

電子印鑑システム

パソコン決裁6

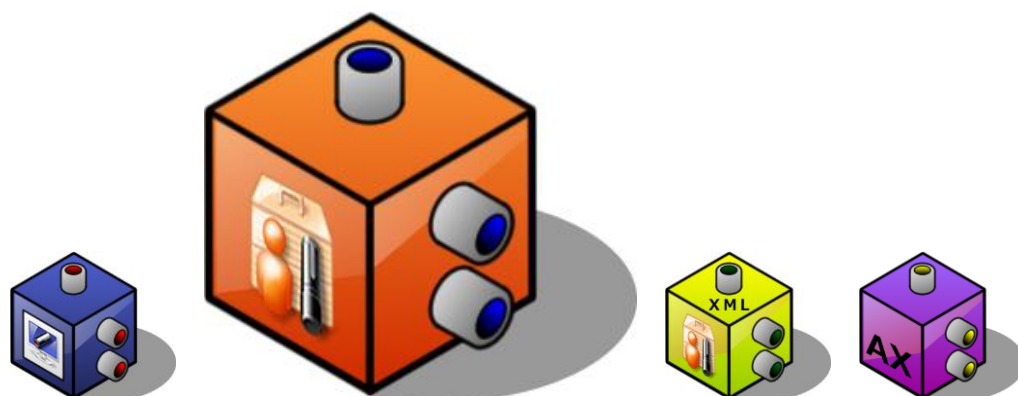
D i g i t a l S t a m p S e r i e s

E x t e n s i o n K i t

サービスリリース 1 試用版

リファレンス マニュアル

電子印鑑 Extension 編



内容

1	はじめに	1
2	電子印鑑 Extension の概要	1
3	電子印鑑 Extension のファイル名	1
4	64ビット版に関する注意事項	1
5	セットアップ	1
6	電子印鑑 Extension の操作	3
6.1	VBScript (Visual Basic Script) の場合	3
6.2	Visual Basic (Microsoft Visual Basic 2008) の場合	3
6.3	Visual C# (Microsoft Visual C# 2008) の場合	4
7	電子印鑑 Extension のコード例	6
8	IExtension インタフェース	10
	DsmFile	10
	GetStampCount	11
	GetXStamperCode	12
	Login	13
	Logout	14
	SelectStamp	15
	SaveBmp	16
	SaveJpg	17
	SavePng	18
	HimetricWidth	19
	HimetricHeight	20
	DrawStamp	21
	PixelWidth	22
	PixelHeight	23
	GenStampData	24
	GetVersion	25
	SetCurrentDate	26
	LoginByDeviceID	27
	GetLoginUserName	28
	GetStampSerialNumber	29
	GetBmpData	30
	GetPngData	31
	GetJpgData	32
	GetUserLevel	33
	GetUserLevelName	34

GetLevelName	35
GetParentUserCount.....	36
GetParentUserName.....	37
GetUserFirstName.....	38
GetUserLastName	39
GetUserMiddleName.....	40
GetUserExplanation.....	41
GetParentUserFirstName.....	42
GetParentUserLastName	43
GetParentUserMiddleName	44
GetParentUserExplanation	45
GenStampDataBase64.....	46
ResetError.....	47
GetLastErrorNumber	48
GetLastErrorMessage	49
GetUserEmailAddress	50
GetUserWindowsAccount.....	51
EsdFile.....	52
EsdBase64	53
GetEsdMejorVersion	54
GetEsdMinorVersion	55
GetEsdRevisionVersion	56
GetEsdUserName.....	57
GetEsdUserLastName	58
GetEsdUserFirstName.....	59
GetEsdUserCid.....	60
GetEsdStampSerialNumber	61
GetEsdXStamperCode.....	62
GetEsdStampCid	63
GetEsdImpressDate	64
GetEsdImpressCount.....	65
GetEsdImpressId	66
SaveEsdImage.....	67
GetDeviceID.....	68
GetStampAngle.....	69
SetStampAngle	70
GetEdition	71
SetUserLockMode	72
GetUserLockMode	73

GetEsdUserMiddleName.....	74
DocFileName	75
DocFileTitle	76
DocFileName	77
DocItem	78
9 IStmpDat インタフェイス	79
SF_DeleteUser	79
SF_RenewUserName	80
SF_DeleteStamp	81
SF_RenewParentUser	82
SF_MoveFirst.....	83
SF_MoveLast	84
SF_MoveNext	85
SF_MovePrevious.....	86
SF_GetCurrentPosition.....	87
SF_SeekUser	88
SF_LockUser	89
SF_UnlockUser	90
SF_GetCurrentUserName	91
SF_GetCurrentStampCount	92
SF_GetUserCount.....	93
SF_GetStampMaxCount	94
SF_GetRootName.....	95
SF_SetRootName.....	96
SF_GetRootID	97
SF_GetAdminPassword	98
SF_SetAdminPassword	99
SF_GetExplanation.....	100
SF_SetExplanation	101
SF_IsBOF	102
SF_IsEOF	103
SU_GetPosition	104
SU_GetStampCount	105
SU_GetUserListupFlag	106
SU_SetUserListupFlag	107
SU_GetUserName	108
SU_GetUserPassword	109
SU_SetUserPassword	110
SU_GetUserPasswordMinLength	111

SU_SetUserPasswordMinLength.....	112
SU_GetUserFirstName	113
SU_SetUserFirstName	114
SU_GetUserMiddleName	115
SU_SetUserMiddleName	116
SU_GetUserLastName	117
SU_SetUserLastName.....	118
SU_GetUserExplanation	119
SU_SetUserExplanation	120
SU_GetUserID	121
SF_DsmOpen	122
SD_GetIndex.....	123
SD_GetIndex.....	124
SD_GetStampSerialNumber.....	125
SD_GetXStamperCode.....	126
SD_GetStampID	127
SD_GetStampFilePath.....	128
SD_GetShowStampFlag	129
SD_SetShowStampFlag	130
SD_GetStretchBlitMode	131
SD_SetStretchBlitMode.....	132
SD_GetDrawStep	134
SD_SetDrawStep	135
SD_GetDrawMode.....	136
SD_SetDrawMode	137
SD_GetRopCode	139
SD_SetRopCode.....	140
SD_GetDrawBits	142
SD_SetDrawBits.....	143
SD_GetPaletteColorFlag.....	145
SD_SetPaletteColorFlag	146
SD_GetCommentMarkFlag	147
SD_SetCommentMarkFlag.....	148
SD_GetCommentMarkFlag	149
SD_SetImpressBitmapFontFlag.....	150
SD_GetSaveStampBitsFlag.....	151
SD_SetSaveStampBitsFlag	152
SD_GetAdditionTextStatus	153
SD_SetAdditionTextStatus.....	154

SD_GetAdditionText	155
SD_SetAdditionText.....	156
SD_GetStampBitsStatus.....	157
SD_SetStampBitsStatus	158
SD_GetStampBitsX.....	159
SD_GetStampBitsY.....	160
SD_GetStampBitsXPerMeter	161
SD_GetStampBitsYPerMeter	162
SD_GetItemSizeWidth	163
SD_GetItemSizeHeight	164
SD_GetItemDisplaySizeWidth	165
SD_GetItemDisplaySizeHeight	166
SD_GetItemHimetricSizeWidth	167
SD_GetItemHimetricSizeHeight	168
SD_GetAdditionDateStatus.....	169
SD_SetAdditionDateStatus	170
SD_GetAdditionDateFormat	171
SD_SetAdditionDateFormat.....	172
SD_GetImpressCountStatus	173
SD_SetImpressCountStatus	174
SD_GetImpressCount	175
SD_SetImpressCount	176
SD_GetLogStatus.....	177
SD_SetLogStatus	178
SD_GetBackDateFlag.....	179
SD_SetBackDateFlag.....	180
SD_GetSaveLogFlag	181
SD_SetSaveLogFlag	182
SD_GetDocImpressStatus	183
SD_SetDocImpressStatus	184
GF_DeleteGroup	185
GF_RenewGroupName	186
GF_Move	187
GF_MoveRoot.....	188
GF_MoveParent	189
GF_MoveFirst	190
GF_MoveLast.....	191
GF_MoveNext.....	192
GF_MovePrevious	193

GF_GetCurrentPosition	194
GF_GetCurrentGroupName	195
GF_GetCurrentGroupNamePath	196
GF_IsBOF	197
GF_IsEOF.....	198
GC_GetPosition.....	199
GC_GetGroupName.....	200
GC_GetGroupExplanation.....	201
GC_SetGroupExplanation	202
GC_GetGroupID.....	203
GC_GetApprovalPassword	204
GC_SetApprovalPassword	205
GF_AppendGroup	206
GF_PutGroup.....	207
GF_RenewParentGroup.....	208
GF_GetGroupCount.....	209
SF_AppendUser	210
SF_PutUser	211
SF_GetStamp	212
SF_PutStamp.....	213
SD_DrawStamp	214
SD_GetAdditionDate.....	215
SF_DsmClose	216
SD_SetAdditionDate.....	217
SD_GetImpressTime	218
GF_SeekGroupByUserName	219
GetVersion	220
SU_GetUserLevel	221
SU_GetUserLevelName	222
SFI_GetPackageSerialNumber	223
SFI_GetUserFirstName	224
SFI_IdxOpen	225
SFI_GetUserCount	226
SU_GetCheckOutStatus	227
SF_GetTimeServerStatus	228
SF_GetTimeServerName	229
SF_SetTimeServerStatus.....	230
SF_SetTimeServerName	231
SF_GetTimeServerFailedStatus	232

SF_SetTimeServerFailedStatus	233
SF_SeekUserByInpplet	234
SU_GetEnableUserConfigStatus.....	235
SU_SetEnableUserConfigStatus	236
SU_GetEnableCheckOutStatus.....	237
SU_SetEnableCheckOutStatus	238
SU_ReleaseCheckOut	239
SU_GetIntroducePackageSerialNumber	240
SU_GetPackageSerialNumber	241
SFI_IsBOF	242
SFI_IsEOF	243
SFI_MoveFirst	244
SFI_MoveLast.....	245
SFI_MoveNext	246
SFI_MovePrevious.....	247
SFI_SeekUser	248
SFI_SeekUserByInpplet	249
SFI_GetLastErrorType.....	250
SFI_GetLastErrorMessage	251
SFI_GetStampSerialNumber	252
SFI_GetUserID	253
SFI_GetXStamperCode	254
SFI_GetUserLastName.....	255
SF_GetLogSaveSpanMode.....	256
SF_SetLogSaveSpanMode	257
SF_GetLogSaveMaxCount	258
SF_SetLogSaveMaxCount.....	259
SFI_SaveESD.....	260
SFI_SavePng	261
SFI_GetPngData	262
SFI_GetIntroducePackageSerialNumber	263
SFI_InstallUser	264
SF_GetParentGroup.....	265
SFI_IdxClose	266
SF_GetLastUserFileException	267
SF_GetLastStampFileException.....	268
SF_GetLastGroupFileException	269
GetLastErrorNumber	270
GetLastErrorMessage	271

SF_GetDsmHeader	272
SU_GetUserEmailAddress	273
SU_SetUserEmailAddress	274
SU_GetUserEnableStatus	275
SU_SetUserEnableStatus	276
SU_GetWindowsAccount	277
SU_SetWindowsAccount	278
SU_RemoveWindowsAccount	279
SF_GetUserCacheMode	280
SF_SetUserCacheMode	281
GetEdition	282
SD_GetStampAngle	283
SD_SetStampAngle	284
SFI_SetExtractPath	285
SFI_GetExtractPath	286
ResetError	287
10 IStmpLog インタフェイス	288
GetEdition	288
LF_MoveFirst	289
LF_MoveLast	290
LF_MoveNext	291
LF_MovePrevious	292
LF_GetLogCount	293
LF_GetStampSerialNumberFlag	294
LF_SetStampSerialNumberFlag	295
LF_GetAdditionTextFlag	296
LF_SetAdditionTextFlag	297
LF_GetAdditionDateFlag	298
LF_SetAdditionDateFlag	299
LF_GetImpressCountFlag	300
LF_SetImpressCountFlag	301
LF_GetGroupNamePathFlag	302
LF_SetGroupNamePathFlag	303
LF_GetUserIDFlag	304
LF_SetUserIDFlag	305
LF_GetStampIDFlag	306
LF_SetStampIDFlag	307
LF_GetImpressIDFlag	308
LF_SetImpressIDFlag	309

LF_GetDocFileNameFlag	310
LF_SetDocFileNameFlag	311
LF_GetDocFileTitleFlag	312
LF_SetDocFileTitleFlag.....	313
LF_GetDocNumberFlag	314
LF_SetDocNumberFlag	315
LF_GetDocItemFlag	316
LF_SetDocItemFlag.....	317
LF_GetDomainNameFlag	318
LF_SetDomainNameFlag.....	319
LF_GetComputerNameFlag	320
LF_SetComputerNameFlag.....	321
LF_GetLoginNameFlag.....	322
LF_SetLoginNameFlag	323
LF_IsBOF.....	324
LF_IsEOF.....	325
LC_GetImpressCount	326
LC_GetImpressID	327
LC_GetAdditionText.....	328
LC_GetStampSerialNumber	329
LC_GetStampID	330
LC_GetGroupNamePath	331
LC_GetUserName.....	332
LC_GetUserID	333
LC_GetDocFileName.....	334
LC_GetDocTitle	335
LC_GetDocNumber.....	336
LC_GetDocItem	337
LC_GetDomainName	338
LC_GetComputerName	339
LC_GetLoginName	340
LF_DsmOpen.....	341
LC_GetImpressTime.....	342
LC_GetAdditionDate	343
LF_DeleteAllLog.....	344
LF_DeleteLog	345
LF_DsmOpen.....	346
GetVersion	347
ResetError.....	348

GetLastErrorNumber	349
GetLastErrorMessgae	350
LF_GetCurrentLogFileName.....	351
LF_GetLogFileCount	352
LF_GetLogFileName	353
LF_SetCurrentLogFile	354
11 戻り値定数一覧	355

1 はじめに

このドキュメントには、[パソコン決裁6 Extension Kit]の電子印鑑 Extension モジュールについての説明がされています。また、本ドキュメントには Windows オペレーティングシステムの操作方法やパソコン決裁6についての操作方法などを理解している方を対象に記載されています。

試用版のご利用について

ダウンロードいただいた、モジュールおよび関連するドキュメントは、ご評価を目的としたモジュールになります。そのため、本モジュールを利用して発生したコンピュータについての障害や現象については、当社は一切の保障を行いません。また、本モジュールにつきましてのご質問やお問い合わせについてはご回答を差し上げることができませんので、あらかじめご了承ください。

2 電子印鑑 Extension の概要

電子印鑑 Extension は参照された捺印用印鑑データファイル(拡張子.DSM)から、グループやユーザ、捺印ログなどの情報を設定・取得が行えます。

3 電子印鑑 Extension のファイル名

電子印鑑 Extension は次のファイルで構成されています。

製品名	ファイル名	機能
電子印鑑 Extension (x86)	DstmpLib.dll	電子印鑑の管理状態を設定・取得するモジュール 32ビット
電子印鑑 Extension (x64)	DstmpLib.dll	電子印鑑の管理状態を設定・取得するモジュール 64ビット

32ビット版、64ビット版、それぞれにモジュールが分かれています。製品名を確認するには、ファイルのプロパティから確認することができます。

4 64 ビット版に関する注意事項

64 ビット版をご利用される場合には、以下の事項に注意してご利用ください。

- ・32ビットオペレーティングシステム上では、動作しません。
- ・登録を行うコマンド("Regsvr32"など)は、64ビット用をご利用ください。
- ・Windows サービスからの呼び出しや Web サービスからの呼び出しなど、呼び出し元による制約を受ける場合があります。

5 セットアップ

電子印鑑 Extension のインストールは次のように行います。

1. メディア内の[bin]フォルダから[DstmpLib.dll]を適当なフォルダにコピーします。

登録の際に、英数字以外の文字が利用されている場合(例:"C:¥サンプル" フォルダなど)では正常に動作しない場合がございますので必ず半角英数字(例:"C:¥ComModule" フォルダなど)が利用されたフォルダを指定してください。

2. コマンドプロンプトを開きます。

Windows Vista の場合には管理権限で操作を行う必要があります。

3. 表示されたコマンドプロンプトの画面に次のように入力します。

```
regsvr32 <コピーしたモジュールのパス>¥DstmpLib.dll
```

4. 登録が成功した旨のメッセージが表示されます。

6 電子印鑑 Extension の操作

電子印鑑 Extension で処理を行うためには次のような実装を行います。

6.1 VBScript (Visual Basic Script) の場合

電子印鑑 Extension を VBScript で実装するためには次のような手順で行います。以下に手順またはコードは、実装を理解しやすくするための例示になります。開発される環境などによって操作方法や名称などが異なる場合がございますので、あらかじめご了承ください。

1. メモ帳を起動します。
2. 起動したメモ帳に以下のコードを入力します。

```
Dim objDstmpLib
Dim DSM_SUCCESS: DSM_SUCCESS = 1
Set objDstmpLib = CreateObject("DstmpLib.Extension")
objDstmpLib.DsmFile = "<捺印用印鑑データファイルの場所>"
if objDstmpLib.Login("<ユーザ名>","<パスワード>") = DSM_SUCCESS then
    `ログインに成功
    objDstmpLib.Logout()
else
    `ログインに失敗
end if
Set objDstmpLib = Nothing
msgbox "プログラムの実行が終了しました"
```

3. [ファイル]メニュー内の[名前を付けて保存]メニューを選択します。
4. 表示された[名前を付けて保存]画面で保存する場所を選択して[ファイル名]に"sample.vbs"と入力します。
5. [保存]ボタンを選択します。
6. 保存されたファイルをダブルクリックして実行を行います。
7. "プログラムの実行が終了しました"というメッセージが表示されます。

6.2 Visual Basic(Microsoft Visual Basic 2008)の場合

電子印鑑 Extension を Visual BasicV で実装するためには次のような手順で行います。以下に手順またはコードは、実装を理解しやすくするための例示になります。開発される環境などによって操作方法や名称などが異なる場合がございますので、あらかじめご了承ください。

1. Visual Basic を起動します。
2. [ファイル | 新規作成 | プロジェクト]の順にメニューを選択します。
3. [新しいプロジェクト]画面で[プロジェクトの種類]に[Visual Basic]が選択されていることを確認します。
(選択されていない場合には、[他の言語]などから探して選択します)
4. [テンプレート]から[Windows フォーム アプリケーション]を選択します。
5. [プロジェクト名]に"DstmpWindowsApplication"と入力して[OK]ボタンを選択します。

6. 新しいプロジェクトファイルが作成され、[Form1.vb]のデザイン画面が表示されます。
7. [プロジェクト | 参照の追加]メニューを選択します。
8. 表示された[参照の追加]画面で[COM]タブを選択し表示されている[コンポーネント名]一覧から"Dstmp Extension ライブラリ"を選択して[OK]ボタンを選択します。
9. [表示 | ツールボックス]メニューを選択します。
10. 表示された[ツールボックス]から[コモン コントロール]内の[Button]をマウスの左ボタンで選択し、マウスの左ボタンを押したままの状態ですべての[Form1]フォーム内に移動します。
11. [Form1]フォーム内の適当な場所でマウスの左ボタンを離します。
(フォーム内にボタンコントロールが追加されます)
12. 追加されたボタンコントロールをマウスの左ボタンでダブルクリックします。
13. 次のようなコードが追加されます。

```
Public Class Form1
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

        End Sub
    End Class
```

14. 追加されたコード内の"Private Sub" と、"End Sub" の間に次のコードを挿入します。

```
Dim objDstmpLib As DSTMPLIBLib.StmpDat
objDstmpLib = CreateObject("DstmpLib.StmpDat")
objDstmpLib.SF_DsmOpen("D:\データ\sample\STMPDAT.DSM", "")
objDstmpLib.SF_MoveFirst(0)
Do While objDstmpLib.SF_IsEOF() = False
    MsgBox(objDstmpLib.SF_GetCurrentUserName())
    objDstmpLib.SF_MoveNext(0)
Loop
objDstmpLib.SF_DsmClose()
MsgBox("プログラムの実行が終了しました")
```

15. [デバッグ | デバッグ開始]メニューを選択します。
16. ビルドが開始され、フォームが表示されます。
17. 表示された[Button1]ボタンを選択します。
18. 指定した捺印用印鑑データファイルに追加されているユーザ名が順にメッセージボックスで表示され、最後に"プログラムの実行が終了しました"というメッセージが表示されます。

6.3 Visual C#(Microsoft Visual C# 2008)の場合

電子印鑑 Extension を C#で実装するためには次のような手順で行います。以下に手順またはコードは、実装を理解しやすくするための例示になります。開発される環境などによって操作方法や名称などが異なる場合がございますので、あらかじめご了承ください。

1. Visual C#を起動します。
2. [ファイル | 新規作成 | プロジェクト]の順にメニューを選択します。
3. [新しいプロジェクト]画面で[プロジェクトの種類]に[Visual C#]が選択されていることを確認します。

(選択されていない場合には、[他の言語]などから探して選択します)

4. [テンプレート]から[Windows フォーム アプリケーション]を選択します。
5. [プロジェクト名]に"DstmpWindowsApplication"と入力して[OK]ボタンを選択します。
6. 新しいプロジェクトファイルが作成され、[Form1.cs]のデザイン画面が表示されます。
7. [プロジェクト | 参照の追加]メニューを選択します。
8. 表示された[参照の追加]画面で[COM]タブを選択し表示されている[コンポーネント名]一覧から"DstmpXml"を選択して[OK]ボタンを選択します。
9. [表示 | ツールボックス]メニューを選択します。
10. 表示された[ツールボックス]から[コモン コントロール]内の[Button]をマウスの左ボタンで選択し、マウスの左ボタンを押したままの状態ですべての[Form1]フォーム内に移動します。
11. [Form1]フォーム内の適当な場所でマウスの左ボタンを離します。
(フォーム内にボタンコントロールが追加されます)
12. 追加されたボタンコントロールをマウスの左ボタンでダブルクリックします。
13. 次のようなコードが追加されます。

```
private partial class Form1 : Form
{
    ...省略...
    private void button1_Click(object sender, EventArgs e)
    {

    }
}
```

14. 追加されたコード内の"private void button1_Click(...{" と、"}" の間に次のコードを挿入します。

```
Type type;
int DSM_SUCCESS = 1;
type = Type.GetTypeFromProgID("DstmpLib.StmpDat");
DSTMPLIBLib.IStmpDat objDstmpLib = (DSTMPLIBLib.IStmpDat)Activator.CreateInstance(type);
objDstmpLib.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>");
objDstmpLib.SF_MoveFirst(0);
while(objDstmpLib.SF_IsEOF() != DSM_SUCCESS)
{
    MessageBox.Show(objDstmpLib.SF_GetCurrentUserName());
    objDstmpLib.SF_MoveNext(0);
}
objDstmpLib.SF_DsmClose();
```

15. [デバッグ | デバッグ開始]メニューを選択します。
16. ビルドが開始され、フォームが表示されます。
17. 表示された[Button1]ボタンを選択します。
18. 指定した捺印用印鑑データファイルに追加されているユーザ名が順にメッセージボックスで表示され、最後に"プログラムの実行が終了しました"というメッセージが表示されます。

7 電子印鑑 Extension のコード例

よく利用される電子印鑑 Extension の VBScript での実装例として以下に記載します。

パソコン決裁にログインしたユーザのユーザ名を取得する (IExtension)

```
Dim objDstmpLib
Dim DSM_SUCCESS:DSM_SUCCESS = 1
Set objDstmpLib = CreateObject("DstmpLib.Extension")
objDstmpLib.DsmFile = "D:¥データ¥sample¥STMPDAT.DSM"
if objDstmpLib.Login("kian", "1111") = DSM_SUCCESS then
    msgbox objDstmpLib.GetLoginUserName()
    objDstmpLib.Logout()
end if
Set objDstmpLib = Nothing
```

パソコン決裁にログインしたユーザが所有する印鑑を ESD 形式のファイルとして出力する (IExtension)

```
Dim objDstmpLib
Dim DSM_SUCCESS:DSM_SUCCESS = 1
Set objDstmpLib = CreateObject("DstmpLib.Extension")
objDstmpLib.DsmFile = "D:¥データ¥sample¥STMPDAT.DSM"
if objDstmpLib.Login("kian", "1111") = DSM_SUCCESS then
    objDstmpLib.GenStampData("D:¥データ¥sample.esd")
    objDstmpLib.Logout()
end if
Set objDstmpLib = Nothing
```

パソコン決裁にログインしたユーザが所有する印鑑を PNG 形式のファイルとして出力する (IExtension)

```
Dim objDstmpLib
Dim DSM_SUCCESS:DSM_SUCCESS = 1
Set objDstmpLib = CreateObject("DstmpLib.Extension")
objDstmpLib.DsmFile = "D:¥データ¥sample¥STMPDAT.DSM"
if objDstmpLib.Login("kian", "1111") = DSM_SUCCESS then
    objDstmpLib.SavePng("D:¥データ¥sample.png")
    objDstmpLib.Logout()
end if
Set objDstmpLib = Nothing
```

出力された ESD 形式のファイルを読み込みファイル内のプロパティ情報を取得する (IExtension)

```
Dim objDstmpLib
Dim DSM_SUCCESS:DSM_SUCCESS = 1
Set objDstmpLib = CreateObject("DstmpLib.Extension")
objDstmpLib.EsdFile = "D:¥データ¥sample.esd"
msgbox "ユーザ名: " & objDstmpLib.GetEsdUserName()
msgbox "印鑑シリアル: " & objDstmpLib.GetEsdStampSerialNumber()
Set objDstmpLib = Nothing
```

捺印用印鑑データファイルに追加されているユーザをすべて表示する (IStmpDat)

```
Dim objDstmpLib
Dim DSM_SUCCESS:DSM_SUCCESS = 1
Set objDstmpLib = CreateObject("DstmpLib.StmpDat")
if objDstmpLib.SF_DsmOpen("D:¥データ¥sample¥STMPDAT.DSM", "") = DSM_SUCCESS then
objDstmpLib.SF_MoveFirst(0)
Do while objDstmpLib.SF_IsEOF() <> DSM_SUCCESS
msgbox objDstmpLib.SF_GetCurrentUserName()
objDstmpLib.SF_MoveNext(0)
Loop
objDstmpLib.SF_DsmClose()
end if
Set objDstmpLib = Nothing
```

捺印用印鑑データファイルの特定のユーザが追加されているグループを表示する (IStmpDat)

```
im objDstmpLib
Dim DSM_SUCCESS:DSM_SUCCESS = 1
Dim nGroupPosition:nGroupPosition = 0
Set objDstmpLib = CreateObject("DstmpLib.StmpDat")
if objDstmpLib.SF_DsmOpen("D:¥データ¥sample¥STMPDAT.DSM", "") = DSM_SUCCESS then
    if objDstmpLib.SF_SeekUser("katyou") = DSM_SUCCESS then
        nGroupPosition = objDstmpLib.SF_GetParentGroup()
        objDstmpLib.GF_Move(nGroupPosition)
        msgbox objDstmpLib.GF_GetCurrentGroupName()
    end if
    objDstmpLib.SF_DsmClose()
end if
Set objDstmpLib = Nothing
```

捺印用印鑑データファイルに追加されているユーザのプロパティを表示する (IStmpDat)

```
Dim objDstmpLib
Dim DSM_SUCCESS:DSM_SUCCESS = 1
Set objDstmpLib = CreateObject("DstmpLib.StmpDat")
if objDstmpLib.SF_DsmOpen("D:¥データ¥sample¥STMPDAT.DSM", "") = DSM_SUCCESS then
    if objDstmpLib.SF_SeekUser("katyou") = DSM_SUCCESS then
        msgbox "ユーザ名:" & objDstmpLib.SU_GetUserName
    end if
    objDstmpLib.SF_DsmClose()
end if
Set objDstmpLib = Nothing
```

ユーザに追加されている印鑑データのプロパティを表示する (IStmpDat)

```
Dim objDstmpLib
Dim DSM_SUCCESS:DSM_SUCCESS = 1
Set objDstmpLib = CreateObject("DstmpLib.StmpDat")
if objDstmpLib.SF_DsmOpen("D:¥データ¥sample¥STMPDAT.DSM", "") = DSM_SUCCESS then
    if objDstmpLib.SF_SeekUser("katyou") = DSM_SUCCESS then
        if objDstmpLib.SF_GetStamp(0) = DSM_SUCCESS then
            msgbox "印鑑シリアル:" & objDstmpLib.SD_GetStampSerialNumber()
        end if
    end if
    objDstmpLib.SF_DsmClose()
end if
Set objDstmpLib = Nothing
```

捺印履歴を表示する (IStmpLog)

```
Dim objDstmpLib
Dim DSM_SUCCESS:DSM_SUCCESS = 1
Dim nGroupPosition:nGroupPosition = 0
Set objDstmpLib = CreateObject("DstmpLib.StmpLog")
if objDstmpLib.LF_DsmOpen("D:¥データ¥sample¥STMPDAT.DSM", "") = DSM_SUCCESS then
    objDstmpLib.LF_MoveFirst()
    Do while objDstmpLib.LF_IsEOF() <> DSM_SUCCESS
        msgbox "ユーザ名:" & objDstmpLib.LC_GetUserName() & " 捺印時間:" & objDstmpLib.LC_GetImpressTime()
        if msgbox("処理を続行しますか?", 3) = 7 then
            Exit Do
        end if
        objDstmpLib.LF_MoveNext()
    Loop
    objDstmpLib.LF_DsmClose()
end if
Set objDstmpLib = Nothing
```

8 IExtension インタフェース

電子印鑑 Extension > IExtension	プロパティ
DsmFile	
参照する捺印用印鑑データファイルのフルパスを設定します。値の取得も可能です。	
設定 <code>object.DsmFile = string</code> 取得 <code>string = object.DsmFile</code>	
<p>パラメータ ありません</p> <p>戻り値 設定：対象となる捺印用印鑑データファイルの場所 取得：設定されている捺印用印鑑データファイルの場所</p> <p>備考</p>	
<p>使用例 設定 <code>object.DsmFile = "捺印用印鑑データファイルのフルパス"</code> 取得 <code>strDsmPath = object.DsmFile</code></p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを設定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" '設定されている捺印用印鑑データファイルの場所を取得 dsmFile = stamp.DsmFile '結果表示 WScript.Echo dsmFile 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '結果表示 If res = 1 Then WScript.Echo "ログインしました。" Else WScript.Echo "ログインできませんでした。" & vbCrLf & stamp.GetLastErrorMessage WScript.Quit End If 'ログアウト stamp.Logout() WScript.Echo "ログアウトしました。" Set stamp = Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetStampCount	
ログインしたユーザに追加されている印鑑の数を取得します。	
long GetStampCount(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には0以上の登録されている印鑑数が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開いていません E_NOT_CERTIFICATION(-7):ユーザがログインしていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nStampCount = object.GetStampCount()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile ="<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '印鑑データのシリアル番号を取得 stampSerialNumber = stamp.GetStampSerialNumber() msg = "シリアル番号=" & stampSerialNumber & vbCrLf WScript.Echo stamp.LastErrorMessage 'ログインしているユーザに追加されている印鑑数を取得 stampConut = stamp.GetStampCount() msg = msg & "印鑑数=" & stampConut & vbCrLf For i = 0 To stampConut-1 '印鑑の機種名を取得 xstamperCode = stamp.GetXstamperCode(i) msg = msg & "登録 Index=" & i & vbTab & "印鑑機種名=" & xstamperCode & vbCrLf Next '結果表示 WScript.Echo msg 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetXStamperCode	
ログインしたユーザに追加されている印鑑で引数 nIndex で指定した登録インデックスの印鑑機種名を取得します。	
string GetXStamperCode(long nIndex)	
<p>パラメータ nIndex 取得する印鑑データの登録インデックス（管理ツールで印鑑データのプロパティで確認できます）</p> <p>戻り値 string 成功した場合には、印鑑データの機種名が戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 機種名は、印鑑データ作成時に追加され変更することはできません。</p>	
<p>使用例 string strXStamperCode = object.GetXStamperCode(0)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile ="<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '印鑑データのシリアル番号を取得 stampSerialNumber = stamp.GetStampSerialNumber() msg = "シリアル番号=" & stampSerialNumber & vbCrLf WScript.Echo stamp.LastErrorMessage 'ログインしているユーザに追加されている印鑑数を取得 stampConut = stamp.GetStampCount() msg = msg & "印鑑数=" & stampConut & vbCrLf For i = 0 To stampConut-1 '印鑑の機種名を取得 xstamperCode = stamp.GetXstamperCode(i) msg = msg & "登録 Index=" & i & vbTab & "印鑑機種名=" & xstamperCode & vbCrLf Next '結果表示 WScript.Echo msg 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
Login	
引数で指定したユーザ名とパスワードで捺印用印鑑データファイル内のユーザにログイン認証を行います。	
long Login(string strUserName, string strUserPassword)	
<p>パラメータ strUserName: ユーザ名 strUserPassword: パスワード</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_CANNOT_OPEN(-3): 捺印用印鑑データファイルが開けません E_USER_NOT_FOUND(-5): ユーザが見つかりません E_UNMATCH_PASSWORD(-6): パスワードが一致しません E_DSM_PATH(-8): 指定された捺印用印鑑データファイルの場所が不正です E_EXPIRATION_TRIAL(-20): 試用期限が超過しています (試用版のみ) E_USER_ENABLED(-201): ユーザが無効になっています E_DSM_USING(-18): 他のアプリケーションでユーザが利用されています Windows 認証でログインする場合には以下のエラーが戻される場合があります。 E_WA_PRIVILEGE_NOT_HELD(-210) : 要求された特権を保有していません E_WA_ACCESS_DENIED(-211): アクセスが拒否されました E_WA_LOGON_FAILURE(-212): ユーザ名がパスワードが間違っています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 引数で指定したユーザ名に¥が含まれる場合には、ドメイン名¥アカウント名として Windows 認証を行います。</p>	
<p>使用例 long nResult = object.Login("ユーザ名", "パスワード") long nResult = object.Login("ドメイン名¥アカウント名", "パスワード")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.Extension") '捺印用印鑑データファイルを設定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" '設定されている捺印用印鑑データファイルの場所を取得 dsmFile = stamp.DsmFile '結果表示 WScript.Echo dsmFile 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '結果表示 If res = 1 Then WScript.Echo "ログインしました。" Else WScript.Echo "ログインできませんでした。" & vbCrLf & stamp.GetLastErrorMessage WScript.Quit End If 'ログアウト stamp.Logout() WScript.Echo "ログアウトしました。" Set stamp = Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
Logout	
現在ログインしているユーザからログアウトします。	
void Logput(void)	
<p>パラメータ ありません</p> <p>戻り値 ありません</p> <p>備考</p>	
<p>使用例 object.Logput()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを設定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" '設定されている捺印用印鑑データファイルの場所を取得 dsmFile = stamp.DsmFile '結果表示 WScript.Echo dsmFile 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '結果表示 If res = 1 Then WScript.Echo "ログインしました。" Else Wscript.Echo "ログインできませんでした。" & vbCrLf & stamp.GetLastErrorMessage WScript.Quit End If 'ログアウト stamp.Logout() WScript.Echo "ログアウトしました。" Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
SelectStamp	
引数で指定された登録インデックスまたは機種名でログインされているユーザ内の印鑑データを選択します	
long SelectStamp(long nIndex, string strXStamperCode)	
<p>パラメータ</p> <p>nIndex 選択する印鑑データの登録インデックス（管理ツールで印鑑データのプロパティで確認できます）</p> <p>strXStamperCode 選択する印鑑データの機種名</p> <p>戻り値</p> <p>long 成功した場合には、DSM_SUCCESS(1)が戻されます。 失敗した場合には次の値が戻されます。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開いていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p> <p>引数 nIndex が正の値の場合には、登録インデックスによる指定が優先されます。機種名で指定する場合には必ず引数 nIndex に負の値を指定してください。</p>	
<p>使用例</p> <p>long nResult = object.SelectStamp(0, ""), long nResult = object.SelectStamp(-1, "XL-09")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '操作対象の印鑑を選択 i=0 res = stamp.SelectStamp(i, stamp.GetXstamperCode(i)) '選択している印鑑データをビットマップ形式で出力 res = stamp.SaveBmp("C:\test¥" & i & ".bmp") 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
SaveBmp	
引数で指定された場所を選択されている印鑑データをビットマップ形式で出力します	
long SaveBmp(string strSavePath)	
<p>パラメータ strSavePath 出力するファイルの場所を指定します。</p> <p>戻り値 long 成功した場合には、DSM_SUCCESS(1)が戻されます。 失敗した場合には次の値が戻されます。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開いていません E_NOT_CERTIFICATION(-7):ユーザがログインしていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SaveBmp("C:¥sample.bmp")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '操作対象の印鑑を選択 i=0 res = stamp.SelectStamp(i, stamp.GetXstamperCode(i)) '選択している印鑑データをビットマップ形式で出力 res = stamp.SaveBmp("C:¥test¥" & i & ".bmp") 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
SaveJpg	
引数で指定された場所を選択されている印鑑データを JPEG 形式で出力します	
long SaveJpg(string strSavePath)	
<p>パラメータ strSavePath 出力するファイルの場所を指定します。</p> <p>戻り値 long 成功した場合には、DSM_SUCCESS(1)が戻されます。 失敗した場合には次の値が戻されます。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開いていません E_NOT_CERTIFICATION(-7):ユーザがログインしていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SaveJpg("C:%sample.jpg")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '操作対象の印鑑を選択 i=0 res = stamp.SelectStamp(i, stamp.GetXstamperCode(i)) '選択している印鑑データを JPEG 形式で出力 res = stamp.SaveJpg("C:%test%" & i & ".jpg") 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
SavePng	
引数で指定された場所を選択されている印鑑データを PNG 形式で出力します	
long SavePng(string strSavePath)	
<p>パラメータ strSavePath 出力するファイルの場所を指定します。</p> <p>戻り値 long 成功した場合には、DSM_SUCCESS(1)が戻されます。 失敗した場合には次の値が戻されます。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開いていません E_NOT_CERTIFICATION(-7):ユーザがログインしていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SavePng("C:¥sample.png")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '操作対象の印鑑を選択 i=0 res = stamp.SelectStamp(i, stamp.GetXstamperCode(i)) '選択している印鑑データを PNG 形式で出力 stamp.SavePng("C:¥test¥" & i & ".png") 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	プロパティ
HimetricWidth	
選択されている印鑑データの幅方向のサイズを HiMetric (0.01 ミリメートル単位) 単位で取得します。このプロパティは読み取り専用です。	
設定 <code>object.HimetricWidth = long</code> 取得 <code>long = object.HimetricWidth</code>	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には、0 以上の値が戻されます。</p> <p>備考</p>	
<p>使用例 long nResult = object.HimetricWidth()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '操作対象の印鑑を選択 i=0 res = stamp.SelectStamp(i, stamp.GetXstamperCode(i)) '選択している印鑑データの高さを HiMetric 単位で取得 himetricHeight = stamp.HimetricHeight '選択している印鑑データの幅を HiMetric 単位で取得 himetricWidth = stamp.HimetricWidth '結果表示 WScript.Echo "高さ: " & himetricHeight & " HiMetric (= " & himetricHeight / 100 & " mm)" & vbCrLf & "幅: " & himetricWidth & " HiMetric (= " & himetricWidth / 100 & " mm)" 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	プロパティ
HimetricHeight	
選択されている印鑑データの高さ方向のサイズを HiMetric (0.01 ミリメートル単位) 単位で取得します。このプロパティは読み取り専用です。	
設定 <code>object.HimetricHeight = long</code> 取得 <code>long = object.HimetricHeight</code>	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には、0 以上の値が戻されます。</p> <p>備考</p>	
<p>使用例 long nResult = object.HimetricHeight()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '操作対象の印鑑を選択 i=0 res = stamp.SelectStamp(i, stamp.GetXstamperCode(i)) '選択している印鑑データの高さを HiMetric 単位で取得 himetricHeight = stamp.HimetricHeight '選択している印鑑データの幅を HiMetric 単位で取得 himetricWidth = stamp.HimetricWidth '結果表示 WScript.Echo "高さ: " & himetricHeight & "HiMetric (= " & himetricHeight / 100 & "mm)" & vbCrLf & "幅: " & himetricWidth & "HiMetric (= " & himetricWidth / 100 & "mm)" 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
DrawStamp	
引数 nHandle で指定されたデバイスコンテキストの引数 nXPosition, nYPosition の場所に印鑑データを描画します	
long DrawStamp(long nHandle, long nXPosition, long nYPosition, long nDrawMode)	
<p>パラメータ</p> <p>n Handle:描画先のデバイスコンテキストハンドル nXPosition:描画する左方向の頂点 nYPosition:描画する上方向の頂点 nDrawMode:将来のために予約されています(0 を指定します)</p> <p>戻り値</p> <p>long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません E_NOT_CERTIFICATION(-7):ユーザがログインしていません E_DRAWMODE_VALUE(-11):引数 nDrawMode の値が不正です 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <p>long nResult = object.DrawStamp(nHandle, 0, 0, nDrawMode)</p>	
サンプルコード	

