 BASE64 编码，此函数内部进行转码

返回值：成功时返回 0，
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# 7.7 哈希签名操作方法

## SignHash

函数声明：BSTR SignHash (BSTR sSourceData, BSTR HashAlgo, BSTR pPin);

说明：进行哈希签名数据操作 (使用客户端证书)

参数：

- sSourceData：原文数据
- HashAlgo： 哈希算法

  szOID_OIWSEC_sha1 = “1.3.14.3.2.26” 产生 176 字节(BASE64 编码)

  szOID_RSA_MD5 = “1.2.840.113549.2.5” 产生 176 字节(BASE64 编码)

  szOID_RSA_MD2 = “1.2.840.113549.2.2” 产生 176 字节(BASE64 编码)
- pPin：用户 Key 口令

如果输入正确的口令，则不弹出输入口令窗口，直接签名哈希数据。

如果输入错误的口令，则弹出输入口令窗口

返回值：成功时返回 BASE64 编码签名哈希值，

失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## VerifySignHash

函数声明：long VerifySignHash (BSTR sSourceData, BSTR sHashData, BSTR HashAlgo, long nCertIndex);

说明：验证哈希签名数据操作

参数：

- sSourceData：原文数据
- sHashData： 原 BASE64 编码哈希签名值
- HashAlgo： 哈希算法

  szOID_OIWSEC_sha1 = “1.3.14.3.2.26”

  szOID_RSA_MD5 = “1.2.840.113549.2.5”

  szOID_RSA_MD2 = “1.2.840.113549.2.2”
- nCertIndex： 指定验证用的公钥证书

返回值：成功时返回 0，

失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## SignHashFile

函数声明：BSTR SignHashFile (BSTR sSourceFile, BSTR HashAlgo, BSTR pPin);

说明：进行哈希签名文件操作 (使用客户端证书)

参数：

- sSourceFile： 原文文件
- HashAlgo： 哈希算法

  szOID_OIWSEC_sha1 = “1.3.14.3.2.26” 产生 176 字节(BASE64 编码)

  szOID_RSA_MD5 = “1.2.840.113549.2.5” 产生 176 字节(BASE64 编码)

  szOID_RSA_MD2 = “1.2.840.113549.2.2” 产生 176 字节(BASE64 编码)

- pPin：用户 Key 口令

如果输入正确的口令，则不弹出输入口令窗口，直接签名哈希数据。
如果输入错误的口令，则弹出输入口令窗口

其中：通过 InputDataType 属性来指定原文文件格式
0 = 输入原文文件为二进制编码，此函数内部不进行转码
1 = 输入原文文件为 BASE64 编码，此函数内部进行转码

返回值：成功时返回 BASE64 编码签名哈希值，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## VerifySignHashFile

函数声明：long VerifySignHashFile (BSTR sSourceFile,
BSTR sHashData,
BSTR HashAlgo,
long nCertIndex);

说明：验证哈希签名文件操作

参数：

- sSourceFile： 原文文件
- sHashData： 原 BASE64 编码哈希签名值
- HashAlgo： 哈希算法
  szOID_OIWSEC_sha1 = “1.3.14.3.2.26”
  szOID_RSA_MD5 = “1.2.840.113549.2.5”
  szOID_RSA_MD2 = “1.2.840.113549.2.2”
- nCertIndex： 指定验证用的公钥证书

其中：通过 InputDataType 属性来指定原文文件格式
0 = 输入原文文件为二进制编码，此函数内部不进行转码
1 = 输入原文文件为 BASE64 编码，此函数内部进行转码

返回值：成功时返回 0，
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# 7.8 签名操作方法

## SignFromHash

函数声明：BSTR SignFromHash (BSTR sHashData, BSTR HashAlgo, BSTR pPin);

说明：对哈希数据进行签名操作 (使用客户端证书)

参数：

- sSourceData：哈希数据
- HashAlgo： 哈希算法

  szOID_OIWSEC_sha1 = “1.3.14.3.2.26” 产生 176 字节(BASE64 编码)

  szOID_RSA_MD5 = “1.2.840.113549.2.5” 产生 176 字节(BASE64 编码)

  szOID_RSA_MD2 = “1.2.840.113549.2.2” 产生 176 字节(BASE64 编码)
- pPin：用户 Key 口令

如果输入正确的口令，则不弹出输入口令窗口，直接签名哈希数据。

如果输入错误的口令，则弹出输入口令窗口

返回值：成功时返回 BASE64 编码签名哈希值，

失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## VerifySignFromHash

函数声明：long VerifySignFromHash (BSTR sHashData, BSTR sSignData, BSTR HashAlgo, long nCertIndex);

说明：用哈希数据验证签名操作

参数：

- sHashData：哈希数据
- sSignData： 原 BASE64 编码签名值
- HashAlgo： 哈希算法

  szOID_OIWSEC_sha1 = “1.3.14.3.2.26”

  szOID_RSA_MD5 = “1.2.840.113549.2.5”

  szOID_RSA_MD2 = “1.2.840.113549.2.2”
- nCertIndex： 指定验证用的公钥证书

返回值：成功时返回 0，

失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# 7.9 PKCS7 签名数据操作方法

## SignDataEx

函数声明： BSTR SignDataEx (BSTR szSrc,

BSTR sSignAlgo,

long IsAddSignCert,

long IsAddSrcData,

BSTR szInnerOid,
long IsAddTime,
BSTR pPin);

说明：进行签名数据操作(使用客户端证书)。

参数：

- szSrc ：原文数据
- sSignAlgo：指定签名算法
  szOID_OIWSEC_sha1 = “1.3.14.3.2.26”
  szOID_RSA_MD5 = “1.2.840.113549.2.5”
  szOID_RSA_MD2 = “1.2.840.113549.2.2”
- IsAddSignCert 是否在结果中携带证书
  0 = 不携带证书
  1 = 携带证书
- IsAddSrcData 是否在结果中携带原文
  0 = 不携带原文
  1 = 携带原文
- szInnerOid：数据类型 OID，(默认:NULL)
  szOID_TSP_TSTInfo = "1.2.840.113549.1.9.16.1.4"
- IsAddTime:是否添加签名时间
  0: 不进行时间编码
  1: 取当前系统时间，进行时间编码
  2、从时间戳服务器取得时间，必须首先设置时间戳服务器 URL。请参考 7.14 章节的时间戳操作。
- pPin：用户 Key 口令

如果输入正确的口令，则不弹出输入口令窗口，直接签名数据。
如果输入错误的口令，则弹出输入口令窗口

其中：通过 InputDataType 属性来指定原文数据格式
0 = 输入原文为二进制编码，此函数内部不进行转码
1 = 输入原文为 BASE64 编码，此函数内部进行转码

返回值：成功时返回签名数据(BASE64 编码)，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## SignData //吉大兼容

函数声明： BSTR SignData(BSTR szSrc);

说明：进行签名数据操作（携带证书，携带原文）(使用客户端证书)。
证书如有私钥保护口令，弹出输入口令窗口，手工输入私钥保护口令

参数：

- szSrc ：原文数据

其他默认参数: 签名算法:szOID_RSA_MD5 = “1.2.840.113549.2.5”,

携带证书,
携带原文,
数据类型 OID=NULL,
是否附加时间编码: 0: 不进行时间编码

其中：通过 InputDataType 属性来指定原文数据格式
0 = 输入原文为二进制编码，此函数内部不进行转码
1 = 输入原文为 BASE64 编码，此函数内部进行转码

返回值：成功时返回签名数据(BASE64 编码)，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## VerifySignEx

函数声明： BSTR VerifySignEx (BSTR szSrc, BSTR szSign, long nCertIndex);
说明：验证签名数据操作。验证由 SignData 或者 SignDataEx 生成的签名数据。
参数：

- szSrc：原文数据(如果为空，表示此签名数据中携带原文，否则输入原文数据
- szSign：签名数据(BASE64 编码)
- nCertIndex：指定使用的验证签名公钥证书
  0 = 自己的签名公钥证书
  2 = 服务器的公钥证书
  4 = 第三方的公钥证书
  -1 = 一个特例，即不用指定证书，签名数据内包含签名者证书