電子印鑑 Extension > IExtension	プロパティ
PixelWidth	
選択されている印鑑データの幅方向のサイズをピクセル単位で取得します。このプロパティは読み取り専用です。	
設定 object.PixelWidth = long 取得 long = object.PixelWidth	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には、0 以上の値が戻されます。</p> <p>備考</p>	
<p>使用例 long nResult = object.PixelWidth()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '操作対象の印鑑を選択 i=0 res = stamp.SelectStamp(i, stamp.GetXstamperCode(i)) '選択している印鑑データの高さをピクセル単位で取得 pixelHeight= stamp.PixelHeight '選択している印鑑データの幅をピクセル単位で取得 pixelWidth = stamp.PixelWidth '結果表示 WScript.Echo "高さ: " & pixelHeight & "ピクセル" & vbCrLf & _ "幅: " & pixelWidth & "ピクセル" 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	プロパティ
PixelHeight	
選択されている印鑑データの高さ方向のサイズをピクセル単位で取得します。このプロパティは読み取り専用です。	
設定 <code>object.PixelHeight = long</code> 取得 <code>long = object.PixelHeight</code>	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には、0 以上の値が戻されます。</p> <p>備考</p>	
<p>使用例 long nResult = object.PixelHeight()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '操作対象の印鑑を選択 i=0 res = stamp.SelectStamp(i, stamp.GetXstamperCode(i)) '選択している印鑑データの高さをピクセル単位で取得 pixelHeight= stamp.PixelHeight '選択している印鑑データの幅をピクセル単位で取得 pixelWidth = stamp.PixelWidth '結果表示 WScript.Echo "高さ: " & pixelHeight & "ピクセル" & vbCrLf & _ "幅: " & pixelWidth & "ピクセル" 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GenStampData	
引数で指定された場所を選択されている印鑑データを専用（ESD）形式で出力します。出力されたファイルは、専用 ActiveX コントロール（電子印鑑コントロール）の DataBlockFile プロパティに設定することで表示されるようになります。	
long GenStampData(string strSavePath)	
<p>パラメータ</p> <p>strSavePath 出力する ESD ファイルの場所 引数にフォルダを指定（終端を¥にした場合）には捺印 ID をファイル名としてファイルを出力します。</p> <p>戻り値</p> <p>string 成功した場合には捺印 ID を 16 進に変換した文字列が戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMesssage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <pre>long nResult = object.GenStampData("C:¥sample.esd")</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" '操作対象ユーザでログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '操作対象の印鑑を選択 i=0 res = stamp.SelectStamp(i, stamp.GetXstamperCode(i)) '出力ファイル名を指定して印鑑データを ESD 形式で出力 res1 = stamp.GenStampData("C:¥test¥" & i & ".esd¥") '出力先フォルダを指定して印鑑データを ESD 形式で出力 res2 = stamp.GenStampData("C:¥test¥") '結果表示 WScript.Echo "出力ファイル名を指定: 捺印 ID=" & res1 & vbCrLf & _ "出力先フォルダを指定: 捺印 ID=" & res2 'ログアウト stamp.Logout() Set stamp =Nothing</pre>	

電子印鑑 Extension > IExtension	メソッド
GetVersion	
モジュールのバージョン情報を文字列で取得します	
string GetVersion(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には 0.00.000.0000 形式で DstmpLib ライブラリのバージョン情報が文字列で戻されます。</p> <p>備考</p>	
<p>使用例 string strVersion = object.GetVersion()</p>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.Extension") 'ライブラリモジュールのバージョンを取得 version = stamp.GetVersion() 'ライブラリモジュールのエディションを取得 edition = stamp.GetEdition() '結果表示 WScript.Echo "バージョン" & vbTab & version & vbCrLf & _ "エディション" & vbTab & edition Set stamp = Nothing</pre>	

電子印鑑 Extension > IExtension	メソッド
SetCurrentDate	
印影を描画する際に利用する日付の値を設定します。設定されていない場合にはモジュールが動作しているシステムの日付が使用されます。	
void SetCurrentDate(nYear, nMonth, nDay, nHour, nMinute, nSecond)	
<p>パラメータ</p> <p>nYear:設定する年（有効な 4 桁の数値 1900～2038） nMonth:設定する月（有効な 4 桁の数値 1～12） nDay:設定する日（有効な 4 桁の数値 1～31） nHour:設定する時（有効な 4 桁の数値 0～23） nMinute:設定する分（有効な 4 桁の数値 0～59） nSecond:設定する秒（有効な 4 桁の数値 0～59）</p> <p>戻り値 ありません</p> <p>備考</p>	
<p>使用例</p> <pre>object.SetCurrentDate(2009, 1, 2, 3, 4, 5)</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" '操作対象ユーザでログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '日付の値を設定 stamp.SetCurrentDate 2007, 8, 9, 12, 34, 56 '結果表示 '印鑑を選択 i=0 res = stamp.SelectStamp(i, stamp.GetXstamperCode(i)) '出力ファイル名を指定して印鑑データを ESD 形式で出力 res = stamp.GenStampData("C:\test\%date.esd") '出力した ESD ファイルを指定 stamp.EsdFile = "C:\test\%date.esd" '捺印日時を表示 WScript.Echo "捺印日時:" & stamp.GetEsdImpressDate() 'ログアウト stamp.Logout() Set stamp =Nothing</pre>	

電子印鑑 Extension > IExtension	メソッド
LoginByDeviceID	
引数で指定されたデバイス ID でログインを行います。引数で指定するデバイス ID 値は、別コントロール（電子印鑑コントロール）の DeviceID プロパティ値や GetDeviceID メソッドで取得された値を利用します。	
long LoginByDeviceID(strDeviceID)	
<p>パラメータ strDeviceID: GetDeviceID メソッドなどを利用して取得したインプレットのデバイス ID</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_USER_NOT_FOUND(-5): 指定されたユーザが見つかりません E_USER_ENABLED(-201): ユーザが無効になっています E_EXPIRATION_TRIAL(-20): 試用期限が経過しています（試用版のみ） 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMesssage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LoginByDeviceID(strDeviceID)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" WScript.Echo "インプレットで[OK]ボタンをクリックしてください。" 'インプレットのデバイス ID を取得 deviceID = stamp.GetDeviceID() '取得したデバイス ID でログイン res = stamp.LoginByDeviceID(deviceID) '結果表示 If res = 1 Then WScript.Echo stamp.GetLoginUserName() & "でログインしました。" Else WScript.Echo stamp.GetLoginUserName() & "ログインできませんでした。" & vbCrLf & _ stamp.GetLastErrorMessage WScript.Quit End If 'ログアウト stamp.Logout() Set stamp=Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetLoginUserName	
現在ログインしているユーザのユーザ名を取得します。	
string GetLoginUserName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には、現在ログインしているユーザのユーザ名を戻します。 失敗した場合には空文字が戻されます。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strUserName = object.GetLoginUserName()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '現在ログインしているユーザのユーザ名を取得 userName = stamp.GetLoginUserName() '現在ログインしているユーザの姓を取得 userLastName = stamp.GetUserLastName() '現在ログインしているユーザの名を取得 userFirstName = stamp.GetUserFirstName() '現在ログインしているユーザのミドルネームを取得 userMiddleName = stamp.GetUserMiddleName() '結果表示 WScript.Echo "ユーザ名" & userName & vbCrLf & _ "姓:" & userLastName & vbCrLf & _ "名:" & userFirstName & vbCrLf & _ "ミドルネーム:" & userMiddleName 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetStampSerialNumber	
現在選択されている印鑑データのシリアル番号を取得します。	
string GetStampSerialNumber(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には、印鑑データのシリアル番号が戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strStampSerialNumber = object.GetStampSerialNumber()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile ="<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '印鑑データのシリアル番号を取得 stampSerialNumber = stamp.GetStampSerialNumber() msg = "シリアル番号=" & stampSerialNumber & vbCrLf WScript.Echo stamp.LastErrorMessage 'ログインしているユーザに追加されている印鑑数を取得 stampConut = stamp.GetStampCount() msg = msg & "印鑑数=" & stampConut & vbCrLf For i = 0 To stampConut-1 '印鑑の機種名を取得 xstamperCode = stamp.GetXstamperCode(i) msg = msg & "登録 Index=" & i & vbCrLf & "印鑑機種名=" & xstamperCode & vbCrLf Next '結果表示 WScript.Echo msg 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetBmpData	
印鑑データのビットマップ形式のポインタを取得します。引数 nBuffer に NULL を指定した場合にはバッファサイズのみを返します。	
long GetBmpData(long nBuffer)	
<p>パラメータ nBuffer ポインタを格納するバッファアドレス</p> <p>戻り値 long 正常に取得された場合にはバッファに必要なサイズが返ります。 エラーの場合には負の値が返ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 本メソッドは引数にポインタアドレスを想定しているため C++などのポインタが取り扱える開発環境で利用するために作成されています。</p>	
<p>使用例 long nBufferSize = object.GetBmpData(NULL)</p>	
サンプルコード	

電子印鑑 Extension > IExtension	メソッド
GetPngData	
印鑑データの PNG 形式のポインタを取得します。引数 nBuffer に NULL を指定した場合にはバッファサイズのみを返します。	
long GetPngData(long nBuffer)	
<p>パラメータ nBuffer ポインタを格納するバッファアドレス</p> <p>戻り値 long 正常に取得された場合にはバッファに必要なサイズが返ります。 エラーの場合には負の値が返ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 本メソッドは引数にポインタアドレスを想定しているため C++などのポインタが取り扱える開発環境で利用するために作成されています。</p>	
<p>使用例 long nBufferSize = object.GetPngData(NULL)</p>	
サンプルコード	

電子印鑑 Extension > IExtension	メソッド
GetJpgData	
印鑑データの JPG 形式のポインタを取得します。引数 nBuffer に NULL を指定した場合にはバッファサイズのみを返します。	
long GetJpgData(long nBuffer)	
<p>パラメータ nBuffer ポインタを格納するバッファアドレス</p> <p>戻り値 long 正常に取得された場合にはバッファに必要なサイズが返ります。 エラーの場合には負の値が返ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessag メソッドを使用します。</p> <p>備考 本メソッドは引数にポインタアドレスを想定しているため C++などのポインタが取り扱える開発環境で利用するために作成されています。</p>	
<p>使用例 long nBufferSize = object.GetJpgData(NULL)</p>	
サンプルコード	

電子印鑑 Extension > IExtension	メソッド
GetUserLevel	
ログインしているユーザの役職 ID を取得します。	
long GetUserLevel(void)	
<p>パラメータ ありません</p> <p>戻り値 long 正常に取得された場合には設定されている役職 ID が戻ります。 エラーの場合には負の値が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 役職 ID は、パソコン決裁 管理ツールでユーザに対して設定される項目です。</p>	
<p>使用例 long nUserLevel = object.GetUserLevel()</p>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '現在ログインしているユーザの役職 ID を取得 userLevel = stamp.GetUserLevel() '現在ログインしているユーザの役職名を取得 userLevelName = stamp.GetUserLevelName() '結果表示 WScript.Echo "役職:" & userLevel & " " & userLevelName 'ログアウト stamp.Logout() Set stamp =Nothing</pre>	

電子印鑑 Extension > IExtension	メソッド
GetUserLevelName	
ログインしているユーザの役職名を取得します。	
string GetUserLevelName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 正常に取得された場合には設定されている役職名が戻ります。 エラーの場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strUserLevelName = object.GetUserLevelName()</p>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '現在ログインしているユーザの役職レベル (役職 ID) を取得 userLevel = stamp.GetUserLevel() '現在ログインしているユーザの役職名を取得 userLevelName = stamp.GetUserLevelName() '結果表示 WScript.Echo "役職:" & userLevel & " " & userLevelName 'ログアウト stamp.Logout() Set stamp =Nothing</pre>	

電子印鑑 Extension > IExtension	メソッド
GetLevelName	
引数で指定された役職レベルに対応した役職名を取得します。	
string GetLevelName(long nLevel)	
<p>パラメータ</p> <p>nLevel 取得する役職レベル</p> <p>戻り値</p> <p>string 正常に取得された場合には指定された役職レベルに対応した役職名が戻ります。 設定が無い場合やエラーの場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMesssage メソッドを使用します。</p> <p>備考</p> <p>このメソッドは、ユーザがログインしていない状態でも利用可能です。</p>	
<p>使用例</p> <pre>string strLevelName = object.GetLevelName(nLevel)</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" '任意のユーザでログイン stamp.Login "<ユーザ名>" "<パスワード>" '指定した役職 ID に対応する役職名を取得 i=1 levelName = stamp.GetLevelName(i) WScript.Echo "役職 ID=" & i & vbCrLf & "役職名=" & levelName 'ログアウト stamp.Logout() Set stamp =Nothing</pre>	

電子印鑑 Extension > IExtension	メソッド
GetParentUserCount	
ログインしているユーザの上階層に追加されているユーザ数を取得します。	
long GetParentUserCount(void)	
<p>パラメータ ありません</p> <p>戻り値 long 正常に取得された場合にはログインしているユーザの上階層に追加されているユーザ数が取得されます。 エラーの場合には負の値が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nParentUserCount = object.GetParentUserCount()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") 'ログインしているユーザの上階層に追加されているユーザ数を取得 parentUserCount = stamp.GetParentUserCount() '結果表示 WScript.Echo "上階層ユーザ数=" & parentUserCount For i = 0 To parentUserCount-1 Step 1 '上階層ユーザのユーザ名を取得 parentUserName = stamp.GetParentUserName(i) '上階層ユーザの姓を取得 parentUserLastName = stamp.GetParentUserLastName(i) '上階層ユーザの名を取得 parentUserFirstName = stamp.GetParentUserFirstName(i) '上階層ユーザのミドルネームを取得 parentUserMiddleName = stamp.GetParentUserMiddleName(i) '上階層ユーザの説明を取得 parentUserExplanation = stamp.GetParentUserExplanation(i) '結果表示 WScript.Echo i & "/" & parentUserCount-1 & vbCrLf & "ユーザ名:" & parentUserName & vbCrLf & _ "姓:" & parentUserLastName & vbCrLf & "名:" & parentUserFirstName & vbCrLf & _ "ミドルネーム:" & parentUserMiddleName & vbCrLf & "説明:" & parentUserExplanation Next 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetParentUserName	
引数で指定されたインデックスのログインしているユーザの上階層のユーザ名を取得します。	
string GetParentUserName(long nIndex)	
<p>パラメータ nIndex 取得するユーザのインデックス</p> <p>戻り値 string 正常に取得された場合には引数で指定されたインデックスを持つユーザ名が戻ります。 エラーの場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strUserName = object.GetParentUserName(0)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") 'ログインしているユーザの上階層に追加されているユーザ数を取得 parentUserCount = stamp.GetParentUserCount() '結果表示 WScript.Echo "上階層ユーザ数=" & parentUserCount For i = 0 To parentUserCount-1 Step 1 '上階層ユーザのユーザ名を取得 parentUserName = stamp.GetParentUserName(i) '上階層ユーザの姓を取得 parentUserLastName = stamp.GetParentUserLastName(i) '上階層ユーザの名を取得 parentUserFirstName = stamp.GetParentUserFirstName(i) '上階層ユーザのミドルネームを取得 parentUserMiddleName = stamp.GetParentUserMiddleName(i) '上階層ユーザの説明を取得 parentUserExplanation = stamp.GetParentUserExplanation(i) '結果表示 WScript.Echo i & "/" & parentUserCount-1 & vbCrLf & "ユーザ名:" & parentUserName & vbCrLf & _ "姓:" & parentUserLastName & vbCrLf & "名:" & parentUserFirstName & vbCrLf & _ "ミドルネーム:" & parentUserMiddleName & vbCrLf & "説明:" & parentUserExplanation Next 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetUserFirstName	
ログインしているユーザの名を取得します。	
string GetUserFirstName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 正常に取得された場合にはユーザの名が戻ります。 エラーの場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strUserFirstName = object.GetUserFirstName()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '現在ログインしているユーザのユーザ名を取得 userName = stamp.GetLoginUserName() '現在ログインしているユーザの姓を取得 userLastName = stamp.GetUserLastName() '現在ログインしているユーザの名を取得 userFirstName = stamp.GetUserFirstName() '現在ログインしているユーザのミドルネームを取得 userMiddleName = stamp.GetUserMiddleName() '結果表示 WScript.Echo "ユーザ名" & userName & vbCrLf & _ "姓:" & userLastName & vbCrLf & _ "名:" & userFirstName & vbCrLf & _ "ミドルネーム:" & userMiddleName 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetUserLastName	
ログインしているユーザの姓を取得します。	
string GetUserLastName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 正常に取得された場合にはユーザの姓が戻ります。 エラーの場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strUserLastName = object.GetUserLastName()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '現在ログインしているユーザのユーザ名を取得 userName = stamp.GetLoginUserName() '現在ログインしているユーザの姓を取得 userLastName = stamp.GetUserLastName() '現在ログインしているユーザの名を取得 userFirstName = stamp.GetUserFirstName() '現在ログインしているユーザのミドルネームを取得 userMiddleName = stamp.GetUserMiddleName() '結果表示 WScript.Echo "ユーザ名" & userName & vbCrLf & _ "姓:" & userLastName & vbCrLf & _ "名:" & userFirstName & vbCrLf & _ "ミドルネーム:" & userMiddleName 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetUserMiddleName	
ログインしているユーザのミドルネームを取得します。	
string GetUserMiddleName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 正常に取得された場合にはユーザのミドルネームが戻ります。 エラーの場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strUserMiddleName = object.GetUserMiddleName()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '現在ログインしているユーザのユーザ名を取得 userName = stamp.GetLoginUserName() '現在ログインしているユーザの姓を取得 userLastName = stamp.GetUserLastName() '現在ログインしているユーザの名を取得 userFirstName = stamp.GetUserFirstName() '現在ログインしているユーザのミドルネームを取得 userMiddleName = stamp.GetUserMiddleName() '結果表示 WScript.Echo "ユーザ名" & userName & vbCrLf & _ "姓:" & userLastName & vbCrLf & _ "名:" & userFirstName & vbCrLf & _ "ミドルネーム:" & userMiddleName 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetUserExplanation	
ログインしているユーザの説明を取得します。	
string GetUserExplanation(void)	
<p>パラメータ ありません</p> <p>戻り値 string 正常に取得された場合にはユーザの説明が戻ります。 エラーの場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strUserExplanation = object.GetUserExplanation()</p>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>" "<パスワード>") '現在ログインしているユーザの説明を取得 userExplanation = stamp.GetUserExplanation() '結果表示 WScript.Echo "説明:" & userExplanation 'ログアウト stamp.Logout() Set stamp =Nothing</pre>	

電子印鑑 Extension > IExtension	メソッド
GetParentUserFirstName	
引数で指定されたインデックスのログインしているユーザの上階層のユーザの名を取得します。	
string GetParentUserFirstName(long nIndex)	
<p>パラメータ nIndex 取得するユーザのインデックス</p> <p>戻り値 string 正常に取得された場合には引数で指定されたインデックスを持つユーザの名が戻ります。 エラーの場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strParentUserFirstName = object.GetParentUserFirstName(0)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") 'ログインしているユーザの上階層に追加されているユーザ数を取得 parentUserCount = stamp.GetParentUserCount() '結果表示 WScript.Echo "上階層ユーザ数=" & parentUserCount For i = 0 To parentUserCount-1 Step 1 '上階層ユーザのユーザ名を取得 parentUserName = stamp.GetParentUserName(i) '上階層ユーザの姓を取得 parentUserLastName = stamp.GetParentUserLastName(i) '上階層ユーザの名を取得 parentUserFirstName = stamp.GetParentUserFirstName(i) '上階層ユーザのミドルネームを取得 parentUserMiddleName = stamp.GetParentUserMiddleName(i) '上階層ユーザの説明を取得 parentUserExplanation = stamp.GetParentUserExplanation(i) '結果表示 WScript.Echo i & "/" & parentUserCount-1 & vbCrLf & "ユーザ名:" & parentUserName & vbCrLf & _ "姓:" & parentUserLastName & vbCrLf & "名:" & parentUserFirstName & vbCrLf & _ "ミドルネーム:" & parentUserMiddleName & vbCrLf & "説明:" & parentUserExplanation Next 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetParentUserLastName	
引数で指定されたインデックスのログインしているユーザの上階層のユーザの姓を取得します。	
string GetParentUserLastName(long nIndex)	
<p>パラメータ</p> <p>nIndex 取得するユーザのインデックス</p> <p>戻り値</p> <p>string 正常に取得された場合には引数で指定されたインデックスを持つユーザの姓が戻ります。 エラーの場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <pre>string strParentUserLastName = object.GetParentUserLastName(0)</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") 'ログインしているユーザの上階層に追加されているユーザ数を取得 parentUserCount = stamp.GetParentUserCount() '結果表示 WScript.Echo "上階層ユーザ数=" & parentUserCount For i = 0 To parentUserCount-1 Step 1 '上階層ユーザのユーザ名を取得 parentUserName = stamp.GetParentUserName(i) '上階層ユーザの姓を取得 parentUserLastName = stamp.GetParentUserLastName(i) '上階層ユーザの名を取得 parentUserFirstName = stamp.GetParentUserFirstName(i) '上階層ユーザのミドルネームを取得 parentUserMiddleName = stamp.GetParentUserMiddleName(i) '上階層ユーザの説明を取得 parentUserExplanation = stamp.GetParentUserExplanation(i) '結果表示 WScript.Echo i & "/" & parentUserCount-1 & vbCrLf & "ユーザ名:" & parentUserName & vbCrLf & _ "姓:" & parentUserLastName & vbCrLf & "名:" & parentUserFirstName & vbCrLf & _ "ミドルネーム:" & parentUserMiddleName & vbCrLf & "説明:" & parentUserExplanation Next 'ログアウト stamp.Logout() Set stamp =Nothing</pre>	

電子印鑑 Extension > IExtension	メソッド
GetParentUserMiddleName	
引数で指定されたインデックスのログインしているユーザの上階層のユーザのミドルネームを取得します。	
string GetParentUserMiddleName(long nIndex)	
<p>パラメータ nIndex 取得するユーザのインデックス</p> <p>戻り値 string 正常に取得された場合には引数で指定されたインデックスを持つユーザのミドルネームが戻ります。 エラーの場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strParentUserMiddleName = object.GetParentUserMiddleName(0)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") 'ログインしているユーザの上階層に追加されているユーザ数を取得 parentUserCount = stamp.GetParentUserCount() '結果表示 WScript.Echo "上階層ユーザ数=" & parentUserCount For i = 0 To parentUserCount-1 Step 1 '上階層ユーザのユーザ名を取得 parentUserName = stamp.GetParentUserName(i) '上階層ユーザの姓を取得 parentUserLastName = stamp.GetParentUserLastName(i) '上階層ユーザの名を取得 parentUserFirstName = stamp.GetParentUserFirstName(i) '上階層ユーザのミドルネームを取得 parentUserMiddleName = stamp.GetParentUserMiddleName(i) '上階層ユーザの説明を取得 parentUserExplanation = stamp.GetParentUserExplanation(i) '結果表示 WScript.Echo i & "/" & parentUserCount-1 & vbCrLf & "ユーザ名:" & parentUserName & vbCrLf & _ "姓:" & parentUserLastName & vbCrLf & "名:" & parentUserFirstName & vbCrLf & _ "ミドルネーム:" & parentUserMiddleName & vbCrLf & "説明:" & parentUserExplanation Next 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetParentUserExplanation	
引数で指定されたインデックスのログインしているユーザの上階層のユーザの説明を取得します。	
string GetParentUserExplanation(long nIndex)	
<p>パラメータ nIndex 取得するユーザのインデックス</p> <p>戻り値 string 正常に取得された場合には引数で指定されたインデックスを持つユーザの説明が戻ります。 エラーの場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strParentUserExplanation = object.GetParentUserExplanation(0)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") 'ログインしているユーザの上階層に追加されているユーザ数を取得 parentUserCount = stamp.GetParentUserCount() '結果表示 WScript.Echo "上階層ユーザ数=" & parentUserCount For i = 0 To parentUserCount-1 Step 1 '上階層ユーザのユーザ名を取得 parentUserName = stamp.GetParentUserName(i) '上階層ユーザの姓を取得 parentUserLastName = stamp.GetParentUserLastName(i) '上階層ユーザの名を取得 parentUserFirstName = stamp.GetParentUserFirstName(i) '上階層ユーザのミドルネームを取得 parentUserMiddleName = stamp.GetParentUserMiddleName(i) '上階層ユーザの説明を取得 parentUserExplanation = stamp.GetParentUserExplanation(i) '結果表示 WScript.Echo i & "/" & parentUserCount-1 & vbCrLf & "ユーザ名:" & parentUserName & vbCrLf & _ "姓:" & parentUserLastName & vbCrLf & "名:" & parentUserFirstName & vbCrLf & _ "ミドルネーム:" & parentUserMiddleName & vbCrLf & "説明:" & parentUserExplanation Next 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GenStampDataBase64	
選択されている印鑑データを Base64 でエンコードした文字列を取得します。	
long GenStampDataBase64(void)	
<p>パラメータ ありません</p> <p>戻り値 string 正常に取得された場合にはエンコードされた ESD データ文字列が戻ります。 エラーの場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strResult = object.GenStampDataBase64()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" '操作対象ユーザでログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '操作対象の印鑑を選択 i=0 res = stamp.SelectStamp(i, stamp.GetXstamperCode(i)) '印鑑データを Base64 形式で出力 resBase64 = stamp.GenStampDataBase64() '結果表示 WScript.Echo "出力された Base64 文字列:" & resBase64 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
ResetError	
GetLastErrorNumber メソッドおよび GetLastErrorMessage メソッドで利用する内部エラー変数を初期化します。	
void ResetError(void)	
<p>パラメータ ありません</p> <p>戻り値 ありません</p> <p>備考 初期化後の GetLastErrorNumber メソッドは DSM_SUCCESS(1)を GetLastErrorMessage は"エラーはありません"を戻します。</p>	
<p>使用例 object.ResetError()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "" 'DMS ファイルを指定しないことでエラーを発生させる 'ログイン res = stamp.Login("", "") 'エラー情報取得 If res <> 0 Then 'エラー番号を取得 msg= "ErrorNumber:" & stamp.getLastErrorNumber() & vbCrLf 'エラーメッセージを取得 msg = msg & "ErrorMessage:" & stamp.GetLastErrorMessage() '結果表示 Wscript.Echo msg End If 'エラー情報を初期化 stamp.ResetError() '結果表示 Wscript.Echo "ErrorNumber:" & stamp.getLastErrorNumber() & vbCrLf & " ErrorMessage:" & stamp.GetLastErrorMessage() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetLastErrorNumber	
IExtension インタフェイスで発生した最終エラー番号を取得します。	
long GetLastErrorNumber(void)	
<p>パラメータ ありません</p> <p>戻り値 long 最終エラー番号が戻ります。</p> <p>備考</p>	
<p>使用例 long nResult = object.GetLastErrorNumber()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "" 'DMS ファイルを指定しないことでエラーを発生させる 'ログイン res = stamp.Login("", "") 'エラー情報取得 If res <> 0 Then 'エラー番号を取得 msg = "ErrorNumber:" & stamp.GetLastErrorNumber() & vbCrLf 'エラーメッセージを取得 msg = msg & "ErrorMessage:" & stamp.GetLastErrorMassage() '結果表示 Wscript.Echo msg End If 'エラー情報を初期化 stamp.ResetError() '結果表示 Wscript.Echo "ErrorNumber:" & stamp.GetLastErrorNumber() & vbCrLf & " ErrorMessage:" & stamp.GetLastErrorMassage() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetLastErrorMessage	
IExtension インタフェイスで発生した最終エラーメッセージを取得します。	
long GetLastErrorMessage(void)	
<p>パラメータ ありません</p> <p>戻り値 string 最終エラーメッセージが戻ります。</p> <p>備考</p>	
<p>使用例 string strResult = object.GetLastErrorMessage()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "" 'DMS ファイルを指定しないことでエラーを発生させる 'ログイン res = stamp.Login("", "") 'エラー情報取得 If res <> 0 Then 'エラー番号を取得 msg = "ErrorNumber:" & stamp.getLastErrorNumber() & vbCrLf 'エラーメッセージを取得 msg = msg & "ErrorMessage:" & stamp.GetLastErrorMessage() '結果表示 Wscript.Echo msg End If 'エラー情報を初期化 stamp.ResetError() '結果表示 Wscript.Echo "ErrorNumber:" & stamp.getLastErrorNumber() & vbCrLf & " ErrorMessage:" & stamp.GetLastErrorMessage() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetUserEmailAddress	
ログインしているユーザの電子メールアドレスを取得します。	
string GetUserEmailAddress(void)	
<p>パラメータ ありません</p> <p>戻り値 string 正常に取得された場合には設定されている電子メールアドレスが戻ります。 設定されていない場合やエラーの場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMesssage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strResult = object.GetUserEmailAddress()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>" "<パスワード>") '現在ログインしているユーザの電子メールアドレスを取得 userEmailAddress = stamp.GetUserEmailAddress() '結果表示 WScript.Echo "電子メールアドレス:" & userEmailAddress 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetUserWindowsAccount	
ログインしているユーザの Windows 認証アカウントを取得します。	
string GetUserWindowsAccount(void)	
<p>パラメータ ありません</p> <p>戻り値 string 正常に取得された場合には Windows 認証で設定されているアカウント名が戻ります。 設定されていない場合やエラーの場合には、空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strResult = object.GetUserWindowsAccount()</p>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '現在ログインしているユーザの Windows アカウントを取得 userWindowsAccount = stamp.GetUserWindowsAccount() '結果表示 WScript.Echo "Windows アカウント:" & userWindowsAccount 'ログアウト stamp.Logout() Set stamp =Nothing</pre>	

電子印鑑 Extension > IExtension	プロパティ
EsdFile	
ESD 形式の印鑑データに関するメソッドを利用する際に、GenStampData メソッドなどで出力された対象となる ESD ファイルの場所を設定します。値の取得も可能です。	
設定 object.EsdFile = string 取得 string = object.EsdFile	
パラメータ ありません 戻り値 設定：対象となる ESD ファイルの場所 取得：設定されている ESD ファイルの場所 備考	
使用例 設定 object.EsdFile = <ESD 形式ファイルのフルパス> 取得 strEsdPath = object.EsdFile	
サンプルコード <pre> Set stamp = CreateObject("DstmpLib.Extension") 'ESD ファイルを指定 stamp.EsdFile = "C:\Ytest\stamp.esd" 'ESD データ内のメジャーバージョン番号を取得 esdMajorVersion = stamp.GetEsdMajorVersion() 'ESD データ内のマイナーバージョン番号を取得 esdMinorVersion = stamp.GetEsdMinorVersion() 'ESD データ内のリビジョン番号を取得 esdRevisionVersion = stamp.GetEsdRevisionVersion() '結果表示 WScript.Echo "メジャーバージョン:" & esdMajorVersion & vbCrLf & "マイナーバージョン:" & esdMinorVersion & vbCrLf & "リビジョン:" & esdRevisionVersion Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	プロパティ
EsdBase64	
ESD 形式の印鑑データに関するメソッドを利用する際に、GenStampDataBase64 メソッドなどで出力された対象となるエンコード文字列を設定します。値の取得も可能です。	
設定 object.EsdBase64 = string 取得 string = object.EsdBase64	
<p>パラメータ ありません</p> <p>戻り値 設定：対象となるエンコード文字列 取得：設定されているエンコード文字列</p> <p>備考</p>	
<p>使用例 設定 object.EsdBase64 = <Base64 でエンコードされた文字列> 取得 strBase64 = object.EsdBase64</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" '操作対象ユーザでログイン stamp.Login "<ユーザ名>", "<パスワード>" 'ログインしているユーザに追加されている印鑑数を取得 stampConut = stamp.GetStampCount() '印鑑インデックスを指定して、印鑑を選択 i=0 res = stamp.SelectStamp(i, stamp.GetXstamperCode(i)) 'ログインしているユーザの印鑑データの ESD 形式データを Base64 エンコード文字列で出力 stampData = stamp.GenStampDataBase64() '現在ログインしているユーザをログアウト stamp.Logout() '印鑑データの Base64 エンコード文字列を設定 stamp.EsdBase64 = stampData '結果表示(ESD データから捺印 ID(10 進数)を取得して 16 進数で表示) WScript.Echo "捺印 ID:" & Hex(stamp.GetEsdImpressId) Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetEsdMejorVersion	
EsdFile プロパティまたは EsdBase64 プロパティで指定された ESD 形式ファイルのメジャーバージョン情報を取得します。	
long GetEsdMejorVersion(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には、ESD データ内のメジャーバージョン番号が戻ります。失敗した場合には次の値が戻ります。 E_ESD_PATH(-1101):指定された ESD ファイルの場所に問題があります。 E_ESD_DATA(-1102):指定された ESD データに問題があります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.GetEsdMajorVersion()</p>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.Extension") 'ESD ファイルを指定 stamp.EsdFile = "C:\test\stamp.esd" 'ESD データ内のメジャーバージョン番号を取得 esdMajorVersion = stamp.GetEsdMajorVersion() 'ESD データ内のマイナーバージョン番号を取得 esdMinorVersion = stamp.GetEsdMinorVersion() 'ESD データ内のリビジョン番号を取得 esdRevisionVersion = stamp.GetEsdRevisionVersion() '結果表示 WScript.Echo "メジャーバージョン:" & esdMajorVersion & vbCrLf & _ "マイナーバージョン:" & esdMinorVersion & vbCrLf & _ "リビジョン:" & esdRevisionVersion Set stamp =Nothing</pre>	

電子印鑑 Extension > IExtension	メソッド
GetEsdMinorVersion	
EsdFile プロパティまたは EsdBase64 プロパティで指定された ESD 形式ファイルのマイナーバージョン情報を取得します。	
long GetEsdMinorVersion(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には、ESD データ内のマイナーバージョン番号が戻ります。失敗した場合には次の値が戻ります。 E_ESD_PATH(-1101):指定された ESD ファイルの場所に問題があります。 E_ESD_DATA(-1102):指定された ESD データに問題があります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.GetEsdMinorVersion()</p>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.Extension") 'ESD ファイルを指定 stamp.EsdFile = "C:\test\stamp.esd" 'ESD データ内のメジャーバージョン番号を取得 esdMajorVersion = stamp.GetEsdMajorVersion() 'ESD データ内のマイナーバージョン番号を取得 esdMinorVersion = stamp.GetEsdMinorVersion() 'ESD データ内のリビジョン番号を取得 esdRevisionVersion = stamp.GetEsdRevisionVersion() '結果表示 WScript.Echo "メジャーバージョン:" & esdMajorVersion & vbCrLf &_ "マイナーバージョン:" & esdMinorVersion & vbCrLf &_ "リビジョン:" & esdRevisionVersion Set stamp =Nothing</pre>	

電子印鑑 Extension > IExtension	メソッド
GetEsdRevisionVersion	
EsdFile プロパティまたは EsdBase64 プロパティで指定された ESD 形式ファイルのリビジョン情報を取得します。	
long GetEsdRevisionVersion(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には、ESD データ内のリビジョン番号が戻ります。失敗した場合には次の値が戻ります。 E_ESD_PATH(-1101):指定された ESD ファイルの場所に問題があります。 E_ESD_DATA(-1102):指定された ESD データに問題があります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.GetEsdRevisionVersion()</p>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.Extension") 'ESD ファイルを指定 stamp.EsdFile = "C:\test\stamp.esd" 'ESD データ内のメジャーバージョン番号を取得 esdMajorVersion = stamp.GetEsdMajorVersion() 'ESD データ内のマイナーバージョン番号を取得 esdMinorVersion = stamp.GetEsdMinorVersion() 'ESD データ内のリビジョン番号を取得 esdRevisionVersion = stamp.GetEsdRevisionVersion() '結果表示 WScript.Echo "メジャーバージョン:" & esdMajorVersion & vbCrLf &_ "マイナーバージョン:" & esdMinorVersion & vbCrLf &_ "リビジョン:" & esdRevisionVersion Set stamp =Nothing</pre>	

電子印鑑 Extension > IExtension	メソッド
GetEsdUserName	
EsdFile プロパティまたは EsdBase64 プロパティで指定された ESD 形式ファイルに設定されているユーザ名を取得します。	
string GetEsdUserName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には、ESD データ内のユーザ名が返ります。失敗した場合には空文字が返ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strResult = object.GetEsdUserName()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") 'ESD ファイルを指定 stamp.EsdFile = "C:\test\stamp.esd" 'ESD データ内のユーザ名を取得 esdUserName= stamp.GetEsdUserName() 'ESD データ内のユーザ CID(10 進数) を取得→結果表示は 16 進数 esdUserCID= stamp.GetEsdUserCid() 'ESD データ内のユーザの姓を取得 esdUserLastName= stamp.GetEsdUserLastName() 'ESD データ内のユーザのミドルネームを取得 esdUserMiddleName = stamp.GetEsdUserMiddleName() 'ESD データ内のユーザの名を取得 esdUserFirstName = stamp.GetEsdUserFirstName() '結果表示 WScript.Echo "ユーザ名 : " & esdUserName & vbCrLf &_ "ユーザ CID : " & Hex(esdUserCID) & vbCrLf &_ "姓:" & esdUserLastName & vbCrLf &_ "名:" & esdUserFirstName Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetEsdUserLastName	
EsdFile プロパティまたは EsdBase64 プロパティで指定された ESD 形式ファイルに設定されているユーザの姓を取得します。	
string GetEsdUserLastName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には、ESD データ内のユーザの姓が戻ります。失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strResult = object.GetEsdUserLastName()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") 'ESD ファイルを指定 stamp.EsdFile = "C:\test\stamp.esd" 'ESD データ内のユーザ名を取得 esdUserName= stamp.GetEsdUserName() 'ESD データ内のユーザ CID(10 進数) を取得→結果表示は 16 進数 esdUserCID= stamp.GetEsdUserCid() 'ESD データ内のユーザの姓を取得 esdUserLastName= stamp.GetEsdUserLastName() 'ESD データ内のユーザのミドルネームを取得 esdUserMiddleName = stamp.GetEsdUserMiddleName() 'ESD データ内のユーザの名を取得 esdUserFirstName = stamp.GetEsdUserFirstName() '結果表示 WScript.Echo "ユーザ名：" & esdUserName & vbCrLf &_ "ユーザ CID：" & Hex(esdUserCID) & vbCrLf &_ "姓：" & esdUserLastName & vbCrLf &_ "名：" & esdUserFirstName Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetEsdUserFirstName	
EsdFile プロパティまたは EsdBase64 プロパティで指定された ESD 形式ファイルに設定されているユーザの名を取得します。	
string GetEsdUserFirstName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には、ESD データ内のユーザの名が戻ります。失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strResult = object.GetEsdUserFirstName()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") 'ESD ファイルを指定 stamp.EsdFile = "C:\test\stamp.esd" 'ESD データ内のユーザ名を取得 esdUserName= stamp.GetEsdUserName() 'ESD データ内のユーザ CID(10 進数) を取得→結果表示は 16 進数 esdUserCID= stamp.GetEsdUserCid() 'ESD データ内のユーザの姓を取得 esdUserLastName= stamp.GetEsdUserLastName() 'ESD データ内のユーザのミドルネームを取得 esdUserMiddleName = stamp.GetEsdUserMiddleName() 'ESD データ内のユーザの名を取得 esdUserFirstName = stamp.GetEsdUserFirstName() '結果表示 WScript.Echo "ユーザ名：" & esdUserName & vbCrLf &_ "ユーザ CID：" & Hex(esdUserCID) & vbCrLf &_ "姓：" & esdUserLastName & vbCrLf &_ "名：" & esdUserFirstName Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetEsdUserCid	
EsdFile プロパティまたは EsdBase64 プロパティで指定された ESD 形式ファイルに設定されているユーザの CID（管理ツールで登録された段階で生成される数値）値を取得します。	
long GetEsdUserCid(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には、ESD データ内のユーザ CID が戻ります。失敗した場合には次の値が戻ります。 E_ESD_PATH(-1101):指定された ESD ファイルの場所に問題があります。 E_ESD_DATA(-1102):指定された ESD データに問題があります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.GetEsdUserCid()</p>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.Extension") 'ESD ファイルを指定 stamp.EsdFile = "C:\test\stamp.esd" 'ESD データ内のユーザ名を取得 esdUserName= stamp.GetEsdUserName() 'ESD データ内のユーザ CID(10 進数) を取得→結果表示は 16 進数 esdUserCID= stamp.GetEsdUserCid() 'ESD データ内のユーザの姓を取得 esdUserLastName= stamp.GetEsdUserLastName() 'ESD データ内のユーザのミドルネームを取得 esdUserMiddleName = stamp.GetEsdUserMiddleName() 'ESD データ内のユーザの名を取得 esdUserFirstName = stamp.GetEsdUserFirstName() '結果表示 WScript.Echo "ユーザ名：" & esdUserName & vbCrLf & _ "ユーザ CID：" & Hex(esdUserCID) & vbCrLf & _ "姓：" & esdUserLastName & vbCrLf & _ "名：" & esdUserFirstName Set stamp =Nothing</pre>	

電子印鑑 Extension > IExtension	メソッド
GetEsdStampSerialNumber	
EsdFile プロパティまたは EsdBase64 プロパティで指定された ESD 形式ファイルに設定されている印鑑のシリアル番号（生産時に生成される文字列）値を取得します。	
string GetEsdStampSerialNumber(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には、ESD データ内の印鑑シリアルが戻ります。失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strResult = object.GetEsdStampSerialNumber()</p>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.Extension") 'ESD ファイルを指定 stamp.EsdFile = "C:\test\stamp.esd" 'ESD データ内の印鑑シリアルを取得 esdStampSerialNumber = stamp.GetEsdStampSerialNumber() 'ESD データ内の印鑑機種名を取得 esdXStamperCode = stamp.GetEsdXStamperCode() 'ESD データ内の印鑑 CID(10 進数)を取得→結果表示は 16 進数 esdStampCID = stamp.GetEsdStampCid() '結果表示 WScript.Echo "印鑑シリアル" & vbTab & esdStampSerialNumber & vbCrLf & _ "印鑑機種名" & vbTab & esdXStamperCode & vbCrLf & _ "印鑑 CID" & vbTab & Hex(esdStampCID) Set stamp =Nothing</pre>	