其中：通过 InputDataType 属性来指定原文数据格式
0 = 输入原文数据为二进制编码，此函数内部不进行转码
1 = 输入原文数据为 BASE64 编码，此函数内部进行转码

通过 OutputDataType 属性来指定返回数据格式
0 = 输出返回数据为二进制编码，此函数内部不进行转码
1 = 输出返回数据为 BASE64 编码，此函数内部进行转码

返回值：成功时返回原文，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## VerifySign //吉大兼容

函数声明： long VerifySign(BSTR szSrc, BSTR szSign);
说明：验证签名数据操作。此签名数据中携带证书，验证由 SignData 或者 SignDataEx 生成的签名数据。

注意：不能验证签名数据中不携带证书，进行时间编码，并且不能返回原文。

参数：

- szSrc：原文数据(如果为空，表示此签名数据中携带原文，否则输入原文数据
- szSign：签名数据(BASE64 编码)

其中：通过 InputDataType 属性来指定原文数据格式

0 = 输入原文数据为二进制编码，此函数内部不进行转码

1 = 输入原文数据为 BASE64 编码，此函数内部进行转码

返回值：成功时返回 0，

失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## SignFileEx

函数声明： BSTR SignFileEx (BSTR szSrcFile, BSTR szEvpFile,
BSTR sSignAlgo, long IsAddSignCert,
long IsAddTime, BSTR pPin);

说明：进行签名数据文件操作，使用客户端证书，产生包含原文、签名数据文件。

参数：

- szSrcFile ： 输入原文数据文件名(全路径)，如果此参数为空，弹出窗口由用户选择原文文件。
- szEvpFile ：输出签名数据文件名
- sSignAlgo：指定签名算法

  szOID_OIWSEC_sha1 = “1.3.14.3.2.26”

  szOID_RSA_MD5 = “1.2.840.113549.2.5”

  szOID_RSA_MD2 = “1.2.840.113549.2.2”
- IsAddSignCert 是否在结果中携带证书

  0 = 不携带证书

  1 = 携带证书
- IsAddTime:是否添加签名时间

  0: 不进行时间编码

  1: 取当前系统时间，进行时间编码

  2、从时间戳服务器取得时间，必须首先设置时间戳服务器 URL。请参考 7.14 章节的时间戳操作。
- pPin：用户 Key 口令

如果输入正确的口令，则不弹出输入口令窗口，直接签名数据。

如果输入错误的口令，则弹出输入口令窗口

其中：通过 InputDataType 属性来指定原文数据格式

0 = 输入原文数据为二进制编码，此函数内部不进行转码

1 = 输入原文数据为 BASE64 编码，此函数内部进行转码

通过 OutputDataType 属性来指定返回文件数据格式

0 = 输出返回文件数据为二进制编码，此函数内部不进行转码

1 = 输出返回文件数据为 BASE64 编码，此函数内部进行转码

返回值：成功是分两种情况，

如果 szEvpFile 参数为空，则返回签名的 BASE64 编码数据，

如果 szEvpFile 参数不为空，则将签名信封保存到文件，同时返回签名的 BASE64 编码数据。

失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 中取错误信息。

# VerifySignFileEx

函数声明： BSTR VerifySignFileEx (BSTR szEvpFile,
BSTR szSrcFile,
long nSignCertIndex);

说明：验证签名数据文件操作，验证由 SignFileEx 生成的包含原文、签名数据文件。

参数：

- szEvpFile：输入签名数据文件名(BIN 编码) ，如果此参数为空，弹出窗口由用户选择签名数据文件。
- szSrcFile： 输出原文数据的文件名
- nCertIndex：指定使用的验证签名公钥证书

0 = 自己的签名公钥证书

2 = 服务器的公钥证书

4 = 第三方的公钥证书

-1 = 一个特例，即不用指定证书，签名数据内包含签名者证书

其中：通过 InputDataType 属性来指定签名数据文件格式

0 = 输入签名数据文件为二进制编码，此函数内部不进行转码

1 = 输入签名数据文件为 BASE64 编码，此函数内部进行转码

通过 OutputDataType 属性来指定返回文件数据格式

0 = 输出返回文件数据为二进制编码，此函数内部不进行转码

1 = 输出返回文件数据为 BASE64 编码，此函数内部进行转码

返回值：成功是分两种情况，

如果 szSrcFile 参数为空，则返回原文数据，

如果 szSrcFile 参数不为空，则将原文数据保存到文件，同时返回原文数据。

失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 中取错误信息。

# VerifySignDataToFileEx

函数声明： BSTR VerifySignDataToFileEx (BSTR szBase64Data,
BSTR szSrcFile,

long nSignCertIndex);

说明：验证签名数据操作，验证由 SignEx 生成的包含原文、BASE64 编码签名数据。

参数：

- szBase64Data：输入签名数据(BASE64 编码)
- szSrcFile： 输出原文数据的文件名
- nCertIndex：指定使用的验证签名公钥证书

  0 = 自己的签名公钥证书

  2 = 服务器的公钥证书

  4 = 第三方的公钥证书

  -1 = 一个特例，即不用指定证书，签名数据内包含签名者证书

其中：通过 OutputDataType 属性来指定返回文件数据格式

0 = 输出返回文件数据为二进制编码，此函数内部不进行转码

1 = 输出返回文件数据为 BASE64 编码，此函数内部进行转码

返回值：成功是分两种情况，

如果 szSrcFile 参数为空，则返回原文数据，

如果 szSrcFile 参数不为空，则将原文数据保存到文件，同时返回原文数据。

失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 中取错误信息。

# ReturnSignerCertData

函数声明：BSTR ReturnSignerCertData ();

说明：返回验证签名数据后的公钥证书数据，前提条件：已经成功地验证签名数据。

注意：如果签名数据中不包含签名者证书，则无法返回证书数据。

参数：

返回值：成功时返回 BASE64 编码证书数据，

失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# ReturnSignerCertSN

函数声明：BSTR ReturnSignerCertSN (BSTR pbBase64SignedMessage);

说明：返回签名数据后的签名者证书序列号。

注意：如果签名数据中不包含证书，则无法返回证书序列号。

参数：

- pbBase64SignedMessage： BASE64 编码签名数据

返回值：成功时返回签名者证书序列号，

失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReturnSigningTime

函数声明：BSTR ReturnSigningTime (BSTR pbBase64SignedMessage);
说明：返回签名数据后的签名时间。(格式 YYYY-MM-DD hh:mm:ss)。
注意：如果签名数据时不包含签名时间，则无法返回签名时间。
参数：

- pbBase64SignedMessage： BASE64 编码签名数据

返回值：成功时返回签名时间字符串，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# 7.10 PKCS7 加密信封操作方法

## EncryptDataEx

函数声明： BSTR EncryptDataEx (BSTR szSrc, long EncType , BSTR EncryptAlgo);
说明：进行加密信封操作。
注：1. 此函数通过 UseDefaultCSP 属性来控制加密 CSP 类型。
2. 通过 FreeEncCertArray()、AddEncCert()、AddEncCertFromFile()函数来控制多人加密证书的设置。当加密完成后，应该立即调用 FreeEncCertArray()函数来释放加密证书所占用的内存空间，否则在控件对象失效时，才自动释放该内存。

参数：

- szSrc：原文数据
- EncType：指定使用的公钥证书
  1 = 自己的公钥证书
  2 = 服务器的公钥证书
  3 = 第三方的公钥证书
  4 = 同时使用服务器的公钥证书和第三方的公钥证书
  5 = 使用多人证书加密数据
- EncryptAlgo：加密算法
  通用算法：根据软硬件算法支持
  szOID_RSA_DES_EDE3_CBC = “1.2.840.113549.3.7”
  szOID_OIWSEC_desCBC = “1.3.14.3.2.7”
  szOID_RSA_RC4 = "1.2.840.113549.3.4"
  szOID_RSA_RC2CBC = “1.2.840.113549.3.2”
  专用算法：根据 Key 硬件算法支持
  szOID_SSF33_CBC = “1.2.840.113549.3.81”
  szOID_SCB2_SPECIAL_ECB = “1.2.840.113549.3.83”

其中：通过 InputDataType 属性来指定原文数据格式

0 = 输入原文为二进制编码，此函数内部不进行转码
1 = 输入原文为 BASE64 编码，此函数内部进行转码

返回值：成功时返回加密信封数据(BASE64 编码)，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## EncryptData　　//吉大兼容

函数声明： BSTR EncryptData(BSTR szSrc, long EncType);
说明：进行加密信封操作。
注：1. 此函数通过 UseDefaultCSP 属性来控制加密 CSP 类型。
2. 通过 FreeEncCertArray()、AddEncCert()、AddEncCertFromFile()函数来控制多人加密证书的设置。当加密完成后，应该立即调用 FreeEncCertArray()函数来释放加密证书所占用的内存空间，否则在控件对象失效时，才自动释放该内存。
参数：

- szSrc：原文数据
- EncType：指定使用的公钥证书
  1 = 自己的公钥证书
  2 = 服务器的公钥证书
  3 = 第三方的公钥证书
  4 = 同时使用服务器的公钥证书和第三方的公钥证书
  5 = 使用多人证书加密数据

默认加密算法: szOID_RSA_DES_EDE3_CBC = “1.2.840.113549.3.7”

其中：通过 InputDataType 属性来指定原文数据格式
0 = 输入原文为二进制编码，此函数内部不进行转码
1 = 输入原文为 BASE64 编码，此函数内部进行转码

返回值：成功时返回加密信封数据(BASE64 编码)，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## DecryptData　　//吉大兼容

函数声明：BSTR DecryptData(BSTR szEnc);
说明：进行加密信封的解密操作。
参数：

- szEnc：加密信封(BASE64 编码)

其中：通过 OutputDataType 属性来指定返回数据格式
0 = 输出返回数据为二进制编码，此函数内部不进行转码

1 = 输出返回数据为 BASE64 编码，此函数内部进行转码

返回值：成功时返回解密原文，

失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# DecryptDataEx

函数声明：BSTR DecryptDataEx(BSTR szEnc, long IsReturnBase64);

说明：进行加密信封的解密操作。

参数：

- szEnc：加密信封(BASE64 编码)
- IsReturnBase64：是否返回 BASE64 编码

0：不转码，1：转 BASE64 编码

返回值：成功时返回解密原文，

失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# EncryptFileEx

函数声明： BSTR EncryptFileEx (BSTR szSrcFile,
BSTR szEncFile,
long EncType,
BSTR EncryptAlgo);

说明：对文件进行加密信封操作。产生二进制编码的密文信封数据。

注：1. 此函数通过 UseDefaultCSP 属性来控制加密 CSP 类型。

2. 通过 FreeEncCertArray()、AddEncCert()、AddEncCertFromFile()函数来控制多人加密证书的设置。当加密完成后，应该立即调用 FreeEncCertArray()函数来释放加密证书所占用的内存空间，否则在控件对象失效时，才自动释放该内存。

参数：

- szSrcFile：输入原文数据文件名(全路径)，如果此参数为空，弹出窗口由用户选择原文文件。
- szEncFile：输出密文数据文件名全路径(BIN 编码)
- EncType：指定使用的公钥证书

1 = 自己的公钥证书
2 = 服务器的公钥证书
3 = 第三方的公钥证书
4 = 同时使用服务器的公钥证书和第三方的公钥证书
5 = 使用多人证书加密数据

- EncryptAlgo：加密算法

通用算法：根据软硬件算法支持

szOID_RSA_DES_EDE3_CBC = “1.2.840.113549.3.7”

szOID_OIWSEC_desCBC = “1.3.14.3.2.7”
szOID_RSA_RC4 = "1.2.840.113549.3.4"
szOID_RSA_RC2CBC = “1.2.840.113549.3.2”

专用算法：根据 Key 硬件算法支持
szOID_SSF33_CBC = “1.2.840.113549.3.81”
szOID_SCB2_SPECIAL_ECB = “1.2.840.113549.3.83”

其中：通过 InputDataType 属性来指定原文数据格式
0 = 输入原文数据为二进制编码，此函数内部不进行转码
1 = 输入原文数据为 BASE64 编码，此函数内部进行转码

通过 OutputDataType 属性来指定返回文件数据格式
0 = 输出返回文件数据为二进制编码，此函数内部不进行转码
1 = 输出返回文件数据为 BASE64 编码，此函数内部进行转码

返回值：成功是分两种情况，
如果 szEncFile 参数为空，则返回密文的 BASE64 编码数据，
如果 szEncFile 参数不为空，则将加密信封保存到文件，同时返回加密的 BASE64 编码数据。
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 中取错误信息。

# DecryptFile

函数声明：BSTR DecryptFile(BSTR szEncFile, BSTR szSrcFile);
说明：对文件进行加密信封的解密操作。
参数：

- szEncFile：输入加密信封文件名全路径，如果此参数为空，弹出窗口由用户选择加密信封文件。
- szSrcFile：输出解密后的原文文件名全路径

其中：通过 InputDataType 属性来指定加密信封文件格式
0 = 输入加密信封文件为二进制编码，此函数内部不进行转码
1 = 输入加密信封文件为 BASE64 编码，此函数内部进行转码

通过 OutputDataType 属性来指定返回文件数据格式
0 = 输出返回文件数据为二进制编码，此函数内部不进行转码
1 = 输出返回文件数据为 BASE64 编码，此函数内部进行转码

返回值：成功是分两种情况，
如果 szSrcFile 参数为空，则返回解密后的原文数据，
如果 szSrcFile 参数不为空，则将原文数据保存到文件，同时返回原文数据。
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 中取错误信息。

## DecryptDataToFile

函数声明：BSTR DecryptDataToFile(BSTR szBase64Data, BSTR szSrcFile);
说明：对文件进行加密信封的解密操作。
参数：

- szBase64Data：输入加密信封数据(BASE64 编码)
- szSrcFile：输出解密后的原文文件名全路径

其中：通过 OutputDataType 属性来指定返回文件数据格式
0 = 输出返回文件数据为二进制编码，此函数内部不进行转码
1 = 输出返回文件数据为 BASE64 编码，此函数内部进行转码

返回值：成功是分两种情况，
如果 szSrcFile 参数为空，则返回解密后的原文数据，
如果 szSrcFile 参数不为空，则将原文数据保存到文件，同时返回原文数据。
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 中取错误信息。

# 7.11 PKCS7 签名加密信封操作方法

## SignEnvDataEx

函数声明： BSTR SignEnvDataEx (BSTR szSrc,
long nEncCertType,
BSTR SignAlgo,
BSTR EncryptAlgo,
long IsAddSignCert,
long IsAddTime,
BSTR pPin);
说明：进行签名加密信封操作，默认携带原文数据。
注：1. 此函数通过 UseDefaultCSP 属性来控制加密 CSP 类型。
2. 通过 FreeEncCertArray()、AddEncCert()、AddEncCertFromFile()函数来控制多人加密证书的设置。当加密完成后，应该立即调用 FreeEncCertArray()函数来释放加密证书所占用的内存空间，否则在控件对象失效时，才自动释放该内存。

参数：

- szSrc：待签名加密的数据
- nEncCertType: 指定公钥证书
1 = 客户加密证书
2 = 服务器加密证书
3 = 第三方加密证书
4 = 同时使用服务器的公钥证书和第三方的公钥证书

5 = 使用多人证书加密数据

- SignAlgo:签名算法

szOID_OIWSEC_sha1 = “1.3.14.3.2.26”

szOID_RSA_MD5 = “1.2.840.113549.2.5”

szOID_RSA_MD2 = “1.2.840.113549.2.2”

- EncryptAlgo：加密算法

通用算法：根据软硬件算法支持

szOID_RSA_DES_EDE3_CBC = “1.2.840.113549.3.7”

szOID_OIWSEC_desCBC = “1.3.14.3.2.7”

szOID_RSA_RC4 = "1.2.840.113549.3.4"

szOID_RSA_RC2CBC = “1.2.840.113549.3.2”

专用算法：根据 Key 硬件算法支持

szOID_SSF33_CBC = “1.2.840.113549.3.81”

szOID_SCB2_SPECIAL_ECB = “1.2.840.113549.3.83”