電子印鑑 Extension > IExtension	メソッド
GetEsdXStamperCode	
EsdFile プロパティまたは EsdBase64 プロパティで指定された ESD 形式ファイルに設定されている印鑑の機種名を取得します。	
string GetEsdXStamperCode(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には、ESD データ内の機種名が戻ります。失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMesssage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strResult = object.GetEsdXStamperCode()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") 'ESD ファイルを指定 stamp.EsdFile = "C:\test\stamp.esd" 'ESD データ内の印鑑シリアルを取得 esdStampSerialNumber = stamp.GetEsdStampSerialNumber() 'ESD データ内の印鑑機種名を取得 esdXStamperCode = stamp.GetEsdXStamperCode() 'ESD データ内の印鑑 CID(10 進数)を取得→結果表示は 16 進数 esdStampCID = stamp.GetEsdStampCid() '結果表示 WScript.Echo "印鑑シリアル" & vbTab & esdStampSerialNumber & vbCrLf &_ "印鑑機種名" & vbTab & esdXStamperCode & vbCrLf &_ "印鑑 CID" & vbTab & Hex(esdStampCID) Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetEsdStampCid	
EsdFile プロパティまたは EsdBase64 プロパティで指定された ESD 形式ファイルに設定されている印鑑の CID（管理ツールで登録される段階で生成される数値）を取得します。	
long GetEsdStampCid(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には、ESD データ内の印鑑 CID が戻ります。失敗した場合には次の値が戻ります。 E_ESD_PATH(-1101):指定された ESD ファイルの場所に問題があります。 E_ESD_DATA(-1102):指定された ESD データに問題があります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 このメソッドでは印鑑 CID は数値で取得されますが、管理ツールなどでは印鑑データの CID は 16 進で表示されていますのでご注意ください。</p>	
<p>使用例</p> <pre>long nResult = object.GetEsdStampCid()</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.Extension") 'ESD ファイルを指定 stamp.EsdFile = "C:\test\stamp.esd" 'ESD データ内の印鑑シリアルを取得 esdStampSerialNumber = stamp.GetEsdStampSerialNumber() 'ESD データ内の印鑑機種名を取得 esdXStamperCode = stamp.GetEsdXStamperCode() 'ESD データ内の印鑑 CID(10 進数)を取得→結果表示は 16 進数 esdStampCID = stamp.GetEsdStampCid() '結果表示 WScript.Echo "印鑑シリアル" & vbTab & esdStampSerialNumber & vbCrLf &_ "印鑑機種名" & vbTab & esdXStamperCode & vbCrLf &_ "印鑑 CID" & vbTab & Hex(esdStampCID) Set stamp =Nothing</pre>	

電子印鑑 Extension > IExtension	メソッド
GetEsdImpressDate	
EsdFile プロパティまたは EsdBase64 プロパティで指定された ESD 形式ファイルに設定されている捺印日時を取得します。	
Date GetEsdImpressDate(void)	
<p>パラメータ ありません</p> <p>戻り値 日付型 成功した場合には、ESD データ内の捺印時間が戻ります。失敗した場合には 0 が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 Date dtResult = object.GetEsdImpressDate()</p>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstampLib.Extension") 'ESD ファイルを指定 stamp.EsdFile = "C:\test\stamp.esd" 'ESD データ内の捺印時間を取得 esdImpressDate = stamp.GetEsdImpressDate() 'ESD データ内の捺印カウンタの値を取得 esdImpressCount = stamp.GetEsdImpressCount() 'ESD データ内の捺印 ID(10 進数)を取得→結果表示は 16 進数 esdImpressId = stamp.GetEsdImpressId() '結果表示 WScript.Echo "捺印時間" & vbTab & esdImpressDate & vbCrLf & _ "捺印カウンタ" & vbTab & esdImpressCount & vbCrLf & _ "捺印 ID" & vbTab & Hex(esdImpressId) Set stamp = Nothing</pre>	

電子印鑑 Extension > IExtension	メソッド
GetEsdImpressCount	
EsdFile プロパティまたは EsdBase64 プロパティで指定された ESD 形式ファイルに設定されている捺印カウンタを取得します。	
long GetEsdImpressCount(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には、ESD データ内の捺印カウンタが戻ります。失敗した場合には次の値が戻ります。 E_ESD_PATH(-1101):指定された ESD ファイルの場所に問題があります。 E_ESD_DATA(-1102):指定された ESD データに問題があります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.GetEsdImpressCount()</p>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.Extension") 'ESD ファイルを指定 stamp.EsdFile = "C:\test\stamp.esd" 'ESD データ内の捺印時間を取得 esdImpressDate = stamp.GetEsdImpressDate() 'ESD データ内の捺印カウンタの値を取得 esdImpressCount = stamp.GetEsdImpressCount() 'ESD データ内の捺印 ID(10 進数)を取得→結果表示は 16 進数 esdImpressId = stamp.GetEsdImpressId() '結果表示 WScript.Echo "捺印時間" & vbTab & esdImpressDate & vbCrLf & _ "捺印カウンタ" & vbTab & esdImpressCount & vbCrLf & _ "捺印 ID" & vbTab & Hex(esdImpressId) Set stamp =Nothing</pre>	

電子印鑑 Extension > IExtension	メソッド
GetEsdImpressId	
EsdFile プロパティまたは EsdBase64 プロパティで指定された ESD 形式ファイルに設定されている捺印 ID（捺印毎に発生する数値）を取得します。	
long GetEsdImpressId(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には、ESD データ内の捺印 ID が数値で返ります。 失敗した場合には次の値が返ります。 E_ESD_PATH(-1101):指定された ESD ファイルの場所に問題があります。 E_ESD_DATA(-1102):指定された ESD データに問題があります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 このメソッドでは捺印 ID は数値で取得されますが、管理ツールなどで表示される捺印 ID は 16 進で表示されていますのでご注意ください。</p>	
<p>使用例 long nResult = object.GetEsdImpressId()</p>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmLib.Extension") 'ESD ファイルを指定 stamp.EsdFile = "C:\test\stamp.esd" 'ESD データ内の捺印時間を取得 esdImpressDate = stamp.GetEsdImpressDate() 'ESD データ内の捺印カウンタの値を取得 esdImpressCount = stamp.GetEsdImpressCount() 'ESD データ内の捺印 ID(10 進数)を取得→結果表示は 16 進数 esdImpressId = stamp.GetEsdImpressId() '結果表示 WScript.Echo "捺印時間" & vbTab & esdImpressDate & vbCrLf & _ "捺印カウンタ" & vbTab & esdImpressCount & vbCrLf & _ "捺印 ID" & vbTab & Hex(esdImpressId) Set stamp =Nothing</pre>	

電子印鑑 Extension > IExtension	メソッド
SaveEsdImage	
引数で指定された画像形式で EsdFile プロパティまたは EsdBase64 プロパティで設定されている ESD データ内の印影イメージを出力します。	
long SaveEsdImage(string strSavePath, long nImageFormat)	
<p>パラメータ</p> <p>strSavePath: 出力するファイルの場所 nImageFormat: 画像形式（下記のいずれかの値が利用できます） DSTMP_IMAGE_BITMAP(101): Windows ビットマップ形式 DSTMP_IMAGE_JPEG(102): Jpeg 形式 DSTMP_IMAGE_PNG(103): PNG 形式</p> <p>戻り値</p> <p>long 成功した場合には、DSM_SUCCESS(1)が戻ります。 失敗した場合には次の値が戻ります。 E_UNMATCH_VALUE(-22): 引数の値が不正です。 E_ESD_PATH(-1101): 指定された ESD ファイルの場所に問題があります。 E_ESD_DATA(-1102): 指定された ESD データに問題があります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <p>long nResult = object.SaveEsdImage("C:¥Sample.jpg", nImageFormat)</p>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.Extension") 'ESD ファイルを指定 stamp.EsdFile = "C:¥test¥stamp.esd" 'ESD データ内の印影イメージをビットマップ形式で出力 res = stamp.SaveEsdImage("C:¥test¥esdImage.bmp", 101) 'ESD データ内の印影イメージを JPEG 形式で出力 res = stamp.SaveEsdImage("C:¥test¥esdImage.jpg", 102) 'ESD データ内の印影イメージを PNG 形式で出力 res = stamp.SaveEsdImage("C:¥test¥esdImage.png", 103) Set stamp = Nothing</pre>	

電子印鑑 Extension > IExtension	メソッド
GetDeviceID	
e-tablet ドライバを経由して、エンコードされたインプレット内のデバイス ID を取得します。取得されたデバイス ID は LoginByDeviceID メソッドなどの引数として利用します。	
string GetDeviceID(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には、デバイスから取得された ID 値をエンコードした文字列が戻ります。失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 このメソッドを利用する場合には、ライブラリモジュールが動作するコンピュータに e-tablet ドライバがセットアップされている必要があります。</p>	
<p>使用例 string strResult = object.GetDeviceID()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" WScript.Echo "インプレットで[OK]ボタンをクリックしてください。" 'インプレットのデバイス ID を取得 deviceID = stamp.GetDeviceID() '取得したデバイス ID でログイン res = stamp.LoginByDeviceID(deviceID) '結果表示 If res = 1 Then WScript.Echo stamp.GetLoginUserName() & "でログインしました。" Else WScript.Echo stamp.GetLoginUserName() & "ログインできませんでした。" & vbCrLf & _ stamp.GetLastErrorMessage WScript.Quit End If 'ログアウト stamp.Logout() Set stamp=Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetStampAngle	
選択された印鑑データに設定されている角度を取得します。	
long GetStampAngle(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には 0 以上の設定されている印面の角度が取得されます。 失敗した場合には次の値が戻されます。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開いていません E_NOT_CERTIFICATION(-7):ログインされていません</p> <p>備考</p>	
<p>使用例 long nResult = object.GetStampAngle()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" '操作対象ユーザでログイン stamp.Login "<ユーザ名>", "<パスワード>" '操作対象の印鑑を選択 i=0 stamp.SelectStamp i, stamp.GetXstamperCode(i) '選択している印鑑の印面角度を取得 stampAngel = stamp.GetStampAngle() '結果表示 WScript.Echo "印面角度=" & stampAngel & "度" '選択している印鑑の印面角度 50 度に設定 stamp.SetStampAngle(50) '結果表示 WScript.Echo "印面角度=" & stamp.GetStampAngle() & "度" 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
SetStampAngle	
選択された印鑑データに捺印角度を設定します。	
long SetStampAngle(long nAngle)	
<p>パラメータ long nAngle 設定する印面角度（0～359 の整数値）</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻されます。 失敗した場合には次の値が戻されます。 E_UNMATCH_VALUE(-22):引数の値が不正です E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開いていません E_NOT_CERTIFICATION(-7):ログインされていません</p> <p>備考 このメソッドで設定した印面確度は、GenStampData メソッドおよび GenStampDataBase64 メソッドで出力される ESD データには影響を与えません。（ESD データには、常に垂直方向で印影が描画されるようになります）</p>	
<p>使用例 long nResult = object.SetStampAngle(60)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" '操作対象ユーザでログイン stamp.Login "<ユーザ名>", "<パスワード>" '操作対象の印鑑を選択 i=0 stamp.SelectStamp i, stamp.GetXstamperCode(i) '選択している印鑑の印面角度を取得 stampAngel = stamp.GetStampAngle() '結果表示 WScript.Echo "印面角度=" & stampAngel & "度" '選択している印鑑の印面角度 50 度に設定 stamp.SetStampAngle(50) '結果表示 WScript.Echo "印面角度=" & stamp.GetStampAngle() & "度" 'ログアウト stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetEdition	
DstmpLib ライブラリのエディションを取得します。	
long GetEdition(void)	
<p>パラメータ ありません</p> <p>戻り値 long DstmpLib ライブラリのエディションが戻ります。 LIB_EDITION_TRIAL(0) : 試用版 LIB_EDITION_STANDARD(1) : 製品版 LIB_EDITION_TRIAL_X64(2) : 64 ビット製品版 LIB_EDITION_STANDARD_X64(3) : 64 ビット試用版</p> <p>備考 製品のエディションは、ファイルのプロパティのバージョン情報の製品名から確認することができます。</p>	
<p>使用例 long nResult = object.GetEdition()</p>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.Extension") 'ライブラリモジュールのバージョンを取得 version = stamp.GetVersion() 'ライブラリモジュールのエディションを取得 edition = stamp.GetEdition() '結果表示 WScript.Echo "バージョン" & vbTab & version & vbCrLf & _ "エディション" & vbTab & edition Set stamp = Nothing</pre>	

電子印鑑 Extension > IExtension	メソッド
SetUserLockMode	
ユーザログイン時のユーザロック設定を行います。 ユーザロックを有効にしてログイン操作を行った場合には、他のアプリケーションからログインできなくなります。	
long SetUserLockMode(long nUserLockMode)	
<p>パラメータ</p> long nUserLockMode 0:ユーザロック無効 1:ユーザロック有効	
<p>戻り値</p> long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には、次のいずれかが戻ります。 E_DSM_ALREADY_OPEN(-23):既にユーザがログイン中です。 E_UNMATCH_VALUE(-22):引数の値が不正です。	
<p>備考</p> このメソッドは、ログイン操作（Login メソッドなど）を行う前にに設定する必要があります。	
<p>使用例</p> long nResult = object.SetUserLockMode(0)	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.Extension") 'ユーザログイン時のユーザロックの状況を取得 lockMode = stamp.GetUserLockMode() '結果表示 If lockMode = 0 Then WScript.Echo "ユーザロック無効" Else WScript.Echo "ユーザロック有効" End If 'ユーザログイン時のユーザロックの状況を変更 If lockMode = 0 Then stamp.SetUserLockMode(1) 'ユーザロックを有効にする Else stamp.SetUserLockMode(0) 'ユーザロックを無効にする End If 'ユーザログイン時のユーザロックの状況を取得 lockMode = stamp.GetUserLockMode() '結果表示 If lockMode = 0 Then WScript.Echo "変更後：ユーザロック無効" Else WScript.Echo "変更後：ユーザロック有効" End If Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	メソッド
GetUserLockMode	
<p>ユーザログイン時のユーザロック設定を取得します。</p> <p>設定が有効の場合には、ログイン操作（Login メソッドなど）を行った場合に他のアプリケーションからのログインができなくなります。</p>	
long GetUserLockMode(void)	
<p>パラメータ ありません</p> <p>戻り値 long 0:ユーザロック無効 1:ユーザロック有効（初期値）</p> <p>備考</p>	
<p>使用例</p> <pre>long nResult = object.GetUserLockMode()</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.Extension") 'ユーザログイン時のユーザロックの状況を取得 lockMode = stamp.GetUserLockMode() '結果表示 If lockMode = 0 Then WScript.Echo "ユーザロック無効" Else WScript.Echo "ユーザロック有効" End If 'ユーザログイン時のユーザロックの状況を変更 If lockMode = 0 Then stamp.SetUserLockMode(1) 'ユーザロックを有効にする Else stamp.SetUserLockMode(0) 'ユーザロックを無効にする End If 'ユーザログイン時のユーザロックの状況を取得 lockMode = stamp.GetUserLockMode() '結果表示 If lockMode = 0 Then WScript.Echo "変更後：ユーザロック無効" Else WScript.Echo "変更後：ユーザロック有効" End If Set stamp =Nothing</pre>	

電子印鑑 Extension > IExtension	メソッド
GetEsdUserMiddleName	
EsdFile プロパティまたは EsdBase64 プロパティで指定された ESD 形式ファイルに設定されているユーザのミドルネームを取得します。	
string GetEsdUserMiddleName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には、ESD データ内のユーザのミドルネームが戻ります。失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strResult = object.GetEsdUserMiddleName()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.Extension") 'ESD ファイルを指定 stamp.EsdFile = "C:\test\stamp.esd" 'ESD データ内のユーザ名を取得 esdUserName= stamp.GetEsdUserName() 'ESD データ内のユーザ CID(10 進数) を取得→結果表示は 16 進数 esdUserCID= stamp.GetEsdUserCid() 'ESD データ内のユーザの姓を取得 esdUserLastName= stamp.GetEsdUserLastName() 'ESD データ内のユーザのミドルネームを取得 esdUserMiddleName = stamp.GetEsdUserMiddleName() 'ESD データ内のユーザの名を取得 esdUserFirstName = stamp.GetEsdUserFirstName() '結果表示 WScript.Echo "ユーザ名 : " & esdUserName & vbCrLf & _ "ユーザ CID : " & Hex(esdUserCID) & vbCrLf & _ "姓:" & esdUserLastName & vbCrLf & _ "名:" & esdUserFirstName Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	プロパティ
DocFileName	
捺印履歴の「文書ファイル名」項目に追加される文字列を設定します。設定された文字列は GenStampData メソッドなどで捺印操作を行った場合に、反映されます。このプロパティは取得も可能です。	
設定 object.DocFileName = string 取得 string = object.DocFileName	
パラメータ ありません 戻り値 string 捺印操作を行った場合に履歴上の項目「文書ファイル名」に追加する文字列を入力します。 備考	
使用例 string strResult = object.DocFileName object.DocFileName = string	
サンプルコード <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '文書情報の設定 stamp.DocFileName = "<文書ファイル名>" stamp.DocFileTitle = "<文書タイトル名>" stamp.DocNumber = "<文書番号>" stamp.DocItem (0) = "<承認項目>" '結果表示 WScript.Echo "文書ファイル名: " & stamp.DocFileName stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	プロパティ
DocFileTitle	
捺印履歴の「文書タイトル」項目に追加される文字列を設定します。設定された文字列は GenStampData メソッドなどで捺印操作を行った場合に、反映されます。このプロパティは取得も可能です。	
設定 object.DocFileTitle = string 取得 string = object.DocFileTitle	
パラメータ ありません 戻り値 string 捺印操作を行った場合に履歴上の項目「文書タイトル」に追加する文字列を入力します。 備考	
使用例 string strResult = object.DocFileTitle object.DocFileTitle = string	
サンプルコード <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '文書情報の設定 stamp.DocFileName = "<文書ファイル名>" stamp.DocFileTitle = "<文書タイトル名>" stamp.DocNumber = "<文書番号>" stamp.DocItem (0) = "<承認項目>" '結果表示 WScript.Echo "文書ファイル名: " & stamp.DocFileName stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	プロパティ
DocFileName	
捺印履歴の「文書番号」項目に追加される文字列を設定します。設定された文字列は GenStampData メソッドなどで捺印操作を行った場合に、反映されます。このプロパティは取得も可能です。	
設定 object.DocNumber = string 取得 string = object.DocNumber	
パラメータ ありません 戻り値 string 捺印操作を行った場合に履歴上の項目「文書番号」に追加する文字列を入力します。 備考	
使用例 string strResult = object.DocNumber object.DocNumber = string	
サンプルコード <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '文書情報の設定 stamp.DocFileName = "<文書ファイル名>" stamp.DocFileTitle = "<文書タイトル名>" stamp.DocNumber = "<文書番号>" stamp.DocItem (0) = "<承認項目>" '結果表示 WScript.Echo "文書ファイル名: " & stamp.DocFileName stamp.Logout() Set stamp =Nothing </pre>	

電子印鑑 Extension > IExtension	プロパティ
DocItem	
引数で指定したインデックスの捺印履歴の「承認項目」項目に追加される文字列を設定します。設定された文字列は GenStampData メソッドなどで捺印操作を行った場合に、反映されます。このプロパティは取得も可能です。	
設定 object.DocItem(int nIndex) = string 取得 string = object.DocItem(int nIndex)	
パラメータ nIndex : 承認項目を示すインデックス 戻り値 string 捺印操作を行った場合に履歴上の項目「承認項目」に追加する文字列を入力します。 備考	
使用例 string strResult = object.DocItem(1) object.DocItem (1) = string	
サンプルコード <pre> Set stamp = CreateObject("DstmpLib.Extension") '捺印用印鑑データファイルを指定 stamp.DsmFile = "<捺印用印鑑データファイルの場所>" 'ログイン res = stamp.Login("<ユーザ名>", "<パスワード>") '文書情報の設定 stamp.DocFileName = "<文書ファイル名>" stamp.DocFileTitle = "<文書タイトル名>" stamp.DocNumber = "<文書番号>" stamp.DocItem (0) = "<承認項目>" '結果表示 WScript.Echo "文書ファイル名: " & stamp.DocFileName stamp.Logout() Set stamp =Nothing </pre>	