- IsAddSignCert：是否在结果中携带证书

0 = 不携带证书

1 = 携带证书

- IsAddTime：是否添加签名时间

0 = 不进行时间编码

1 = 取当前系统时间，进行时间编码

2、从时间戳服务器取得时间，必须首先设置时间戳服务器 URL。请参考 7.14 章节的时间戳操作。

- pPin：用户 Key 口令

如果输入正确的口令，则不弹出输入口令窗口，直接哈希签名数据。

如果输入错误的口令，则弹出输入口令窗口

其中：通过 InputDataType 属性来指定原文数据格式

0 = 输入原文为二进制编码，此函数内部不进行转码

1 = 输入原文为 BASE64 编码，此函数内部进行转码

返回值：成功时返回签名加密信封(BASE64 编码)，

失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## SignEnvData //吉大兼容

函数声明： BSTR SignEnvData(BSTR szSrc);

说明：进行签名加密信封操作，默认携带原文数据。

注：1. 此函数通过 UseDefaultCSP 属性来控制加密 CSP 类型。

参数：

- szSrc：待签名加密的数据

默认加密证书: 使用服务器加密证书 = 2

默认算法:
签名算法: szOID_RSA_MD5 = “1.2.840.113549.2.5”
加密算法: szOID_RSA_DES_EDE3_CBC = “1.2.840.113549.3.7”
默认是否携带证书: 1 = 携带证书
默认是否添加签名时间: 0 = 不进行时间编码
默认用户 Pin 码: 为空

其中：通过 InputDataType 属性来指定原文数据格式
0 = 输入原文为二进制编码，此函数内部不进行转码
1 = 输入原文为 BASE64 编码，此函数内部进行转码

返回值：成功时返回签名加密信封(BASE64 编码)，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## VfyDecDataEx

函数声明： BSTR VfyDecDataEx(BSTR szEnc, long nSignCertIndex);
说明：验证解密由 SignEncData 或者 SignEncDataEx 产生的签名加密信封。
参数：

- szEnc：签名加密信封(BASE64 编码)
- nSignCertIndex：指定验证签名证书在证书数组中的索引(从 0 开始)
  0 = 自己的签名公钥证书
  2 = 服务器的签名公钥证书
  4 = 第三方的签名公钥证书
  -1 = 一个特例，即不用指定证书，签名数据内包含签名者证书

其中：通过 OutputDataType 属性来指定返回数据格式
0 = 输出返回数据为二进制编码，此函数内部不进行转码
1 = 输出返回数据为 BASE64 编码，此函数内部进行转码

返回值：成功时返回原文数据，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## VfyDecData //吉大兼容

函数声明： BSTR VfyDecData(BSTR szEnc);
说明：验证解密由 SignEncData 或者 SignEncDataEx 产生的签名加密信封。
注意：不能验证不携带签名证书，进行时间编码的签名加密信封。
参数：

- szEnc：签名加密信封(BASE64 编码)

其中：通过 OutputDataType 属性来指定返回数据格式

0 = 输出返回数据为二进制编码，此函数内部不进行转码

1 = 输出返回数据为 BASE64 编码，此函数内部进行转码

返回值：成功时返回验证解密后的原文数据，

失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## SignEncFileEx

函数声明：BSTR SignEncFileEx (BSTR szSrcFile,
BSTR szEvpFile,
long nEncCertType,
BSTR SignAlgo,
BSTR EncryptAlgo,
long IsAddSignCert,
long IsAddTime,
BSTR pPin);

说明：对文件进行签名加密信封操作，返回携带原文的信封数据文件。

注：1. 此函数通过 UseDefaultCSP 属性来控制加密 CSP 类型。

2. 通过 FreeEncCertArray()、AddEncCert()、AddEncCertFromFile()函数来控制多人加密证书的设置。当加密完成后，应该立即调用 FreeEncCertArray()函数来释放加密证书所占用的内存空间，否则在控件对象失效时，才自动释放该内存。

参数：

- szSrcFile：输入原文文件名(全路径)，如果此参数为空，弹出窗口由用户选择原文文件。
- szEvpFile：输出的信封文件名(全路径)
- nEncCertType: 指定公钥证书
  1 = 客户加密证书
  2 = 服务器加密证书
  3 = 第三方加密证书
  4 = 同时使用服务器的公钥证书和第三方的公钥证书
  5 = 使用多人证书加密数据
- SignAlgo:签名算法
  szOID_OIWSEC_sha1 = “1.3.14.3.2.26”
  szOID_RSA_MD5 = “1.2.840.113549.2.5”
  szOID_RSA_MD2 = “1.2.840.113549.2.2”
- EncryptAlgo：加密算法
  通用算法：根据软硬件算法支持
  szOID_RSA_DES_EDE3_CBC = “1.2.840.113549.3.7”
  szOID_OIWSEC_desCBC = “1.3.14.3.2.7”
  szOID_RSA_RC4 = "1.2.840.113549.3.4"
  szOID_RSA_RC2CBC = “1.2.840.113549.3.2”
  专用算法：根据 Key 硬件算法支持

szOID_SSF33_CBC = “1.2.840.113549.3.81”

szOID_SCB2_SPECIAL_ECB = “1.2.840.113549.3.83”

- IsAddSignCert：是否在结果中携带证书

  0 = 不携带证书

  1 = 携带证书

- IsAddTime：是否添加签名时间

  0 = 不进行时间编码

  1 = 取当前系统时间，进行时间编码

  2 =从时间戳服务器取得时间，必须首先设置时间戳服务器 URL。请参考 7.14 章节的时间戳操作。

- pPin：用户 Key 口令

如果输入正确的口令，则不弹出输入口令窗口，直接签名数据。
如果输入错误的口令，则弹出输入口令窗口

其中：通过 InputDataType 属性来指定原文数据格式

0 = 输入原文数据为二进制编码，此函数内部不进行转码

1 = 输入原文数据为 BASE64 编码，此函数内部进行转码

通过 OutputDataType 属性来指定返回文件数据格式

0 = 输出返回文件数据为二进制编码，此函数内部不进行转码

1 = 输出返回文件数据为 BASE64 编码，此函数内部进行转码

返回值：成功是分两种情况，

如果 szEvpFile 参数为空，则返回签名加密信封的 BASE64 编码数据，

如果 szEvpFile 参数不为空，则将签名加密信封保存到文件，同时返回签名加密的 BASE64 编码数据。

失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 中取错误信息。

## SignEncFile　　//吉大兼容

函数声明：BSTR SignEncFile(BSTR szSrcFile, BSTR szEvpFile);

说明：对文件进行签名加密信封操作，返回携带原文的二进制编码信封数据文件。

注：1. 此函数通过 UseDefaultCSP 属性来控制加密 CSP 类型。

参数：

- szSrcFile：输入原文文件名(全路径) ，如果此参数为空，弹出窗口由用户选择原文文件。
- szEvpFile：输出的信封文件名(全路径)

默认加密证书:　使用服务器加密证书 = 2

默认签名算法:　　szOID_RSA_MD5 = “1.2.840.113549.2.5”

默认加密算法:　　szOID_RSA_DES_EDE3_CBC = “1.2.840.113549.3.7”

默认是否携带证书:　　1 = 携带证书

默认是否添加签名时间: 0 = 不进行时间编码

默认用户 Pin 码:　　为空

其中：通过 InputDataType 属性来指定原文数据格式

0 = 输入原文数据为二进制编码，此函数内部不进行转码

1 = 输入原文数据为 BASE64 编码，此函数内部进行转码

通过 OutputDataType 属性来指定返回文件数据格式

0 = 输出返回文件数据为二进制编码，此函数内部不进行转码

1 = 输出返回文件数据为 BASE64 编码，此函数内部进行转码

返回值：成功是分两种情况，

如果 szEvpFile 参数为空，则返回签名加密信封的 BASE64 编码数据，

如果 szEvpFile 参数不为空，则将二进制签名加密信封保存到文件，同时返回签名加密的 BASE64 编码数据。

失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 中取错误信息。

# VfyDecFileEx

函数声明： BSTR VfyDecFileEx(BSTR szEvpFile, BSTR szSrcFile, long nSignCertIndex);