9 IStmpDat インタフェイス

電子印鑑 Extension > IStmpDat	メソッド
SF_DeleteUser	
選択中のユーザを削除します。削除対象のユーザに印鑑データが追加されている場合にはすべて削除されます。	
long SF_DeleteUser (void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS (1) が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 ユーザの削除に成功した場合は、すべてのユーザ内の最初のユーザが選択されます。</p>	
<p>使用例 long nResult = object.SF_DeleteUser()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") userName = "<検索するユーザ名>" 'ユーザを検索 res = stamp.SF_SeekUser(userName) Select Case res Case 1 '指定したユーザ名のユーザが存在する(そのユーザが選択されている) '選択中のユーザを削除 res = stamp.SF_DeleteUser() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If Case -5 '指定したユーザ名のユーザが存在しない '指定したユーザ名でユーザを追加 res = stamp.SF_AppendUser(userName) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_RenewUse r Name	
選択中のユーザのユーザ名を変更します。	
long SF_RenewUserName(string strOldUserName, string strNewUserName)	
<p>パラメータ</p> <p>strOldUserName:変更前のユーザ名 strNewUserName:変更後のユーザ名</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS (1) が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません E_USER_NOT_FOUND(-5):strOldUserName で指定されたユーザが見つかりません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessag メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <p>long nResult = object.SF_RenewUserName("変更前のユーザ名", "変更後のユーザ名")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") oldUserName = "<検索するユーザ名>" newUserName = "<新しいユーザ名>" '操作対象のユーザを選択 res = stamp.SF_SeekUser(oldUserName) '選択中のユーザのユーザ名を変更 res = stamp.SF_RenewUserName(oldUserName, newUserName) '結果表示 If res = 1 Then WScript.Echo "ユーザ名を変更しました。" & vbCrLf Else WScript.Echo "ユーザ名を変更できませんでした。" & stamp.GetLastErrorMessag End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_DeleteStamp	
選択中のユーザに追加されている印鑑データを削除します。	
long SF_DeleteStamp (long nIndex)	
<p>パラメータ nIndex:削除する印鑑データの登録インデックス</p> <p>戻り値 long 成功した場合には DSM_SUCCESS (1) が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SF_DeleteStamp(0)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザの指定した登録インデックスの印鑑データを削除 res = stamp.SF_DeleteStamp(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_RenewParentUse r	
引数で指定した登録インデックスの印鑑データの追加先をユーザを変更します。	
long SF_RenewParentUser(long nIndex, string strNewUserName)	
<p>パラメータ</p> <p>nIndex:変更前のユーザに追加されている印鑑データの登録インデックス strNewUserName:変更後のユーザ名</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS (1) が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません E_USER_NOT_FOUND(-5):strNewUserName で指定されたユーザが見つかりません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <p>現在選択中のユーザの登録インデックス 1 の印鑑データを"変更後のユーザ名"に変更する場合 long nResult = object.SF_RenewParentUser(0, "変更後のユーザ名")</p>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザ（元の所有ユーザ）を選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザの指定した登録インデックスの印鑑データを指定したユーザに移動 res = stamp.SF_RenewParentUser(0, "<新しい所有ユーザ名>") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_MoveFirst	
捺印用印鑑データファイルに追加されている最初のユーザを選択します。	
long SF_MoveFirst(long nGroupPosition)	
<p>パラメータ</p> <p>long nGroupPosition: 移動するグループのポジション 0 を指定した場合にはすべてのユーザ内で移動します</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS (1) が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21): 読み取り専用で開かれています E_DSM_NOT_OPEN(-1): 捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessages メソッドを使用します。</p> <p>備考</p> <p>最初のユーザとは管理ツールなどを使って追加を行った最初のユーザになります。 移動するグループのポジション値は、目的のユーザに移動後 SF_GetParentGroup メソッドで戻された値を利用します。</p>	
<p>使用例</p> <p>long nResult = object.SF_MoveFirst(0)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '捺印用印鑑データファイルに追加されている最初のユーザを選択 'すべてのユーザ内の最初のユーザを選択 res = stamp.SF_MoveFirst(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザが最後のユーザであるか judge = stamp.SF_IsEOF() If judge < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If msg = "" i = 1 While judge = 0 '選択中のユーザが最後のユーザでない '選択中のユーザのユーザ名を取得 msg = msg & i & ":" & stamp.SF_GetCurrentUserName() & vbCrLf 'すべてのユーザ内で、選択中のユーザの次のユーザを選択 res = stamp.SF_MoveNext(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If i = i + 1 judge = stamp.SF_IsEOF() Wend '結果表示 WScript.Echo msg '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_MoveLast	
捺印用印鑑データファイルに追加されている末尾のユーザを選択します。	
long SF_MoveLast(long nGroupPosition)	
<p>パラメータ</p> <p>long nGroupPosition: 移動するグループのポジション 0 を指定した場合にはすべてのユーザ内で移動します</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS (1) が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21): 読み取り専用で開かれています E_DSM_NOT_OPEN(-1): 捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorErrorMessage メソッドを使用します。</p> <p>備考</p> <p>末尾のユーザとは管理ツールなどを使って追加を行った最近のユーザになります。 移動するグループのポジション値は、目的のユーザに移動後 SF_GetParentGroup メソッドで戻された値を利用します。</p>	
<p>使用例</p> <p>long nResult = object.SF_MoveLast(0)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '捺印用印鑑データファイルに追加されている最後のユーザを選択 'すべてのユーザ内の最後のユーザを選択 res = stamp.SF_MoveLast(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorErrorMessage() End If '捺印用印鑑データファイルに追加されているユーザ数を取得 i = stamp.SF_GetUserCount() If i < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorErrorMessage() End If '選択中のユーザが最初のユーザであるか judge = stamp.SF_IsBOF() If judge < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorErrorMessage() End If msg = "" While judge = 0 '選択中のユーザが最初のユーザでない '選択中のユーザのユーザ名を取得 msg = msg & i & ":" & stamp.SF_GetCurrentUserName() & vbCrLf 'すべてのユーザ内で、選択中のユーザの前のユーザを選択 res = stamp.SF_MovePrevious(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorErrorMessage() End If judge = stamp.SF_IsBOF() i = i - 1 Wend '結果表示 WScript.Echo msg '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_MoveNext	
選択されているユーザの次ユーザを選択します。	
long SF_MoveNext(long nGroupPosition)	
<p>パラメータ</p> <p>long nGroupPosition: 移動するグループのポジション 0 を指定した場合にはすべてのユーザ内で移動します</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS (1) が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21): 読み取り専用で開かれています E_DSM_NOT_OPEN(-1): 捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessages メソッドを使用します。</p> <p>備考</p> <p>移動するグループのポジション値は、目的のユーザに移動後 SF_GetParentGroup メソッドで戻された値を利用します。</p>	
<p>使用例</p> <p>long nResult = object.SF_MoveNext(0)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '捺印用印鑑データファイルに追加されている最初のユーザを選択 'すべてのユーザ内の最初のユーザを選択 res = stamp.SF_MoveFirst(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザが最後のユーザであるか judge = stamp.SF_IsEOF() If judge < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If msg = "" i = 1 While judge = 0 '選択中のユーザが最後のユーザでない '選択中のユーザのユーザ名を取得 msg = msg & i & ":" & stamp.SF_GetCurrentUserName() & vbCrLf 'すべてのユーザ内で、選択中のユーザの次のユーザを選択 res = stamp.SF_MoveNext(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If i = i + 1 judge = stamp.SF_IsEOF() Wend '結果表示 WScript.Echo msg '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_MovePrevious	
選択されているユーザの前ユーザを選択します。	
long SF_MovePrevious(long nGroupPosition)	
<p>パラメータ</p> <p>long nGroupPosition: 移動するグループのポジション 0 を指定した場合にはすべてのユーザ内で移動します</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS (1) が戻ります。</p> <p>失敗した場合には以下の値が戻ります。</p> <p>DSM_SUCCESS_READONLY(-21): 読み取り専用で開かれています</p> <p>E_DSM_NOT_OPEN(-1): 捺印用印鑑データファイルが開かれています</p> <p>詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p> <p>移動するグループのポジション値は、目的のユーザに移動後 SF_GetParentGroup メソッドで戻された値を利用します。</p>	
<p>使用例</p> <p>long nResult = object.SF_MovePrevious(0)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '捺印用印鑑データファイルに追加されている最後のユーザを選択 'すべてのユーザ内の最後のユーザを選択 res = stamp.SF_MoveLast(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '捺印用印鑑データファイルに追加されているユーザ数を取得 i = stamp.SF_GetUserCount() If i < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザが最初のユーザであるか judge = stamp.SF_IsBOF() If judge < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If msg = "" While judge = 0 '選択中のユーザが最初のユーザでない '選択中のユーザのユーザ名を取得 msg = msg & i & ":" & stamp.SF_GetCurrentUserName() & vbCrLf 'すべてのユーザ内で、選択中のユーザの前のユーザを選択 res = stamp.SF_MovePrevious(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If judge = stamp.SF_IsBOF() i = i - 1 Wend '結果表示 WScript.Echo msg '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetCurrentPosition	
選択されているユーザのポジション値（捺印用印鑑データファイル内でユーザデータが保存されている場所）を取得します	
long SF_GetCurrentPosition(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には選択中のユーザのポジション値が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 取得される値は、SU_GetPosition メソッドで取得される値と同一です。</p>	
<p>使用例 long nResult = object.SF_GetCurrentUserPosition()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択されているユーザが追加されているグループの位置を取得 groupPosition = stamp.SF_GetParentGroup() If groupPosition < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザが追加されているグループを選択 res = stamp.GF_Move(groupPosition) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザが追加されているグループのグループ名を取得 groupName = stamp.GF_GetCurrentGroupName() 'エラーの場合設定なしの場合ともに groupName は空文字なので、 'GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザのポジション値を取得 userPosition = stamp.SF_GetCurrentPosition() If userPosition < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザに登録されている印鑑数を取得 stampCount = stamp.SF_GetCurrentStampCount() If stampCount < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "ユーザ名:" & stamp.SF_GetCurrentUserName & vbCrLf & _ "グループポジション値:" & groupPosition & vbCrLf & _ "グループポジション名:" & groupName & vbCrLf & _ "ユーザポジション値:" & userPosition & vbCrLf & _ "ユーザに登録されている印鑑数:" & stampCount '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_SeekUser	
指定されたユーザ名で完全一致検索を行います。 ユーザが見つかった場合、そのユーザが選択されます。	
long SF_SeekUser(string strUserName)	
<p>パラメータ strUserName:検索するユーザ名</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません E_USER_NOT_FOUND(-5):指定されたユーザが見つかりません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 パソコン決裁のユーザ名は大文字・小文字を区別します。 Windows 認証が設定されている場合には、"ドメイン名¥アカウント名"で検索を行うこともできます。</p>	
<p>使用例 long nResult = object.SF_SeekUser("検索対象のユーザ名")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") userName = "<検索するユーザ名>" '指定した userName で、捺印用印鑑データファイル内のユーザを検索 res = stamp.SF_SeekUser(userName) '結果表示 Select Case res Case 1 '指定したユーザ名のユーザが存在する WScript.Echo userName & "は登録されています。" Case -5 '指定したユーザ名のユーザが存在しない WScript.Echo userName & "は登録されていません。" Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_LockUser	
選択されたユーザをロック状態（他のアプリケーションからデータの変更を禁止にする状態）にします。	
long SF_LockUser(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SF_LockUser()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザをロックする res = stamp.SF_LockUser() '結果表示 If res = 1 Then WScript.Echo stamp.SF_GetCurrentUserName & "をロックしました。" Else WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '選択中のユーザのロックを解除する res = stamp.SF_UnlockUser() '結果表示 If res = 1 Then WScript.Echo stamp.SF_GetCurrentUserName & "のロックを解除しました。" Else WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_UnlockUser	
選択されたユーザをロック状態（他のアプリケーションからデータの変更を禁止にする状態）を解除します。	
long SF_UnlockUser(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 SF_LockUser メソッドでユーザの変更禁止状態にした場合には、必ず本メソッドを呼び出してロック状態を解除してください。</p>	
<p>使用例 long nResult = object.SF_UnlockUser()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザをロックする res = stamp.SF_LockUser() '結果表示 If res = 1 Then WScript.Echo stamp.SF_GetCurrentUserName & "をロックしました。" Else WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessaqe() End If '選択中のユーザのロックを解除する res = stamp.SF_UnlockUser() '結果表示 If res = 1 Then WScript.Echo stamp.SF_GetCurrentUserName & "のロックを解除しました。" Else WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessaqe() End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetCurrentUserName	
選択されているユーザのユーザ名を取得します	
string SF_GetCurrentUserName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には選択中のユーザのユーザ名が戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 取得された値は、SU_GetUserName メソッドで取得される値と同一です</p>	
<p>使用例</p> <pre>string strResult = object.SF_GetCurrentUserName()</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択されているユーザが追加されているグループの位置を取得 groupPosition = stamp.SF_GetParentGroup() If groupPosition < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザが追加されているグループを選択 res = stamp.GF_Move(groupPosition) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザが追加されているグループのグループ名を取得 groupName = stamp.GF_GetCurrentGroupName() 'エラーの場合設定なしの場合ともに groupName は空文字なので、 'GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザのポジション値を取得 userPosition = stamp.SF_GetCurrentPosition() If userPosition < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザに登録されている印鑑数を取得 stampCount = stamp.SF_GetCurrentStampCount() If stampCount < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "ユーザ名:" & stamp.SF_GetCurrentUserName & vbCrLf & _ "グループポジション値:" & groupPosition & vbCrLf & _ "グループポジション名:" & groupName & vbCrLf & _ "ユーザポジション値:" & userPosition & vbCrLf & _ "ユーザに登録されている印鑑数:" & stampCount '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetCurrentStampCount	
選択されているユーザに追加されている印鑑データの数を取得します	
long SF_GetCurrentStampCount(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には選択中のユーザに追加されている印鑑データの数が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 取得された値は、SU_GetStampCount メソッドで取得される値と同一です</p>	
<p>使用例 long nResult = object.SF_GetCurrentStampCount()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択されているユーザが追加されているグループの位置を取得 groupPosition = stamp.SF_GetParentGroup() If groupPosition < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '選択されているユーザが追加されているグループを選択 res = stamp.GF_Move(groupPosition) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '選択されているユーザが追加されているグループのグループ名を取得 groupName = stamp.GF_GetCurrentGroupName() 'エラーの場合設定なしの場合ともに groupName は空文字なので、 'GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '選択されているユーザのポジション値を取得 userPosition = stamp.SF_GetCurrentPosition() If userPosition < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '選択されているユーザに登録されている印鑑数を取得 stampCount = stamp.SF_GetCurrentStampCount() If stampCount < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '結果表示 WScript.Echo "ユーザ名:" & stamp.SF_GetCurrentUserName & vbCrLf & _ "グループポジション値:" & groupPosition & vbCrLf & _ "グループポジション名:" & groupName & vbCrLf & _ "ユーザポジション値:" & userPosition & vbCrLf & _ "ユーザに登録されている印鑑数:" & stampCount '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetUserCount	
追加されているユーザの総数を取得します	
long SF_GetUserCount(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には追加されているユーザ数が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1): 捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessages メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SF_GetUserCount()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '捺印用印鑑データファイルに追加されている最後のユーザを選択 'すべてのユーザ内の最後のユーザを選択 res = stamp.SF_MoveLast(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & "." & stamp.GetLastErrorMessages() End If '捺印用印鑑データファイルに追加されているユーザ数を取得 i = stamp.SF_GetUserCount() If i < 0 Then WScript.Echo stamp.GetLastErrorNumber() & "." & stamp.GetLastErrorMessages() End If '選択中のユーザが最初のユーザであるか judge = stamp.SF_IsBOF() If judge < 0 Then WScript.Echo stamp.GetLastErrorNumber() & "." & stamp.GetLastErrorMessages() End If msg = "" While judge = 0 '選択中のユーザが最初のユーザでない '選択中のユーザのユーザ名を取得 msg = msg & i & "." & stamp.SF_GetCurrentUserName() & vbCrLf 'すべてのユーザ内で、選択中のユーザの前のユーザを選択 res = stamp.SF_MovePrevious(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & "." & stamp.GetLastErrorMessages() End If judge = stamp.SF_IsBOF() i = i - 1 Wend '結果表示 WScript.Echo msg '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetStampMaxCount	
1 ユーザに追加可能な印鑑データ数を取得します。	
long SF_GetStampMaxCount(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には 1 ユーザに追加可能な印鑑データ数が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 本メソッドで取得される値は、管理ツールで捺印用印鑑データファイルを作成した際に設定された値（既定値は 10）が戻ります。また、1 ユーザに追加可能な印鑑データ数は変更することはできません。</p>	
<p>使用例 long nResult = object.SF_GetStampMaxCount()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '1 ユーザに追加可能な印鑑数を取得 stampMaxCount = stamp.SF_GetStampMaxCount() If stampMaxCount < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() Else WScript.Echo "1 ユーザに追加可能な印鑑数:" & stampMaxCount End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetRootName	
捺印用印鑑データファイルに設定されているルート名を取得します。	
string SF_GetRootName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合にはルート名が戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <pre>string strResult = object.SF_GetRootName()</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '捺印用印鑑データファイルのルート ID(=クリエイト ID、10 進数)を取得→結果表示は 16 進数 rootID = stamp.SF_GetRootID() "rootID は符号付き Long 型なので、rootID の取得に成功してもその値が負となることがある "そのため GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '捺印用印鑑データファイルのルート名を取得 rootName = stamp.SF_GetRootName() If rootName = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "ルート ID(クリエイト ID):" & Hex(rootID) & vbCrLf & _ "ルート名:" & rootName '捺印用印鑑データファイルにルート名を設定 res = stamp.SF_SetRootName("統括事業部")rootName & "1") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "ルート ID(クリエイト ID):" & Hex(stamp.SF_GetRootID()) & vbCrLf & _ "ルート名:" & stamp.SF_GetRootName() '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_SetRootName	
捺印用印鑑データファイルのルート名を設定します。	
long SF_SetRootName(string strRootName)	
<p>パラメータ strRootName:設定するルート名</p> <p>戻り値 long 成功した場合には DSM_SUCCESS (1) が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SF_SetRootName("ルート名")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '捺印用印鑑データファイルのルート ID(=クリエイト ID、10 進数)を取得→結果表示は 16 進数 rootID = stamp.SF_GetRootID() "rootID は符号付き Long 型なので、rootID の取得に成功してもその値が負となることがある "そのため GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '捺印用印鑑データファイルのルート名を取得 rootName = stamp.SF_GetRootName() If rootName = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '結果表示 WScript.Echo "ルート ID(クリエイト ID):" & Hex(rootID) & vbCrLf & _ "ルート名:" & rootName '捺印用印鑑データファイルにルート名を設定 res = stamp.SF_SetRootName("統括事業部")rootName & "1") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '結果表示 WScript.Echo "ルート ID(クリエイト ID):" & Hex(stamp.SF_GetRootID()) & vbCrLf & _ "ルート名:" & stamp.SF_GetRootName() '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetRootID	
捺印用印鑑データファイルに設定されているクリエイト ID（管理ツールで生成される値）を取得します。	
long SF_GetRootID(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には 0 以上の数値が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 取得される値は、管理ツールの「捺印用印鑑データファイルのプロパティ」で確認することができます。 （管理ツールで表示される値は 16 進数に変換されて表示されています）</p>	
<p>使用例 long nResult = object.SF_GetRootID()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '捺印用印鑑データファイルのルート ID(=クリエイト ID、10 進数)を取得→結果表示は 16 進数 rootID = stamp.SF_GetRootID() "rootID は符号付き Long 型なので、rootID の取得に成功してもその値が負となることがある "そのため GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '捺印用印鑑データファイルのルート名を取得 rootName = stamp.SF_GetRootName() If rootName = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "ルート ID(クリエイト ID):" & Hex(rootID) & vbCrLf & _ "ルート名:" & rootName '捺印用印鑑データファイルにルート名を設定 res = stamp.SF_SetRootName("統括事業部")rootName & "1") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "ルート ID(クリエイト ID):" & Hex(stamp.SF_GetRootID()) & vbCrLf & _ "ルート名:" & stamp.SF_GetRootName() '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetAdminPassword	
捺印用印鑑データファイルに設定されている開くパスワードを取得します。	
string SF_GetAdminPassword(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には取得された開くパスワードが戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strResult = object.SF_GetAdminPassword()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '捺印用印鑑データファイルに設定されている開くパスワードを取得 password = stamp.SF_GetAdminPassword() 'エラーの場合、パスワードなしの場合ともに firstName は空文字なので、 "GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If If password = "" Then '捺印用印鑑データファイルに開くパスワードを設定 res = stamp.SF_SetAdminPassword("<開くパスワード>") WScript.Echo "捺印用印鑑データファイルに開くパスワードを設定しました。" Else '捺印用印鑑データファイルの開くパスワードをクリア res = stamp.SF_SetAdminPassword("") WScript.Echo "捺印用印鑑データファイルの開くパスワードをクリアしました。" End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_SetAdminPassword	
捺印用印鑑データファイルの開くパスワードを設定します。	
long SF_SetAdminPassword(string strAdminPassword)	
<p>パラメータ strAdminPassword:設定する開くパスワード</p> <p>戻り値 long 成功した場合には DSM_SUCCESS (1) が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SF_SetAdminPassword("開くパスワード")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '捺印用印鑑データファイルに設定されている開くパスワードを取得 password = stamp.SF_GetAdminPassword() 'エラーの場合、パスワードなしの場合ともに firstName は空文字なので、 'GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If If password = "" Then '捺印用印鑑データファイルに開くパスワードを設定 res = stamp.SF_SetAdminPassword("<開くパスワード>") WScript.Echo "捺印用印鑑データファイルに開くパスワードを設定しました。" Else '捺印用印鑑データファイルの開くパスワードをクリア res = stamp.SF_SetAdminPassword("") WScript.Echo "捺印用印鑑データファイルの開くパスワードをクリアしました。" End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetExplanation	
捺印用印鑑データファイルに設定されている説明を取得します。	
string SF_GetExplanation(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には取得された捺印用印鑑データファイルの説明が戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <pre>string strResult = object.SF_GetExplanation()</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '捺印用印鑑データファイルに設定されている説明を取得 explanation = stamp.SF_GetExplanation() "エラーの場合、設定なしの場合ともに explanation は空文字なので、 "GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If If explanation = "" Then '捺印用印鑑データファイルに設定されている説明を設定 res = stamp.SF_SetExplanation("テスト用") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If Else '捺印用印鑑データファイルに設定されている説明を設定 res = stamp.SF_SetExplanation("") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_SetExplanation	
捺印用印鑑データファイルの説明を設定します。	
long SF_SetExplanation(string strExplanation)	
<p>パラメータ strExplanation: 設定する捺印用印鑑データファイルの説明</p> <p>戻り値 long 成功した場合には DSM_SUCCESS (1) が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21): 読み取り専用で開かれています E_DSM_NOT_OPEN(-1): 捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SF_SetExplanation("捺印用印鑑データファイルの説明")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '捺印用印鑑データファイルに設定されている説明を取得 explanation = stamp.SF_GetExplanation() 'エラーの場合、設定なしの場合ともに explanation は空文字なので、 'GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If If explanation = "" Then '捺印用印鑑データファイルに設定されている説明を設定 res = stamp.SF_SetExplanation("テスト用") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If Else '捺印用印鑑データファイルに設定されている説明を設定 res = stamp.SF_SetExplanation("") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_IsBOF	
選択されているユーザが最初のユーザであるか判断します	
long SF_IsBOF(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS_FALSE(0)または DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 最初のユーザとは管理ツールなどを使って追加を行った最初のユーザになります。</p>	
<p>使用例 long nResult = object.SF_IsBOF()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '捺印用印鑑データファイルに追加されている最後のユーザを選択 'すべてのユーザ内の最後のユーザを選択 res = stamp.SF_MoveLast(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '捺印用印鑑データファイルに追加されているユーザ数を取得 i = stamp.SF_GetUserCount() If i < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中のユーザが最初のユーザであるか judge = stamp.SF_IsBOF() If judge < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If msg = "" While judge = 0 '選択中のユーザが最初のユーザでない '選択中のユーザのユーザ名を取得 msg = msg & i & ":" & stamp.SF_GetCurrentUserName() & vbCrLf 'すべてのユーザ内で、選択中のユーザの前のユーザを選択 res = stamp.SF_MovePrevious(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If judge = stamp.SF_IsBOF() i = i - 1 Wend '結果表示 WScript.Echo msg '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_IsEOF	
選択されているユーザが末尾のユーザであるか判断します	
long SF_IsEOF(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS_FALSE(0)または DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 末尾のユーザとは管理ツールなどを使って追加を行った最近のユーザになります。</p>	
<p>使用例 long nResult = object.SF_IsEOF()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '捺印用印鑑データファイルに追加されている最初のユーザを選択 'すべてのユーザ内の最初のユーザを選択 res = stamp.SF_MoveFirst(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中のユーザが最後のユーザであるか judge = stamp.SF_IsEOF() If judge < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If msg = "" i = 1 While judge = 0 '選択中のユーザが最後のユーザでない '選択中のユーザのユーザ名を取得 msg = msg & i & ":" & stamp.SF_GetCurrentUserName() & vbCrLf 'すべてのユーザ内で、選択中のユーザの次のユーザを選択 res = stamp.SF_MoveNext(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If i = i + 1 judge = stamp.SF_IsEOF() Wend '結果表示 WScript.Echo msg '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetPosition	
選択されているユーザのポジション値（捺印用印鑑データファイル内でユーザデータが保存されている場所）を取得します	
long SU_GetPosition(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には選択中のユーザのポジション値が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 取得される値は、SF_GetCurrentUserPosition メソッドで取得される値と同一です。</p>	
<p>使用例 long nResult = object.SU_GetUserPosition()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザのポジション値を取得 position = stamp.SU_GetPosition() '結果表示 If res < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() Else WScript.Echo stamp.SU_GetUserName() & "のポジション=" & position End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetStampCount	
選択されているユーザに追加されている印鑑データの数を取得します	
long SU_GetStampCount(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には選択中のユーザに追加されている印鑑データの数が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 取得された値は、SU_GetCurrentStampCount メソッドで取得される値と同一です</p>	
<p>使用例 long nResult = object.SU_GetStampCount()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザの CID を取得 CID= stamp.SU_GetUserID() "CID は符号付き Long 型なので、CID の取得に成功してもその値が負となることがある "そのため GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザのユーザ名を取得 userName= stamp.SU_GetUserName() If userName ="" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザの印鑑データ数を取得 count= stamp.SU_GetStampCount() If count < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "ユーザ名" & userName & vbCrLf &_ "ユーザ CID:" & Hex(CID) & vbCrLf &_ "印鑑データ数:" & count '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetUserListupFlag	
ユーザの [パソコン決裁ログイン画面でユーザ名リストに表示する] プロパティを取得します。	
long SU_GetUserListupFlag(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS_FALSE(0)または DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SU_GetUserListupFlag()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザの[パソコン決裁ログイン画面でユーザ名リストに表示する]の設定を取得 enableUserStatus = stamp.SU_GetEnableUserConfigStatus() Select Case enableUserStatus Case 1 '有効 '[パソコン決裁ログイン画面でユーザ名リストに表示する]を無効にする res = stamp.SU_SetEnableUserConfigStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if '選択中のユーザを更新 res = stamp.SF_PutUser() '結果表示 WScript.Echo enableUserStatus & "->" & stamp.SU_GetEnableUserConfigStatus() & ":" & stamp.SF_GetCurrentUserName & "をユーザ名リストに表示しません。" Case 0 '無効 '[パソコン決裁ログイン画面でユーザ名リストに表示する]を有効にする res = stamp.SU_SetEnableUserConfigStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if '選択中のユーザを更新 res = stamp.SF_PutUser() '結果表示 WScript.Echo enableUserStatus & "->" & stamp.SU_GetEnableUserConfigStatus() & ":" & stamp.SF_GetCurrentUserName & "をユーザ名リストに表示します。" Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_SetUserListupFlag	
引数で指定したユーザの「パソコン決裁ログイン画面でユーザ名リストに表示する」プロパティを設定します。	
long SU_SetUserListupFlag(BOOL bListupFlag)	
<p>パラメータ bListupFlag:0（表示しない）,1（表示する）</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 設定の変更を捺印用印鑑データに反映させるには、SF_PutUser()を実行して選択されているユーザを更新します。</p>	
<p>使用例 long nResult = object.SU_SetUserListupFlag(1)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザの[パソコン決裁ログイン画面でユーザ名リストに表示する]の設定を取得 enableUserStatus = stamp.SU_GetEnableUserConfigStatus() Select Case enableUserStatus Case 1 '有効 '[パソコン決裁ログイン画面でユーザ名リストに表示する]を無効にする res = stamp.SU_SetEnableUserConfigStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() '結果表示 WScript.Echo enableUserStatus & "->" & stamp.SU_GetEnableUserConfigStatus() & ":" & stamp.SF_GetCurrentUserName & "をユーザ名リストに表示しません。" Case 0 '無効 '[パソコン決裁ログイン画面でユーザ名リストに表示する]を有効にする res = stamp.SU_SetEnableUserConfigStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() '結果表示 WScript.Echo enableUserStatus & "->" & stamp.SU_GetEnableUserConfigStatus() & ":" & stamp.SF_GetCurrentUserName & "をユーザ名リストに表示します。" Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetUserName	
選択されているユーザのユーザ名を取得します	
string SU_GetUserName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には選択中のユーザのユーザ名が戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 取得された値は、SF_GetCurrentUserName メソッドで取得される値と同一です</p>	
<p>使用例 string strResult = object.SU_GetUserName()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("マツモト") '選択中のユーザの CID を取得 CID= stamp.SU_GetUserID() "CID は符号付き Long 型なので、CID の取得に成功してもその値が負となることがある "そのため GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザのユーザ名を取得 userName= stamp.SU_GetUserName() If userName = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザの印鑑データ数を取得 count= stamp.SU_GetStampCount() If count < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "ユーザ名" & userName & vbCrLf &_ "ユーザ CID:" & Hex(CID) & vbCrLf &_ "印鑑データ数:" & count '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetUserPassword	
選択されているユーザのパスワードを取得します	
string SU_GetUserPassword(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には選択中のユーザのパスワードが戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strResult = object.SU_GetUserPassword()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザのパスワードを取得 password = stamp.SU_GetUserPassword() 'エラーの場合、パスワードなしの場合ともに password は空文字なので、 'GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() WScript.Quit End If If password = "" Then '選択中のユーザのパスワードを設定 res = stamp.SU_SetUserPassword("<ユーザパスワード>") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() Else '選択中のユーザのパスワードを設定 res = stamp.SU_SetUserPassword("") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_SetUserPassword	
引数で指定したユーザのパスワードを設定します。	
long SU_SetUserPassword(string strUserPassword)	
<p>パラメータ strUserPassword:ユーザパスワード</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 設定の変更を捺印用印鑑データに反映させるには、SF_PutUser()を実行して選択されているユーザを更新します。</p>	
<p>使用例 long nResult = object.SU_SetUserPassword("ユーザパスワード")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザのパスワードを取得 password = stamp.SU_GetUserPassword() 'エラーの場合、パスワードなしの場合ともに password は空文字なので、 "GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() WScript.Quit End If If password = "" Then '選択中のユーザのパスワードを設定 res = stamp.SU_SetUserPassword("<開くパスワード>") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() Else '選択中のユーザのパスワードを設定 res = stamp.SU_SetUserPassword("") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetUserPasswordMinLength	
選択されているユーザのパスワード最小文字数を取得します。	
long SU_GetUserPasswordMinLength(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には 0 以上のパスワード最小文字数が戻ります。(0 の場合には最小文字数の設定がありません) 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1): 捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SU_GetUserPasswordMinLength()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザのパスワードの最小文字数を取得 length = stamp.SU_GetUserPasswordMinLength() If length < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() WScript.Quit End If If length = 0 Then '選択中のユーザのパスワードの最小文字数を設定 res = stamp.SU_SetUserPasswordMinLength(4) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() Else '選択中のユーザのパスワードの最小文字数を設定 res = stamp.SU_SetUserPasswordMinLength(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_SetUserPasswordMinLength	
選択されているユーザのパスワード最小文字数を設定します。	
long SU_SetUserPasswordMinLength(long nUserPasswordMinLength)	
<p>パラメータ</p> <p>nUserPasswordMinLength:パスワード最小文字数 (0 を指定した場合には最小文字数を設定しません)</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS(1)が戻ります。</p> <p>失敗した場合には以下の値が戻ります。</p> <p>E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません</p> <p>詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p> <p>設定の変更を捺印用印鑑データに反映させるには、SF_PutUser()を実行して選択されているユーザを更新します。</p>	
<p>使用例</p> <pre>long nResult = object.SU_SetUserPasswordMinLength(15)</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザのパスワードの最小文字数を取得 length = stamp.SU_GetUserPasswordMinLength() If length < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() WScript.Quit End If If length = 0 Then '選択中のユーザのパスワードの最小文字数を設定 res = stamp.SU_SetUserPasswordMinLength(4) If res <= 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() Else '選択中のユーザのパスワードの最小文字数を設定 res = stamp.SU_SetUserPasswordMinLength(0) If res <= 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetUserFirstName	
選択されているユーザの名を取得します。	
string SU_GetUserFirstName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合にはユーザの名が戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SU_GetUserFirstName()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザの名を取得 firstName = stamp.SU_GetUserFirstName() 'エラーの場合、名なしの場合ともに firstName は空文字なので、 'GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If If firstName = "太郎" Then '選択中のユーザの姓を設定 res = stamp.SU_SetUserFirstName("たろう") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '選択中のユーザを更新 res = stamp.SF_PutUser() Else '選択中のユーザの姓を設定 res = stamp.SU_SetUserFirstName("太郎") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '選択中のユーザを更新 res = stamp.SF_PutUser() End if '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_SetUserFirstName	
選択されているユーザの名を設定します。	
long SU_SetUserFirstName(string strUserFirstName)	
<p>パラメータ strUserFirstName: ユーザの氏名</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1): 捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 設定の変更を捺印用印鑑データに反映させるには、SF_PutUser()を実行して選択されているユーザを更新します。</p>	
<p>使用例 long nResult = object.SU_SetUserFirstName("ユーザの名")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザの名を取得 firstName = stamp.SU_GetUserFirstName() 'エラーの場合、名なしの場合ともに firstName は空文字なので、 "GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If If firstName = "太郎" Then '選択中のユーザの姓を設定 res = stamp.SU_SetUserFirstName("たろう") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() Else '選択中のユーザの姓を設定 res = stamp.SU_SetUserFirstName("太郎") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetUserMiddleName	
選択されているユーザのミドルネームを取得します。	
string SU_GetUserMiddleName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合にはユーザのミドルネームが戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SU_GetUserMiddleName()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザのミドルネームを取得 middleName = stamp.SU_GetUserMiddleName() "エラーの場合、ミドルネームなしの場合ともに middleName は空文字なので、 "GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If If middleName = "" Then '選択中のユーザのミドルネームを設定 res = stamp.SU_SetUserMiddleName("D") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() Else '選択中のユーザのミドルネームを設定 res = stamp.SU_SetUserMiddleName("") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_SetUserMiddleName	
選択されているユーザのミドルネームを設定します。	
long SU_SetUserMiddleName(string strUserMiddleName)	
<p>パラメータ strUserMiddleName:ユーザのミドルネーム</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 設定の変更を捺印用印鑑データに反映させるには、SF_PutUser()を実行して選択されているユーザを更新します。</p>	
<p>使用例 long nResult = object.SU_SetUserMiddleName("ユーザのミドルネーム")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザのミドルネームを取得 middleName = stamp.SU_GetUserMiddleName() 'エラーの場合、ミドルネームなしの場合ともに middleName は空文字なので、 'GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If If middleName = "" Then '選択中のユーザのミドルネームを設定 res = stamp.SU_SetUserMiddleName("D") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() Else '選択中のユーザのミドルネームを設定 res = stamp.SU_SetUserMiddleName("") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetUserLastName	
選択されているユーザの姓を取得します。	
string SU_GetUserLastName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合にはユーザの姓が戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SU_GetUserLastName()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザの姓を取得 lastName = stamp.SU_GetUserLastName() 'エラーの場合、姓なしの場合ともに firstName は空文字なので、 'GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If If lastName = "鈴木" Then '選択中のユーザの姓を設定 res = stamp.SU_SetUserLastName("すずき") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() Else '選択中のユーザの姓を設定 res = stamp.SU_SetUserLastName("鈴木") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_SetUserLastName	
選択されているユーザの姓を設定します。	
long SU_SetUserLastName(string strUserLastName)	
<p>パラメータ strUserLastName:ユーザの姓名</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 設定の変更を捺印用印鑑データに反映させるには、SF_PutUser()を実行して選択されているユーザを更新します。</p>	
<p>使用例 long nResult = object.SU_SetUserLastName("ユーザの姓")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザの姓を取得 lastName = stamp.SU_GetUserLastName() 'エラーの場合、姓なしの場合ともに firstName は空文字なので、 'GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If If lastName = "鈴木" Then '選択中のユーザの姓を設定 res = stamp.SU_SetUserLastName("すずき") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() Else '選択中のユーザの姓を設定 res = stamp.SU_SetUserLastName("鈴木") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetUserExplanation	
選択されているユーザの説明を取得します。	
string SU_GetUserExplanation(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合にはユーザの説明が戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SU_GetUserExplanation()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザの説明を取得 explanation = stamp.SU_GetUserExplanation() 'エラーの場合、設定なしの場合ともに explanation は空文字なので、 'GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() WScript.Quit End If If explanation = "" Then '選択中のユーザの説明を設定 res = stamp.SU_SetUserExplanation("テスト用ユーザ") '選択中のユーザを更新 res = stamp.SF_PutUser() Else '選択中のユーザの説明を設定 res = stamp.SU_SetUserExplanation("") '選択中のユーザを更新 res = stamp.SF_PutUser() End if '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_SetUserExplanation	
選択されているユーザの説明を設定します。	
long SU_SetUserExplanation(string strUserExplanation)	
<p>パラメータ strUserExplanation: ユーザの説明</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1): 捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 設定の変更を捺印用印鑑データに反映させるには、SF_PutUser()を実行して選択されているユーザを更新します。</p>	
<p>使用例 long nResult = object.SU_SetUserExplanation("ユーザの説明")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザの説明を取得 explanation = stamp.SU_GetUserExplanation() 'エラーの場合、設定なしの場合ともに explanation は空文字なので、 'GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() WScript.Quit End If If explanation = "" Then '選択中のユーザの説明を設定 res = stamp.SU_SetUserExplanation("テスト用ユーザ") '選択中のユーザを更新 res = stamp.SF_PutUser() Else '選択中のユーザの説明を設定 res = stamp.SU_SetUserExplanation("") '選択中のユーザを更新 res = stamp.SF_PutUser() End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetUserID	
ユーザに設定されている CID（管理ツールで生成される値）を取得します。	
long SU_GetUserID(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には 0 以上の数値が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 取得されるユーザ CID は、管理ツールの「ユーザのプロパティ」で確認することができます。 （管理ツールで表示されるユーザ CID は 16 進数に変換されて表示されています）</p>	
<p>使用例 long nResult = object.SU_GetUserID()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("マツモト") '選択中のユーザの CID を取得 CID= stamp.SU_GetUserID() "CID は符号付き Long 型なので、CID の取得に成功してもその値が負となることがある" 'そのため GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザのユーザ名を取得 userName= stamp.SU_GetUserName() If userName = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザの印鑑データ数を取得 count= stamp.SU_GetStampCount() If count < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "ユーザ名" & userName & vbCrLf &_ "ユーザ CID:" & Hex(CID) & vbCrLf &_ "印鑑データ数:" & count '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_DsmOpen	
引数で指定した場所の捺印用印鑑データファイルを開きます。	
long SF_DsmOpen(string strFilePath, string strPassword)	
<p>パラメータ strFilePath:捺印用印鑑データファイルへのパス文字列 strPassword:開くパスワード</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 ファイルを開いた時点では、捺印用印鑑データファイルの ・ ルートグループ ・ すべてのユーザ内で最初のユーザ が選択されています。</p>	
<p>使用例 long nResult = object.SF_DsmOpen("捺印用印鑑データファイルの場所", "開くパスワード")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '結果表示 If res = 1 Then WScript.Echo "捺印用印鑑データファイルを開きました。" Else WScript.Echo "捺印用印鑑データファイルを開けませんでした。" & vbCrLf & _ stamp.GetLastErrorNumber() & "、" & stamp.GetLastErrorMessag() End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() WScript.Echo "捺印用印鑑データファイルを閉じました。" Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetIndex	
選択中の印鑑データの登録インデックスを取得します。	
long SD_GetIndex(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には 0 以上の数値が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します</p>	
<p>使用例 long nResult = object.SD_GetIndex()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") msg = "登録インデックス, シリアル番号, CID" For i = 0 To stamp.SU_GetStampCount() - 1 '操作対象の印鑑データを選択 res = stamp.SF_GetStamp(i) If res <> 1 Then WScript.Echo "GetStamp" & res & ":" & stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの登録インデックスを取得 index = stamp.SD_GetIndex() If index < 0 Then WScript.Echo "GetIndex" & res & ":" & stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データのシリアル番号を取得 serial = stamp.SD_GetStampSerialNumber() If serial = "" Then WScript.Echo "GetStampSerial" & res & ":" & stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの CID(10 進)を取得→結果表示は 16 進 CID = stamp.SD_GetStampID() "CID は符号付き Long 型なので、CID の取得に成功してもその値が負となることがある "そのため GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If msg = msg & vbCrLf & index & ", " & serial & ", " & vbTab & Hex(CID) Next '結果表示 WScript.Echo msg '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetIndex	
選択中の印鑑データの種類を取得します。	
long SD_GetStampType(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には氏名印(1) または日付印(2)が戻ります。 ※カスタム（社判やサイン）の場合は、氏名印(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します</p>	
<p>使用例 long nResult = object.SD_GetStampType()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データで捺印したときに表示する日時書式を取得 format = stamp.SD_GetAdditionDateFormat() If stampType < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '選択中の印鑑データの種類を取得 stampType = stamp.SD_GetStampType() If stampType < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If Select Case stampType Case 1 '氏名印 '選べる書式は 10 種 (1 ~10) nextFormat = (format Mod 10) + 1 Case 2 '日付印 '選べる書式は 8 種 (1 ~8) nextFormat = (format Mod 8) + 1 Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End Select '選択中の印鑑データで捺印したときに表示する日時書式を設定 res = stamp.SD_SetAdditionDateFormat(nextFormat) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetStampSerialNumber	
選択中の印鑑データのシリアル番号を取得します。	
string SD_GetStampSerialNumber(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には印鑑データのシリアル番号が戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します</p>	
<p>使用例 string strResult = object.SD_GetStampSerialNumber()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") msg = "登録インデックス, シリアル番号, CID" For i = 0 To stamp.SU_GetStampCount() - 1 '操作対象の印鑑データを選択 res = stamp.SF_GetStamp(i) If res <> 1 Then WScript.Echo "GetStamp" & res & ":" & stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessaqe() End If '選択中の印鑑データの登録インデックスを取得 index = stamp.SD_GetIndex() If index < 0 Then WScript.Echo "GetIndex" & res & ":" & stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessaqe() End If '選択中の印鑑データのシリアル番号を取得 serial = stamp.SD_GetStampSerialNumber() If serial = "" Then WScript.Echo "GetStampSerial" & res & ":" & stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessaqe() End If '選択中の印鑑データの CID(10 進)を取得→結果表示は 16 進 CID = stamp.SD_GetStampID() "CID は符号付き Long 型なので、CID の取得に成功してもその値が負となることがある "そのため GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessaqe() End If msg = msg & vbCrLf & index & ", " & serial & ", " & vbTab & Hex(CID) Next '結果表示 WScript.Echo msg '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetXStamperCode	
選択されている印鑑データの機種名を取得します。	
string SD_GetXStamperCode(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には選択されている印鑑データの機種名が戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します</p>	
<p>使用例 string strResult = object.SD_GetXStamperCode()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの機種名を取得 xStamperCode = stamp.SD_GetXStamperCode() If xStamperCode= "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '結果表示 WScript.Echo "印鑑機種名:" & xStamperCode '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetStampID	
選択中の「印鑑データ」に設定されている CID（管理ツールで生成される値）を取得します。	
long SD_GetStampID(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には 0 以上の数値が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します 取得される印鑑 CID は、管理ツールの「印鑑データのプロパティ」で確認することができます。 （管理ツールで表示される印鑑 CID は 16 進数に変換されて表示されています） 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetStampID()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") msg = "登録インデックス, シリアル番号, CID" For i = 0 To stamp.SU_GetStampCount() - 1 '操作対象の印鑑データを選択 res = stamp.SF_GetStamp(i) If res <> 1 Then WScript.Echo "GetStamp" & res & ":" & stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの登録インデックスを取得 index = stamp.SD_GetIndex() If index < 0 Then WScript.Echo "GetIndex" & res & ":" & stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データのシリアル番号を取得 serial = stamp.SD_GetStampSerialNumber() If serial = "" Then WScript.Echo "GetStampSerial" & res & ":" & stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの CID(10 進)を取得→結果表示は 16 進 CID = stamp.SD_GetStampID() "CID は符号付き Long 型なので、CID の取得に成功してもその値が負となることがある "そのため GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If msg = msg & vbCrLf & index & ", " & serial & ", " & vbCrLf & Hex(CID) Next '結果表示 WScript.Echo msg '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetStampFilePath	
印鑑データが保存されている実ファイルへのパスを取得します。	
string SD_GetStampFilePath(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には印鑑データが保持されている実ファイルへのパスが戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します</p>	
<p>使用例 string strResult = object.SD_GetStampFilePath()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの実ファイルパスを取得 path = stamp.SD_GetStampFilePath() If path = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() Else WScript.Echo path End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetShowStampFlag	
捺印アプリケーションでの印鑑データの表示設定を取得します。	
long SD_GetShowStampFlag(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には 0 以上（1：表示 0：非表示）の数値が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します</p>	
<p>使用例 long nResult = object.SD_GetShowStampFlag()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの捺印ツールでの表示状態を取得 status = stamp.SD_GetShowStampFlag() Select Case status Case 0 '表示しない '捺印ツールで表示するように設定 res = stamp.SD_SetShowStampFlag(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 '表示する '捺印ツールで表示しないようにに設定 res = stamp.SD_SetShowStampFlag(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetShowStampFlag	
捺印アプリケーションでの印鑑データの表示、非表示を設定します。	
long SD_SetShowStampFlag(long nShowFlag)	
<p>パラメータ nShowFlag: 非表示 (0) 表示 (1)</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します</p>	
<p>使用例 long nResult = object.SD_SetShowStampFlag(1)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの捺印ツールでの表示状態を取得 status = stamp.SD_GetShowStampFlag() Select Case status Case 0 '表示しない '捺印ツールで表示するように設定 res = stamp.SD_SetShowStampFlag(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 '表示する '捺印ツールで表示しないようにに設定 res = stamp.SD_SetShowStampFlag(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetStretchBlitMode	
印面の描画時の伸縮（ストレッチ）状態を取得します。	
long SD_GetStretchBlitMode(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 BLACKONWHITE(1)：残す点の色と、取り除く点の色を論理 AND 演算子で結合して描画します COLORONCOLOR (3)：ピクセルを削除します HALFTONE (4)：取り除く点の色を削除して描画します WHITEONBLACK(2)：残す点の色と、取り除く点の色を論理 OR 演算子で結合して描画します 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21)：読み取り専用で開かれています E_DSM_NOT_</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetStretchBlitMode()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの描画時の伸縮状態を取得 status = stamp.SD_GetStretchBlitMode() Select Case status Case 1 '論理 AND 演算子で結合 '捺論理 OR 演算子で結合して描画するように設定 res = stamp.SD_SetStretchBlitMode(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 2 '論理 OR 演算子で結合 'ピクセルを削除して描画するように設定 res = stamp.SD_SetStretchBlitMode(3) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 3 'ピクセル削除 '色を削除して描画するように設定 res = stamp.SD_SetStretchBlitMode(4) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 4 '色削除 '捺論理 AND 演算子で結合して描画するように設定 res = stamp.SD_SetStretchBlitMode(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetStretchBlitMode	
印面の描画時の伸縮（ストレッチ）状態を設定します。	
long SD_SetStretchBlitMode(long nStretchMode)	
<p>パラメータ</p> <p>nStretchMode:</p> <p>BLACKONWHITE(1): 残す点の色と、取り除く点の色を論理 AND 演算子で結合して描画します</p> <p>COLORONCOLOR (3): ピクセルを削除します</p> <p>HALFTONE (4): 取り除く点の色を削除して描画します</p> <p>WHITEONBLACK(2): 残す点の色と、取り除く点の色を論理 OR 演算子で結合して描画します</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS(1)が戻ります。</p> <p>失敗した場合には以下の値が戻ります。</p> <p>DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています</p> <p>E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません</p> <p>詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p> <p>このメソッドは、データを保持する機能のみを提供して、実際の描画には影響を与えません。</p> <p>印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。</p>	
<p>使用例</p> <pre>long nResult = object.SD_SetStretchBlitMode(1)</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの描画時の伸縮状態を取得 status = stamp.SD_GetStretchBlitMode() Select Case status Case 1 '論理 AND 演算子で結合 '捺論理 OR 演算子で結合して描画するように設定 res = stamp.SD_SetStretchBlitMode(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 2 '論理 OR 演算子で結合 'ピクセルを削除して描画するように設定 res = stamp.SD_SetStretchBlitMode(3) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 3 'ピクセル削除 '色を削除して描画するように設定 res = stamp.SD_SetStretchBlitMode(4) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 es = stamp.SF_PutStamp(0) Case 4 '色削除 '捺論理 AND 演算子で結合して描画するように設定 res = stamp.SD_SetStretchBlitMode(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End Select</pre>	

捺印用印鑑データファイルを閉じる
stamp.SF_DsmClose()
Set stamp =Nothing

電子印鑑 Extension > IStmpDat	メソッド
SD_GetDrawStep	
印面の描画する際のビット転送方法を取得します。	
long SD_GetDrawStep(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 経由なし(1)：直接描画を行います DDB 経由 (2)：デバイス依存ビットマップを生成してから描画を行います DIB (3)：デバイス独立ビットマップを生成してから描画を行います 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または</p> <p>備考 本メソッドは以前のバージョンとの互換性を保つために残されていますが、現在でも利用することはできます。 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetDrawStep()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データのビット転送方法を取得 status = stamp.SD_GetDrawStep() Select Case status Case 1 '直接描画 'ビット転送方法をデバイス依存ビットマップを生成してから描画に設定 res = stamp.SD_SetDrawStep(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 2 'デバイス依存ビットマップを生成してから描画 'ビット転送方法をデバイス独立ビットマップを生成してから描画に設定 res = stamp.SD_SetDrawStep(3) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 3 'デバイス独立ビットマップを生成してから描画 'ビット転送方法を直接描画に設定 res = stamp.SD_SetDrawStep(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetDrawStep	
印面の描画する際のビット転送方法を設定します。	
long SD_SetDrawStep(long nDrawStep)	
<p>パラメータ</p> <p>long</p> <p>経由なし(1)：直接描画を行います DDB 経由 (2)：デバイス依存ビットマップを生成してから描画を行います DIB (3)：デバイス独立ビットマップを生成してから描画を行います</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p> <p>このメソッドは、データを保持する機能のみを提供して、実際の描画には影響を与えません。 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。</p>	
<p>使用例</p> <p>long nResult = object.SD_SetDrawStep(1)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データのビット転送方法を取得 status = stamp.SD_GetDrawStep() Select Case status Case 1 '直接描画 'ビット転送方法をデバイス依存ビットマップを生成してから描画に設定 res = stamp.SD_SetDrawStep(2) If res <> 1 Then WScript.Echo stamp.LastErrorNumber() & ":" & stamp.LastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 2 'デバイス依存ビットマップを生成してから描画 'ビット転送方法をデバイス独立ビットマップを生成してから描画に設定 res = stamp.SD_SetDrawStep(3) If res <> 1 Then WScript.Echo stamp.LastErrorNumber() & ":" & stamp.LastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 3 'デバイス独立ビットマップを生成してから描画 'ビット転送方法を直接描画に設定 res = stamp.SD_SetDrawStep(1) If res <> 1 Then WScript.Echo stamp.LastErrorNumber() & ":" & stamp.LastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.LastErrorNumber() & ":" & stamp.LastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetDrawMode	
印面の描画する際の描画モードを取得します。	
long SD_GetDrawMode(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 背景透過(0)：背景（印面以外の部分）を透過した状態で描画を行います 透過なし（1）：背景（印面以外の部分）を白で塗りつぶした状態で描画を行います 全透過（2）：背景以外（印面の部分）も含めて透過した状態で描画を行います ROP 指定（3）：描画方法をラスターオペレーションコードで指定します 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetDrawMode()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの描画モードを取得 status = stamp.SD_GetDrawMode() Select Case status Case 0 '印面外を背景透過 '選択中の印鑑データの描画モードを背景透過なしに設定 res = stamp.SD_SetDrawMode(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 '背景透過なし '選択中の印鑑データの描画モードを印面内も背景透過に設定 res = stamp.SD_SetDrawMode(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 2 '印面外を背景透過 '選択中の印鑑データの描画モードを描画方法をラスターオペレーションコードで指定に設定 res = stamp.SD_SetDrawMode(3) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 3 '描画方法をラスターオペレーションコードで指定 '選択中の印鑑データの描画モードを印面外を背景透過に設定 res = stamp.SD_SetDrawMode(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetDrawMode	
印面の描画する際の描画モードを設定します。	
long SD_SetDrawMode(long nDrawMode)	
<p>パラメータ</p> <p>long</p> <p>背景透過(0)：背景（印面以外の部分）を透過した状態で描画を行います 透過なし（1）：背景（印面以外の部分）を白で塗りつぶした状態で描画を行います 全透過（2）：背景以外（印面の部分）も含めて透過した状態で描画を行います ROP 指定（3）：描画方法をラスターオペレーションコードで指定します</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p> <p>このメソッドは、データを保持する機能のみを提供して、実際の描画には影響を与えません。 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。 ROP 指定を行う場合には、SD_SetRopCode を利用します。</p>	
<p>使用例</p> <pre>long nResult = object.SD_SetDrawMode(1)</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの描画モードを取得 status = stamp.SD_GetDrawMode() Select Case status Case 0 '印面外を背景透過 '選択中の印鑑データの描画モードを背景透過なしに設定 res = stamp.SD_SetDrawMode(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 '背景透過なし '選択中の印鑑データの描画モードを印面内も背景透過に設定 res = stamp.SD_SetDrawMode(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 2 '印面外を背景透過 '選択中の印鑑データの描画モードを描画方法をラスターオペレーションコードで指定に設定 res = stamp.SD_SetDrawMode(3) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 3 '描画方法をラスターオペレーションコードで指定 '選択中の印鑑データの描画モードを印面外を背景透過に設定 res = stamp.SD_SetDrawMode(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose()</pre>	