说明：对文件进行验证解密操作。验证解密由 SignEncFile 或者 SignEncFileEx 产生的签名加密信封文件。

参数：

- szEvpFile：输入信封文件名全路径(BIN 编码)，如果此参数为空，弹出窗口由用户选择签名加密信封文件。
- szSrcFile： 输出原文文件名全路径
- nSignCertIndex：指定验证签名证书在证书数组中的索引(从 0 开始)

  0 = 自己的签名公钥证书

  2 = 服务器的签名公钥证书

  4 = 第三方的签名公钥证书

  -1 = 一个特例，即不用指定证书，签名数据内包含签名者证书

其中：通过 InputDataType 属性来指定签名加密信封文件格式

0 = 输入加密信封文件为二进制编码，此函数内部不进行转码

1 = 输入加密信封文件为 BASE64 编码，此函数内部进行转码

通过 OutputDataType 属性来指定返回文件数据格式

0 = 输出返回文件数据为二进制编码，此函数内部不进行转码

1 = 输出返回文件数据为 BASE64 编码，此函数内部进行转码

返回值：成功是分两种情况，

如果 szSrcFile 参数为空，则返回原文数据，

如果 szEvpFile 参数不为空，则将原文数据保存到文件，同时返回原文数据。

失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 中取错误信息。

# VfyDecFile　　//吉大兼容

函数声明：BSTR VfyDecFile(BSTR szEvpFile, BSTR szSrcFile);
说明：对文件进行签名加密信封的解密验证操作，验证解密由 SignEncFile 产生的签名加密信封文件。

注意：不能验证不携带签名证书或者进行时间编码的签名加密信封。

参数：

- szEvpFile：输入信封文件名全路径(BIN 编码)，如果此参数为空，弹出窗口由用户选择签名加密信封文件。
- szSrcFile：输出原文文件名全路径

其中：通过 InputDataType 属性来指定签名加密信封文件格式

0 = 输入加密信封文件为二进制编码，此函数内部不进行转码
1 = 输入加密信封文件为 BASE64 编码，此函数内部进行转码

通过 OutputDataType 属性来指定返回文件数据格式

0 = 输出返回文件数据为二进制编码，此函数内部不进行转码
1 = 输出返回文件数据为 BASE64 编码，此函数内部进行转码

返回值：成功是分两种情况，

如果 szSrcFile 参数为空，则返回原文数据，
如果 szEvpFile 参数不为空，则将原文数据保存到文件，同时返回原文数据。

失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 中取错误信息。

# VfyDecDataToFileEx

函数声明：　BSTR VfyDecDataToFileEx(BSTR szBase64Data,
BSTR szSrcFile,
long nSignCertIndex);

说明：对文件进行验证解密操作。验证解密由 SignEncFile 或者 SignEncFileEx 产生的签名加密信封文件。

参数：

- szBase64Data：输入信封数据(BASE64 编码)
- szSrcFile：　输出原文文件名全路径
- nSignCertIndex：指定验证签名证书在证书数组中的索引(从 0 开始)

0 = 自己的签名公钥证书
2 = 服务器的签名公钥证书
4 = 第三方的签名公钥证书
-1 = 一个特例，即不用指定证书，签名数据内包含签名者证书

其中：通过 OutputDataType 属性来指定返回文件数据格式

0 = 输出返回文件数据为二进制编码，此函数内部不进行转码
1 = 输出返回文件数据为 BASE64 编码，此函数内部进行转码

返回值：成功是分两种情况，
如果 szSrcFile 参数为空，则返回原文数据，
如果 szEvpFile 参数不为空，则将原文数据保存到文件，同时返回原文数据。
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 中取错误信息。

## 7.12 RSA 加解密操作方法

### RSAEncryptData

函数声明：BSTR RSAEncryptData(BSTR SourceData, long nCertIndex);
说明：RSA 加密数据，原文数据长度根据加密密钥的长度而决定，如加密密钥长度为 128 字节，则原文数据长度不能超过 117 字节。
参数：

- SourceData： 原文数据
- nCertIndex：指定证书在证书数组中的索引(从 0 开始)

其中：通过 InputDataType 属性来指定原文数据格式
0 = 输入原文为二进制编码，此函数内部不进行转码
1 = 输入原文为 BASE64 编码，此函数内部进行转码

返回值：成功时返回 BASE64 编码密文数据，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

### RSADecryptData

函数声明：BSTRRSADecryptData(BSTR CryptoData, long nCertIndex);
说明：RSA 解密数据。
参数：

- CryptoData： BASE64 编码密文数据
- nCertIndex：指定证书在证书数组中的索引(从 0 开始)

其中：通过 OutputDataType 属性来指定返回数据格式
0 = 输出返回数据为二进制编码，此函数内部不进行转码
1 = 输出返回数据为 BASE64 编码，此函数内部进行转码

返回值：成功时返回解密后的原文数据，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# 7.13 对称加解密操作方法

## SymCipherInit

函数声明：long SymCipherInit(BSTR CipherAlgo, BSTR pbKey);
说明：对称加解密初始化，必须进行此初始化后，才可以进行对称加解密操作。必须输入口令。

注：1. 此函数通过 UseDefaultCSP 属性来控制加密 CSP 类型。

参数：

- CipherAlgo： 加解密算法

  1. 通用算法：根据系统软硬件算法支持有关。

  3DES 算法: "1.2.840.113549.3.7" = szOID_RSA_DES_EDE3_CBC
  DES 算法: "1.3.14.3.2.7" = szOID_OIWSEC_desCBC,
  RC4 算法: "1.2.840.113549.3.4" = szOID_RSA_RC4,
  RC2 算法: "1.2.840.113549.3.2" = szOID_RSA_RC2CBC

  2. 专用算法：由于系统默认不支持专用算法，因此，需要根据具体使用的 Key 硬件提供者支持。

  SSF33: "1.2.840.113549.3.81" = szOID_SSF33_CBC
  SCB2: "1.2.840.113549.3.83" = szOID_SCB2_SPECIAL_ECB

- pbKey： 密钥口令(4-8) ，必须输入口令

返回值：成功时返回 0，
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## SymEncryptData

函数声明：BSTR SymEncryptData(BSTR SourceData);
说明：对称加密数据，必须调用 SymCipherInit()初始化后，才可以进行对称加密操作。
参数：

- SourceData： 原文数据

其中：通过 InputDataType 属性来指定原文数据格式
0 = 输入原文为二进制编码，此函数内部不进行转码
1 = 输入原文为 BASE64 编码，此函数内部进行转码

返回值：成功时返回 BASE64 编码密文数据，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# SymDecryptData

函数声明：BSTR SymDecryptData(BSTR CryptoData);
说明：对称解密数据，必须调用 SymCipherInit()初始化后，才可以进行对称解密操作。
参数：

- CryptoData：　　BASE64 编码密文数据

其中：通过 OutputDataType 属性来指定返回数据格式
　　0 = 输出返回数据为二进制编码，此函数内部不进行转码
　　1 = 输出返回数据为 BASE64 编码，此函数内部进行转码

返回值：成功时返回解密后的原文数据，
　　失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# SymEncryptFile

函数声明：long SymEncryptFile(BSTR SourceFile, BSTR EncFile);
说明：文件对称加密，必须调用 SymCipherInit()初始化后，才可以进行对称加密操作。
　　注：本函数内部不进行转码处理，产生的密文为二进制格式。
参数：

- SourceFile：　　原文文件名
- EncFile：　　产生的密文文件名

返回值：成功时返回 0，
　　失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# SymDecryptFile

函数声明：long SymDecryptFile(BSTR EncFile, BSTR RetSourceFile);
说明：文件对称解密，必须调用 SymCipherInit()初始化后，才可以进行对称解密操作。
　　注：本函数内部不进行转码处理，产生的密文为二进制格式。
参数：

- EncFile：　　密文文件名
- RetSourceFile：　　返回的解密原文文件名

返回值：成功时返回 0，
　　失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# 7.14 LDAP 相关操作方法