Set stamp =Nothing

電子印鑑 Extension > IStmpDat	メソッド
SD_GetRopCode	
印面の描画する際のラスタオペレーションコードを取得します。	
long SD_GetRopCode(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には設定されているラスタオペレーションコードの値が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessages メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetRopCode()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) ropCode = stamp.SD_GetRopCode() res = stamp.SD_SetRopCode(1)'SRCCOPY stamp.SF_PutStamp(0) WScript.Echo res & ":" & ropCode & "-> " & stamp.SD_GetRopCode() ropCode = stamp.SD_GetRopCode() res = stamp.SD_SetRopCode(2)'SRCAND stamp.SF_PutStamp(0) WScript.Echo res & ":" & ropCode & "-> " & stamp.SD_GetRopCode() '選択中の印鑑データのラスタオペレーションコードを取得 ropCode = stamp.SD_GetRopCode() If ropCode < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End If 'Select Case status ' Case 0 'パレットカラーを指定しない ' 'パレットカラーを指定するように設定 ' res = stamp.SD_SetPaletteColorFlag(1) ' If res <> 1 Then ' WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() ' End If ' '印鑑データを更新 ' res = stamp.SF_PutStamp(0) ' Case 1 'パレットカラーを指定する ' 'テキストとして描画に設定 ' res = stamp.SD_SetPaletteColorFlag(0) ' If res <> 1 Then ' WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() ' End If ' '印鑑データを更新 ' res = stamp.SF_PutStamp(0) ' Case Else 'エラー ' WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() 'End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetRopCode	
<p>印面の描画する際のラスタオペレーションコードを設定します。</p> <p>ラスタオペレーションコードとは、出力処理の中で、現在のブラシ、転送先のビットマップ、転送元のビットマップにそれぞれ含まれる色をグラフィックスデバイスインタフェイス（GDI）が結合する方法を定義したものです。</p>	
long SD_SetRopCode(long nRopCode)	
<p>パラメータ</p> <p>long 描画方法をラスタオペレーションコードを指定します</p> <p>戻り値</p> <p>long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessages メソッドを使用します。</p> <p>備考</p> <p>このメソッドは、データを保持する機能のみを提供して、実際の描画には影響を与えません。 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。 ROP 指定を行う場合には、SD_SetRopCode を利用します。</p>	
<p>使用例</p> <pre>long nResult = object.SD_SetRopCode(1)</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) ropCode = stamp.SD_GetRopCode() res = stamp.SD_SetRopCode(1)'SRCCOPY) stamp.SF_PutStamp(0) WScript.Echo res & ":" & ropCode & "-> " & stamp.SD_GetRopCode() ropCode = stamp.SD_GetRopCode() res = stamp.SD_SetRopCode(2)'SRCAND) stamp.SF_PutStamp(0) WScript.Echo res & ":" & ropCode & "-> " & stamp.SD_GetRopCode() '選択中の印鑑データのラスタオペレーションコードを取得 ropCode = stamp.SD_GetRopCode() If ropCode < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End If 'Select Case status ' Case 0 'パレットカラーを指定しない ' 'パレットカラーを指定するように設定 ' res = stamp.SD_SetPaletteColorFlag(1) ' If res <> 1 Then ' WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() ' End If ' '印鑑データを更新 ' res = stamp.SF_PutStamp(0) ' Case 1 'パレットカラーを指定する ' 'テキストとして描画に設定 ' res = stamp.SD_SetPaletteColorFlag(0) ' If res <> 1 Then ' WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() ' End If ' '印鑑データを更新 ' res = stamp.SF_PutStamp(0) ' Case Else 'エラー ' WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() 'End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetDrawBits	
印面の描画する際の描画ビットを取得します。	
long SD_GetDrawBits(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 1 ビット (1) : 2 色で印影を描画します 4 ビット (4) : 16 色で印影を描画します 8 ビット (8) : 256 色で印影を描画します 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。 この値は、[印面プロパティ]-[描画]-[描画パターン]の[詳細設定]-[カラーパレット]プロパティに設定されています。</p>	
<p>使用例 long nResult = object.SD_GetDrawBits()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの[印面プロパティ]-[描画]-[描画パターン]の[詳細設定]-[カラーパレット]プロパティを取得 status = stamp.SD_GetDrawBits() Select Case status Case 1 '2 色(1 ビット) '16 色(4 ビット)に設定 res = stamp.SD_SetDrawBits(4) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 4 '16 色(4 ビット) '256 色(8 ビット)に設定 res = stamp.SD_SetDrawBits(8) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 8 '256 色(8 ビット) '2 色(1 ビット)に設定 res = stamp.SD_SetDrawBits(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetDrawBits	
印面の描画する際の描画ビットを設定します。	
long SD_SetDrawBits(long nDrawBits)	
<p>パラメータ</p> <p>long 印影の描画ビット 1 ビット (1) : 2 色で印影を描画します 4 ビット (4) : 16 色で印影を描画します 8 ビット (8) : 256 色で印影を描画します</p> <p>戻り値</p> <p>long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p> <p>このメソッドは、データを保持する機能のみを提供して、実際の描画には影響を与えません。 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。 ROP 指定を行う場合には、SD_SetRopCode を利用します。 この値は、[印面プロパティ]-[描画]-[描画パターン]の[詳細設定]-[カラーパレット]プロパティに設定されてます。</p>	
<p>使用例</p> <pre>long nResult = object.SD_SetDrawBits(1)</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの[印面プロパティ]-[描画]-[描画パターン]の[詳細設定]-[カラーパレット]プロパティを取得 status = stamp.SD_GetDrawBits() Select Case status Case 1 '2 色(1 ビット) '16 色(4 ビット)に設定 res = stamp.SD_SetDrawBits(4) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 4 '16 色(4 ビット) '256 色(8 ビット)に設定 res = stamp.SD_SetDrawBits(8) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 8 '256 色(8 ビット) '2 色(1 ビット)に設定 res = stamp.SD_SetDrawBits(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetPaletteColorFlag	
印面の描画する際のパレットカラーの状態を取得します。	
long SD_GetPaletteColorFlag メソッド(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 1 : パレットにカラーを指定して描画します 0 : パレットにカラーを指定しないで描画します 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetPaletteColorFlag()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データのパレットカラーの状態を取得 status = stamp.SD_GetPaletteColorFlag() Select Case status Case 0 'パレットカラーを指定しない 'パレットカラーを指定するように設定 res = stamp.SD_SetPaletteColorFlag(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 'パレットカラーを指定する 'テキストとして描画に設定 res = stamp.SD_SetPaletteColorFlag(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetPaletteColorFlag	
印面の描画する際のパレットカラーの状態を設定します。	
long SD_SetPaletteColorFlag(long nPaletteColorFlag)	
<p>パラメータ</p> <p>long</p> <p>成功した場合には次のいずれかの値が戻ります。</p> <p>1 : パレットにカラーを指定して描画します</p> <p>0 : パレットにカラーを指定しないで描画します</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS(1)が戻ります。</p> <p>失敗した場合には以下の値が戻ります。</p> <p>DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています</p> <p>E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません</p> <p>詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p> <p>このメソッドは、データを保持する機能のみを提供して、実際の描画には影響を与えません。</p> <p>印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。</p>	
<p>使用例</p> <p>long nResult = object.SD_SetPaletteColorFlag(1)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データのパレットカラーの状態を取得 status = stamp.SD_GetPaletteColorFlag() Select Case status Case 0 'パレットカラーを指定しない 'パレットカラーを指定するように設定 res = stamp.SD_SetPaletteColorFlag(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 'パレットカラーを指定する 'テキストとして描画に設定 res = stamp.SD_SetPaletteColorFlag(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetCommentMarkFlag	
印面の描画する際のコメントや添付ファイルがある場合に印影の右上にアスタリスク（*）マークを描画する状態を取得します。	
long SD_GetCommentMarkFlag メソッド(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 1：コメントや添付ファイルがある場合に印影の右上にアスタリスク（*）マークを描画します 0：設定なし 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessages メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetCommentMarkFlag()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データのコメントフラグ表示設定を取得 status = stamp.SD_GetCommentMarkFlag() Select Case status Case 0 '設定なし '選択中の印鑑データのコメントフラグを表示するように設定 res = stamp.SD_SetCommentMarkFlag(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 '設定あり '選択中の印鑑データのコメントフラグを表示しないように設定 res = stamp.SD_SetCommentMarkFlag(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetCommentMarkFlag	
印面の描画する際のコメントや添付ファイルがある場合に印影の右上にアスタリスク（*）マークを描画する状態を設定します。	
long SD_SetCommentMarkFlag(long nCommentMarkFlag)	
<p>パラメータ</p> <p>long</p> <p>1：コメントや添付ファイルがある場合に印影の右上にアスタリスク（*）マークを描画します</p> <p>0：設定なし</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS(1)が戻ります。</p> <p>失敗した場合には以下の値が戻ります。</p> <p>DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています</p> <p>E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています</p> <p>詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p> <p>このメソッドは、データを保持する機能のみを提供して、実際の描画には影響を与えません。</p> <p>印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。</p>	
<p>使用例</p> <p>long nResult = object.SD_SetCommentMarkFlag(1)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データのコメントフラグ表示設定を取得 status = stamp.SD_GetCommentMarkFlag() Select Case status Case 0 '設定なし '選択中の印鑑データのコメントフラグを表示するように設定 res = stamp.SD_SetCommentMarkFlag(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 '設定あり '選択中の印鑑データのコメントフラグを表示しないように設定 res = stamp.SD_SetCommentMarkFlag(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetCommentMarkFlag	
印影を描画する際のテキスト情報を描画する状態を取得します。	
long SD_GetCommentMarkFlag メソッド(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 1: 印影に描画されるフォント（印面テキストや日付など）をビットマップに変換して描画します 0: 印影に描画されるフォント（印面テキストや日付など）をテキストとして描画します 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または Ge</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetImpressBitmapFontFlag()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データのテキスト描画方法を取得 status = stamp.SD_GetImpressBitmapFontFlag() Select Case status Case 0 'テキストとして描画 'ビットマップに変換して描画に設定 res = stamp.SD_SetImpressBitmapFontFlag(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 'ビットマップに変換して描画 'テキストとして描画に設定 res = stamp.SD_SetImpressBitmapFontFlag(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetImpressBitmapFontFlag	
印影を描画する際のテキスト情報を描画する状態を設定します。	
long SD_SetImpressBitmapFontFlag(long nImpressBitmapFontFlag)	
<p>パラメータ</p> <p>long</p> <p>1: 印影に描画されるフォント（印面テキストや日付など）をビットマップに変換して描画します 0: 印影に描画されるフォント（印面テキストや日付など）をテキストとして描画します</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p> <p>このメソッドは、データを保持する機能のみを提供して、実際の描画には影響を与えません。 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。</p>	
<p>使用例</p> <p>long nResult = object.SD_SetImpressBitmapFontFlag(1)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データのテキスト描画方法を取得 status = stamp.SD_GetImpressBitmapFontFlag() Select Case status Case 0 'テキストとして描画 'ビットマップに変換して描画に設定 res = stamp.SD_SetImpressBitmapFontFlag(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 'ビットマップに変換して描画 'テキストとして描画に設定 res = stamp.SD_SetImpressBitmapFontFlag(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetSaveStampBitsFlag	
印影を描画する際の再描画用ビットマップの状態を取得します。	
long SD_GetSaveStampBitsFlag メソッド(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 1：再描画用の印影ビットマップを付加して描画します 0：設定なし 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetSaveStampBitsFlag()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの再描画用ビットマップの状態を取得 status = stamp.SD_GetSaveStampBitsFlag() Select Case status Case 0 '再描画用ビットマップを付加しない '再描画用ビットマップを付加するように設定 res = stamp.SD_SetSaveStampBitsFlag(1) If res <> 1 Then WScript.Echo stamp.LastErrorNumber() & ":" & stamp.LastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 '再描画用ビットマップを付加する '印影にログを保存しないに設定 res = stamp.SD_SetSaveStampBitsFlag(0) If res <> 1 Then WScript.Echo stamp.LastErrorNumber() & ":" & stamp.LastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.LastErrorNumber() & ":" & stamp.LastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetSaveStampBitsFlag	
印影を描画する際の再描画用ビットマップの状態を設定します。	
long SD_SetSaveStampBitsFlag(long nSaveStampBitsFlag)	
<p>パラメータ</p> <p>long</p> <p>1: 再描画用の印影ビットマップを付加して描画します</p> <p>0: 設定なし</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS(1)が戻ります。</p> <p>失敗した場合には以下の値が戻ります。</p> <p>DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています</p> <p>E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています</p> <p>詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p> <p>このメソッドは、データを保持する機能のみを提供して、実際の描画には影響を与えません。</p> <p>印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。</p>	
<p>使用例</p> <pre>long nResult = object.SD_SetSaveStampBitsFlag(1)</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの再描画用ビットマップの状態を取得 status = stamp.SD_GetSaveStampBitsFlag() Select Case status Case 0 '再描画用ビットマップを付加しない '再描画用ビットマップを付加するように設定 res = stamp.SD_SetSaveStampBitsFlag(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 '再描画用ビットマップを付加する '印影にログを保存しないに設定 res = stamp.SD_SetSaveStampBitsFlag(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetAdditionTextStatus	
印影を描画する際の印面テキストの描画状態を取得します。	
long SD_GetAdditionTextStatus メソッド(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 IS_NONE (0) : 設定なし IS_BLACK (1) : 黒色で描画 IS_COLOR (2) : カラーで描画 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetAdditionTextStatus()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データで捺印したときのテキスト表示の設定を取得 status = stamp.SD_GetAdditionTextStatus() Select Case status Case 0 '表示しない '選択中の印鑑データのテキスト表示を黒に変更 res = stamp.SD_SetAdditionTextStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 '黒 '選択中の印鑑データのテキスト表示をカラーに変更 res = stamp.SD_SetAdditionTextStatus(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 2 'カラー '選択中の印鑑データのテキスト表示をなしに変更 res = stamp.SD_SetAdditionTextStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetAdditionTextStatus	
印影を描画する際の印面テキストの描画状態を設定します。	
long SD_SetAdditionTextStatus(long nStatus)	
<p>パラメータ</p> <p>long</p> <p>IS_NONE (0) : 設定なし</p> <p>IS_BLACK (1) : 黒色で描画</p> <p>IS_COLOR (2) : カラーで描画</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS(1)が戻ります。</p> <p>失敗した場合には以下の値が戻ります。</p> <p>DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています</p> <p>E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています</p> <p>詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p> <p>このメソッドは、データを保持する機能のみを提供して、実際の描画には影響を与えません。</p> <p>印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。</p>	
<p>使用例</p> <p>long nResult = object.SD_SetAdditionTextStatus(2)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データで捺印したときのテキスト表示の設定を取得 status = stamp.SD_GetAdditionTextStatus() Select Case status Case 0 '表示しない '選択中の印鑑データのテキスト表示を黒に変更 res = stamp.SD_SetAdditionTextStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 '黒 '選択中の印鑑データのテキスト表示をカラーに変更 res = stamp.SD_SetAdditionTextStatus(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 2 'カラー '選択中の印鑑データのテキスト表示をなしに変更 res = stamp.SD_SetAdditionTextStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetAdditionText	
印影を描画する際の印面テキストの内容を取得します。	
string SD_GetAdditionText(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には印面テキストの内容が戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例</p> <pre>string strResult = object.SD_GetAdditionText()</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの印面テキストを取得 additonText = stamp.SD_GetAdditionText() "エラーの場合、設定なしの場合ともに"エラーの場合、名なしの場合ともに firstName は空文字なので、 "GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If If additonText = "" Then '選択中の印鑑データの印面テキストを設定 res = stamp.SD_SetAdditionText("確認") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if '印鑑データを更新 res = stamp.SF_PutStamp(0) Else '選択中の印鑑データの印面テキストを設定 res = stamp.SD_SetAdditionText("") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if '印鑑データを更新 res = stamp.SF_PutStamp(0) End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetAdditionText	
印影を描画する際の印面テキストの内容を設定します。	
long SD_SetAdditionText(string strAdditionText)	
<p>パラメータ strAdditionText: 印面テキスト</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_SetAdditionText("印面テキストの内容")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの印面テキストを取得 additionText = stamp.SD_GetAdditionText() 'エラーの場合、設定なしの場合ともに"エラーの場合、名なしの場合ともに firstName は空文字なので、" GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If If additionText = "" Then '選択中の印鑑データの印面テキストを設定 res = stamp.SD_SetAdditionText("確認") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Else '選択中の印鑑データの印面テキストを設定 res = stamp.SD_SetAdditionText("") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetStampBitsStatus	
印影を描画する際の印影色のタイプを取得します。	
long SD_GetStampBitsStatus メソッド(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 IS_BLACK (1) : 黒色で描画 IS_COLOR (2) : カラーで描画 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetStampBitsStatus()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの印影色のタイプを取得 status = stamp.SD_GetStampBitsStatus() Select Case status Case 1 '黒 '選択中の印鑑データの印影色のタイプをカラーに変更 res = stamp.SD_SetStampBitsStatus(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 2 'カラー '選択中の印鑑データの印影色のタイプを黒に変更 res = stamp.SD_SetStampBitsStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetStampBitsStatus	
印影を描画する際の印影色のタイプを設定します。	
long SD_SetStampBitsStatus(long nStatus)	
<p>パラメータ</p> <p>long IS_BLACK (1) : 黒色で描画 IS_COLOR (2) : カラーで描画</p> <p>戻り値</p> <p>long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p> <p>印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。</p>	
<p>使用例</p> <p>long nResult = object.SD_SetStampBitsStatus(2)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの印影色のタイプを取得 status = stamp.SD_GetStampBitsStatus() Select Case status Case 1 '黒 '選択中の印鑑データの印影色のタイプをカラーに変更 res = stamp.SD_SetStampBitsStatus(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 2 'カラー '選択中の印鑑データの印影色のタイプを黒に変更 res = stamp.SD_SetStampBitsStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetStampBitsX	
印影ビットマップの幅をピクセル単位で取得します。取得されるサイズは、印影のみの幅で印面テキストなどの付加情報が追加されている状態のサイズを取得するには、SD_GetItemSizeWidth メソッドを利用します。	
long SD_GetStampBitsX メソッド(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には印影ビットマップ幅の値が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessages メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetStampBitsX()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(4) '選択中の印鑑データの印影の幅をピクセル単位で取得 widthPxcel = stamp.SD_GetStampBitsX() If widthPxcel < 0 Then WScript.Echo stamp.GetLastErrorNumber() & "、" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの印影の幅を(1/1000mm)で取得 width = stamp.SD_GetStampBitsXPerMeter() If width < 0 Then WScript.Echo stamp.GetLastErrorNumber() & "、" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態での幅をピクセル単位で取得 itemWidthPxcel = stamp.SD_GetItemSizeWidth() If itemWidthPxcel < 0 Then WScript.Echo stamp.GetLastErrorNumber() & "、" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態で、実際に捺印される解像度に合わせたときの幅を Himetric 単位(1/100mm)で取得 itemWidthHimetric = stamp.SD_GetItemHimetricSizeWidth() If itemWidthHimetric < 0 Then WScript.Echo stamp.GetLastErrorNumber() & "、" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態での幅をピクセル単位で取得 itemDisplayWidth = stamp.SD_GetItemDisplaySizeWidth() If itemDisplayWidth < 0 Then WScript.Echo stamp.GetLastErrorNumber() & "、" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "印影の幅" & vbCrLf & vbTab & _ widthPxcel & "ピクセル、" & width / 1000 & "mm" & vbCrLf & _ "付加情報を含む幅" & vbCrLf & vbTab & _ itemWidthPxcel & "ピクセル、" & itemWidthHimetric / 100 & "mm" & vbCrLf & vbTab & _ "ディスプレイに表示したとき：" & itemDisplayWidth & "ピクセル" '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetStampBitsY	
印影ビットマップの高さをピクセル単位で取得します。取得されるサイズは、印影のみの高さで印面テキストなどの付加情報が追加されている状態のサイズを取得するには、SD_GetItemSizeHeight メソッドを利用します。	
long SD_GetStampBitsY メソッド(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には印影ビットマップ高さの値が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetStampBitsY()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの印影の高さをピクセル単位で取得 heightPxcel = stamp.SD_GetStampBitsY() If heightPxcel < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの印影の高さを(1/1000mm)で取得 height = stamp.SD_GetStampBitsYPerMeter() If height < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態での高さをピクセル単位で取得 itemHeightPxcel = stamp.SD_GetItemSizeHeight() If itemHeightPxcel < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態で、実際に捺印される解像度に合わせたときの高さを Himetric 単位(1/100mm)で取得 itemHeightHimetric = stamp.SD_GetItemHimetricSizeHeight() If itemHeightHimetric < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態での高さをピクセル単位で取得 itemDisplayHeight = stamp.SD_GetItemDisplaySizeHeight() If itemDisplayHeight < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "印影の高さ" & vbCrLf & vbCrLf & _ heightPxcel & "ピクセル, " & height / 1000 & "mm" & vbCrLf & _ "付加情報を含む高さ" & vbCrLf & vbCrLf & _ itemHeightPxcel & "ピクセル, " & itemHeightHimetric / 100 & "mm" & vbCrLf & vbCrLf & _ "ディスプレイに表示したとき: " & itemDisplayHeight & "ピクセル" '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetStampBitsXPerMeter	
印影ビットマップの幅を 1/1000mm 単位で取得します。	
long SD_GetStampBitsXPerMeter メソッド(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には印影ビットマップ幅の値が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 9 ミリの印鑑の場合には 9000 が取得されるべきですが、余白が考慮されますので実際に取得される値は 11,808 になります。 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetStampBitsXPerMeter()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(4) '選択中の印鑑データの印影の幅をピクセル単位で取得 widthPxcel = stamp.SD_GetStampBitsX() If widthPxcel < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの印影の幅を(1/1000mm)で取得 width = stamp.SD_GetStampBitsXPerMeter() If width < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態での幅をピクセル単位で取得 itemWidthPxcel = stamp.SD_GetItemSizeWidth() If itemWidthPxcel < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態で、実際に捺印される解像度に合わせたときの幅を Himetric 単位(1/100mm)で取得 itemWidthHimetric = stamp.SD_GetItemHimetricSizeWidth() If itemWidthHimetric < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態での幅をピクセル単位で取得 itemDisplayWidth = stamp.SD_GetItemDisplaySizeWidth() If itemDisplayWidth < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "印影の幅" & vbCrLf & vbTab & _ widthPxcel & "ピクセル," & width / 1000 & "mm" & vbCrLf & _ "付加情報を含む幅" & vbCrLf & vbTab & _ itemWidthPxcel & "ピクセル," & itemWidthHimetric / 100 & "mm" & vbCrLf & vbTab & _ "ディスプレイに表示したとき:" & itemDisplayWidth & "ピクセル" '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetStampBitsYPerMeter	
印影ビットマップの高さを 1/1000mm 単位で取得します。	
long SD_GetStampBitsYPerMeter メソッド(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には印影ビットマップ高さの値が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 9 ミリの印鑑の場合には 9000 が取得されるべきですが、余白が考慮されますので実際に取得される値は 11,808 になります。 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetStampBitsYPerMeter()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの印影の高さをピクセル単位で取得 heightPxcel = stamp.SD_GetStampBitsY() If heightPxcel < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中の印鑑データの印影の高さを(1/1000mm)で取得 height = stamp.SD_GetStampBitsYPerMeter() If height < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中の印鑑データの付加情報が追加されている状態での高さをピクセル単位で取得 itemHeightPxcel = stamp.SD_GetItemSizeHeight() If itemHeightPxcel < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中の印鑑データの付加情報が追加されている状態で、実際に捺印される解像度に合わせたときの高さを Himetric 単位(1/100mm)で取得 itemHeightHimetric = stamp.SD_GetItemHimetricSizeHeight() If itemHeightHimetric < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中の印鑑データの付加情報が追加されている状態での高さをピクセル単位で取得 itemDisplayHeight = stamp.SD_GetItemDisplaySizeHeight() If itemDisplayHeight < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '結果表示 WScript.Echo "印影の高さ" & vbCrLf & vbCrLf & _ heightPxcel & "ピクセル, " & height / 1000 & "mm" & vbCrLf & _ "付加情報を含む高さ" & vbCrLf & vbCrLf & _ itemHeightPxcel & "ピクセル, " & itemHeightHimetric / 100 & "mm" & vbCrLf & _ "ディスプレイに表示したとき: " & itemDisplayHeight & "ピクセル" '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetItemSizeWidth	
印面テキストなどの付加情報が追加されている状態の幅をピクセル単位で取得します。印影のみの幅でサイズを取得するには、SD_GetStampBitsX メソッドを利用します。	
long SD_GetItemSizeWidth メソッド(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には既定値で描画される幅の値が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetItemSizeWidth()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(4) '選択中の印鑑データの印影の幅をピクセル単位で取得 widthPxcel = stamp.SD_GetStampBitsX() If widthPxcel < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの印影の幅を(1/1000mm)で取得 width = stamp.SD_GetStampBitsXPerMeter() If width < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態での幅をピクセル単位で取得 itemWidthPxcel = stamp.SD_GetItemSizeWidth() If itemWidthPxcel < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態で、実際に捺印される解像度に合わせたときの幅を Himetric 単位(1/100mm)で取得 itemWidthHimetric = stamp.SD_GetItemHimetricSizeWidth() If itemWidthHimetric < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態での幅をピクセル単位で取得 itemDisplayWidth = stamp.SD_GetItemDisplaySizeWidth() If itemDisplayWidth < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "印影の幅" & vbCrLf & vbTab & _ widthPxcel & "ピクセル, " & width / 1000 & "mm" & vbCrLf & _ "付加情報を含む幅" & vbCrLf & vbTab & _ itemWidthPxcel & "ピクセル, " & itemWidthHimetric / 100 & "mm" & vbCrLf & vbTab & _ "ディスプレイに表示したとき: " & itemDisplayWidth & "ピクセル" '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetItemSizeHeight	
印面テキストなどの付加情報が追加されている状態の高さをピクセル単位で取得します。印影のみの高さでサイズを取得するには、SD_GetStampBitsY メソッドを利用します。	
long SD_GetItemSizeHeight メソッド(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には既定値で描画される高さの値が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessages メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetItemSizeHeight()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの印影の高さをピクセル単位で取得 heightPixel = stamp.SD_GetStampBitsY() If heightPixel < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの印影の高さを(1/1000mm)で取得 height = stamp.SD_GetStampBitsYPerMeter() If height < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態の高さをピクセル単位で取得 itemHeightPixel = stamp.SD_GetItemSizeHeight() If itemHeightPixel < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態で、実際に捺印される解像度に合わせたときの高さを Himetric 単位(1/100mm)で取得 itemHeightHimetric = stamp.SD_GetItemHimetricSizeHeight() If itemHeightHimetric < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態の高さをピクセル単位で取得 itemDisplayHeight = stamp.SD_GetItemDisplaySizeHeight() If itemDisplayHeight < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "印影の高さ" & vbCrLf & vbTab & _ heightPixel & "ピクセル, " & height / 1000 & "mm" & vbCrLf & _ "付加情報を含む高さ" & vbCrLf & vbTab & _ itemHeightPixel & "ピクセル, " & itemHeightHimetric / 100 & "mm" & vbCrLf & vbTab & _ "ディスプレイに表示したとき: " & itemDisplayHeight & "ピクセル" '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetItemDisplaySizeWidth	
印面テキストなどの付加情報が追加されている状態の幅をデバイス単位値で取得します。印影のみの幅でサイズを取得するには、SD_GetStampBitsX メソッドを利用します。	
long SD_GetItemDisplaySizeWidth メソッド(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には実際に画面に描画される幅の値が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetItemDisplaySizeWidth()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(4) '選択中の印鑑データの印影の幅をピクセル単位で取得 widthPxcel = stamp.SD_GetStampBitsX() If widthPxcel < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中の印鑑データの印影の幅を(1/1000mm)で取得 width = stamp.SD_GetStampBitsXPerMeter() If width < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中の印鑑データの付加情報が追加されている状態での幅をピクセル単位で取得 itemWidthPxcel = stamp.SD_GetItemSizeWidth() If itemWidthPxcel < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中の印鑑データの付加情報が追加されている状態で、実際に捺印される解像度に合わせたときの幅を Himetric 単位(1/100mm)で取得 itemWidthHimetric = stamp.SD_GetItemHimetricSizeWidth If itemWidthHimetric < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中の印鑑データの付加情報が追加されている状態での幅をピクセル単位で取得 itemDisplayWidth = stamp.SD_GetItemDisplaySizeWidth() If itemDisplayWidth < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '結果表示 WScript.Echo "印影の幅" & vbCrLf & vbTab & _ widthPxcel & "ピクセル, " & width / 1000 & "mm" & vbCrLf & _ "付加情報を含む幅" & vbCrLf & vbTab & _ itemWidthPxcel & "ピクセル, " & itemWidthHimetric / 100 & "mm" & vbCrLf & vbTab & _ "ディスプレイに表示したとき: " & itemDisplayWidth & "ピクセル" '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetItemDisplaySizeHeight	
印面テキストなどの付加情報が追加されている状態の高さをデバイス単位で取得します。印影のみの高さでサイズを取得するには、SD_GetStampBitsX メソッドを利用します。	
long SD_GetItemDisplaySizeHeight メソッド(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には実際に画面に描画される高さの値が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetItemDisplaySizeHeight()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの印影の高さをピクセル単位で取得 heightPxcel = stamp.SD_GetStampBitsY() If heightPxcel < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorErrorMessage() End If '選択中の印鑑データの印影の高さを(1/1000mm)で取得 height = stamp.SD_GetStampBitsYPerMeter() If height < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態の高さをピクセル単位で取得 itemHeightPxcel = stamp.SD_GetItemSizeHeight() If itemHeightPxcel < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態で、実際に捺印される解像度に合わせたときの高さを Himetric 単位(1/100mm)で取得 itemHeightHimetric = stamp.SD_GetItemHimetricSizeHeight() If itemHeightHimetric < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態の高さをピクセル単位で取得 itemDisplayHeight = stamp.SD_GetItemDisplaySizeHeight() If itemDisplayHeight < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorErrorMessage() End If '結果表示 WScript.Echo "印影の高さ" & vbCrLf & vbCrLf & _ heightPxcel & "ピクセル, " & height / 1000 & "mm" & vbCrLf & _ "付加情報を含む高さ" & vbCrLf & vbCrLf & _ itemHeightPxcel & "ピクセル, " & itemHeightHimetric / 100 & "mm" & vbCrLf & _ "ディスプレイに表示したとき: " & itemDisplayHeight & "ピクセル" '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetItemHimetricSizeWidth	
実際に捺印ドキュメントの解像度に合わせた印面テキストなどの付加情報が追加されている状態の幅を Himetric 単位 (1/100mm) で取得します。印影のみの幅でサイズを取得するには、SD_GetStampBitsX メソッドを利用します。	
long SD_GetItemHimetricSizeWidth メソッド(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には実際に捺印ドキュメントの解像度に合わせた描画される幅の値が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetItemHimetricSizeWidth()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(4) '選択中の印鑑データの印影の幅をピクセル単位で取得 widthPxcel = stamp.SD_GetStampBitsX() If widthPxcel < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessaqe() End If '選択中の印鑑データの印影の幅を(1/1000mm)で取得 width = stamp.SD_GetStampBitsXPerMeter() If width < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessaqe() End If '選択中の印鑑データの付加情報が追加されている状態での幅をピクセル単位で取得 itemWidthPxcel = stamp.SD_GetItemSizeWidth() If itemWidthPxcel < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessaqe() End If '選択中の印鑑データの付加情報が追加されている状態で、実際に捺印される解像度に合わせたときの幅を Himetric 単位(1/100mm)で取得 itemWidthHimetric = stamp.SD_GetItemHimetricSizeWidth If itemWidthHimetric < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessaqe() End If '選択中の印鑑データの付加情報が追加されている状態での幅をピクセル単位で取得 itemDisplayWidth = stamp.SD_GetItemDisplaySizeWidth() If itemDisplayWidth < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessaqe() End If '結果表示 WScript.Echo "印影の幅" & vbCrLf & vbTab & _ widthPxcel & "ピクセル, " & width / 1000 & "mm" & vbCrLf & _ "付加情報を含む幅" & vbCrLf & vbTab & _ itemWidthPxcel & "ピクセル, " & itemWidthHimetric / 100 & "mm" & vbCrLf & vbTab & _ "ディスプレイに表示したとき: " & itemDisplayWidth & "ピクセル" '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetItemHimetricSizeHeight	
実際に捺印ドキュメントの解像度に合わせた印面テキストなどの付加情報が追加されている状態の高さを Himetric 単位 (1/100mm) で取得します。印影のみの高さでサイズを取得するには、SD_GetStampBitsY メソッドを利用します。	
long SD_GetItemHimetricSizeHeight メソッド(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には実際に捺印ドキュメントの解像度に合わせた描画される高さの値が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessages メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetItemHimetricSizeHeight()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの印影の高さをピクセル単位で取得 heightPixel = stamp.SD_GetStampBitsY() If heightPixel < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの印影の高さを(1/1000mm)で取得 height = stamp.SD_GetStampBitsYPerMeter() If height < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態の高さをピクセル単位で取得 itemHeightPixel = stamp.SD_GetItemSizeHeight() If itemHeightPixel < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態で、実際に捺印される解像度に合わせたときの高さを Himetric 単位(1/100mm)で取得 itemHeightHimetric = stamp.SD_GetItemHimetricSizeHeight() If itemHeightHimetric < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの付加情報が追加されている状態の高さをピクセル単位で取得 itemDisplayHeight = stamp.SD_GetItemDisplaySizeHeight() If itemDisplayHeight < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "印影の高さ" & vbCrLf & vbTab & _ heightPixel & "ピクセル, " & height / 1000 & "mm" & vbCrLf & _ "付加情報を含む高さ" & vbCrLf & vbTab & _ itemHeightPixel & "ピクセル, " & itemHeightHimetric / 100 & "mm" & vbCrLf & vbTab & _ "ディスプレイに表示したとき: " & itemDisplayHeight & "ピクセル" '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetAdditionDateStatus	
印影を描画する際の印面日付の描画状態を取得します。	
long SD_GetAdditionDateStatus メソッド(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 IS_NONE (0) : 設定なし IS_BLACK (1) : 黒色で描画 IS_COLOR (2) : カラーで描画 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetAdditionDateStatus()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データで捺印したときの日時表示の設定を取得 status = stamp.SD_GetAdditionDateStatus() Select Case status Case 0 '表示なし '選択中の印鑑データで捺印したときの日時表示を黒に設定 res = stamp.SD_SetAdditionDateStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 '黒表示 '選択中の印鑑データで捺印したときの日時表示をカラーに設定 res = stamp.SD_SetAdditionDateStatus(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 2 'カラー表示 '選択中の印鑑データで捺印したときの日時表示をなしに設定 res = stamp.SD_SetAdditionDateStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetAdditionDateStatus	
印影を描画する際の印面日時の描画状態を設定します。	
long SD_SetAdditionDateStatus(long nStatus)	
<p>パラメータ nStatus IS_NONE (0) : 設定なし IS_BLACK (1) : 黒色で描画 IS_COLOR (2) : カラーで描画</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。 日付印の場合には、IS_NONE (0) を設定することはできません。また、SD_GetStampBitsStatus で取得される印影色と異なる設定の場合には描画時に SD_GetStampBitsStatus の状態が優先されます。</p>	
<p>使用例 long nResult = object.SD_SetAdditionTextStatus(2)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データで捺印したときの日時表示の設定を取得 status = stamp.SD_GetAdditionDateStatus() Select Case status Case 0 '表示なし '選択中の印鑑データで捺印したときの日時表示を黒に設定 res = stamp.SD_SetAdditionDateStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 '黒表示 '選択中の印鑑データで捺印したときの日時表示をカラーに設定 res = stamp.SD_SetAdditionDateStatus(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 2 'カラー表示 '選択中の印鑑データで捺印したときの日時表示をなしに設定 res = stamp.SD_SetAdditionDateStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetAdditionDateFormat	
印影を描画する際の印面日付の書式を取得します。	
long SD_GetAdditionDateFormat メソッド(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 氏名印の場合 1 : 'yy.mm.dd 2 : 'yy/mm/dd 3 : ee.mm.dd 4 : ee/mm/dd 5 : gee.mm.dd 6 : gee/mm/dd 7 : 'yy/mm/dd hh:mm:ss 8 : hh:mm:ss 9 : yyyy.mm.dd 10 : yyyy/mm/dd 日付印の場合 1 : 'yy.mm.dd 2 : 'yy/mm/dd 3 : ee.mm.dd 4 : ee/mm/dd 5 : gee.mm.dd 6 : gee/mm/dd 7 : yyyy.mm.dd 8</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetAdditionDateFormat()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データで捺印したときに表示する日時の書式を取得 format = stamp.SD_GetAdditionDateFormat() If stampType < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの種別を取得 stampType = stamp.SD_GetStampType() If stampType < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If Select Case stampType Case 1 '氏名印 '選べる書式は 10 種 (1 ~10) nextFormat = (format Mod 10) + 1 Case 2 '日付印 '選べる書式は 8 種 (1 ~8) nextFormat = (format Mod 8) + 1 Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '選択中の印鑑データで捺印したときに表示する日時の書式を設定 res = stamp.SD_SetAdditionDateFormat(nextFormat) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetAdditionDateFormat	
印影を描画する際の印面日付の書式を取得します。書式は、SD_GetStampType メソッドで取得される印鑑種別によって異なります。	
long SD_SetAdditionDateFormat メソッド(long nFormat)	
<p>パラメータ</p> <p>nFormat</p> <p>氏名印の場合</p> <p>1 : 'yy.mm.dd 2 : 'yy/mm/dd 3 : ee.mm.dd 4 : ee/mm/dd</p> <p>5 : gee.mm.dd 6 : gee/mm/dd 7 : 'yy/mm/dd hh:mm:ss 8 : hh:mm:ss</p> <p>9 : yyyy.mm.dd 10 : yyyy/mm/dd</p> <p>日付印の場合</p> <p>1 : 'yy.mm.dd 2 : 'yy/mm/dd 3 : ee.mm.dd 4 : ee/mm/dd</p> <p>5 : gee.mm.dd 6 : gee/mm/dd 7 : yyyy.mm.dd 8 : yyyy/mm/dd</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS(1)が戻ります。</p> <p>失敗した場合には以下の値が戻ります。</p> <p>DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています</p> <p>E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています</p> <p>詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p> <p>印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例</p> <pre>long nResult = object.SD_SetAdditionDateFormat(1)</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データで捺印したときに表示する日時書式を取得 format = stamp.SD_GetAdditionDateFormat() If stampType < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中の印鑑データの種別を取得 stampType = stamp.SD_GetStampType() If stampType < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If Select Case stampType Case 1 '氏名印 '選べる書式は 10 種 (1 ~10) nextFormat = (format Mod 10) + 1 Case 2 '日付印 '選べる書式は 8 種 (1 ~8) nextFormat = (format Mod 8) + 1 Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End Select '選択中の印鑑データで捺印したときに表示する日時書式を設定 res = stamp.SD_SetAdditionDateFormat(nextFormat) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetImpressCountStatus	
印影を描画する際の捺印カウンタの描画状態を取得します。	
long SD_GetImpressCountStatus メソッド(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 IS_NONE (0) : 設定なし IS_BLACK (1) : 黒色で描画 IS_COLOR (2) : カラーで描画 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessages メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetImpressCountStatus()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データで捺印したときの捺印カウンタ表示の設定を取得 status = stamp.SD_GetImpressCountStatus() Select Case status Case 0 '設定なし (表示しない) '選択中の印鑑データで捺印したときの捺印カウンタ表示を黒に設定 res = stamp.SD_SetImpressCountStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 '黒で表示 '選択中の印鑑データで捺印したときの捺印カウンタ表示をカラーに設定 res = stamp.SD_SetImpressCountStatus(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 2 'カラーで表示 '選択中の印鑑データで捺印したときの捺印カウンタ表示をなしに設定 res = stamp.SD_SetImpressCountStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetImpressCountStatus	
印影を描画する際の捺印カウンタの描画状態を設定します。	
long SD_SetImpressCountStatus(long nStatus)	
<p>パラメータ</p> <p>nStatus IS_NONE (0) : 設定なし IS_BLACK (1) : 黒色で描画 IS_COLOR (2) : カラーで描画</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。</p>	
<p>使用例</p> <pre>long nResult = object.SD_SetImpressCountStatus(2)</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データで捺印したときの捺印カウンタ表示の設定を取得 status = stamp.SD_GetImpressCountStatus() Select Case status Case 0 '設定なし (表示しない) '選択中の印鑑データで捺印したときの捺印カウンタ表示を黒に設定 res = stamp.SD_SetImpressCountStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 '黒で表示 '選択中の印鑑データで捺印したときの捺印カウンタ表示をカラーに設定 res = stamp.SD_SetImpressCountStatus(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 2 'カラーで表示 '選択中の印鑑データで捺印したときの捺印カウンタ表示をなしに設定 res = stamp.SD_SetImpressCountStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetImpressCount	
現在の捺印カウンタを取得します。	
long SD_GetImpressCount メソッド(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には捺印カウンタ（正の整数）値が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessages メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetImpressCount()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの捺印カウンタの値を取得 count = stamp.SD_GetImpressCount() If count < 0 Then WScript.Echo stamp.GetLastErrorNumber() & "." & stamp.GetLastErrorMessage() End If If count > 99 Then '選択中の印鑑データの捺印カウンタの値を設定 res = stamp.SD_SetImpressCount(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & "." & stamp.GetLastErrorMessage() Else '印鑑データを更新 res = stamp.SF_PutStamp(0) End If End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetImpressCount	
捺印カウンタを設定します。	
long SD_SetImpressCount(long nCount)	
<p>パラメータ nCount:捺印カウンタ</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_SetImpressCount(10)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの捺印カウンタの値を取得 count = stamp.SD_GetImpressCount() If count < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If If count > 99 Then '選択中の印鑑データの捺印カウンタの値を設定 res = stamp.SD_SetImpressCount(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() Else '印鑑データを更新 res = stamp.SF_PutStamp(0) End If End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetLogStatus	
印鑑データに設定されている[ログ収集あり]項目を取得します。	
long SD_GetLogStatus(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 1 : ログ収集あり 0 : ログ収集なし 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetLogStatus()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの[ログ収集あり]の設定を取得 logFlag = stamp.SD_GetLogStatus() Select Case logFlag Case 0 '収集なし 'ログ収集あり]を有効にする res = stamp.SD_SetLogStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 '収集あり 'ログ収集あり]を無効にする res = stamp.SD_SetLogStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetLogStatus	
印鑑データに設定されている[ログ収集あり]項目を設定します。	
long SD_SetLogStatus(long nStatus)	
<p>パラメータ</p> <p>nStatus 1 : ログ収集あり 0 : ログ収集なし</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessages メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。</p>	
<p>使用例</p> <pre>long nResult = object.SD_SetLogStatus(1)</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの[ログ収集あり]の設定を取得 logFlag = stamp.SD_GetLogStatus() Select Case logFlag Case 0 '収集なし 'ログ収集あり]を有効にする res = stamp.SD_SetLogStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 '収集あり 'ログ収集あり]を無効にする res = stamp.SD_SetLogStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetBackDateFlag	
印鑑データに設定されている[バックデート可]項目を取得します。	
long SD_GetBackDateFlag(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 1 : 本日より以前に日付を戻せる 0 : 本日より以前に日付を戻せない 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetBackDateFlag()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの[バックデート可]の設定を取得 backDateFlag = stamp.SD_GetBackDateFlag() Select Case backDateFlag Case 0 'バックデート不可 '[バックデート可]を有効にする res = stamp.SD_SetBackDateFlag(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 'バックデート可 '[バックデート可]を無効にする res = stamp.SD_SetBackDateFlag(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetBackDateFlag	
印鑑データに設定されている[バックデート可]項目を設定します。	
long SD_SetBackDateFlag(long nFlag)	
<p>パラメータ nFlag 1 : 本日より以前に日付を戻せる 0 : 本日より以前に日付を戻せない</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_SetBackDateFlag(1)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの[バックデート可]の設定を取得 backDateFlag = stamp.SD_GetBackDateFlag() Select Case backDateFlag Case 0 'バックデート不可 '[バックデート可]を有効にする res = stamp.SD_SetBackDateFlag(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 'バックデート可 '[バックデート可]を無効にする res = stamp.SD_SetBackDateFlag(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetSaveLogFlag	
捺印ログに保存する情報を印影に保存する状態を取得します。	
long SD_GetSaveLogFlag(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 1：捺印ログと印鑑オブジェクトに監査情報を保存する 0：捺印ログと印鑑オブジェクトに監査情報を保存しない 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessages メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetSaveLogFlag()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの印影にログを保存する設定を取得 status = stamp.SD_GetSaveLogFlag() Select Case status Case 0 '印影にログを保存しない '印影にログを保存するように設定 res = stamp.SD_SetSaveLogFlag(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 '印影にログを保存する '印影にログを保存しないように設定 res = stamp.SD_SetSaveLogFlag(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetSaveLogFlag	
捺印ログに保存する情報を印影に保存する状態を設定します。	
long SD_SetSaveLogFlag(long nFlag)	
<p>パラメータ</p> <p>nFlag</p> <p>1 : 印影オブジェクトと捺印ログに監査情報を保存する</p> <p>0 : 印鑑オブジェクトと捺印ログに監査情報を保存しない</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS(1)が戻ります。</p> <p>失敗した場合には以下の値が戻ります。</p> <p>DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています</p> <p>E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています</p> <p>詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p> <p>印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。</p>	
<p>使用例</p> <p>long nResult = object.SD_SetSaveLogFlag(1)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの印影にログを保存する設定を取得 status = stamp.SD_GetSaveLogFlag() Select Case status Case 0 '印影にログを保存しない '印影にログを保存するように設定 res = stamp.SD_SetSaveLogFlag(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 '印影にログを保存する '印影にログを保存しないように設定 res = stamp.SD_SetSaveLogFlag(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetDocImpressStatus	
印鑑データの[文書情報入力促進]プロパティを取得します。	
long SD_GetDocImpressStatus(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合にはつぎのいずれかの値が戻ります。 0：文書情報なしでも捺印可能 1：文書情報なしの場合には確認後に捺印可能 2：文書情報なしでは捺印不可 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetDocImpressStatus()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの[文書情報入力促進]プロパティを取得 status = stamp.SD_GetDocImpressStatus() Select Case status Case 0 '設定なし '[文書情報入力促進]プロパティを[警告]に設定 res = stamp.SD_SetDocImpressStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 '警告 '[文書情報入力促進]プロパティを[強制]に設定 res = stamp.SD_SetDocImpressStatus(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 2 '強制 '[文書情報入力促進]プロパティを[なし]に設定 res = stamp.SD_SetDocImpressStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetDocImpressStatus	
印鑑データの[文書情報入力促進]プロパティを設定します。	
long SD_SetDocImpressStatus(long nStatus)	
<p>パラメータ</p> <p>nStatus</p> <p>0 : 文書情報なしでも捺印可能</p> <p>1 : 文書情報なしの場合には確認後に捺印可能</p> <p>2 : 文書情報なしでは捺印不可</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS(1)が戻ります。</p> <p>失敗した場合には以下の値が戻ります。</p> <p>DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています</p> <p>E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています</p> <p>詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p> <p>印鑑データの選択を行うには SF_GetStamp メソッドを利用します。設定した内容を保存するには、SF_PutStamp メソッドを利用します。</p>	
<p>使用例</p> <p>long nResult = object.SD_SetDocImpressStatus(1)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの[文書情報入力促進]プロパティを取得 status = stamp.SD_SetDocImpressStatus() Select Case status Case 0 '設定なし '[文書情報入力促進]プロパティを[警告]に設定 res = stamp.SD_SetDocImpressStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 1 '警告 '[文書情報入力促進]プロパティを[強制]に設定 res = stamp.SD_SetDocImpressStatus(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case 2 '強制 '[文書情報入力促進]プロパティを[なし]に設定 res = stamp.SD_SetDocImpressStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GF_DeleteGroup	
選択されているグループを削除します。	
long GF_DeleteGroup(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 削除対象のグループにユーザが追加されている場合にはこのメソッドは失敗します。また、このメソッドは実行後に先頭グループが選択されます。</p>	
<p>使用例 long nResult = object.GF_DeleteGroup()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループを選択 res = stamp.GF_SeekGroupByUserName("test") '選択しているグループを削除 res = stamp.GF_DeleteGroup() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GF_RenewGroupName	
選択されているグループの名前を変更します。	
long GF_RenewGroupName(string strNewGroupName)	
<p>パラメータ strNewGroupName</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 変更するグループの名前は、同一の親グループを持つグループ内では重複は許可されませんが、異なる親グループでは設定を行うことが可能です。</p>	
<p>使用例 long nResult = object.GF_RenewGroupName("変更後のグループ名")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループを選択 res = stamp.GF_SeekGroupByUserName("<検索するユーザ名>") '選択しているグループ名を変更 res = stamp.GF_RenewGroupName(stamp.GF_GetCurrentGroupName & "1") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '結果表示 WScript.Echo "変更後のグループ名" & ":" & stamp.GF_GetCurrentGroupName() '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GF_Move	
引数で指定された位置のグループを選択します。	
long GF_Move(long nPosition)	
<p>パラメータ nPosition: グループの位置</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 引数で指定するグループの位置は、GC_GetPosition メソッド、GF_GetCurrentPosition メソッドや SF_GetParentGroup メソッドなどで取得された値である必要があります。</p>	
<p>使用例 long nResult = object.GF_Move(グループの位置)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択されているユーザが追加されているグループの位置を取得 groupPosition = stamp.SF_GetParentGroup() If groupPosition < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザが追加されているグループを選択 res = stamp.GF_Move(groupPosition) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザが追加されているグループのグループ名を取得 groupName = stamp.GF_GetCurrentGroupName() 'エラーの場合設定なしの場合ともに groupName は空文字なので、 'GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザのポジション値を取得 userPosition = stamp.SF_GetCurrentPosition() If userPosition < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザに登録されている印鑑数を取得 stampCount = stamp.SF_GetCurrentStampCount() If stampCount < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "ユーザ名:" & stamp.SF_GetCurrentUserName & vbCrLf &_ "グループポジション値:" & groupPosition & vbCrLf &_ "グループポジション名:" & groupName & vbCrLf &_ "ユーザポジション値:" & userPosition & vbCrLf &_ "ユーザに登録されている印鑑数:" & stampCount '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GF_MoveRoot	
ルートグループ（捺印用印鑑データファイル内のグループ階層の最上部）を選択します。	
long GF_MoveRoot(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.GF_MoveRoot()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループを選択 res = stamp.GF_SeekGroupByUserName("<検索するユーザ名>") '選択しているグループの親グループを選択 res = stamp.GF_MoveParent() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "親グループ:" & stamp.GF_GetCurrentGroupName() 'ルートグループを選択 res = stamp.GF_MoveRoot() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "選択中のグループ:" & stamp.GF_GetCurrentGroupName() '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GF_MoveParent	
選択されているグループの親グループ（一つ上の階層）を選択します。	
long GF_MoveParent(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.GF_MoveParent()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループを選択 res = stamp.GF_SeekGroupByUserName("<検索するユーザ名>") '選択しているグループの親グループを選択 res = stamp.GF_MoveParent() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '結果表示 WScript.Echo "親グループ:" & stamp.GF_GetCurrentGroupName() 'ルートグループを選択 res = stamp.GF_MoveRoot() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '結果表示 WScript.Echo "選択中のグループ:" & stamp.GF_GetCurrentGroupName() '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GF_MoveFirst	
引数で指定されたグループを親に持つグループ内で、先頭のグループに移動します。	
long GF_MoveFirst(long nPosition)	
<p>パラメータ nPosition:親グループの位置</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessages メソッドを使用します。</p> <p>備考 引数で指定するグループの位置は、GC_GetPosition メソッド、GF_GetCurrentPosition メソッドや SF_GetParentGroup メソッドなどで取得された値である必要があります。</p>	
<p>使用例 long nResult = object.GF_MoveFirst(0)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く(ルートグループが選択されている) res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループの位置を取得 res = stamp.GF_MoveRoot() groupPosition = stamp.GF_GetCurrentPosition() '指定したグループ以下にあるグループの最初のグループを選択 res = stamp.GF_MoveFirst(groupPosition) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のグループが最後のグループであるか judge = stamp.GF_IsEOF() If judge < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If i = 1 msg = "" While judge = 0 '最後のグループでない '選択中のグループのグループ名を取得 msg = msg & i & ":" & stamp.GF_GetCurrentGroupname() & vbCrLf '指定したグループ以下にあるグループ内で、選択中のグループの次のグループを選択 res = stamp.GF_MoveNext(groupPosition) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If judge = stamp.GF_IsEOF() i = i + 1 Wend '結果表示 WScript.Echo msg '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GF_MoveLast	
引数で指定されたグループを親に持つグループ内で、末尾のグループに移動します。	
long GF_MoveLast(long nPosition)	
<p>パラメータ nPosition:親グループの位置</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessages メソッドを使用します。</p> <p>備考 引数で指定するグループの位置は、GC_GetPosition メソッド、GF_GetCurrentPosition メソッドや SF_GetParentGroup メソッドなどで取得された値である必要があります。</p>	
<p>使用例 long nResult = object.GF_MoveLast(0)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループの位置を取得 res = stamp.GF_MoveRoot() groupPosition = stamp.GF_GetCurrentPosition() '指定したグループ以下にあるグループの最後のグループを選択 res = stamp.GF_MoveLast(groupPosition) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のグループが最初のグループであるか judge = stamp.GF_IsBOF() If judge < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If i = 1 msg = "" While judge = 0 '最初のグループでない '選択中のグループのグループ名を取得 msg = msg & i & ":" & stamp.GF_GetCurrentGroupname() & vbCrLf 'ルートグループ以下で、選択中のグループの前のグループを選択 res = stamp.GF_MovePrevious(groupPosition) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If judge = stamp.GF_IsBOF() i = i + 1 Wend '結果表示 WScript.Echo msg '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GF_MoveNext	
引数で指定されたグループを親に持つグループ内で、次のグループに移動します。	
long GF_MoveNext(long nPosition)	
<p>パラメータ nPosition:親グループの位置</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 引数で指定するグループの位置は、GC_GetPosition メソッド、GF_GetCurrentPosition メソッドや SF_GetParentGroup メソッドなどで取得された値である必要があります。</p>	
<p>使用例 long nResult = object.GF_MoveNext(0)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く(ルートグループが選択されている) res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループの位置を取得 res = stamp.GF_MoveRoot() groupPosition = stamp.GF_GetCurrentPosition() '指定したグループ以下にあるグループの最初のグループを選択 res = stamp.GF_MoveFirst(groupPosition) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '選択中のグループが最後のグループであるか judge = stamp.GF_IsEOF() If judge < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If i = 1 msg = "" While judge = 0 '最後のグループでない '選択中のグループのグループ名を取得 msg = msg & i & ":" & stamp.GF_GetCurrentGroupname() & vbCrLf '指定したグループ以下にあるグループ内で、選択中のグループの次のグループを選択 res = stamp.GF_MoveNext(groupPosition) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If judge = stamp.GF_IsEOF() i = i + 1 Wend '結果表示 WScript.Echo msg '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GF_MovePrevious	
引数で指定されたグループを親に持つグループ内で、前のグループに移動します。	
long GF_MovePrevious(long nPosition)	
<p>パラメータ nPosition:親グループの位置</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 引数で指定するグループの位置は、GC_GetPosition メソッド、GF_GetCurrentPosition メソッドや SF_GetParentGroup メソッドなどで取得された値である必要があります。</p>	
<p>使用例 long nResult = object.GF_MovePrevious(0)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループの位置を取得 res = stamp.GF_MoveRoot() groupPosition = stamp.GF_GetCurrentPosition() '指定したグループ以下にあるグループの最後のグループを選択 res = stamp.GF_MoveLast(groupPosition) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '選択中のグループが最初のグループであるか judge = stamp.GF_IsBOF() If judge < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If i = 1 msg = "" While judge = 0 '最初のグループでない '選択中のグループのグループ名を取得 msg = msg & i & ":" & stamp.GF_GetCurrentGroupname() & vbCrLf 'ルートグループ以下で、選択中のグループの前のグループを選択 res = stamp.GF_MovePrevious(groupPosition) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If judge = stamp.GF_IsBOF() i = i + 1 Wend '結果表示 WScript.Echo msg '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GF_GetCurrentPosition	
選択されているグループの位置を取得します。	
long GF_GetCurrentPosition(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には選択されているグループの位置が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 取得されたグループの位置は、GF_Move メソッドなどで利用します。</p>	
<p>使用例 long nResult = object.GF_GetCurrentPosition()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループを選択 res = stamp.GF_SeekGroupByUserName("<検索するユーザ名>") '選択中のグループのグループ名を取得 1 nameGF = stamp.GF_GetCurrentGroupName() If nameGF = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中のグループのグループ名を取得 2 nameGC = stamp.GC_GetGroupName() If nameGC = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中のグループの位置を取得 1 positionGF = stamp.GF_GetCurrentPosition() If positionGF < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中のグループの位置を取得 2 positionGC = stamp.GC_GetPosition() If positionGC < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中のグループのグループパスを取得 path = stamp.GF_GetCurrentGroupNamePath() If path = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択されているグループの CID(10 進)を設定→結果表示は 16 進 groupID = stamp.GC_GetGroupID() "CID は符号付き Long 型なので、CID の取得に成功してもその値が負となることがある "そのため GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '結果表示 WScript.Echo "グループ名:" & vbCrLf &_ " GF_GetCurrentGroupName:" & nameGF & vbCrLf &_ " GC_GetGroupName:" & nameGC & vbCrLf &_ "位置:" & vbCrLf &_ " GF_GetCurrentPosition:" & positionGF & vbCrLf &_ " GC_GetPosition:" & positionGC & vbCrLf &_ "グループパス : " & path & vbCrLf &_ "グループの CID:" & Hex(groupID) '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GF_GetCurrentGroupName	
選択されているグループの名前を取得します。	
string GF_GetCurrentGroupName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には選択されているグループの名前（または、Active Directory の組織単位名）が戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 GF_MoveRoot メソッドなどで最上位のグループに移動した場合にこのメソッドは、既定値"root"を戻します。管理ツールなどで設定された最上位のグループ名は SF_GetRootName メソッドを利用します。</p>	
<p>使用例 string strResult = object.GF_GetCurrentGroupName()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループを選択 res = stamp.GF_SeekGroupByUserName("<検索するユーザ名>") '選択中のグループのグループ名を取得 1 nameGF = stamp.GF_GetCurrentGroupName() If nameGF = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のグループのグループ名を取得 2 nameGC = stamp.GC_GetGroupName() If nameGC = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のグループの位置を取得 1 positionGF = stamp.GF_GetCurrentPosition() If positionGF < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のグループの位置を取得 2 positionGC = stamp.GC_GetPosition() If positionGC < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のグループのグループパスを取得 path = stamp.GF_GetCurrentGroupNamePath() If path = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているグループの CID(10 進)を設定→結果表示は 16 進 groupID = stamp.GC_GetGroupID() "CID は符号付き Long 型なので、CID の取得に成功してもその値が負となることがある "そのため GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "グループ名:" & vbCrLf & _ " GF_GetCurrentGroupName:" & nameGF & vbCrLf & _ " GC_GetGroupName:" & nameGC & vbCrLf & _ "位置:" & vbCrLf & _ " GF_GetCurrentPosition:" & positionGF & vbCrLf & _ " GC_GetPosition:" & positionGC & vbCrLf & _ "グループパス : " & path & vbCrLf & _ "グループの CID:" & Hex(groupID) '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GF_GetCurrentGroupNamePath	
選択されているグループのグループパス（ルートグループからのグループ名を¥マークで結合した文字列）を取得します。	
string GF_GetCurrentGroupNamePath(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には選択されているグループのグループパス（または、Active Directory の組織単位名のパス）が戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <pre>string strResult = object.GF_GetCurrentGroupNamePath()</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループを選択 res = stamp.GF_SeekGroupByUserName("<検索するユーザ名>") '選択中のグループのグループ名を取得 1 nameGF = stamp.GF_GetCurrentGroupName() If nameGF = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessagel() End If '選択中のグループのグループ名を取得 2 nameGC = stamp.GC_GetGroupName() If nameGC = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessagel() End If '選択中のグループの位置を取得 1 positionGF = stamp.GF_GetCurrentPosition() If positionGF < 0Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessagel() End If '選択中のグループの位置を取得 2 positionGC = stamp.GC_GetPosition() If positionGC < 0Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessagel() End If '選択中のグループのグループパスを取得 path = stamp.GF_GetCurrentGroupNamePath() If path = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessagel() End If '選択されているグループの CID(10 進)を設定→結果表示は 16 進 groupID = stamp.GC_GetGroupID() "CID は符号付き Long 型なので、CID の取得に成功してもその値が負となることがある "そのため GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessagel() End If '結果表示 WScript.Echo "グループ名:" & vbCrLf &_ " GF_GetCurrentGroupName:" & nameGF & vbCrLf &_ " GC_GetGroupName:" & nameGC & vbCrLf &_ "位置:" & vbCrLf &_ " GF_GetCurrentPosition:" & positionGF & vbCrLf &_ " GC_GetPosition:" & positionGC & vbCrLf &_ "グループパス : " & path & vbCrLf &_ "グループの CID:" & Hex(groupID) '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
GF_IsBOF	
選択されているグループが、そのグループの親グループを親に持つグループ内の先頭グループであるか判断します。	
long GF_IsBOF(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 1：選択されているグループは先頭グループです 0：選択されているグループは先頭グループではありません 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessages メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.GF_IsBOF()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループの位置を取得 res = stamp.GF_MoveRoot() groupPosition = stamp.GF_GetCurrentPosition() '指定したグループ以下にあるグループの最後のグループを選択 res = stamp.GF_MoveLast(groupPosition) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のグループが最初のグループであるか judge = stamp.GF_IsBOF() If judge < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If i = 1 msg = "" While judge = 0 '最初のグループでない '選択中のグループのグループ名を取得 msg = msg & i & ":" & stamp.GF_GetCurrentGroupname() & vbCrLf 'ルートグループ以下で、選択中のグループの前のグループを選択 res = stamp.GF_MovePrevious(groupPosition) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If judge = stamp.GF_IsBOF() i = i + 1 Wend '結果表示 WScript.Echo msg '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GF_IsEOF	
選択されているグループが、そのグループの親グループを親に持つグループ内の末尾グループであるか判断します。	
long GF_IsEOF(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 1: 選択されているグループは末尾グループです 0: 選択されているグループは末尾グループではありません 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.GF_IsEOF()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く(ルートグループが選択されている) res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループの位置を取得 res = stamp.GF_MoveRoot() groupPosition = stamp.GF_GetCurrentPosition() '指定したグループ以下にあるグループの最初のグループを選択 res = stamp.GF_MoveFirst(groupPosition) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中のグループが最後のグループであるか judge = stamp.GF_IsEOF() If judge < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If i = 1 msg = "" While judge = 0 '最後のグループでない '選択中のグループのグループ名を取得 msg = msg & i & ":" & stamp.GF_GetCurrentGroupname() & vbCrLf '指定したグループ以下にあるグループ内で、選択中のグループの次のグループを選択 res = stamp.GF_MoveNext(groupPosition) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If judge = stamp.GF_IsEOF() i = i + 1 Wend '結果表示 WScript.Echo msg '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GC_GetPosition	
選択されているグループの位置を取得します。	
long GC_GetPosition(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には選択されているグループの位置が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 取得されたグループの位置は、GF_Move メソッドなどの引数で利用します。</p>	
<p>使用例 long nResult = object.GC_GetPosition()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループを選択 res = stamp.GF_SeekGroupByUserName("<検索するユーザ名>") '選択中のグループのグループ名を取得 1 nameGF = stamp.GF_GetCurrentGroupName() If nameGF = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中のグループのグループ名を取得 2 nameGC = stamp.GC_GetGroupName() If nameGC = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中のグループの位置を取得 1 positionGF = stamp.GF_GetCurrentPosition() If positionGF < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中のグループの位置を取得 2 positionGC = stamp.GC_GetPosition() If positionGC < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中のグループのグループパスを取得 path = stamp.GF_GetCurrentGroupNamePath() If path = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択されているグループの CID(10 進)を設定→結果表示は 16 進 groupID = stamp.GC_GetGroupID() "CID は符号付き Long 型なので、CID の取得に成功してもその値が負となることがある "そのため GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '結果表示 WScript.Echo "グループ名:" & vbCrLf & _ " GF_GetCurrentGroupName:" & nameGF & vbCrLf & _ " GC_GetGroupName:" & nameGC & vbCrLf & _ "位置:" & vbCrLf & _ " GF_GetCurrentPosition:" & positionGF & vbCrLf & _ " GC_GetPosition:" & positionGC & vbCrLf & _ "グループパス : " & path & vbCrLf & _ "グループの CID:" & Hex(groupID) '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GC_GetGroupName	
選択されているグループの名前を取得します。	
string GC_GetGroupName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には選択されているグループの名前が戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strResult = object.GC_GetGroupName()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループを選択 res = stamp.GF_SeekGroupByUserName("<検索するユーザ名>") '選択中のグループのグループ名を取得 1 nameGF = stamp.GF_GetCurrentGroupName() If nameGF = "" Then WScript.Echo stamp.LastErrorNumber() & "." & stamp.LastErrorMessage() End If '選択中のグループのグループ名を取得 2 nameGC = stamp.GC_GetGroupName() If nameGC = "" Then WScript.Echo stamp.LastErrorNumber() & "." & stamp.LastErrorMessage() End If '選択中のグループの位置を取得 1 positionGF = stamp.GF_GetCurrentPosition() If positionGF < 0 Then WScript.Echo stamp.LastErrorNumber() & "." & stamp.LastErrorMessage() End If '選択中のグループの位置を取得 2 positionGC = stamp.GC_GetPosition() If positionGC < 0 Then WScript.Echo stamp.LastErrorNumber() & "." & stamp.LastErrorMessage() End If '選択中のグループのグループパスを取得 path = stamp.GF_GetCurrentGroupNamePath() If path = "" Then WScript.Echo stamp.LastErrorNumber() & "." & stamp.LastErrorMessage() End If '選択されているグループの CID(10 進)を設定→結果表示は 16 進 groupID = stamp.GC_GetGroupID() "CID は符号付き Long 型なので、CID の取得に成功してもその値が負となることがある "そのため GetLastErrorNumber の値でエラーを判定する If stamp.LastErrorNumber <> 1 Then WScript.Echo stamp.LastErrorNumber() & "." & stamp.LastErrorMessage() End If '結果表示 WScript.Echo "グループ名:" & vbCrLf &_ " GF_GetCurrentGroupName:" & nameGF & vbCrLf &_ " GC_GetGroupName:" & nameGC & vbCrLf &_ "位置:" & vbCrLf &_ " GF_GetCurrentPosition:" & positionGF & vbCrLf &_ " GC_GetPosition:" & positionGC & vbCrLf &_ "グループパス : " & path & vbCrLf &_ "グループの CID:" & Hex(groupID) '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GC_GetGroupExplanation	