## ReadRootCertFromLdap

函数声明：BSTR ReadRootCertFromLdap (BSTR szLDAPServerAddress, long dwPort);
说明：从 LDAP 服务器获得根证书(BASE64 编码)，同时初始化根证书对象，并清空证书数组中的其他对象。

注：每次连接 LDAP 服务器，需要 5 秒时间，请耐心等待。

参数：

- szLDAPServerAddress：LDAP 服务器地址，如果输入 NULL = 默认从 ldap.ln-ca.com 下载
- dwPort：LDAP 服务器端口，如果输入 0 = 默认从 391 下载

返回值：成功时返回 BASE64 编码公钥根证书数据，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReadCRLFromLdap

函数声明：BSTR ReadCRLFromLdap (BSTR szLDAPServerAddress, long dwPort);
说明：从 LDAP 服务器获得 CRL(BASE64 编码)，同时初始化 CRL 对象。

注：每次连接 LDAP 服务器，需要 5 秒时间，请耐心等待。

参数：

- szLDAPServerAddress：LDAP 服务器地址，如果输入 NULL = 默认从 ldap.ln-ca.com 下载
- dwPort：LDAP 服务器端口，如果输入 0 = 默认从 391 下载

返回值：成功时返回 BASE64 编码 CRL 数据，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReadCertFromLdap

函数声明：BSTR ReadCertFromLdap (BSTR szCertSerialNumber,
BSTR szLDAPServerAddress,
long dwPort);
说明：从 LDAP 服务器获得指定证书序列号的公钥证书(BASE64 编码)。

注：每次连接 LDAP 服务器，需要 5 秒时间，请耐心等待。

参数：

- szCertSerialNumber：证书序列号
- szLDAPServerAddress：LDAP 服务器地址，如果输入 NULL = 默认从 ldap.ln-ca.com

下载

- dwPort：LDAP 服务器端口，如果输入 0 = 默认从 391 下载

返回值：成功时返回 BASE64 编码公钥证书数据，

失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReturnCertType

函数声明：long ReturnCertType ();

说明：从 LDAP 获得查询证书后，返回证书类型号。

注: 前提条件是已经成功地从 LDAP 服务器上查询到证书。

目前设定的证书种类如下：

4：基本 CA 单证书
5：基本 CA 双证书
102：个人身份证书
103：个人代码签名证书
104：单位身份证书
106：个人电子商务证书
107：单位电子商务证书
108：设备证书
109：CRL 验证个人身份证书
110：辽宁个人身份证书
111：辽宁个人代码签名证书
112：辽宁个人电子商务证书
113：辽宁 CRL 验证个人身份证书
114：辽宁单位身份证书
115：辽宁单位代码签名证书
116：辽宁单位电子商务证书
117：辽宁设备证书
118：单位代码签名证书
127：国家根 CA 测试

参数：

返回值：成功时返回证书类型号，

失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReturnCertStatus

函数声明：long ReturnCertStatus ();

说明：从 LDAP 获得查询证书后，返回证书状态。

注: 前提条件是已经成功地从 LDAP 服务器上查询证书。

目前设定的证书状态如下：

3：新申请（证书未下载）
5：证书使用中
6：证书注销

参数：

返回值：成功时返回证书状态，
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReturnCertTypeStr

函数声明：BSTR ReturnCertTypeStr (long CertType);

说明：证书类型码转换为字符串。
目前设定的证书种类如下：
4：基本 CA 单证书
5：基本 CA 双证书
102：个人身份证书
103：个人代码签名证书
104：单位身份证书
106：个人电子商务证书
107：单位电子商务证书
108：设备证书
109：CRL 验证个人身份证书
110：辽宁个人身份证书
111：辽宁个人代码签名证书
112：辽宁个人电子商务证书
113：辽宁 CRL 验证个人身份证书
114：辽宁单位身份证书
115：辽宁单位代码签名证书
116：辽宁单位电子商务证书
117：辽宁设备证书
118：单位代码签名证书
127：国家根 CA 测试

参数：

- CertType：证书类型码

返回值：成功时返回证书类型字符串，否则返回空。

## ReturnCertStatusStr

函数声明：BSTR ReturnCertStatusStr (long CertStatus);

说明：证书状态编码转换为字符串。
目前设定的证书状态如下：
3：新申请（证书未下载）
5：证书使用中

6：证书注销

参数：

- CertStatus：证书状态编码

返回值：成功时返回证书状态字符串，否则返回空。

## 7.15 时间戳操作方法

### SetTimeStampServerUrl

函数声明：long SetTimeStampServerUrl(BSTR sUrl);

说明：设置时间戳服务器地址。

参数：

- sUrl：时间戳服务器地址

返回值：成功时返回时间戳服务器地址总数。

### AppendTimeStampServerUrl

函数声明：long AppendTimeStampServerUrl(BSTR sUrl);

说明：追加时间戳服务器地址。

参数：

- sUrl：时间戳服务器地址

返回值：成功时返回时间戳服务器地址总数。

### RequestTimeStamp

函数声明：long RequestTimeStamp (BSTR sData, BSTR szHashAlgorithmOID);

说明：发送时间戳请求，同时返回时间戳响应。前提条件：已经设置或追加时间戳服务器地址

参数：

- sData：请求原文数据
- szHashAlgorithmOID：哈希算法

  szOID_RSA_MD5 = "1.2.840.113549.2.5"

  szOID_OIWSEC_sha1 = "1.3.14.3.2.26"

返回值：成功时返回 0，

失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## GetTimeStampVersion

函数声明：BSTR GetTimeStampVersion();
说明：获得时间戳版本。前提条件：已经成功地发送时间戳请求，并成功地获得时间戳响应。
参数：
返回值：成功时返回时间戳版本，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## GetTimeStampTimeStr

函数声明：BSTR GetTimeStampTimeStr ();
说明：获得时间戳签名时间字符串。前提条件：已经成功地发送时间戳请求，并成功地获得时间戳响应。
参数：
返回值：成功时返回时间戳签名时间字符串，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## GetTimeStampSerialNumber

函数声明：BSTR GetTimeStampSerialNumber ();
说明：获得时间戳序列号。前提条件：已经成功地发送时间戳请求，并成功地获得时间戳响应。
参数：
返回值：成功时返回时间戳序列号，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## GetTimeStampMessageImprint

函数声明：BSTR GetTimeStampMessageImprint ();
说明：获得时间戳签名消息拇印。前提条件：已经成功地发送时间戳请求，并成功地获得时间戳响应。
参数：
返回值：成功时返回时间戳签名消息拇印，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# GetTimeStampToken

函数声明：BSTR GetTimeStampToken ();
说明：获得时间戳响应信息。前提条件：已经成功地发送时间戳请求，并成功地获得时间戳响应。
参数：
返回值：成功时返回时间戳响应信息，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# GetTimeStampSrcHash

函数声明：BSTR GetTimeStampSrcHash ();
说明：获得时间戳中包含的原文数据哈希值。前提条件：已经成功地发送时间戳请求，并成功地获得时间戳响应。
参数：
返回值：成功时返回时间戳中包含的原文数据哈希值，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# GetTimeStampSrcAlgo

函数声明：BSTR GetTimeStampSrcAlgo ();
说明：获得时间戳中包含的原文数据哈希算法。前提条件：已经成功地发送时间戳请求，并成功地获得时间戳响应。
参数：
返回值：成功时返回时间戳中包含的原文数据哈希算法，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# ViewTimeStampCertWnd

函数声明：long ViewTimeStampCertWnd ();
说明：显示时间戳签名证书窗体。前提条件：已经成功地发送时间戳请求，并成功地获得时间戳响应。
参数：
返回值：成功时返回 0，
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# 7.16 编码格式转换方法

## StringToByteLength

函数声明：long StringToByteLength(BSTR Str);
说明：获得字符串的 Ansi 字节长度
参数：

- Str：原文数据

## StringToUnicodeLength

函数声明：long StringToUnicodeLength(BSTR Str);
说明：获得字符串的 Unicode 宽字节长度
参数：

- Str：原文数据

返回值：成功返回原文数据长度，失败返回 0

## BinToBase64

函数声明：BSTR BinToBase64 (BSTR sSource, long SourceLen, long IsCert);
说明：将二进制数据转成 BASE64 编码数据，其中二进制数据长度在网页、VB 等程序调用中请输入二进制数据的字节码长度，尤其是中文字符，一个中文字符等于 2 个字节，可以通过本控件提供的 long StringToByteLength (BSTR sSource)函数获得长度。
参数：