選択されているグループの説明を取得します。	
string GC_GetGroupExplanation(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には選択されているグループの説明が戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <pre>string strResult = object.GC_GetGroupExplanation()</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く(ルートグループが選択される) res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループを選択 res = stamp.GF_SeekGroupByUserName("<検索するユーザ名>") '選択されているグループの説明を取得 explanation = stamp.GF_GetCurrentGroupName() 'エラーの場合、説明なしの場合ともに mexplanation は空文字なので、 'GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If If stamp.GetLastErrorNumber = "" Then '選択されているグループの説明を設定 res = stamp.GC_SetGroupExplanation("このグループはテスト用です。") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているグループを更新 res = stamp.GF_PutGroup() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If Else '選択されているグループの説明を設定 res = stamp.GC_SetGroupExplanation("") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているグループを更新 res = stamp.GF_PutGroup() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
GC_SetGroupExplanation	
選択されているグループの説明を設定します。	
long GC_SetGroupExplanation(string strExplanation)	
<p>パラメータ strExplanation: グループの説明</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.GC_SetGroupExplanation("グループの説明")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く(ルートグループが選択される) res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループを選択 res = stamp.GF_SeekGroupByUserName("<検索するユーザ名>") '選択されているグループの説明を取得 explanation = stamp.GF_GetCurrentGroupName() "エラーの場合、説明なしの場合ともに mExplanation は空文字なので、 "GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If If stamp.GetLastErrorNumber = "" Then '選択されているグループの説明を設定 res = stamp.GC_SetGroupExplanation("このグループはテスト用です。") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているグループを更新 res = stamp.GF_PutGroup() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If Else '選択されているグループの説明を設定 res = stamp.GC_SetGroupExplanation("") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているグループを更新 res = stamp.GF_PutGroup() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GC_GetGroupID	
グループに設定されている CID（管理ツールで生成される値）を取得します。	
long GC_GetGroupID(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には 0 以上の数値が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 取得されるユーザ CID は、管理ツールの「グループのプロパティ」で確認することができます。 （管理ツールで表示されるグループ CID は 16 進数に変換されて表示されています）</p>	
<p>使用例 long nResult = object.GC_GetGroupID()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループを選択 res = stamp.GF_SeekGroupByUserName("<検索するユーザ名>") '選択中のグループのグループ名を取得 1 nameGF = stamp.GF_GetCurrentGroupName() If nameGF = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中のグループのグループ名を取得 2 nameGC = stamp.GC_GetGroupName() If nameGC = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中のグループの位置を取得 1 positionGF = stamp.GF_GetCurrentPosition() If positionGF < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中のグループの位置を取得 2 positionGC = stamp.GC_GetPosition() If positionGC < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中のグループのグループパスを取得 path = stamp.GF_GetCurrentGroupNamePath() If path = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択されているグループの CID(10 進)を設定→結果表示は 16 進 groupID = stamp.GC_GetGroupID() "CID は符号付き Long 型なので、CID の取得に成功してもその値が負となることがある "そのため GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '結果表示 WScript.Echo "グループ名:" & vbCrLf & _ " GF_GetCurrentGroupName:" & nameGF & vbCrLf & _ " GC_GetGroupName:" & nameGC & vbCrLf & _ "位置:" & vbCrLf & _ " GF_GetCurrentPosition:" & positionGF & vbCrLf & _ " GC_GetPosition:" & positionGC & vbCrLf & _ "グループパス : " & path & vbCrLf & _ "グループの CID:" & Hex(groupID) '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GC_GetApprovalPassword	
選択されているグループの承認者パスワードを取得します。	
string GC_GetApprovalPassword(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には承認者パスワードを取得します。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessages メソッドを使用します。</p> <p>備考 このメソッドは下位互換性のために残されています。このメソッドで取得されるパスワード情報は以前のバージョンで捺印されたパソコン決裁の捺印オブジェクト（文書ファイルに挿入された電子印鑑）の捺印プロパティの監査情報を表示する際に求められるパスワード情報になります。</p>	
<p>使用例 string strResult = object.GC_GetApprovalPassword()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループを選択 res = stamp.GF_SeekGroupByUserName("<検索するユーザ名>") '選択されているグループの承認者パスワードを取得 password = stamp.GC_GetApprovalPassword() 'エラーの場合、パスワードなしの場合ともに password は空文字なので、 'GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているグループの承認者パスワードを設定 res = stamp.GC_SetApprovalPassword("password") '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GC_SetApprovalPassword	
選択されているグループの承認者パスワードを設定します。	
long GC_SetApprovalPassword(string strApprovalPassword)	
<p>パラメータ strApprovalPassword:承認者パスワード</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 このメソッドは下位互換性のために残されています。このメソッドで設定されるパスワード情報は以前のバージョンで捺印されたパソコン決裁の捺印オブジェクト（文書ファイルに挿入された電子印鑑）の捺印プロパティの監査情報を表示する際に求められるパスワード情報になります。</p>	
<p>使用例 long nResult = object.GC_SetApprovalPassword("承認者パスワード")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループを選択 res = stamp.GF_SeekGroupByUserName("<検索するユーザ名>") '選択されているグループの承認者パスワードを取得 password = stamp.GC_GetApprovalPassword() 'エラーの場合、パスワードなしの場合ともに password は空文字なので、 'GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているグループの承認者パスワードを設定 res = stamp.GC_SetApprovalPassword("password") '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GF_AppendGroup	
選択されているグループを親グループとした、新しいグループを追加します。 成功した場合、追加されたグループが選択された状態になります。	
long GF_AppendGroup(string strGroupName)	
<p>パラメータ strGroupName:グループ名</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません E_STF32V3LIB(-4):グループ名が重複している場合や内部的なエラーが発生しました 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 新しく追加するグループの名前は、同一の親グループを持つグループ内では重複は許可されませんが異なる親グループでは追加を行うことが可能です。</p>	
<p>使用例 long nResult = object.GF_AppendGroup("新しいグループ")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループを選択 res = stamp.GF_SeekGroupByUserName("<検索するユーザ名>") '選択しているグループの下に、新しいグループを追加 res = stamp.GF_AppendGroup("NewGroup") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GF_PutGroup	
選択されているグループを更新します。	
long GF_PutGroup(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 GC_SetGroupExplantion などグループに関する変更を行った場合、その変更を保存します。</p>	
<p>使用例 long nResult = object.GF_PutGroup()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く(ルートグループが選択される) res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループを選択 res = stamp.GF_SeekGroupByUserName("<検索するユーザ名>") '選択されているグループの説明を取得 explanation = stamp.GF_GetCurrentGroupName() "エラーの場合、説明なしの場合ともに mexplanation は空文字なので、 "GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If If stamp.GetLastErrorNumber = "" Then '選択されているグループの説明を設定 res = stamp.GC_SetGroupExplanation("このグループはテスト用です。") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているグループを更新 res = stamp.GF_PutGroup() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If Else '選択されているグループの説明を設定 res = stamp.GC_SetGroupExplanation("") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているグループを更新 res = stamp.GF_PutGroup() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GF_RenewParentGroup	
選択しているグループの親グループを引数で指定された親グループに変更します。	
long GF_RenewParentGroup(long nParentGroupPosition)	
<p>パラメータ nParentGroupPosition: 変更後の親グループの位置</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 引数で指定するグループの位置は SF_GetParentGroup メソッドや GC_GetPosition メソッド、GF_GetCurrentPosition メソッドなどで取得された値である必要があります。</p>	
<p>使用例 long nResult = object.GF_RenewParentGroup(0)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '新しい親グループを追加し、その位置を取得 res = stamp.GF_MoveRoot() res = stamp.GF_AppendGroup("NewParentGroup") parentGroupPosition = stamp.GF_GetCurrentPosition() '操作対象のグループを選択 res = stamp.GF_SeekGroupByUserName("<検索するユーザ名>") groupPosition = stamp.SF_GetParentGroup() res = stamp.GF_Move(groupPosition) '選択しているグループの親グループを変更 res = stamp.GF_RenewParentGroup(parentGroupPosition) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '結果表示 WScript.Echo "変更後のグループパス" & ":" & stamp.GF_GetCurrentGroupNamePath() '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GF_GetGroupCount	
追加されているすべてのグループの数を取得します。	
long GF_GetGroupCount(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には追加されているグループの総数が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.GF_GetGroupCount()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く(ルートグループが選択されている) res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '捺印用印鑑データファイルに追加されているグループの総数を取得 count = stamp.GF_GetGroupCount() If count < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "グループ総数:" & count '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_AppendUser	
引数で指定されたユーザ名を持つ、新しいユーザを追加します。	
long SF_AppendUser(string strUserName)	
<p>パラメータ strUserName:ユーザ名</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessages メソッドを使用します。</p> <p>備考 グループを指定してユーザの追加を行う場合には、GF_Move メソッドなどで本メソッドを利用する前に追加するグループを選択しておく必要があります。新しいユーザの追加に成功した場合には、追加されたユーザが選択されます。</p>	
<p>使用例 long nResult = object.SF_AppendUser("新しいユーザ")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") userName = "testUser" 'ユーザを検索 res = stamp.SF_SeekUser(userName) Select Case res Case 1 '指定したユーザ名のユーザが存在する(そのユーザが選択されている) '選択中のユーザを削除 res = stamp.SF_DeleteUser() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End If Case -5 '指定したユーザ名のユーザが存在しない '指定したユーザ名でユーザを追加 res = stamp.SF_AppendUser(userName) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End If Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_PutUser	
選択されているユーザを更新します。	
long SF_PutUser(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 SU_SetUserExplantation などユーザに関する変更を行った場合、その変更を保存します。</p>	
<p>使用例 long nResult = object.SF_PutUser()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザの[パソコン決裁ログイン画面でユーザ名リストに表示する]の設定を取得 enableUserStatus = stamp.SU_GetEnableUserConfigStatus() Select Case enableUserStatus Case 1 '有効 '[パソコン決裁ログイン画面でユーザ名リストに表示する]を無効にする res = stamp.SU_SetEnableUserConfigStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if '選択中のユーザを更新 res = stamp.SF_PutUser() '結果表示 WScript.Echo enableUserStatus & "->" & stamp.SU_GetEnableUserConfigStatus() & ":" & stamp.SF_GetCurrentUserName & "をユーザ名リストに表示しません。" Case 0 '無効 '[パソコン決裁ログイン画面でユーザ名リストに表示する]を有効にする res = stamp.SU_SetEnableUserConfigStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if '選択中のユーザを更新 res = stamp.SF_PutUser() '結果表示 WScript.Echo enableUserStatus & "->" & stamp.SU_GetEnableUserConfigStatus() & ":" & stamp.SF_GetCurrentUserName & "をユーザ名リストに表示します。" Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetStamp	
選択中のユーザに引数で指定した登録インデックスで追加されている印鑑データを取得します。	
long SF_GetStamp(long nIndex)	
<p>パラメータ nIndex:取得する印鑑データの登録インデックス</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 引数で使用する登録インデックスは、管理ツールの「印鑑データのプロパティ」で確認することができます。ユーザの指定を行うには、本メソッドを利用する前に、SF_Move メソッドや SF_SeekUser メソッドを利用してユーザを選択しておく必要があります。</p>	
<p>使用例 long nResult = object.SF_GetStamp(1)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") msg = "登録インデックス, シリアル番号, CID" For i = 0 To stamp.SU_GetStampCount() - 1 '操作対象の印鑑データを選択 res = stamp.SF_GetStamp(i) If res <> 1 Then WScript.Echo "GetStamp" & res & ":" & stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの登録インデックスを取得 index = stamp.SD_GetIndex() If index < 0 Then WScript.Echo "GetIndex" & res & ":" & stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データのシリアル番号を取得 serial = stamp.SD_GetStampSerialNumber() If serial = "" Then WScript.Echo "GetStampSerial" & res & ":" & stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの CID(10 進)を取得→結果表示は 16 進 CID = stamp.SD_GetStampID() "CID は符号付き Long 型なので、CID の取得に成功してもその値が負となることがある "そのため GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If msg = msg & vbCrLf & index & ", " & serial & ", " & Hex(CID) Next '結果表示 WScript.Echo msg '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_PutStamp	
引数で指定した登録インデックスで追加されている印鑑データを更新します。SD_SetLogStatus メソッドなどで印鑑データに関する変更を行った場合、その変更を保存します。	
long SF_PutStamp(long nIndex)	
<p>パラメータ nIndex:更新する印鑑データの登録インデックス</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 引数で使用する登録インデックスは、管理ツールの [印鑑データのプロパティ] で確認することができます。ユーザの指定を行うには、本メソッドを利用する前に、SF_Move メソッドや SF_SeekUser メソッドを利用してユーザを選択しておく必要があります。</p>	
<p>使用例 long nResult = object.SF_PutStamp(1)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの印影の角度を取得 angle = stamp.SD_GetStampAngle() If angle < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If If angle <> 0 Then '選択中の印鑑データの印影の角度を設定 res = stamp.SD_SetStampAngle(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If End If Else '選択中の印鑑データの印影の角度を設定 res = stamp.SD_SetStampAngle(15) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If End If End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_DrawStamp	
引数で指定されたデバイスコンテキスト内に選択されている印鑑データの印影を描画します。	
long SD_DrawStamp(long hDC, long x, long y)	
<p>パラメータ</p> <p>hDC: デバイスコンテキストハンドル x: 描画時の左座標 y: 描画時の上座標</p> <p>戻り値</p> <p>long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21): 読み取り専用で開かれています E_DSM_NOT_OPEN(-1): 捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <p>long nResult = object.SD_DrawStamp(hDC, 0, 0)</p>	
サンプルコード	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetAdditionDate	
印影に表示される日時を取得します。	
string SD_GetAdditionDate(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には日付文字列が戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <pre>string strResult = object.SD_GetAditionDate()</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの印影に表示される日時を取得 additonDate = stamp.SD_GetAdditionDate() '選択中の印鑑データの捺印日時を取得 impressTime = stamp.SD_GetImpressTime() "エラーの場合、設定なしの場合ともに"エラーの場合、名なしの場合ともに additonDate は空文字なので、 "GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & "." & stamp.GetLastErrorMessgae() End If '選択中の印鑑データの印影に表示される日時を設定 res = stamp.SD_SetAdditionDate(2009,11,22,12,23,56) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & "." & stamp.GetLastErrorMessgae() End If '結果表示 WScript.Echo "印影に表示される日時" & vbCrLf & _ vbTab & "SD_GetAdditionDate=" & additonDate & vbCrLf & _ vbTab & "SD_GetImpressTime=" & impressTime & vbCrLf & _ "変更後：印影に表示される日時"& vbCrLf & _ vbTab & "SD_GetAdditionDate=" & stamp.SD_GetAdditionDate() & vbCrLf & _ vbTab & "SD_GetImpressTime=" & stamp.SD_GetImpressTime() '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_DsmClose	
開いている捺印用印鑑データファイルを閉じます。	
long SF_DsmClose(void)	
<p>パラメータ ありません</p> <p>戻り値 ありません</p> <p>備考</p>	
<p>使用例 object.SF_DsmClose()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '結果表示 If res = 1 Then WScript.Echo "捺印用印鑑データファイルを開きました。" Else WScript.Echo "捺印用印鑑データファイルを開けませんでした。" & vbCrLf & _ stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() WScript.Echo "捺印用印鑑データファイルを閉じました。" Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetAdditionDate	
印影に表示される日時を設定します。	
string SD_SetAdditionDate(long nYear, long nMonth, long nDay, long nHour, long nMinute, long nSecond)	
<p>パラメータ nYear: 年 nMonth: 月 nDay: 日 nHour: 時 nMinute: 分 nSecond: 秒</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SD_SetAdditionDate(2009, 12, 31, 12, 34, 56)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの印影に表示される日時を取得 additonDate = stamp.SD_GetAdditionDate() '選択中の印鑑データの捺印日時を取得 impressTime = stamp.SD_GetImpressTime() "エラーの場合、設定なしの場合ともに"エラーの場合、名なしの場合ともに additonDate は空文字なので、 "GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessaqe() End If '選択中の印鑑データの印影に表示される日時を設定 res = stamp.SD_SetAdditionDate(2009,11,22,12,23,56) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessaqe() End If '結果表示 WScript.Echo "印影に表示される日時" & vbCrLf & _ vbTab & "SD_GetAdditionDate=" & additonDate & vbCrLf & _ vbTab & "SD_GetImpressTime=" & impressTime & vbCrLf & _ "変更後：印影に表示される日時" & vbCrLf & _ vbTab & "SD_GetAdditionDate=" & stamp.SD_GetAdditionDate() & vbCrLf & _ vbTab & "SD_GetImpressTime=" & stamp.SD_GetImpressTime() '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetImpressTime	
捺印時間を取得します。	
string SD_GetImpressTime(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には捺印時間の文字列が戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strResult = object.SD_GetImpressTime()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの印影に表示される日時を取得 additonDate = stamp.SD_GetAdditionDate() '選択中の印鑑データの捺印日時を取得 impressTime = stamp.SD_GetImpressTime() "エラーの場合、設定なしの場合ともに"エラーの場合、名なしの場合ともに additonDate は空文字なので、 "GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中の印鑑データの印影に表示される日時を設定 res = stamp.SD_SetAdditionDate(2009,11,22,12,23,56) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "印影に表示される日時" & vbCrLf & _ vbTab & "SD_GetAdditionDate=" & additonDate & vbCrLf & _ vbTab & "SD_GetImpressTime=" & impressTime & vbCrLf & _ "変更後：印影に表示される日時"& vbCrLf & _ vbTab & "SD_GetAdditionDate=" & stamp.SD_GetAdditionDate() & vbCrLf & _ vbTab & "SD_GetImpressTime=" & stamp.SD_GetImpressTime() '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GF_SeekGroupByUserName	
引数で指定されたユーザ名と一致するユーザが追加されているグループを選択します。	
long GF_SeekGroupByUserName(string strUserName)	
<p>パラメータ strUserName:検索を行うユーザ名</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 成功した場合には、検索されたグループが選択され、失敗した場合には末尾のグループが選択されます。</p>	
<p>使用例 long nResult = object.GF_SeekGroupByUserName("検索ユーザ名")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループを選択 res = stamp.GF_SeekGroupByUserName("<検索するユーザ名>") '選択しているグループの親グループを選択 res = stamp.GF_MoveParent() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '結果表示 WScript.Echo "親グループ:" & stamp.GF_GetCurrentGroupName() 'ルートグループを選択 res = stamp.GF_MoveRoot() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '結果表示 WScript.Echo "選択中のグループ:" & stamp.GF_GetCurrentGroupName() '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GetVersion	
利用モジュールのバージョン情報を取得します。	
long GetVersion(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には利用モジュールのバージョン情報文字列が戻ります。 失敗した場合には空文字が戻ります。</p> <p>備考</p>	
<p>使用例 string strResult = object.GetVersion()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") 'ライブラリモジュールのバージョンを取得 version = stamp.GetVersion() msg = "バージョン:" & version & vbCrLf 'ライブラリモジュールのエディションを取得 edition = stamp.GetEdition() Select Case edition Case 1 '製品版 msg = msg & "エディション:製品版" Case 2 '試用版 msg = msg & "エディション:試用版" Case Else msg = msg & "エディション:不明" End Select '結果表示 WScript.Echo msg set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetUserLevel	
ユーザの役職 ID を取得します。	
long SU_GetUserLevel(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には役職 ID に設定された数値が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 役職 ID は管理ツールで設定することができます。</p>	
<p>使用例 long nResult = object.SU_GetUserLevel()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザの役職 ID を取得 level = stamp.SU_GetUserLevel() If level <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '選択中のユーザの役職名を取得 levelName = stamp.SU_GetUserLevelName(level) 'エラーの場合、設定なしの場合ともに levelName は空文字なので、 'GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '結果表示 WScript.Echo "役職 ID:" & level & vbCrLf &_ "役職名:" & levelName '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetUserLevelName	
ユーザの役職名を取得します。	
string SU_GetUserLevelName(long nLevel)	
<p>パラメータ nLevel 取得する役職レベル</p> <p>戻り値 string 成功した場合には役職名に設定された文字列が戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 役職名は役職 ID に関連付けされた文字列で管理ツールで設定することができます。</p>	
<p>使用例 string strResult = object.SU_GetUserLevelName(long nLevel)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザの役職 ID を取得 level = stamp.SU_GetUserLevel() If level <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザの役職名を取得 levelName = stamp.SU_GetUserLevelName(level) 'エラーの場合、設定なしの場合ともに levelName は空文字なので、 'GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "役職 ID:" & level & vbCrLf & _ "役職名:" & levelName '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_GetPackageSerialNumber	
パッケージ シリアル番号（注文番号）を取得します。	
string SFI_GetPackageSerialNumber(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合にはパッケージ シリアル番号が文字列で戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、SFI_GetLastErrorNumber または SFI_GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <pre>string strResult = object.SFI_GetPackageSerialNumber()</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.StmpDat") '印鑑セットアップ元ファイルを開く res = stamp.SFI_IdxOpen("C:\YSTMP\YSTMP.ipx", "admin") '導入パック シリアル番号を取得 introducePackageSeriall = stamp.SFI_GetIntroducePackageSerialNumber() If introducePackageSeriall = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if 'パッケージ シリアル番号（注文番号）を取得 packageSeriall = stamp.SFI_GetPackageSerialNumber If packageSeriall = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if '結果表示 WScript.Echo "導入パック シリアル番号:" & introducePackageSeriall & vbCrLf &_ "パッケージ シリアル番号(注文番号):" & packageSeriall '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_GetUserFirstName	
選択されている印鑑データの名を取得します。	
string SFI_GetUserFirstName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には名が文字列で戻されます。 失敗した場合には空文字が戻されます。 詳細なエラー情報を取得するには、SFI_GetLastErrorType または SFI_GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑セットアップ元ファイル内の名は、印鑑データ申込時にご記入いただいた文字列です。印鑑セットアップ元ファイル内で印鑑データを選択するには SFI_SeekUser メソッドなどを利用します。</p>	
<p>使用例</p> <pre>string strResult = object.SFI_GetUserFirstName()</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.StmpDat") '印鑑セットアップ元ファイルを開く res = stamp.SFI_IdxOpen("C:\YSTMP\YSTMP.ipx", "admin") '操作対象のユーザを選択 res = stamp.SFI_SeekUser("<検索するユーザ名>") '選択されているユーザのユーザ ID を取得 userID = stamp.SFI_GetUserID() If userID = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザの姓を取得 userLastName = stamp.SFI_GetUserlastName() If userLastName = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザの名を取得 userFirstName = stamp.SFI_GetUserFirstName() If userFirstName = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザの印鑑シリアル番号を取得 stampSerial = stamp.SFI_GetStampSerialNumber() If stampSerial = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザの印鑑機種名を取得 xStamperCode = stamp.SFI_GetXStamperCode() If xStamperCode = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "ユーザ ID:" & userID & vbCrLf & _ "姓:" & userLastName & vbCrLf & _ "名:" & userFirstName & vbCrLf & _ "印鑑シリアル番号:" & stampSerial & vbCrLf & _ "印鑑機種名:" & xStamperCode '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_IdxOpen	
印鑑セットアップ元ファイル（拡張子: IDX または拡張子: IPX）を開きます。	
long SFI_IdxOpen(string strFilePath, string strPassword)	
<p>パラメータ</p> <p>strFilePath: 印鑑セットアップ元ファイル（拡張子: IDX または拡張子: IPX）の場所</p> <p>strPassword: 開くパスワード</p> <p>引数に拡張子: IPX ファイルを指定した場合には、このメソッドは SFI_GetExtractPath メソッドで取得される場所に指定されたファイルの内容を展開した後に読み込みます。（既定値ではユーザの一時ファイルフォルダ）別の場所に展開するには SFI_SetExtractPath メソッドを利用します。</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS(1)が戻ります。</p> <p>失敗した場合には以下の値が戻ります。</p> <p>E_IDX_ALREADYOPEN(-27): 既に開かれています</p> <p>E_IDX_CANNOT_OPEN(-26): 印鑑セットアップ元ファイルが見つかりません</p> <p>E_IDX_FILE_EXT(-28): ファイルの拡張子が異なります</p> <p>E_IDX_EXTRACT(-30): ファイルの展開に失敗しました</p> <p>E_UNMATCH_PASSWORD(-6): 開くパスワードが不正です</p> <p>詳細なエラー情報を取得するには、SFI</p> <p>備考</p> <p>印鑑セットアップ元ファイルの開くパスワードは、工場出荷時には半角英文字"admin"が設定されています。</p>	
<p>使用例</p> <p>long nResult = object.SFI_IdxOpen("印鑑セットアップ元ファイルの場所","admin")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '印鑑セットアップ元ファイルを開く res stamp.SFI_IdxOpen("C:\\$TMP\\$TMP.ipx", "admin") If res <> 1 Then Wscript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessaqe() End If '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_GetUserCount	
印鑑セットアップ元ファイルに登録されているユーザ数を取得します。	
long SFI_GetUserCount(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には印鑑セットアップ元ファイルに追加されているユーザ数が返ります。 失敗した場合には以下の値が返ります。 E_IDX_NOT_OPEN(-25):印鑑セットアップ元ファイルが開かれていません 詳細なエラー情報を取得するには、SFI_GetLastErrorType または SFI_GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SFI_GetUserCount()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '印鑑セットアップ元ファイルを開く res = stamp.SFI_IdxOpen("C:\STMP\STMP.ipx", "admin") '印鑑セットアップ元ファイルに登録されているユーザ数を取得 count = stamp.SFI_GetUserCount() If count < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If msg = "ユーザ数:" & count & vbCrLf '印鑑セットアップ元ファイルに追加されている末尾のユーザを選択 res = stamp.SFI_MoveLast() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザが最初のユーザであるか judge = stamp.SFI_IsBOF() If judge < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If While judge = 0 '最初のユーザでない '選択中のユーザのユーザ ID を取得 msg = msg & count & ":" & stamp.SFI_GetUserID() 'すべてのユーザ内で、選択中のユーザの前のユーザを選択 res = stamp.SFI_MovePrevious() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If judge = stamp.SFI_IsBOF() count = count -1 Wend '結果表示 WScript.Echo msg '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetCheckOutStatus	
ユーザのチェックアウト状態を取得します。	
long SU_GetCheckOutStatus(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 1: 選択されているユーザはチェックアウト状態です。 0: 選択されているユーザはチェックイン（通常）状態です。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessages メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SU_GetCheckOutStatus()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") 'ユーザのチェックアウト状態を取得 checkOutStatus = stamp.SU_GetCheckOutStatus() Select case checkOutStatus Case 1 'チェックアウト中 If MsgBox("チェックアウト中です。" & vbCrCL & "チェックアウトを強制解除しますか?", vbYesNo) = vbYes Then 'チェックアウト状態を解除 res = stamp.SU_ReleaseCheckOut() 'ユーザを更新 res = stamp.SF_PutUser() End If Case 0 WScript.Echo "チェックアウトしていません。" Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetTimeServerStatus	
捺印用印鑑データファイルに設定されている捺印時間の取得先の設定を取得します。	
long SF_GetTimeServerStatus(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 0：ローカル時間を適用 1：捺印用印鑑データファイルの参照先コンピュータの時間を適用 2：指定されたネットワークコンピュータの時間を適用 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage</p> <p>備考 この設定は、管理ツールで表示される [捺印用印鑑データファイルのプロパティ] 内の [捺印時間の取得先] 項目で変更されます。</p>	
<p>使用例 long nResult = object.SF_GetTimeServerStatus()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '[捺印時間の]取得先の設定を取得 status = stamp.SF_GetTimeServerStatus() Select Case status Case 0 '捺印するコンピュータの日付と時刻を使用する '[捺印用印鑑データファイルの参照先コンピュータの日付と時刻を使用する]に設定 res = stamp.SF_SetTimeServerStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End If Case 1 '捺印用印鑑データファイルの参照先コンピュータの日付と時刻を使用する '[次のコンピュータ名または IP アドレス上のの日付と時刻を使用する]に設定 res = stamp.SF_SetTimeServerStatus(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End If '捺印時間取得先のコンピュータ名を設定 res = stamp.SF_SetTimeServerName("TimeServer") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End If '[捺印時間の取得に失敗した場合にローカル時間を適用する]を設定 res = stamp.SF_SetTimeServerFailedStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End If Case 2 '次のコンピュータ名または IP アドレス上のの日付と時刻を使用する 'なしに設定 res = stamp.SF_SetTimeServerStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End If Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetTimeServerName	
捺印用印鑑データファイルに設定されている捺印時間の取得先コンピュータ名を取得します。	
string SF_GetTimeServerName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には捺印時間に利用するコンピュータ名が戻ります。 失敗した場合には空文字が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 この設定は、管理ツールで表示される [捺印用印鑑データファイルのプロパティ] 内の [捺印時間の取得先] 項目で変更されます。</p>	
<p>使用例 string strResult = object.SF_GetTimeServerName()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '[捺印時間の]取得先の設定を取得 status = stamp.SF_GetTimeServerStatus() Select Case status Case 0 '捺印するコンピュータの日付と時刻を使用する '[捺印用印鑑データファイルの参照先コンピュータの日付と時刻を使用する]に設定 res = stamp.SF_SetTimeServerStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If Case 1 '捺印用印鑑データファイルの参照先コンピュータの日付と時刻を使用する '[次のコンピュータ名または IP アドレス上のの日付と時刻を使用する]に設定 res = stamp.SF_SetTimeServerStatus(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '捺印時間取得先のコンピュータ名を設定 res = stamp.SF_SetTimeServerName("TimeServer") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '[捺印時間の取得に失敗した場合にローカル時間を適用する]を設定 res = stamp.SF_SetTimeServerFailedStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If Case 2 '次のコンピュータ名または IP アドレス上のの日付と時刻を使用する 'なしに設定 res = stamp.SF_SetTimeServerStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_SetTimeServerStatus	
捺印用印鑑データファイルに設定されている捺印時間の取得先を設定します。	
long SF_SetTimeServerStatus(long nStatus)	
<p>パラメータ</p> <p>nStatus:</p> <p>0 : ローカル時間を適用</p> <p>1 : 捺印用印鑑データファイルの参照先コンピュータの時間を適用</p> <p>2 : 指定されたネットワークコンピュータの時間を適用</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS(1)が戻ります。</p> <p>失敗した場合には以下の値が戻ります。</p> <p>DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています</p> <p>E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています</p> <p>詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p> <p>この設定は、管理ツールで表示される [捺印用印鑑データファイルのプロパティ] 内の [捺印時間の取得先] 項目で変更されます。</p>	
<p>使用例</p> <pre>long nResult = object.SF_SetTimeServerStatus(0)</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '[捺印時間]の[取得先]の設定を取得 status = stamp.SF_GetTimeServerStatus() Select Case status Case 0 '捺印するコンピュータの日付と時刻を使用する '[捺印用印鑑データファイルの参照先コンピュータの日付と時刻を使用する]に設定 res = stamp.SF_SetTimeServerStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If Case 1 '捺印用印鑑データファイルの参照先コンピュータの日付と時刻を使用する '[次のコンピュータ名または IP アドレス上のの日付と時刻を使用する]に設定 res = stamp.SF_SetTimeServerStatus(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '捺印時間取得先のコンピュータ名を設定 res = stamp.SF_SetTimeServerName("TimeServer") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '[捺印時間の取得に失敗した場合にローカル時間を適用する]を設定 res = stamp.SF_SetTimeServerFailedStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If Case 2 '次のコンピュータ名または IP アドレス上のの日付と時刻を使用する 'なしに設定 res = stamp.SF_SetTimeServerStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_SetTimeServerName	
捺印用印鑑データファイルに設定されている捺印時間の取得先コンピュータ名または IP アドレスを設定します。	
long SF_SetTimeServerName(string strTimeServerName)	
<p>パラメータ strTimeServerName:捺印時間に利用するコンピュータ名</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には次の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 この設定は、管理ツールで表示される [捺印用印鑑データファイルのプロパティ] 内の [捺印時間の取得先] 項目で変更されます。</p>	
<p>使用例 long strResult = object.SF_SetTimeServerName("適用コンピュータ名")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '[捺印時間の]取得先の設定を取得 status = stamp.SF_GetTimeServerStatus() Select Case status Case 0 '捺印するコンピュータの日付と時刻を使用する '[捺印用印鑑データファイルの参照先コンピュータの日付と時刻を使用する]に設定 res = stamp.SF_SetTimeServerStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If Case 1 '捺印用印鑑データファイルの参照先コンピュータの日付と時刻を使用する '[次のコンピュータ名または IP アドレス上のの日付と時刻を使用する]に設定 res = stamp.SF_SetTimeServerStatus(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '捺印時間取得先のコンピュータ名を設定 res = stamp.SF_SetTimeServerName("TimeServer") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '[捺印時間の取得に失敗した場合にローカル時間を適用する]を設定 res = stamp.SF_SetTimeServerFailedStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If Case 2 '次のコンピュータ名または IP アドレス上のの日付と時刻を使用する 'なしに設定 res = stamp.SF_SetTimeServerStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetTimeServerFailedStatus	
捺印用印鑑データファイルに設定されている捺印時間の取得に失敗した場合の状態を取得します。	
long SF_GetTimeServerFailedStatus(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 1：時刻取得に失敗した場合にはローカル時間を適用 0：時刻取得に失敗した場合には捺印を失敗させる 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 この設定は、管理ツールで表示される「捺印用印鑑データファイルのプロパティ」内の「捺印時間の取得に失敗した場合にローカル時間を適用する」項目で変更されます。</p>	
<p>使用例 long nResult = object.SF_GetTimeServerFailedStatus()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '[捺印時間]の[取得先]の設定を取得 status = stamp.SF_GetTimeServerStatus() Select Case status Case 0 '捺印するコンピュータの日付と時刻を使用する '[捺印用印鑑データファイルの参照先コンピュータの日付と時刻を使用する]に設定 res = stamp.SF_SetTimeServerStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If Case 1 '捺印用印鑑データファイルの参照先コンピュータの日付と時刻を使用する '[次のコンピュータ名または IP アドレス上のの日付と時刻を使用する]に設定 res = stamp.SF_SetTimeServerStatus(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '捺印時間取得先のコンピュータ名を設定 res = stamp.SF_SetTimeServerName("TimeServer") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '[捺印時間の取得に失敗した場合にローカル時間を適用する]を設定 res = stamp.SF_SetTimeServerFailedStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If Case 2 '次のコンピュータ名または IP アドレス上のの日付と時刻を使用する 'なしに設定 res = stamp.SF_SetTimeServerStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_SetTimeServerFailedStatus	
捺印用印鑑データファイルに設定されている捺印時間の取得に失敗した場合の状態を設定します。	
long SF_SetTimeServerFailedStatus(long nStatus)	
<p>パラメータ</p> <p>nStatus:</p> <p>1: 時刻取得に失敗した場合にはローカル時間を適用</p> <p>0: 時刻取得に失敗した場合には捺印を失敗させる</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS(1)が戻ります。</p> <p>失敗した場合には以下の値が戻ります。</p> <p>DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています</p> <p>E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれています</p> <p>詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p> <p>この設定は、管理ツールで表示される [捺印用印鑑データファイルのプロパティ] 内の [捺印時間の取得に失敗した場合にローカル時間を適用する] 項目で変更されます。</p>	
<p>使用例</p> <pre>long nResult = object.SF_SetTimeServerFailedStatus(0)</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '[捺印時間]の[取得先]の設定を取得 status = stamp.SF_GetTimeServerStatus() Select Case status Case 0 '捺印するコンピュータの日付と時刻を使用する '[捺印用印鑑データファイルの参照先コンピュータの日付と時刻を使用する]に設定 res = stamp.SF_SetTimeServerStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If Case 1 '捺印用印鑑データファイルの参照先コンピュータの日付と時刻を使用する '[次のコンピュータ名または IP アドレス上のの日付と時刻を使用する]に設定 res = stamp.SF_SetTimeServerStatus(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '捺印時間取得先のコンピュータ名を設定 res = stamp.SF_SetTimeServerName("TimeServer") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '[捺印時間の取得に失敗した場合にローカル時間を適用する]を設定 res = stamp.SF_SetTimeServerFailedStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If Case 2 '次のコンピュータ名または IP アドレス上のの日付と時刻を使用する 'なしに設定 res = stamp.SF_SetTimeServerStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_SeekUserByInpplet	
指定されたインプレットのデバイス ID で検索を行います。 ユーザが見つかった場合、そのユーザが選択されます。	
long SF_SeekUserByInpplet(string strDeviceID)	
<p>パラメータ strDeviceID:検索するインプレットのデバイス ID</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません E_USER_NOT_FOUND(-5):指定されたユーザが見つかりません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 引数で利用するインプレットのデバイス ID は IExtension::GetDeviceID メソッドなどで取得された値を使います。</p>	
<p>使用例 long nResult = object.SF_SeekUserByInpplet("検索対象のデバイス ID")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") WScript.Echo "インプレットで[OK]ボタンをクリックしてください。" 'インプレットのデバイス ID を取得 Set device = CreateObject("DSTMPLib.Extension") deviceID = device.GetDeviceID() Set device = Nothing 'インプレットのデバイス ID で検索 res = stamp.SF_SeekUserByInpplet(deviceID) '結果表示 Select Case res Case 1 '指定したデバイス ID のユーザが存在する(そのユーザが選択されている) WScript.Echo stamp.SF_GetCurrentUserName() Case -5 '指定したデバイス ID のユーザが存在しない WScript.Echo "登録されていません。" Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetEnableUserConfigStatus	
ユーザに設定されている[ユーザによるオプション設定の変更を禁止する]項目を取得します。	
long SU_GetEnableUserConfigStatus(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 1: 設定あり 0: 設定なし 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1): 捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 この設定は、管理ツールで表示される [ユーザのプロパティ] 内の [捺印ツールによる設定変更を無効にする] 項目で変更されます。</p>	
<p>使用例 long nResult = object.SU_GetEnableUserConfigStatus()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザの[ユーザによるオプション設定の変更を禁止する]の設定を取得 status = stamp.SU_GetEnableUserConfigStatus() Select Case status Case 1 '有効 '[ユーザによるオプション設定の変更を禁止する]を無効にする res = stamp.SU_SetEnableUserConfigStatus(0) '選択中のユーザを更新 res = stamp.SF_PutUser() WScript.Echo "I->0:" & stamp.SU_GetEnableUserConfigStatus() Case 0 '無効 '[ユーザによるオプション設定の変更を禁止する]を有効にする res = stamp.SU_SetEnableUserConfigStatus(1) '選択中のユーザを更新 res = stamp.SF_PutUser() WScript.Echo "0->1:" & stamp.SU_GetEnableUserConfigStatus() Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_SetEnableUserConfigStatus	
[ユーザによるオプション設定の変更を禁止する]項目を設定します。	
long SU_SetEnableUserConfigStatus(nStatus)	
<p>パラメータ nStatus: 1: 設定あり 0: 設定なし</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 設定の変更を捺印用印鑑データに反映させるには、SF_PutUser()を実行して選択されているユーザを更新します。</p>	
<p>使用例 long nResult = object.SU_SetEnableUserConfigStatus(1)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザの[ユーザによるオプション設定の変更を禁止する]の設定を取得 status = stamp.SU_GetEnableUserConfigStatus() Select Case status Case 1 '有効 '[ユーザによるオプション設定の変更を禁止する]を無効にする res = stamp.SU_SetEnableUserConfigStatus(0) '選択中のユーザを更新 res = stamp.SF_PutUser() WScript.Echo "I->0:" & stamp.SU_GetEnableUserConfigStatus() Case 0 '無効 '[ユーザによるオプション設定の変更を禁止する]を有効にする res = stamp.SU_SetEnableUserConfigStatus(1) '選択中のユーザを更新 res = stamp.SF_PutUser() WScript.Echo "0->1:" & stamp.SU_GetEnableUserConfigStatus() Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetEnableCheckOutStatus	
[チェックアウトを禁止する] 項目の状態を取得します。	
long SU_GetEnableCheckOutStatus(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 1: 設定あり 0: 設定なし 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1): 捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 この設定は、管理ツールで表示される [ユーザのプロパティ] 内の [チェックアウトを禁止する] 項目で変更されます。</p>	
<p>使用例 long nResult = object.SU_GetEnableCheckOutStatus()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") 'ユーザのチェックアウトの設定状態を取得 enableCheckOutStatus = stamp.SU_GetEnableCheckOutStatus() Select Case enableCheckOutStatus Case 1 'チェックアウト利用禁止 '利用禁止を解除 res = stamp.SU_SetEnableCheckOutStatus(0) 'ユーザを更新 res = stamp.SF_PutUser() Case 0 'チェックアウト利用禁止でない '利用禁止にする res = stamp.SU_SetEnableCheckOutStatus(1) 'ユーザを更新 res = stamp.SF_PutUser() Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_SetEnableCheckOutStatus	
チェックアウトを禁止する状態を設定します。	
long SU_SetEnableCheckOutStatus(nStatus)	
<p>パラメータ nStatus: 1: 設定あり 0: 設定なし</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 この設定は、管理ツールで表示される [ユーザのプロパティ] 内の [チェックアウトを禁止する] 項目で変更されます。 設定の変更を捺印用印鑑データに反映させるには、SF_PutUser()を実行して選択されているユーザを更新します。</p>	
<p>使用例 long nResult = object.SU_SetEnableCheckOutStatus(1)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") 'ユーザのチェックアウトの設定状態を取得 enableCheckOutStatus = stamp.SU_GetEnableCheckOutStatus() Select Case enableCheckOutStatus Case 1 'チェックアウト利用禁止 '利用禁止を解除 res = stamp.SU_SetEnableCheckOutStatus(0) 'ユーザを更新 res = stamp.SF_PutUser() Case 0 'チェックアウト利用禁止でない '利用禁止にする res = stamp.SU_SetEnableCheckOutStatus(1) 'ユーザを更新 res = stamp.SF_PutUser() Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_ReleaseCheckOut	
チェックアウトされているユーザの状態を解除します。	
long SU_ReleaseCheckOut(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 設定の変更を捺印用印鑑データに反映させるには、SF_PutUser()を実行して選択されているユーザを更新します。 この設定は、管理ツールで表示される「ユーザのプロパティ」内の「チェックアウトを解除する」項目と同様の操作を行います。このメソッドを使ってチェックアウト状態を解除した場合に、捺印ツールをからのチェックイン操作が無効となり、チェックアウト先で記録された捺印ログ情報などはすべて無くなりますのでご注意ください。</p>	
<p>使用例 long nResult = object.SU_ReleaseCheckOut()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") 'ユーザのチェックアウト状態を取得 checkOutStatus = stamp.SU_GetCheckOutStatus() Select case checkOutStatus Case 1 'チェックアウト中 If MsgBox("チェックアウト中です。" & vbCrCL & "チェックアウトを強制解除しますか？", vbYesNo) = vbYes Then 'チェックアウト状態を解除 res = stamp.SU_ReleaseCheckOut() 'ユーザを更新 res = stamp.SF_PutUser() End If Case 0 WScript.Echo "チェックアウトしていません。" Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetIntroducePackageSerialNumber	
導入パック シリアル番号を取得します。	
string SU_GetIntroducePackageSerialNumber(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には導入パック シリアル番号が文字列で戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 このメソッドは、下位互換性のために残されています。</p>	
<p>使用例 string strResult = object.SU_GetIntroducePackageSerialNumber()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") 'デバッグ用 stamp.ResetError '導入パックシリアル番号を取得 introducePackageSerial = stamp.SU_GetIntroducePackageSerialNumber() If introducePackageSerial = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If 'パッケージシリアル番号を取得 packageSerial = stamp.SU_GetPackageSerialNumber() If introducePackageSerial = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "導入パックシリアル番号:" & introducePackageSerial & vbCrLf & _ "パッケージシリアル番号:" & packageSerial '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetPackageSerialNumber	
パッケージ シリアル番号（注文番号）を取得します。	
string SU_GetPackageSerialNumber(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合にはパッケージ シリアル番号が文字列で戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strResult = object.SU_GetPackageSerialNumber()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") 'デバッグ用 stamp.ResetError '導入パックシリアル番号を取得 introducePackageSerial = stamp.SU_GetIntroducePackageSerialNumber() If introducePackageSerial = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessagge() End If 'パッケージシリアル番号を取得 packageSerial = stamp.SU_GetPackageSerialNumber() If introducePackageSerial = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessagge() End If '結果表示 WScript.Echo "導入パックシリアル番号:" & introducePackageSerial & vbCrLf & _ "パッケージシリアル番号:" & packageSerial '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_IsBOF	
選択されているユーザが最初のユーザであるか判断します	
long SFI_IsBOF(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS_FALSE(0)または DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_IDX_NOT_OPEN(-1):印鑑セットアップ元ファイルが開かれていません 詳細なエラー情報を取得するには、SFI_GetLastErrorType または SFI_GetLastErrorMessage メソッドを使用します。</p> <p>備考 最初のユーザとは印鑑セットアップ元ファイルに追加されている先頭のユーザです。</p>	
<p>使用例 long nResult = object.SFI_IsBOF()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '印鑑セットアップ元ファイルを開く res = stamp.SFI_IdxOpen("C:\YSTMP\YSTMP.ipx", "admin") '印鑑セットアップ元ファイルに登録されているユーザ数を取得 count = stamp.SFI_GetUserCount() If count < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If msg = "ユーザ数:" & count & vbCrLf '印鑑セットアップ元ファイルに追加されている末尾のユーザを選択 res = stamp.SFI_MoveLast() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザが最初のユーザであるか judge = stamp.SFI_IsBOF() If judge < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If While judge = 0 '最初のユーザでない '選択中のユーザのユーザ ID を取得 msg = msg & count & ":" & stamp.SFI_GetUserID() 'すべてのユーザ内で、選択中のユーザの前のユーザを選択 res = stamp.SFI_MovePrevious() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If judge = stamp.SFI_IsBOF() count = count - 1 Wend '結果表示 WScript.Echo msg '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_IsEOF	
選択されているユーザが末尾のユーザであるか判断します	
long SFI_IsEOF(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS_FALSE(0)または DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_IDX_NOT_OPEN(-1):印鑑セットアップ元ファイルが開かれていません 詳細なエラー情報を取得するには、SFI_GetLastErrorType または SFI_GetLastErrorMessage メソッドを使用します。</p> <p>備考 末尾のユーザとは印鑑セットアップ元ファイルで最後に追加されているユーザです。</p>	
<p>使用例 long nResult = object.SFI_IsEOF()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '印鑑セットアップ元ファイルを開く res = stamp.SFI_IdxOpen("C:\YSTMP\YSTMP.ipx", "admin") '印鑑セットアップ元ファイルに追加されている最初のユーザを選択 res = stamp.SFI_MoveFirst() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザが最後のユーザであるか judge = stamp.SFI_IsEOF() If judge < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If i = 1 msg = "" While judge = 0 '最後のユーザでない '選択中の印鑑データのユーザ ID を取得 msg = msg & i & ":" & stamp.SFI_GetUserID() 'すべての印鑑データ内で、選択中の印鑑データの次のユーザを選択 res = stamp.SFI_Movenext() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If judge = stamp.SFI_IsEOF() i = i + 1 Wend '結果表示 WScript.Echo msg '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_MoveFirst	
印鑑セットアップ元ファイルに追加されている最初のユーザを選択します。	
long SFI_MoveFirst(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS (1) が戻ります。 失敗した場合には以下の値が戻ります。 E_IDX_NOT_OPEN(-1): 印鑑セットアップ元ファイルが開かれていません 詳細なエラー情報を取得するには、SFI_GetLastErrorType または SFI_GetLastErrorMessage メソッドを使用します。</p> <p>備考 最初のユーザとは印鑑セットアップ元ファイルに追加されている先頭のユーザです。</p>	
<p>使用例 long nResult = object.SFI_MoveFirst()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '印鑑セットアップ元ファイルを開く res = stamp.SFI_IdxOpen("C:\STMP\STMP.ipx", "admin") '印鑑セットアップ元ファイルに追加されている最初のユーザを選択 res = stamp.SFI_MoveFirst() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザが最後のユーザであるか judge = stamp.SFI_IsEOF() If judge < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If i = 1 msg = "" While judge = 0 '最後のユーザでない '選択中の印鑑データのユーザ ID を取得 msg = msg & i & ":" & stamp.SFI_GetUserID() 'すべての印鑑データ内で、選択中の印鑑データの次のユーザを選択 res = stamp.SFI_Movenext() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If judge = stamp.SFI_IsEOF() i = i + 1 Wend '結果表示 WScript.Echo msg '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_MoveLast	
印鑑セットアップ元ファイルに追加されている末尾のユーザを選択します。	
long SFI_MoveLast(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS (1) が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_IDX_NOT_OPEN(-1):印鑑セットアップ元ファイルが開かれています 詳細なエラー情報を取得するには、SFI_GetLastErrorType または SFI_GetLastErrorMessage メソッドを使用します。</p> <p>備考 末尾のユーザとは印鑑セットアップ元ファイルで最後に追加されているユーザです。</p>	
<p>使用例 long nResult = object.SFI_MoveLast()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '印鑑セットアップ元ファイルを開く res = stamp.SFI_IdxOpen("C:\STMP\STMP.ipx", "admin") '印鑑セットアップ元ファイルに登録されているユーザ数を取得 count = stamp.SFI_GetUserCount() If count < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If msg = "ユーザ数:" & count & vbCrLf '印鑑セットアップ元ファイルに追加されている末尾のユーザを選択 res = stamp.SFI_MoveLast() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザが最初のユーザであるか judge = stamp.SFI_IsBOF() If judge < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If While judge = 0 '最初のユーザでない '選択中のユーザのユーザ ID を取得 msg = msg & count & ":" & stamp.SFI_GetUserID() 'すべてのユーザ内で、選択中のユーザの前のユーザを選択 res = stamp.SFI_MovePrevious() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If judge = stamp.SFI_IsBOF() count = count - 1 Wend '結果表示 WScript.Echo msg '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_MoveNext	
選択されているユーザの次ユーザを選択します。	
long SFI_MoveNext(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS (1) が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_IDX_NOT_OPEN(-1):印鑑セットアップ元ファイルが開かれています 詳細なエラー情報を取得するには、SFI_GetLastErrorType または SFI_GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SFI_MoveNext()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '印鑑セットアップ元ファイルを開く res = stamp.SFI_IdxOpen("C:\STMP\STMP.ipx", "admin") '印鑑セットアップ元ファイルに追加されている最初のユーザを選択 res = stamp.SFI_MoveFirst() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザが最後のユーザであるか judge = stamp.SFI_IsEOF() If judge < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If i = 1 msg = "" While judge = 0 '最後のユーザでない '選択中の印鑑データのユーザ ID を取得 msg = msg & i & ":" & stamp.SFI_GetUserID() 'すべての印鑑データ内で、選択中の印鑑データの次のユーザを選択 res = stamp.SFI_Movenext() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If judge = stamp.SFI_IsEOF() i = i + 1 Wend '結果表示 WScript.Echo msg '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_MovePrevious	
選択されているユーザの前ユーザを選択します。	
long SFI_MovePrevious(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS (1) が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_IDX_NOT_OPEN(-1):印鑑セットアップ元ファイルが開かれています 詳細なエラー情報を取得するには、SFI_GetLastErrorType または SFI_GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SFI_MovePrevious()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '印鑑セットアップ元ファイルを開く res = stamp.SFI_IdxOpen("C:\STMP\STMP.ipx", "admin") '印鑑セットアップ元ファイルに登録されているユーザ数を取得 count = stamp.SFI_GetUserCount() If count < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If msg = "ユーザ数:" & count & vbCrLf '印鑑セットアップ元ファイルに追加されている末尾のユーザを選択 res = stamp.SFI_MoveLast() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザが最初のユーザであるか judge = stamp.SFI_IsBOF() If judge < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If While judge = 0 '最初のユーザでない '選択中のユーザのユーザ ID を取得 msg = msg & count & ":" & stamp.SFI_GetUserID() 'すべてのユーザ内で、選択中のユーザの前のユーザを選択 res = stamp.SFI_MovePrevious() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If judge = stamp.SFI_IsBOF() count = count - 1 Wend '結果表示 WScript.Echo msg '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_SeekUser	
指定されたユーザ ID で完全一致検索を行います。 ユーザが見つかった場合、そのユーザが選択されます。	
long SFI_SeekUser(string strUserName)	
<p>パラメータ strUserName:検索するユーザ ID</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_IDX_NOT_OPEN(-25):印鑑セットアップ元ファイルが開かれていません E_USER_NOT_FOUND(-5):指定されたユーザが見つかりません 詳細なエラー情報を取得するには、SFI_GetLastErrorType または SFI_GetLastErrorMessage メソッドを使用します。</p> <p>備考 検索するユーザ ID は大文字・小文字を区別します。</p>	
<p>使用例 long nResult = object.SFI_SeekUser("検索対象のユーザ ID")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '印鑑セットアップ元ファイルを開く res = stamp.SFI_IdxOpen("C:\STMP\umeda.idx", "admin") stamp.ResetError userID = "ume" 'ユーザ ID を指定して、印鑑セットアップ元ファイル内のユーザを検索 res = stamp.SFI_SeekUser(userID) '結果表示 Select Case res Case 1 '指定したユーザ ID のユーザが存在する WScript.Echo userID & "は登録されています。" Case -5 '指定したユーザ ID のユーザが存在しない WScript.Echo userID & "は登録されていません。" Case Else WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_SeekUserByInpplet	
指定されたインプレットのデバイス ID で検索を行います。 ユーザが見つかった場合、そのユーザが選択されます。	
long SFI_SeekUserByInpplet(string strDeviceID)	
<p>パラメータ strDeviceID:検索するインプレットのデバイス ID</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_IDX_NOT_OPEN(-25):印鑑セットアップ元ファイルが開かれていません E_USER_NOT_FOUND(-5):指定されたユーザが見つかりません 詳細なエラー情報を取得するには、SFI_GetLastErrorType または SFI_GetLastErrorMessage メソッドを使用します。</p> <p>備考 引数で利用するインプレットのデバイス ID は IExtension::GetDeviceID メソッドなどで取得された値を使います。</p>	
<p>使用例 long nResult = object.SFI_SeekUserByInpplet("検索対象のデバイス ID")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '印鑑セットアップ元ファイルを開く res = stamp.SFI_IdxOpen("C:\STMP\umeda.idx", "admin") 'インプレットのデバイス ID を取得 WScript.Echo "インプレットで[OK]ボタンをクリックしてください。" stamp.ResetError Set device = CreateObject("DSTMPLib.Extension") deviceID = device.GetDeviceID() Set device = Nothing 'インプレットのデバイス ID で検索 res = stamp.SFI_SeekUserByInpplet(deviceID) '結果表示 Select Case res Case 1 '指定したデバイス ID のユーザが存在する WScript.Echo stamp.SFI_GetUserID() Case -5 '指定したデバイス ID のユーザが存在しない WScript.Echo "登録されていません。" Case Else WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_GetLastErrorType	
印鑑セットアップ元ファイルに関連するメソッド（メソッド名の先頭に SFI_が追加されているメソッド）で最後に発生したエラー番号を取得します。	
long SFI_GetLastErrorType(void)	
<p>パラメータ ありません</p> <p>戻り値 long 最後に発生したエラー番号をが戻ります。</p> <p>備考</p>	
<p>使用例 long nResult = object.SFI_GetLastErrorType()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '印鑑セットアップ元ファイルを指定しないことで、エラーを発生させる res = stamp.SFI_IdxOpen("", "") 'エラー情報取得 If res <> 1 Then 'エラー番号を取得 msg = "ErrorType: " & stamp.SFI_GetLastErrorType() & vbCrLf 'エラーメッセージを取得 msg = msg & "ErrorMessage: " & stamp.SFI_GetLastErrorMessage() End If '結果表示 WScript.Echo msg Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_GetLastErrorMessage	
印鑑セットアップ元ファイルに関連するメソッド（メソッド名の先頭に SFI_が追加されているメソッド）で最後に発生したエラーメッセージを取得します。	
string SFI_GetLastErrorMessage(void)	
<p>パラメータ ありません</p> <p>戻り値 string 最後に発生した定義済みのエラーメッセージをが戻ります。</p> <p>備考</p>	
<p>使用例 string strResult = object.SFI_GetLastErrorMessage()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '印鑑セットアップ元ファイルを指定しないことで、エラーを発生させる res = stamp.SFI_IdxOpen("", "") 'エラー情報取得 If res <> 1 Then 'エラー番号を取得 msg = "ErrorType: " & stamp.SFI_GetLastErrorType() & vbCrLf 'エラーメッセージを取得 msg = msg & "ErrorMessage: " & stamp.SFI_GetLastErrorMessage() End If '結果表示 WScript.Echo msg Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_GetStampSerialNumber	
選択されている印鑑のシリアル番号を取得します。	
string SFI_GetStampSerialNumber(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には印鑑データのシリアル番号が文字列で戻されます。 失敗した場合には空文字が戻されます。 詳細なエラー情報を取得するには、SFI_GetLastErrorType または SFI_GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑セットアップ元ファイル内で印鑑データを選択するには SFI_SeekUser メソッドなどを利用します。</p>	
<p>使用例 string strResult = object.SFI_GetStampSerialNumber()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '印鑑セットアップ元ファイルを開く res = stamp.SFI_IdxOpen("C:\STMP\STMP.ipx", "admin") '操作対象のユーザを選択 res = stamp.SFI_SeekUser("<検索するユーザ名>") '選択されているユーザのユーザ ID を取得 userID = stamp.SFI_GetUserID() If userID = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End If '選択されているユーザの姓を取得 userLastName = stamp.SFI_GetUserLastName() If userLastName = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End If '選択されているユーザの名を取得 userFirstName = stamp.SFI_GetUserFirstName() If userFirstName = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End If '選択されているユーザの印鑑シリアル番号を取得 stampSerial = stamp.SFI_GetStampSerialNumber() If stampSerial = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End If '選択されているユーザの印鑑機種名を取得 xStamperCode = stamp.SFI_GetXStamperCode() If xStamperCode = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessages() End If '結果表示 WScript.Echo "ユーザ ID:" & userID & vbCrLf & _ "姓:" & userLastName & vbCrLf & _ "名:" & userFirstName & vbCrLf & _ "印鑑シリアル番号:" & stampSerial & vbCrLf & _ "印鑑機種名:" & xStamperCode '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_GetUserID	
選択されているユーザのユーザ ID を取得します。	
string SFI_GetUserID(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合にはユーザ ID が文字列で戻されます。 失敗した場合には空文字が戻されます。 詳細なエラー情報を取得するには、SFI_GetLastErrorType または SFI_GetLastErrorMessage メソッドを使用します。</p> <p>備考 ユーザ ID は、印鑑データ申込時にご記入いただいたユーザを識別する記号です。印鑑セットアップ元ファイル内で印鑑データを選択するには SFI_SeekUser メソッドなどを利用します。</p>	
<p>使用例 string strResult = object.SFI_GetUserID()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '印鑑セットアップ元ファイルを開く res = stamp.SFI_IdxOpen("C:\YSTMP\YSTMP.ipx", "admin") '操作対象のユーザを選択 res = stamp.SFI_SeekUser("<検索するユーザ名>") '選択されているユーザのユーザ ID を取得 userID = stamp.SFI_GetUserID() If userID = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザの姓を取得 userLastName = stamp.SFI_GetUserlastName() If userLastName = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザの名を取得 userFirstName = stamp.SFI_GetUserFirstName() If userFirstName = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザの印鑑シリアル番号を取得 stampSerial = stamp.SFI_GetStampSerialNumber() If stampSerial = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザの印鑑機種名を取得 xStamperCode = stamp.SFI_GetXStamperCode() If xStamperCode = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "ユーザ ID:" & userID & vbCrLf & _ "姓:" & userLastName & vbCrLf & _ "名:" & userFirstName & vbCrLf & _ "印鑑シリアル番号:" & stampSerial & vbCrLf & _ "印鑑機種名:" & xStamperCode '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_GetXStamperCode	
印鑑データの機種名を取得します。	
string SFI_GetXStamperCode(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には印鑑データの機種名が文字列で戻されます。 失敗した場合には空文字が戻されます。 詳細なエラー情報を取得するには、SFI_GetLastErrorType または SFI_GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑セットアップ元ファイル内で印鑑データを選択するには SFI_SeekUser メソッドなどを利用します。</p>	
<p>使用例 string strResult = object.SFI_GetXStamperCode()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '印鑑セットアップ元ファイルを開く res = stamp.SFI_IdxOpen("C:\STMP\STMP.ipx", "admin") '操作対象のユーザを選択 res = stamp.SFI_SeekUser("<検索するユーザ名>") '選択されているユーザのユーザ ID を取得 userID = stamp.SFI_GetUserID() If userID = "" Then WScript.Echo stamp.GetLastErrorNumber() & "." & stamp.GetLastErrorMessag() End If '選択されているユーザの姓を取得 userLastName = stamp.SFI_GetUserlastName() If userLastName = "" Then WScript.Echo stamp.GetLastErrorNumber() & "." & stamp.GetLastErrorMessag() End If '選択されているユーザの名を取得 userFirstName = stamp.SFI_GetUserFirstName() If userFirstName = "" Then WScript.Echo stamp.GetLastErrorNumber() & "." & stamp.GetLastErrorMessag() End If '選択されているユーザの印鑑シリアル番号を取得 stampSerial = stamp.SFI_GetStampSerialNumber() If stampSerial = "" Then WScript.Echo stamp.GetLastErrorNumber() & "." & stamp.GetLastErrorMessag() End If '選択されているユーザの印鑑機種名を取得 xStamperCode = stamp.SFI_GetXStamperCode() If xStamperCode = "" Then WScript.Echo stamp.GetLastErrorNumber() & "." & stamp.GetLastErrorMessag() End If '結果表示 WScript.Echo "ユーザ ID:" & userID & vbCrLf & _ "姓:" & userLastName & vbCrLf & _ "名:" & userFirstName & vbCrLf & _ "印鑑シリアル番号:" & stampSerial & vbCrLf & _ "印鑑機種名:" & xStamperCode '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IstmpDat	メソッド
SFI_GetUserLastName	
選択されている印鑑データの姓を取得します。	
string SFI_GetUserLastName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には姓が文字列で戻されます。 失敗した場合には空文字が戻されます。 詳細なエラー情報を取得するには、SFI_GetLastErrorType または SFI_GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑セットアップ元ファイル内の姓は、印鑑データ申込時にご記入いただいた文字列です。印鑑セットアップ元ファイル内で印鑑データを選択するには SFI_SeekUser メソッドなどを利用します。</p>	
<p>使用例 string strResult = object.SFI_GetUserLastName()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '印鑑セットアップ元ファイルを開く res = stamp.SFI_IdxOpen("C:\STMP\STMP.ipx", "admin") '操作対象のユーザを選択 res = stamp.SFI_SeekUser("<検索するユーザ名>") '選択されているユーザのユーザ ID を取得 userID = stamp.SFI_GetUserID() If userID = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザの姓を取得 userLastName = stamp.SFI_GetUserLastName() If userLastName = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザの名を取得 userFirstName = stamp.SFI_GetUserFirstName() If userFirstName = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザの印鑑シリアル番号を取得 stampSerial = stamp.SFI_GetStampSerialNumber() If stampSerial = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザの印鑑機種名を取得 xStamperCode = stamp.SFI_GetXStamperCode() If xStamperCode = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "ユーザ ID:" & userID & vbCrLf & _ "姓:" & userLastName & vbCrLf & _ "名:" & userFirstName & vbCrLf & _ "印鑑シリアル番号:" & stampSerial & vbCrLf & _ "印鑑機種名:" & xStamperCode '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetLogSaveSpanMode	
捺印ログが保存される期間を取得します。	
long SF_GetLogSaveSpanMode(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 0：指定なし 1：毎日 2：毎月 3：ログ件数が設定値に到達した場合 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SF_GetLogSaveSpanMode()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '捺印ログが保存される期間を取得 span = stamp.SF_GetLogSaveSpanMode() Select Case span Case 0 '設定なし '捺印ログが保存される期間を毎日に設定 res = stamp.SF_SetLogSaveSpanMode(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if Case 1 '毎日 '捺印ログが保存される期間を毎月に設定 res = stamp.SF_SetLogSaveSpanMode(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if Case 2 '毎月 '捺印ログが保存される期間を指定件数に到達するまでに設定 res = stamp.SF_SetLogSaveSpanMode(3) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if '捺印ログが保存される最大件数を設定 res = stamp.SF_SetLogSaveMaxCount(90) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if Case 3 '指定件数に到達するまで '捺印ログが保存される期間をなしに設定 res = stamp.SF_SetLogSaveSpanMode(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_SetLogSaveSpanMode	
捺印ログが保存される期間を設定します。	
long SF_SetLogSaveSpanMode(long nMode)	
<p>パラメータ</p> <p>nMode: 0 : 指定なし 1 : 毎日 2 : 毎月 3 : ログ件数が設定値に到達した場合</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <pre>long nResult = object.SF_SetLogSaveSpanMode(1)</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '捺印ログが保存される期間を取得 span = stamp.SF_GetLogSaveSpanMode() Select Case span Case 0 '設定なし '捺印ログが保存される期間を毎日に設定 res = stamp.SF_SetLogSaveSpanMode(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if Case 1 '毎日 '捺印ログが保存される期間を毎月に設定 res = stamp.SF_SetLogSaveSpanMode(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if Case 2 '毎月 '捺印ログが保存される期間を指定件数に到達するまでに設定 res = stamp.SF_SetLogSaveSpanMode(3) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if '捺印ログが保存される最大件数を設定 res = stamp.SF_SetLogSaveMaxCount(90) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if Case 3 '指定件数に到達するまで '捺印ログが保存される期間をなしに設定 res = stamp.SF_SetLogSaveSpanMode(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetLogSaveMaxCount	
捺印ログが保存される最大件数を取得します。	
long SF_GetLogSaveMaxCount(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には設定されている最大件数の値が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SF_GetLogSaveMaxCount()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '捺印ログが保存される期間を取得 span = stamp.SF_GetLogSaveSpanMode() Select Case span Case 0 '設定なし '捺印ログが保存される期間を毎日に設定 res = stamp.SF_SetLogSaveSpanMode(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if Case 1 '毎日 '捺印ログが保存される期間を毎月に設定 res = stamp.SF_SetLogSaveSpanMode(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if Case 2 '毎月 '捺印ログが保存される期間を指定件数に到達するまでに設定 res = stamp.SF_SetLogSaveSpanMode(3) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if '捺印ログが保存される最大件数を設定 res = stamp.SF_SetLogSaveMaxCount(90) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if Case 3 '指定件数に到達するまで '捺印ログが保存される期間をなしに設定 res = stamp.SF_SetLogSaveSpanMode(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_SetLogSaveMaxCount	
捺印ログが保存される最大件数を設定します。	
long SF_SetLogSaveMaxCount(long nMaxCount)	
<p>パラメータ nMaxCount: 保存最大件数</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1): 捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SF_SetLogSaveMaxCount(1000)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '捺印ログが保存される期間を取得 span = stamp.SF_GetLogSaveSpanMode() Select Case span Case 0 '設定なし '捺印ログが保存される期間を毎日に設定 res = stamp.SF_SetLogSaveSpanMode(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if Case 1 '毎日 '捺印ログが保存される期間を毎月に設定 res = stamp.SF_SetLogSaveSpanMode(2) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if Case 2 '毎月 '捺印ログが保存される期間を指定件数に到達するまでに設定 res = stamp.SF_SetLogSaveSpanMode(3) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if '捺印ログが保存される最大件数を設定 res = stamp.SF_SetLogSaveMaxCount(90) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if Case 3 '指定件数に到達するまで '捺印ログが保存される期間をなしに設定 res = stamp.SF_SetLogSaveSpanMode(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End if Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_SaveESD	
選択されている印鑑データの印面イメージを ESD 形式で保存します。	
long SFI_SaveESD(string strFilePath)	
<p>パラメータ strFilePath: 保存する場所</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_IDX_NOT_OPEN(-1): 印鑑セットアップ元ファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SFI_SaveESD("保存する場所")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '印鑑セットアップ元ファイルを開く res = stamp.SFI_IdxOpen("C:\STMP\STMP.ipx", "admin") '操作対象のユーザを選択 res = stamp.SFI_SeekUser("p000019") '選択されているユーザの印鑑イメージを ESD 形式で保存 res = stamp.SFI_SaveESD("C:\test\test.esd") '結果表示 If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber & ":" & stamp.GetLastErrorMessage End If '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_SavePng	
選択されている印鑑データの印面イメージを PNG 形式で保存します。	
long SFI_SavePng(string strFilePath)	
<p>パラメータ strFilePath: 保存する場所</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_IDX_NOT_OPEN(-1): 印鑑セットアップ元ファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SFI_SavePng("保存する場所")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '印鑑セットアップ元ファイルを開く res = stamp.SFI_IdxCOpen("C:\STMP\STMP.ipx", "admin") '操作対象のユーザを選択 res = stamp.SFI_SeekUser("p000019") '選択されているユーザの印鑑イメージを PNG 形式で保存 res = stamp.SFI_SavePNG("C:\test\test.png") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_GetPngData	
印鑑データの PNG 形式のポインタを取得します。引数 nBuffer に NULL を指定した場合にはバッファサイズのみを返します。	
long SFI_GetPngData(long nBuffer)	
<p>パラメータ nBuffer ポインタを格納するバッファアドレス</p> <p>戻り値 long 正常に取得された場合にはバッファに必要なサイズが返ります。 エラーの場合には負の値が返ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 本メソッドは引数にポインタアドレスを想定しているため C++などのポインタが取り扱える開発環境で利用するために作成されています。</p>	
<p>使用例 long nBufferSize = object.SFI_GetPngData(NULL)</p>	
サンプルコード	

電子印鑑 Extension > IStmpDat	メソッド
SFI_GetIntroducePackageSerialNumber	
導入パック シリアル番号を取得します。	
string SFI_GetIntroducePackageSerialNumber(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には導入パック シリアル番号が文字列で戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、SFI_GetLastErrorNumber または SFI_GetLastErrorMessage メソッドを使用します。</p> <p>備考 このメソッドは、下位互換性のために残されています。</p>	
<p>使用例 string strResult = object.SFI_GetIntroducePackageSerialNumber()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '印鑑セットアップ元ファイルを開く res = stamp.SFI_IdxOpen("C:\YSTMP\YSTMP.ipx", "admin") '導入パック シリアル番号を取得 introducePackageSerial = stamp.SFI_GetIntroducePackageSerialNumber() If introducePackageSerial = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End if 'パッケージ シリアル番号（注文番号）を取得 packageSerial = stamp.SFI_GetPackageSerialNumber If packageSerial = "" Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End if '結果表示 WScript.Echo "導入パック シリアル番号:" & introducePackageSerial & vbCrLf & _ "パッケージ シリアル番号(注文番号):" & packageSerial '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_InstallUser	
引数で指定した捺印用印鑑データファイルにセットアップを行います。	
long SFI_InstallUser(string strFilePath, string strPassword)	
<p>パラメータ strFilePath:捺印用印鑑データファイルの場所 strPassword:捺印用印鑑データファイルの開くパスワード</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_IDX_NOT_OPEN(-1):印鑑セットアップ元ファイルが開かれていません E_DSM_CANNOT_OPEN(-3):捺印用印鑑データファイルを開くことができません 詳細なエラー情報を取得するには、SFI_GetLastErrorNumber または SFI_GetLastErrorMessage メソッドを使用します。</p> <p>備考 このメソッドは既定値を使って印鑑セットアップ元ファイル内の選択中のユーザを捺印用印鑑データファイルのルートにセットアップします。</p>	
<p>使用例 long nResult = object.SFI_InstallUser("捺印用印鑑データファイルの場所", "開くパスワード")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '印鑑セットアップ元ファイルを開く res = stamp.SFI_IdxOpen("C:\YSTMP\YSTMP.ipx", "admin") '操作対象のユーザを選択 res = stamp.SFI_SeekUser("<検索するユーザ名>") '選択中のユーザを指定した捺印用印鑑データファイルに追加 res = stamp.SFI_InstallUser("<捺印用印鑑データファイルの場所>", "<開くパスワード>") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetParentGroup	
選択されているユーザが追加されているグループの位置を取得します。	
long SF_GetParentGroup(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には追加されているグループの位置が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 このメソッドで取得される値は GF_Move メソッドや SF_Move メソッドなどで選択するグループを特定する際に利用します。</p>	
<p>使用例 long nResult = object.SF_GetParentGroup()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択されているユーザが追加されているグループの位置を取得 groupPosition = stamp.SF_GetParentGroup() If groupPosition < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザが追加されているグループを選択 res = stamp.GF_Move(groupPosition) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザが追加されているグループのグループ名を取得 groupName = stamp.GF_GetCurrentGroupName() 'エラーの場合設定なしの場合ともに groupName は空文字なので、 'GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザのポジション値を取得 userPosition = stamp.SF_GetCurrentPosition() If userPosition < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択されているユーザに登録されている印鑑数を取得 stampCount = stamp.SF_GetCurrentStampCount() If stampCount < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo "ユーザ名:" & stamp.SF_GetCurrentUserName & vbCrLf & _ "グループポジション値:" & groupPosition & vbCrLf & _ "グループポジション名:" & groupName & vbCrLf & _ "ユーザポジション値:" & userPosition & vbCrLf & _ "ユーザに登録されている印鑑数:" & stampCount '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_IdxClose	
印鑑セットアップ元ファイルを閉じます。	
long SFI_IdxClose(void)	
<p>パラメータ ありません</p> <p>戻り値 ありません</p> <p>備考</p>	
<p>使用例 long nResult = object.SFI_IdxClose()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '印鑑セットアップ元ファイルを開く res stamp.SFI_IdxOpen("C:¥STMP¥STMP.ipx", "admin") If res <> 1 Then Wscript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑セットアップ元ファイルを閉じる stamp.SFI_IdxClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetLastUserFileException	
ユーザ関連の操作で最後に発生したエラーの値を取得します。	
long SF_GetLastUserFileException(void)	
<p>パラメータ ありません</p> <p>戻り値 ユーザ関連の操作で最後に発生したエラー値</p> <p>備考 このメソッドは、ユーザに関連した操作を行った場合に、MFC の CFileException::m_cause が示す列挙値を戻します。</p>	
<p>使用例 long nResult = object.SF_GetLastUserFileException()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SU_SetEnableCheckOutStatus(100) '引数に定義されていない値をセット lastErrorNumber = stamp.GetLastErrorNumber() lastErrorMessage = stamp.GetLastErrorMessgae() sfError = stamp.SF_GetLastUserFileException() If res < 0 Then WScript.Echo "SF_GetLastUserFileException=" & sfError & _ vbCrLf & "SU_SetEnableCheckOutStatus = " & res & _ vbCrLf & "GetLastError~ = " & lastErrorNumber & ";" & lastErrorMessage End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetLastStampFileException	
印鑑データ関連の操作で最後に発生したエラーの値を取得します。	
long SF_GetLastStampFileException(void)	
<p>パラメータ ありません</p> <p>戻り値 印鑑データ関連の操作で最後に発生したエラー値</p> <p>備考 このメソッドは、印鑑データに関連した操作を行った場合に、MFC の CFileException::m_cause が示す列挙値を戻します。</p>	
<p>使用例 long nResult = object.SF_GetLastStampFileException()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの種類を取得 res = stamp.SD_SetAdditionDateFormat(11) '引数に定義されていない値をセット lastErrorNumber = stamp.GetLastErrorNumber() lastErrorMessage = stamp.GetLastErrorMessage() sfError = stamp.SF_GetLastStampFileException() If res < 0 Then WScript.Echo "SF_GetLastStampFileException=" & sfError & _ vbCrLf & "SD_SetAdditionDateFormat = " & res & _ vbCrLf & "GetLastError~ = " & lastErrorNumber & "." & lastErrorMessage End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetLastGroupFileException	
グループ関連の操作で最後に発生したエラーの値を取得します。	
long SF_GetLastGroupFileException(void)	
<p>パラメータ ありません</p> <p>戻り値 グループ関連の操作で最後に発生したエラー値</p> <p>備考 このメソッドは、グループに関連した操作を行った場合に、MFC の CFileException::m_cause が示す列挙値を戻します。</p>	
<p>使用例 long nResult = object.SF_GetLastGroupFileException()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のグループを選択 res = stamp.GF_SeekGroupByUserName("<検索するユーザ名>") '選択しているグループの下に、新しいグループを追加 res = stamp.GF_AppendGroup(stamp.GF_GetCurrentGroupName()) '既存のグループ名で新規追加 lastErrorNumber = stamp.GetLastErrorNumber() lastErrorMessage = stamp.GetLastErrorMessgae() sfError = stamp.SF_GetLastGroupFileException() If res < 0 Then WScript.Echo "SF_GetLastGroupFileException=" & sfError & _ vbCrLf & "GF_AppendGroup = " & res & _ vbCrLf & "GetLastError~ = " & lastErrorNumber & "." & lastErrorMessage End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GetLastErrorNumber	
最後に発生したエラー番号を取得します。	
long GetLastErrorNumber(void)	
<p>パラメータ ありません</p> <p>戻り値 long:最後に発生したエラーの番号が戻ります。</p> <p>備考</p>	
<p>使用例 long nResult = object.GetLastErrorNumber()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを指定しないことで、エラーを発生させる res = stamp.SF_DsmOpen("", "") 'エラー情報取得 If res <> 1 Then 'エラー番号を取得 msg = "ErrorNumber: " & stamp.GetLastErrorNumber() & vbCrLf 'エラーメッセージを取得 msg = msg & "ErrorMessage: " & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo msg 'エラー情報を初期化 stamp.ResetError() '結果表示 WScript.Echo "ErrorNumber: " & stamp.GetLastErrorNumber() & vbCrLf & _ "ErrorMessage: " & stamp.GetLastErrorMessage() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GetLastErrorMessage	
最後に発生したエラーメッセージを取得します。	
long GetLastErrorMessage(void)	
<p>パラメータ ありません</p> <p>戻り値 string:最後に発生したエラーメッセージが戻ります。</p> <p>備考 このメソッドは定義済みのエラーメッセージが取得されます。エラーメッセージが定義されていない場合には「エラーメッセージが定義されていません」が戻されます。</p>	
<p>使用例</p> <pre>string strResult = object.GetLastErrorMessage()</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを指定しないことで、エラーを発生させる res = stamp.SF_DsmOpen("", "") 'エラー情報取得 If res <> 1 Then 'エラー番号を取得 msg = "ErrorNumber: " & stamp.GetLastErrorNumber() & vbCrLf 'エラーメッセージを取得 msg = msg & "ErrorMessage: " & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo msg 'エラー情報を初期化 stamp.ResetError() '結果表示 WScript.Echo "ErrorNumber: " & stamp.GetLastErrorNumber() & vbCrLf & _ "ErrorMessage: " & stamp.GetLastErrorMessage() Set stamp =Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetDsmHeader	
捺印用印鑑データファイルの種類を取得します。	
long GetDsmHeader(string strFilePath)	
<p>パラメータ strFilePath: 捺印用印鑑データファイルの場所</p> <p>戻り値 long 成功した場合次のいずれかの値が戻ります。 DSM_NORMAL(1): 通常の捺印用印鑑データファイル DSM_FREE(2): Free Edition で作成された捺印用印鑑データファイル 失敗した場合には次の値が戻ります。 E_DSM_PATH(-8): 捺印用印鑑データファイルが見つかりません DSM_SUCCESS_FALSE(0): 捺印用印鑑データファイルを特定できません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessag メソッド</p> <p>備考 このメソッドは SF_DsmOpen などでは捺印用印鑑データファイルを開く必要はありません。</p>	
<p>使用例 long nResult = object.SF_GetDsmHeader("捺印用印鑑データファイルの場所")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルの種類を取得 res = stamp.SF_GetDsmHeader("<捺印用印鑑データファイルの場所>") '結果表示 Select case res Case 1 WScript.Echo res & " : 製品版のパソコン決裁で作成された捺印用印鑑データファイル" Case 2 WScript.Echo res & " : パソコン決裁 Free Edition で作成された捺印用印鑑データファイル" Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End Select Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetUserEmailAddress	
ユーザに設定されている電子メールアドレスを取得します。	
string SU_GetUserEmailAddress(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には電子メールアドレスが戻ります。 失敗した場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strResult = object.SU_GetUserEmailAddress()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザの電子メールアドレスを取得 email = stamp.SU_GetUserEmailAddress() "エラーの場合、設定なしの場合ともに email は空文字なので、 "GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザの電子メールアドレスを設定 res = stamp.SU_SetUserEmailAddress("suzuki@test.com") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & res & stamp.GetLastErrorMessage() End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_SetUserEmailAddress	
ユーザに電子メールアドレスを設定します。	
long SU_SetUserEmailAddress(string strEmailAddress)	
<p>パラメータ strEmailAddress:電子メールアドレス</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には次の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません E_DSM_SAMEVALUE_ANOTHERUSER(-24):同じ電子メールアドレスが既に利用されています 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SU_SetUserEmailAddress("電子メールアドレス")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザの電子メールアドレスを取得 email = stamp.SU_GetUserEmailAddress() "エラーの場合、設定なしの場合ともに email は空文字なので、 "GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザの電子メールアドレスを設定 res = stamp.SU_SetUserEmailAddress("suzuki@test.com") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":res =" & res & stamp.GetLastErrorMessage() End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetUserEnableStatus	
ユーザに設定されている「ユーザを無効にする」項目を取得します。	
long SU_GetUserEnableStatus(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 DSM_SUCCESS(1):ユーザは有効です DSM_SUCCESS_FALSE(0):ユーザは無効です 失敗した場合には次の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SU_GetUserEnableStatus()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstampLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザの[ユーザを無効にする]の設定を取得 status = stamp.SU_GetUserEnableStatus() Select Case status Case 1 '有効 '[ユーザを無効にする]を無効にする res = stamp.SU_SetUserEnableStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '選択中のユーザを更新 res = stamp.SF_PutUser() Case 0 '無効 '[ユーザを無効にする]を有効にする res = stamp.SU_SetUserEnableStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End If '選択中のユーザを更新 res = stamp.SF_PutUser() Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessgae() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_SetUserEnableStatus	
ユーザの「ユーザを無効にする」項目を設定します。	
long SU_SetUserEnableStatus(nStatus)	
<p>パラメータ</p> <p>nStatus:</p> <p>1:ユーザを有効にする</p> <p>0:ユーザを無効にする</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS(1)が戻ります。</p> <p>失敗した場合には次の値が戻ります。</p> <p>E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません</p> <p>詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p> <p>設定の変更を捺印用印鑑データに反映させるには、SF_PutUser()を実行して選択されているユーザを更新します。</p>	
<p>使用例</p> <p>long nResult = object.SU_SetUserEnableStatus(1)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択中のユーザの[ユーザを無効にする]の設定を取得 status = stamp.SU_GetUserEnableStatus() Select Case status Case 1 '有効 '[ユーザを無効にする]を無効にする res = stamp.SU_SetUserEnableStatus(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() Case 0 '無効 '[ユーザを無効にする]を有効にする res = stamp.SU_SetUserEnableStatus(1) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() Case Else 'エラー WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetWindowsAccount	
ユーザに設定されている [Windows 認証] 項目を取得します。	
string SU_GetWindowsAccount(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には Windows 認証で利用されるアカウント情報が"ドメイン名¥アカウント名"形式の文字列で戻ります。 失敗した場合や設定されていない場合には空文字が戻ります。 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <pre>string strResult = object.SU_GetWindowsAccount()</pre>	
<p>サンプルコード</p> <pre>Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択しているユーザの[Windows 認証]の設定を取得 WindowsAccount = stamp.SU_GetWindowsAccount() "エラーの場合、設定なしの場合ともに WindowsAccount は空文字なので、 "GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() WScript.Quit End If If WindowsAccount = "" Then '選択しているユーザの[Windows 認証]を設定 res = stamp.SU_SetWindowsAccount("testDomain", "testUser") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() Else '選択しているユーザの[Windows 認証]の設定を削除 res = stamp.SU_RemoveWindowsAccount() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing</pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_GetWindowsAccount	
ユーザに設定されている [Windows 認証] 項目を設定します。	
long SU_SetWindowsAccount(string strDomainName, string strAccountName)	
<p>パラメータ</p> <p>strDomainName: ドメイン名 strAccountName: アカウント名</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には次の値が戻ります。 E_DSM_NOT_OPEN(-1): 捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <p>long nResult = object.SU_SetWindowsAccount("ドメイン名", "アカウント名")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択しているユーザの[Windows 認証]の設定を取得 WindowsAccount = stamp.SU_GetWindowsAccount() "エラーの場合、設定なしの場合ともに WindowsAccount は空文字なので、 "GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() WScript.Quit End If If WindowsAccount = "" Then '選択しているユーザの[Windows 認証]を設定 res = stamp.SU_SetWindowsAccount("testDomain", "testUser") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() Else '選択しているユーザの[Windows 認証]の設定を削除 res = stamp.SU_RemoveWindowsAccount() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SU_RemoveWindowsAccount	
ユーザに設定されている [Windows 認証] を削除します。	
long SU_RemoveWindowsAccount(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には次の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.SU_RemoveWindowsAccount()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象のユーザを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") '選択しているユーザの[Windows 認証]の設定を取得 WindowsAccount = stamp.SU_GetWindowsAccount() "エラーの場合、設定なしの場合ともに WindowsAccount は空文字なので、 "GetLastErrorNumber の値でエラーを判定する If stamp.GetLastErrorNumber < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() WScript.Quit End If If WindowsAccount = "" Then '選択しているユーザの[Windows 認証]を設定 res = stamp.SU_SetWindowsAccount("testDomain", "testUser") If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '選択中のユーザを更新 res = stamp.SF_PutUser() Else '選択しているユーザの[Windows 認証]の設定を削除 res = stamp.SU_RemoveWindowsAccount() If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_GetUserCacheMode	
ユーザー時ファイルの利用状態を取得します。(ユーザー時ファイルを利用した場合には、モジュールを利用してユーザのプロパティ情報を取得する場合に、ユーザが別のアプリケーションで利用している場合でもユーザのプロパティを取得できるようになります)	
long SF_GetUserCacheMode(void)	
<p>パラメータ ありません</p> <p>戻り値 long 次のいずれかの値が戻ります。 DSM_SUCCESS(1):ユーザー時ファイルを利用します 既定値 DSM_SUCCESS_FALSE(0):ユーザー時ファイルを利用しません</p> <p>備考 ユーザー時ファイルを利用しない場合には、ユーザが別のアプリケーションで捺印用印鑑データファイルにログインしている場合に SF_SeekUser メソッドなどでエラーが発生します。</p>	
<p>使用例 long nResult = object.SF_GetUserCacheMode()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") 'ユーザー時ファイルを利用状態を取得 res = stamp.SF_GetUserCacheMode() Select Case res Case 0 '利用しない 'ユーザー時ファイルの利用状態を設定 stamp.SF_SetUserCacheMode(1) WScript.Echo "ユーザー時ファイルを利用するように変更しました。" & res Case 1 '利用する 'ユーザー時ファイルの利用状態を設定 stamp.SF_SetUserCacheMode(0) WScript.Echo "ユーザー時ファイルを利用しないように変更しました。" & res Case Else WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SF_SetUserCacheMode	
ユーザー一時ファイルを利用状態を設定します。（ユーザー一時ファイルを利用した場合には、モジュールを利用してユーザのプロパティ情報を取得する場合に、ユーザが別のアプリケーションで利用している場合でもユーザのプロパティを取得できるようになります）	
void SF_SetUserCacheMode(nCacheMode)	
<p>パラメータ nCacheMode: 1:ユーザー一時ファイルを利用します 0:ユーザー一時ファイルを利用しません</p> <p>戻り値 ありません</p> <p>備考 ユーザー一時ファイルを利用しない場合には、ユーザが別のアプリケーションで捺印用印鑑データファイルにログインしている場合に SF_SeekUser メソッドなどでエラーが発生します。また、本メソッドの引数で設定される値は揮発性データとなりモジュールが生成され、破棄されるまでは有効ですがモジュールを生成した場合には既定値（ユーザー一時ファイルを利用する）が初期値となります。</p>	
<p>使用例 object.SF_SetUserCacheMode(1)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") 'ユーザー一時ファイルを利用状態を取得 res = stamp.SF_GetUserCacheMode() Select Case res Case 0 '利用しない 'ユーザー一時ファイルの利用状態を設定 stamp.SF_SetUserCacheMode(1) WScript.Echo "ユーザー一時ファイルを利用するように変更しました。" & res Case 1 '利用する 'ユーザー一時ファイルの利用状態を設定 stamp.SF_SetUserCacheMode(0) WScript.Echo "ユーザー一時ファイルを利用しないように変更しました。" & res Case Else WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End Select '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
GetEdition	
モジュールのエディションを取得します。	
long GetEdition(void)	
<p>パラメータ ありません</p> <p>戻り値 long 次のいずれかの値が戻ります。 LIB_EDITION_TRIAL(0) : 試用版 LIB_EDITION_STANDARD(1) : 製品版 LIB_EDITION_TRIAL_X64(2) : 64 ビット試用版 LIB_EDITION_STANDARD_X64(3) : 64 ビット製品版</p> <p>備考</p>	
<p>使用例 long nResult = object.GetEdition()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") 'ライブラリモジュールのバージョンを取得 version = stamp.GetVersion() msg = "バージョン:" & version & vbCrLf 'ライブラリモジュールのエディションを取得 edition = stamp.GetEdition() Select Case edition Case 1 '製品版 msg = msg & "エディション:製品版" Case 2 '試用版 msg = msg & "エディション:試用版" Case Else msg = msg & "エディション:不明" End Select '結果表示 WScript.Echo msg set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_GetStampAngle	
印鑑データに設定されている角度を取得します。	
long SD_GetStampAngle(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には設定されている角度（0～364 の値）が戻ります。 失敗した場合には次の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 このメソッドは現在選択中 (SD_GetIndex メソッドで取得される登録インデックスで追加されている) の印鑑データの角度を取得します。 選択中の印鑑を変更するには SD_GetStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_GetStampAngle()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの印影の角度を取得 angle = stamp.SD_GetStampAngle() If angle < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If If angle <> 0 Then '選択中の印鑑データの印影の角度を設定 res = stamp.SD_SetStampAngle(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If Else '選択中の印鑑データの印影の角度を設定 res = stamp.SD_SetStampAngle(15) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessag() End If End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SD_SetStampAngle	
印鑑データに角度を設定します。	
long SD_SetStampAngle(long nAngle)	
<p>パラメータ nAngle:設定する印鑑データの角度（0～364 の値）</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には次の値が戻ります。 E_UNMATCH_VALUE(-22):引数の値が不正です E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 このメソッドは現在選択中 (SD_GetIndex メソッドで取得される登録インデックスで追加されている) の印鑑データの角度を設定します。 選択中の印鑑を変更するには SD_GetStamp メソッドを利用します。設定した値を保存するには、SF_PutStamp メソッドを利用します。</p>	
<p>使用例 long nResult = object.SD_SetStampAngle(15)</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmLib.StmpDat") '捺印用印鑑データファイルを開く res = stamp.SF_DsmOpen("<捺印用印鑑データファイルの場所>", "<開くパスワード>") '操作対象の印鑑データを選択 res = stamp.SF_SeekUser("<検索するユーザ名>") res = stamp.SF_GetStamp(0) '選択中の印鑑データの印影の角度を取得 angle = stamp.SD_GetStampAngle() If angle < 0 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If If angle <> 0 Then '選択中の印鑑データの印影の角度を設定 res = stamp.SD_SetStampAngle(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If Else '選択中の印鑑データの印影の角度を設定 res = stamp.SD_SetStampAngle(15) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If '印鑑データを更新 res = stamp.SF_PutStamp(0) If res <> 1 Then WScript.Echo stamp.GetLastErrorNumber() & ":" & stamp.GetLastErrorMessage() End If End If '捺印用印鑑データファイルを閉じる stamp.SF_DsmClose() Set stamp =Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_SetExtractPath	
SFI_Id x Open メソッドで引数として IPX 形式のファイルが指定された場合に一時的なファイルを展開する場所を指定します。またこのメソッドで指定された値は、揮発性のデータであるためオブジェクトが破棄された場合には既定値に戻ります。	
long SFI_SetExtractPath(string strFilePath)	
<p>パラメータ strFilePath: 展開先のフォルダの場所（終端¥記号が必要です）</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_IDX_EXTRACT_PATH(-29): 展開先のフォルダが不正かアクセス権がありません 詳細なエラー情報を取得するには、SFI_GetLastErrorType または SFI_GetLastErrorMessage メソッドを使用します。</p> <p>備考 このメソッドで指定した値は、SFI_Id x Open メソッドで引数として IPX 形式のファイルが指定された場合に一時的なファイルを展開する場所になります。SFI_Id x Open メソッドを行う前に必ず設定を行ってください。また指定されたフォルダ内には一時的なファイルが書き込まれるため、書き込み以上の権限が必要になります。</p>	
<p>使用例 long nResult = object.SFI_SetExtractPath("展開先のフォルダの場所")</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '印鑑セットアップ元ファイル（IPX 形式のみ）の内容を展開する一時フォルダの場所を設定 res = stamp.SFI_SetExtractPath("C:¥temp¥") '印鑑セットアップ元ファイル（IPX 形式のみ）の内容を展開する一時フォルダの場所を取得 path = stamp.SFI_GetExtractPath() '結果表示 WScript.Echo "IPX ファイル展開一時フォルダの場所:" & path Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
SFI_GetExtractPath	
印鑑セットアップ元ファイル（IPX 形式のみ）の内容を展開する一時フォルダの場所を取得します。	
string SFI_SetExtractPath(void)	
<p>パラメータ ありません</p> <p>戻り値 strFilePath:展開先のフォルダの場所</p> <p>備考</p>	
<p>使用例 string strResult = object.SFI_GetExtractPath()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '印鑑セットアップ元ファイル（IPX 形式のみ）の内容を展開する一時フォルダの場所を設定 res = stamp.SFI_SetExtractPath("C:¥temp¥") '印鑑セットアップ元ファイル（IPX 形式のみ）の内容を展開する一時フォルダの場所を取得 path = stamp.SFI_GetExtractPath() '結果表示 WScript.Echo "IPX ファイル展開一時フォルダの場所:" & path Set stamp = Nothing </pre>	