- sSource：二进制数据
- SourceLen：二进制数据长度(字节)
- IsCert：指定是否操作证书(0=不是证书,1=是证书)

返回值：成功时返回 BASE64 编码数据，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## Base64ToBin

函数声明：BSTR Base64ToBin (BSTR sEncoded, long EncodedLen, long IsCert);
说明：将 BASE64 编码数据转成二进制编码数据，其中二进制数据长度在网页、VB 等程序调用中请输入二进制数据的字节码长度，尤其是中文字符，一个中文字符等于 2 个字节，可以通过本控件提供的 long StringToByteLength (BSTR sSource)函数获得长度。
参数：

- sEncoded：BASE64 编码数据
- EncodedLen：BASE64 编码数据长度(字节)
- IsCert：指定是否操作证书(0=不是证书,1=是证书)

返回值：成功时返回二进制数据，

失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## BinToBase64File

函数声明：long BinToBase64File (BSTR sBinFile, BSTR sBase64File);

说明：将二进制编码文件转成 BASE64 编码文件。

参数：

- sBinFile： 二进制编码文件(全路径)
- sBase64File：BASE64 编码文件(全路径)

如果 sBase64File 参数为空，则将产生的 BASE64 编码数据写入 sBinFile 文件中。

返回值：成功时返回 0，

失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## Base64ToBinFile

函数声明：long Base64ToBinFile (BSTR sBase64File, BSTR sBinFile);

说明：将 BASE64 编码文件转成二进制编码文件。

参数：

- sBase64File：BASE64 编码文件(全路径)
- sBinFile： 二进制编码文件(全路径)

如果 sBinFile 参数为空，则将产生的二进制编码数据写入 sBase64File 文件中。

返回值：成功时返回 0，

失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## GB2312ToUTF8String

函数声明：VARIANT GB2312ToUTF8String (VARIANT * inputStr);

说明：将 GB2312 编码数据转成 UTF8 编码数据。

参数：

- inputStr： GB2312 编码数据

返回值：成功时 ErrorCode 属性=0，

失败时 ErrorCode 属性<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## UTF8To GB2312String

函数声明：VARIANT UTF8ToGB2312String (VARIANT * inputStr);

说明：将 UTF8 编码数据转成 GB2312 编码数据。

参数：

- inputStr： UTF8 编码数据

返回值：成功时 ErrorCode 属性=0，

失败时 ErrorCode 属性<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## GB2312ToUTF8File

函数声明：long GB2312ToUTF8File(BSTR inputFile, BSTR outputFile);

说明：将 GB2312 编码文件转成 UTF8 编码文件。

参数：

- inputFile： GB2312 编码文件名
- outputFile： UTF8 编码文件名

如果 outputFile 参数为空，则将输出数据写入 inputFile 文件中。

返回值：成功时返回 0，

失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## UTF8To GB2312File

函数声明：long UTF8To GB2312File(BSTR inputFile, BSTR outputFile);

说明：将 UTF8 编码文件转成 GB2312 编码文件。

参数：

- inputFile： UTF8 编码文件名
- outputFile： GB2312 编码文件名

如果 outputFile 参数为空，则将输出数据写入 inputFile 文件中。

返回值：成功时返回 0，

失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## GB2312ToUnicodeFile

函数声明：long GB2312ToUnicodeFile (BSTR inputFile, BSTR outputFile);

说明：将 GB2312 编码文件转成 Unicode 编码文件。

参数：

- inputFile： GB2312 编码文件名
- outputFile： Unicode 编码文件名

如果 outputFile 参数为空，则将输出数据写入 inputFile 文件中。

返回值：成功时返回 0，

失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## UnicodeTo GB2312File

函数声明：long UnicodeTo GB2312File(BSTR inputFile, BSTR outputFile);
说明：将 Unicode 编码文件转成 GB2312 编码文件。
参数：

- inputFile：　Unicode 编码文件名
- outputFile：　GB2312 编码文件名

如果 outputFile 参数为空，则将输出数据写入 inputFile 文件中。

返回值：成功时返回 0，
失败时返回<0，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

# 7.17 其他公共方法

## GenUUID

函数声明：BSTR GenUUID (long iRetType);
说明：产生 UUID。UUID = universally unique identifier。
参数：

- iRetType：输入返回数据的类型

0:返回两位十六进制数据(中间含"-"符号)(36 位长度)
1:返回两位十六进制数据(中间不含"-"符号)(32 位长度)

返回值：成功时返回 BASE64 编码 UUID，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## CryptGenRandom

函数声明：BSTR CryptGenRandom (long iRandLen);
说明：产生随机数。
参数：

- iRandLen：指定要产生的随机数长度

返回值：成功时返回 BASE64 编码随机数，
失败时返回空，由 ErrorCode 属性中取错误码，由 ErrorMessage 属性中取错误信息。

## ReadDataLength

函数声明：long ReadDataLength ();
说明：读取数据长度，前提条件：成功地调用了 Base64ToBin()函数
参数：
返回值：返回数据长度
返回值：成功返回原文数据长度，失败返回 0

## AboutBox　　//吉大兼容

函数声明：void AboutBox ();
说明：显示该控件的版本窗口，可以用于判断控件是否可以使用
参数：
返回值：无

## GetVersion

函数声明：BSTR GetVersion ();
说明：获得该控件的版本，通过外部脚本，可以用于判断控件是否为最新版本，即是否可以使用。
参数：
返回值：返回获得该控件的版本

## SetShowError

函数声明：void SetShowError(long lValue);
说明：设置显示错误信息方式，该接口函数为在调试时使用而设置的。
默认时，可以不调用此函数。
参数：

- lValue：状态值
  0 = 通过 ErrorCode 属性返回错误代码或者 ErrorMessage 属性返回错误信息
  1 = 除了有 0 状态的功能外，还可通过消息窗口显示错误信息

返回值：无

## SetRDNType

函数声明：void SetRDNType(long lValue);
说明：设置 RDN 表示类型，此项设置是在"获得证书主题、获得证书颁发者和以证书主题初始化证书"时起作用。
默认时，控件自动使用 0 类型，可以不调用此函数。

参数：

- lValue：状态值
  0 =微软类型:以"S="描述省份， 每个子项以"逗号+空格"间隔
  1 = RFC 类型：以"ST="描述省份，每个子项以"逗号"间隔

返回值：无

## CertRDNChange

函数声明：BSTR CertRDNChange(LPCTSTR SourceStr, long ChangeType);
说明：改变证书 RDN 字符串的格式。此函数是为了改变”证书主题”和”证书颁发者”的格式。
参数：

- SourceStr：输入原始 RDN 字符串
- ChangeType：格式类型
  0 = 微软类型：以"S="描述省份， 每个子项以"逗号+空格"间隔
  1 = RFC 类型：以"ST="描述省份，每个子项以"逗号"间隔

返回值：返回新的证书 RDN 字符串。

## GetSystemMemory

函数声明：double GetSystemMemory(long nInfoType, long nUnitType);
说明：获得系统内存信息。
参数：

- nInfoType：要返回的内存信息类型
  0: 系统总物理内存
  1: 系统剩余物理内存
  2: 系统总虚拟内存
  3: 系统剩余虚拟内存
  4: 系统内存已使用率(%)
  5: 系统内存未使用率(%)
- nUnitType：要返回的内存单位
  0: Bytes
  1: KB
  2: MB
  3: GB

返回值：返回内存信息值。

## GetDiskCapacity

函数声明：BSTR GetDiskCapacity(LPCTSTR sDiskName);
说明：获得系统指定磁盘容量。
参数：

- sDiskName：磁盘名称(如: c:)

返回值：成功返回磁盘容量(字节)，失败返回空串。

## GetDiskFreeSpace

函数声明：BSTR GetDiskFreeSpace (LPCTSTR sDiskName);
说明：获得系统指定磁盘剩余容量。
参数：

- sDiskName：磁盘名称(如: c:)

返回值：成功返回磁盘剩余容量(字节)，失败返回空串。

# 8 返回码与错误信息

```
#define E_OK 0L    //成功

#define E_MSG_UNKNOWN_INVALIDATE "未知错误"
#define E_UNKNOWN_INVALIDATE     -6001

#define E_MSG_PARAM_INVALIDATE     "参数错误"
#define E_PARAM_INVALIDATE         -6002

#define E_MSG_OPEN_SESSION_FAILED   "进行网络连接时打开会话失败"
#define E_OPEN_SESSION_FAILED       -6003

#define E_MSG_CONECT_TO_SERVER_FAILED "进行网络连接时连接服务器失败"
#define E_CONECT_TO_SERVER_FAILED  -6004

#define E_MSG_OPEN_REQUEST_FAILED  "进行网络连接时打开请求失败"
#define E_OPEN_REQUEST_FAILED      -6005

#define E_MSG_ADD_HEADER_FAILED    "进行网络连接时添加请求头失败"
#define E_ADD_HEADER_FAILED        -6006

#define E_MSG_SEND_REQUEST_FAILED  "发送请求失败"
#define E_SEND_REQUEST_FAILED      -6007

#define E_MSG_END_REQUEST_FALED    "结束请求失败"
#define E_END_REQUEST_FALED        -6008
```

```
#define E_MSG_OPEN_STORE_FAILED      "打开系统证书库失败"
#define E_OPEN_STORE_FAILED          -6009

#define E_MSG_SELECT_CERT_FAILED     "选择证书失败"
#define E_SELECT_CERT_FAILED         -6010

#define E_MSG_CERT_NOT_INITIALIZED   "证书未初始化"
#define E_CERT_NOT_INITIALIZED       -6011

#define E_MSG_BUFFER_TOO_SMALL       "缓冲区太小"
#define E_BUFFER_TOO_SMALL           -6012

#define E_MSG_CERT_NOT_IN_STORE      "找不到指定的证书"
#define E_CERT_NOT_IN_STORE          -6013

#define E_MSG_ADD_CERT_TO_STORE      "添加证书到证书库失败"
#define E_ADD_CERT_TO_STORE          -6014

#define E_MSG_CREAT_CERT_FAILED      "创建证书上下文失败"
#define E_CREAT_CERT_FAILED          -6015

#define E_MSG_ALLOC_MEM_FAILED       "内存不足!"
#define E_ALLOC_MEM_FAILED           -6016

#define E_MSG_REALLOC_MEM_FAILED     "重新分配内存失败!"
#define E_REALLOC_MEM_FAILED         -6017

#define E_MSG_STREAM_BUFFER_IS_NULL  "流缓存为空!"
#define E_STREAM_BUFFER_IS_NULL      -6018

#define E_MSG_INIT_CERT_NOT_BEGIN_TIME   "证书还未生效!"
#define E_INIT_CERT_NOT_BEGIN_TIME       -6019

#define E_MSG_INIT_CERT_OVER_END_TIME    "证书已经过期!"
#define E_INIT_CERT_OVER_END_TIME        -6020

#define E_MSG_CERT_TYPE_ERROR        "证书类型错误!"
#define E_CERT_TYPE_ERROR            -6021

#define E_MSG_CAN_NOT_FIND_LIBARY    "找不到动态库"
#define E_CAN_NOT_FIND_LIBARY        -6022

#define E_MSG_DATA_CHANGE_CODE_FAILED   "数据转码失败"
```

```
#define E_DATA_CHANGE_CODE_FAILED          -6023

#define E_MSG_CRYPT_FAILED                 "密码类无效"
#define E_CRYPT_FAILED                     -6024

#define E_MSG_SYM_CRYPT_FAILED             "对称密码类无效"
#define E_SYM_CRYPT_FAILED                 -6025

#define E_MSG_VERIFY_HASH_FAILED           "验证哈希失败"
#define E_VERIFY_HASH_FAILED               -6026

#define E_MSG_ENGINE_INVALIDATE            "引擎未初始化"
#define E_ENGINE_INVALIDATE                 -6027

#define E_MSG_VERIFY_PASSWORD_FAILED "校验口令失败"
#define E_VERIFY_PASSWORD_FAILED           -6028

#define E_MSG_VERIFY_CRL_FAILED            "校验 CRL 失败"
#define E_VERIFY_CRL_FAILED                -6029

#define E_MSG_VERIFY_ROOT_FAILED           "校验根证书失败"
#define E_VERIFY_ROOT_FAILED               -6030

#define E_MSG_FILE_NOT_EXIST_FAILED        "文件不存在"
#define E_FILE_NOT_EXIST_FAILED            -6031

#define E_MSG_FILE_NOT_WRITE_FAILED        "不允许写文件"
#define E_FILE_NOT_WRITE_FAILED            -6032

#define E_MSG_STORE_NOT_INITIALIZED        "证书库未初始化"
#define E_STORE_NOT_INITIALIZED            -6033

#define E_MSG_RSA_CRYPT_FAILED             "RSA 密码类无效"
#define E_RSA_CRYPT_FAILED                 -6034

#define E_MSG_INIT_PFX_FAILED              "初始化 PFX 证书失败"
#define E_INIT_PFX_FAILED                  -6082
```

其他返回码和错误信息为操作系统函数返回信息，可以查询微软帮助。

# 9 历史版本

2006.11.20 v2.8.6.0
1. 增加选择证书时的过滤条件设置属性 2 个：FilterCertCN、FilterCertEMail
2. 增加用于控制加密数据时，使用的 CSP 类型属性：UseDefaultCSP
3. 增加从文件初始化公钥证书函数：InitServerCertFromFile()、InitThirdCertFromFile()
4. 增加多人证书加密功能函数：FreeEncCertArray()、AddEncCert()、AddEncCertFromFile ()
5. 增加数据编码转换功能函数：GB2312ToUTF8String()、UTF8ToGB2312String()、GB2312ToUTF8File()、UTF8ToGB2312File ()、GB2312ToUnicodeFile ()、UnicodeToGB2312File ()
6. 增加加密算法的支持：
   szOID_SSF33_CBC = 1.2.840.113549.3.81 SSF33 算法
   szOID_SCB2_SPECIAL_ECB = 1.2.840.113549.3.83 SCB2 算法

2006.10.18 v2.8.5.1
1. 增加 RSA 加解密数据函数：RSAEncryptData()、RSADecryptData()，
2. 修改通过证书编码初始化客户端证书的一个 Bug。

2006.07.10 v2.8.5.0
1. 修改对称加解密数据函数，增加对称加解密文件函数，
2. 增加哈希文件函数，
3. 增加哈希签名文件函数，
4. 增加获得证书密钥用法函数，
5. 增加通过公钥证书初始化含私钥的证书功能。

2006.06.20 v2.8.1.0
1. 增加对证书主题的哈希值返回数据类型的控制属性；
2. 增加写文件操作时，对弹出确认窗口的控制属性；
3. 增加数据输入输出的参数控制属性；
4. 增加获得系统内存和系统磁盘信息的方法。

2006.04.01 v2.8.0.0 增加证书主题的哈希值、BASE64 编解码文件操作方法。

2006.03.15 v2.7.1.0 增加 RSA 解密扩展方法，可以指定返回 BASE64 编码数据。

2006.03.02 v2.7.0.0 修改有关 LDAP 内部查询实现方法，相关接口函数不变；修改有关对文件的选择操作，相关接口函数不变。

2006.01.26 v2.6.0.2 修改内部证书 RDN 转换成字符串的表示法，增加了两个函数：SetRDNType()用于控制 RDN 的表示类型，CertRDNChange()用于改变 RDN 的格式。

2005.12.25 v2.6.0.1 去掉控件中包含的不合理接口函数。

2005.12.01　v2.5.0.2　更新从 LDAP 服务器下载证书中的 BUG。

2005.12.01　v2.5.1.0　增加 4 个接口函数，用于选择证书时，显示证书窗体中证书的类型。分别是：

EnumClientCertEx
InitClientCertEx
InitServerCertFromStoreEx
InitThirdCertFromStoreEx