電子印鑑 Extension > IStmpDat	メソッド
ResetError	
内部エラー情報を初期化します。	
void ResetError(void)	
<p>パラメータ ありません</p> <p>戻り値 ありません</p> <p>備考 初期化後の GetLastErrorNumber メソッドは DSM_SUCCESS(1)を GetLastErrorMessage メソッドは"エラーはありません"を戻します。</p>	
<p>使用例 object.ResetError()</p>	
<p>サンプルコード</p> <pre> Set stamp = CreateObject("DstmpLib.StmpDat") '捺印用印鑑データファイルを指定しないことで、エラーを発生させる res = stamp.SF_DsmOpen("", "") 'エラー情報取得 If res <> 1 Then 'エラー番号を取得 msg = "ErrorNumber: " & stamp.GetLastErrorNumber() & vbCrLf 'エラーメッセージを取得 msg = msg & "ErrorMessage: " & stamp.GetLastErrorMessage() End If '結果表示 WScript.Echo msg 'エラー情報を初期化 stamp.ResetError() '結果表示 WScript.Echo "ErrorNumber: " & stamp.GetLastErrorNumber() & vbCrLf & _ "ErrorMessage: " & stamp.GetLastErrorMessage() Set stamp =Nothing </pre>	

10 IStmpLog インタフェース

電子印鑑 Extension > IStmpLog	メソッド
GetEdition	
DstmpLib ライブラリのエディションを取得します。	
long GetEdition(void)	
<p>パラメータ ありません</p> <p>戻り値 long DstmpLib ライブラリのエディションが戻ります。 LIB_EDITION_TRIAL(0) : 試用版 LIB_EDITION_STANDARD(1) : 製品版 LIB_EDITION_TRIAL_X64(2) : 64 ビット試用版 LIB_EDITION_STANDARD_X64(3) : 64 ビット製品版</p> <p>備考 製品のエディションは、ファイルのプロパティのバージョン情報の製品名から確認することができます。</p>	
<p>使用例 long nResult = object.GetEdition()</p>	
<p>サンプルコード</p> <pre> Set objDstmpLib = CreateObject("DstmpLib.StmpLog") nEdition = objDstmpLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmpLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmpLib.LF_GetLogCount() 'ログの総数 nResult = objDstmpLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmpLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmpLib.LC_GetImpressTime() '取得フラグを確認する If objDstmpLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmpLib.LC_GetStampSerialNumber() End If objDstmpLib.LF_MoveNext() Loop 'objDstmpLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmpLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmpLib.GetLastErrorNumber() & " " & objDstmpLib.GetLastErrorMessgae() End If Set objDstmpLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_MoveFirst	
捺印履歴ファイルに追加されている最初の履歴を選択します。	
long LF_MoveFirst(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS (1) が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 最初の履歴とは捺印ログに追加されている先頭の履歴になります。</p>	
<p>使用例 long nResult = object.LF_MoveFirst()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_MoveLast	
捺印履歴ファイルに追加されている最後の履歴を選択します。	
long LF_MoveLast(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS (1) が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 最後の履歴とは捺印ログに追加されている末尾の履歴になります。</p>	
<p>使用例 long nResult = object.LF_MoveLast()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_MoveNext	
選択されている捺印履歴の次履歴を選択します。	
long LF_MoveNext(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS (1) が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_MoveNext()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_MovePrevious	
選択されている捺印履歴の前履歴を選択します。	
long LF_MovePrevious(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS (1) が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_MovePrevious()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_GetLogCount	
捺印履歴データの総履歴数を取得します。	
long LF_GetLogCount(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS (1) が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_GetLogCount()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_GetStampSerialNumberFlag	
捺印履歴データの印鑑シリアルログ収集状態を取得します。	
long LF_GetStampSerialNumberFlag(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_GetStampSerialNumberFlag()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_SetStampSerialNumberFlag	
捺印履歴データの印鑑シリアルログ収集状態を設定します。	
long LF_SetStampSerialNumberFlag(long nFlag)	
<p>パラメータ</p> <p>nFlag: 次のいずれかの値を指定します。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <p>long nResult = object.LF_SetStampSerialNumberFlag(1)</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_GetAdditionTextFlag	
捺印履歴データの印面テキストのログ収集状態を取得します。	
long LF_GetAdditionTextFlag(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_GetAdditionTextFlag()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_SetAdditionTextFlag	
捺印履歴データの印面テキストのログ収集状態を設定します。	
long LF_SetAdditionTextFlag(long nFlag)	
<p>パラメータ</p> <p>nFlag: 次のいずれかの値を指定します。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <p>long nResult = object.LF_SetAdditionTextFlag(1)</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_GetAdditionDateFlag	
捺印履歴データの印面日時のログ収集状態を取得します。	
long LF_GetAdditionDateFlag(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_GetAdditionDateFlag()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_SetAdditionDateFlag	
捺印履歴データの印面日時のログ収集状態を設定します。	
long LF_SetAdditionDateFlag(long nFlag)	
<p>パラメータ</p> <p>nFlag: 次のいずれかの値を指定します。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <p>long nResult = object.LF_SetAdditionDateFlag(1)</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_GetImpressCountFlag	
捺印履歴データの捺印カウンタのログ収集状態を取得します。	
long LF_GetImpressCountFlag(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_GetImpressCountFlag()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_SetImpressCountFlag	
捺印履歴データの捺印カウンタのログ収集状態を設定します。	
long LF_SetImpressCountFlag(long nFlag)	
<p>パラメータ</p> <p>nFlag: 次のいずれかの値を指定します。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <pre>long nResult = object.LF_SetImpressCountFlag(1)</pre>	
<p>サンプルコード</p> <pre>Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing</pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_GetGroupNamePathFlag	
捺印履歴データのグループパス名のログ収集状態を取得します。	
long LF_GetGroupNamePathFlag(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_GetGroupNamePathFlag()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_SetGroupNamePathFlag	
捺印履歴データのグループパス名のログ収集状態を設定します。	
long LF_SetGroupNamePathFlag(long nFlag)	
<p>パラメータ</p> <p>nFlag: 次のいずれかの値を指定します。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <pre>long nResult = object.LF_SetGroupNamePathFlag(1)</pre>	
<p>サンプルコード</p> <pre>Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing</pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_GetUserIDFlag	
捺印履歴データのユーザ ID のログ収集状態を取得します。	
long LF_GetUserIDFlag(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_GetUserIDFlag()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_SetUserIDFlag	
捺印履歴データのユーザ ID のログ収集状態を設定します。	
long LF_SetUserIDFlag(long nFlag)	
<p>パラメータ</p> <p>nFlag: 次のいずれかの値を指定します。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <pre>long nResult = object.LF_SetUserIDFlag(1)</pre>	
<p>サンプルコード</p> <pre>Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing</pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_GetStampIDFlag	
捺印履歴データの印鑑 CID のログ収集状態を取得します。	
long LF_GetStampIDFlag(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_GetStampIDFlag()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_SetStampIDFlag	
捺印履歴データの印鑑 CID のログ収集状態を設定します。	
long LF_SetStampIDFlag(long nFlag)	
<p>パラメータ</p> <p>nFlag: 次のいずれかの値を指定します。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <p>long nResult = object.LF_SetStampIDFlag(1)</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_GetImpressIDFlag	
捺印履歴データの捺印 ID のログ収集状態を取得します。	
long LF_GetImpressIDFlag(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_GetImpressIDFlag()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_SetImpressIDFlag	
捺印履歴データの捺印 ID のログ収集状態を設定します。	
long LF_SetImpressIDFlag(long nFlag)	
<p>パラメータ</p> <p>nFlag: 次のいずれかの値を指定します。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <p>long nResult = object.LF_SetImpressIDFlag(1)</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_GetDocFileNameFlag	
捺印履歴データの文書ファイル名のログ収集状態を取得します。	
long LF_GetDocFileNameFlag(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_GetDocFileNameFlag()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_SetDocFileNameFlag	
捺印履歴データの文書ファイル名のログ収集状態を設定します。	
long LF_SetDocFileNameFlag(long nFlag)	
<p>パラメータ</p> <p>nFlag: 次のいずれかの値を指定します。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <p>long nResult = object.LF_SetDocFileNameFlag(1)</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_GetDocFileTitleFlag	
捺印履歴データの文書タイトルのログ収集状態を取得します。	
long LF_GetDocFileTitleFlag(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_GetDocFileTitleFlag()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_SetDocFileTitleFlag	
捺印履歴データの文書タイトルのログ収集状態を設定します。	
long LF_SetDocFileTitleFlag(long nFlag)	
<p>パラメータ</p> <p>nFlag: 次のいずれかの値を指定します。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <p>long nResult = object.LF_SetDocFileTitleFlag(1)</p>	
<p>サンプルコード</p> <pre> Set objDstmplib = CreateObject("Dstmplib.StmpLog") nEdition = objDstmplib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmplib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmplib.LF_GetLogCount() 'ログの総数 nResult = objDstmplib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmplib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmplib.LC_GetImpressTime() '取得フラグを確認する If objDstmplib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmplib.LC_GetStampSerialNumber() End If objDstmplib.LF_MoveNext() Loop 'objDstmplib.LF_DeleteAllLog() 'すべてのログの削除 objDstmplib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmplib.GetLastErrorNumber() & " " & objDstmplib.GetLastErrorMessage() End If Set objDstmplib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_GetDocNumberFlag	
捺印履歴データの文書番号のログ収集状態を取得します。	
long LF_GetDocNumberFlag(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_GetDocNumberFlag()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_SetDocNumberFlag	
捺印履歴データの文書番号のログ収集状態を設定します。	
long LF_SetDocNumberFlag(long nFlag)	
<p>パラメータ</p> <p>nFlag: 次のいずれかの値を指定します。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <p>long nResult = object.LF_SetDocNumberFlag(1)</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_GetDocItemFlag	
捺印履歴データの承認項目のログ収集状態を取得します。	
long LF_GetDocItemFlag(long nIndex)	
<p>パラメータ nIndex 承認項目のインデックス (0～4)</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_GetDocItemFlag()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_SetDocItemFlag	
捺印履歴データの承認項目のログ収集状態を設定します。	
long LF_SetDocItemFlag(long nIndex, long nFlag)	
<p>パラメータ</p> <p>nIndex: 承認項目のインデックス (0~4)</p> <p>nFlag: 次のいずれかの値を指定します。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <pre>long nResult = object.LF_SetDocItemFlag(1)</pre>	
<p>サンプルコード</p> <pre>Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing</pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_GetDomainNameFlag	
捺印履歴データのドメイン名のログ収集状態を取得します。	
long LF_GetDomainNameFlag(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_GetDomainNameFlag()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_SetDomainNameFlag	
捺印履歴データのドメイン名のログ収集状態を設定します。	
long LF_SetDomainNameFlag(long nFlag)	
<p>パラメータ</p> <p>nFlag: 次のいずれかの値を指定します。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <pre>long nResult = object.LF_SetDomainNameFlag(1)</pre>	
<p>サンプルコード</p> <pre>Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing</pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_GetComputerNameFlag	
捺印履歴データのコンピュータ名のログ収集状態を取得します。	
long LF_GetComputerNameFlag(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_GetComputerNameFlag()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_SetComputerNameFlag	
捺印履歴データのコンピュータ名のログ収集状態を設定します。	
long LF_SetComputerNameFlag(long nFlag)	
<p>パラメータ</p> <p>nFlag: 次のいずれかの値を指定します。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <p>long nResult = object.LF_SetComputerNameFlag(1)</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_GetLoginNameFlag	
捺印履歴データのログイン名のログ収集状態を取得します。	
long LF_GetLoginNameFlag(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には次のいずれかの値が戻ります。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_GetLoginNameFlag()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_SetLoginNameFlag	
Set objDstmpLib = CreateObject("DstmpLib.StmpLog") nEdition = objDstmpLib.GetEdition() ' 捺印用印鑑データファイルを開く ResetError() ' エラー情報を初期化 ' 捺印用印鑑データファイルを開く nResult = objDstmpLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen:" & nResu	
long LF_SetLoginNameFlag(long nFlag)	
<p>パラメータ</p> nFlag: 次のいずれかの値を指定します。 DSM_SUCCESS (1) : 設定あり DSM_SUCCESS_FALSE (0) : 設定なし <p>戻り値</p> long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。 <p>備考</p>	
<p>使用例</p> long nResult = object.LF_SetLoginNameFlag(1)	
<p>サンプルコード</p> aaaaaa	

電子印鑑 Extension > IStmpLog	メソッド
LF_IsBOF	
選択されているユーザが最初のユーザであるか判断します	
long LF_IsBOF(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS_FALSE(0)または DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 最初のユーザとは管理ツールなどを使って追加を行った最初のユーザになります。</p>	
<p>使用例 long nResult = object.LF_IsBOF()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_IsEOF	
選択されているユーザが末尾のユーザであるか判断します	
long LF_IsEOF(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS_FALSE(0)または DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 末尾のユーザとは管理ツールなどを使って追加を行った最近のユーザになります。</p>	
<p>使用例 long nResult = object.LF_IsEOF()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LC_GetImpressCount	
捺印履歴の捺印カウンタを取得します。	
long LC_GetImpressCount(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には選択されている捺印履歴の捺印カウンタの値が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 捺印カウンタの履歴を取得するには LF_SetImpressCountFlag メソッドなどで捺印カウンタの履歴が収集されている必要があります。</p>	
<p>使用例 long nResult = object.LC_GetImpressCount()</p>	
<p>サンプルコード</p> <pre> Set objDstmpLib = CreateObject("DstmpLib.StmpLog") nEdition = objDstmpLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmpLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmpLib.LF_GetLogCount() 'ログの総数 nResult = objDstmpLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmpLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmpLib.LC_GetImpressTime() '取得フラグを確認する If objDstmpLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmpLib.LC_GetStampSerialNumber() End If objDstmpLib.LF_MoveNext() Loop 'objDstmpLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmpLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmpLib.GetLastErrorNumber() & " " & objDstmpLib.GetLastErrorMessgae() End If Set objDstmpLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LC_GetImpressID	
捺印履歴の捺印 ID を取得します。	
long LC_GetImpressID(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には選択されている捺印履歴の捺印 ID の値が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 捺印 ID の履歴を取得するには LF_SetImpressIDFlag メソッドなどで捺印 ID の履歴が収集されている必要があります。</p>	
<p>使用例 long nResult = object.LC_GetImpressID()</p>	
<p>サンプルコード</p> <pre> Set objDstmpLib = CreateObject("DstmpLib.StmpLog") nEdition = objDstmpLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmpLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmpLib.LF_GetLogCount() 'ログの総数 nResult = objDstmpLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmpLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmpLib.LC_GetImpressTime() '取得フラグを確認する If objDstmpLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmpLib.LC_GetStampSerialNumber() End If objDstmpLib.LF_MoveNext() Loop 'objDstmpLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmpLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmpLib.GetLastErrorNumber() & " " & objDstmpLib.GetLastErrorMessage() End If Set objDstmpLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LC_GetAdditionText	
捺印履歴の印面テキストを取得します。	
string LC_GetAdditionText(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には選択されている捺印履歴の印鑑シリアル値が戻ります。 失敗した場合には空文字の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 印面テキストの履歴を取得するには LF_SetAdditionTextFlag メソッドなどで印面テキストの履歴が収集されている必要があります。</p>	
<p>使用例 string strResult = object.LC_GetAdditionText()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LC_GetStampSerialNumber	
捺印履歴の印鑑シリアルを取得します。	
string LC_GetStampSerialNumber(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には選択されている捺印履歴の印鑑シリアルの値が戻ります。 失敗した場合には空文字の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑シリアルの履歴を取得するには LF_SetStampSerialNumberFlag メソッドなどで印鑑シリアルの履歴が収集されている必要があります。</p>	
<p>使用例 string strResult = object.LC_GetStampSerialNumber()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LC_GetStampID	
捺印履歴の印鑑 ID を取得します。	
long LC_GetStampID(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には選択されている捺印履歴の印鑑 ID の値が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 印鑑 ID の履歴を取得するには LF_SetStampIDFlag メソッドなどで印鑑 ID の履歴が収集されている必要があります。</p>	
<p>使用例 long nResult = object.LC_GetStampID()</p>	
<p>サンプルコード</p> <pre> Set objDstmpLib = CreateObject("DstmpLib.StmpLog") nEdition = objDstmpLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmpLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmpLib.LF_GetLogCount() 'ログの総数 nResult = objDstmpLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmpLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmpLib.LC_GetImpressTime() '取得フラグを確認する If objDstmpLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmpLib.LC_GetStampSerialNumber() End If objDstmpLib.LF_MoveNext() Loop 'objDstmpLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmpLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmpLib.GetLastErrorNumber() & " " & objDstmpLib.GetLastErrorMessage() End If Set objDstmpLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LC_GetGroupNamePath	
捺印履歴のグループパス名を取得します。	
string LC_GetGroupNamePath(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には選択されている捺印履歴のグループパス名の値が戻ります。 失敗した場合には空文字の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 グループパス名の履歴を取得するには LF_SetGroupNamePathFlag メソッドなどでグループパス名の履歴が収集されている必要があります。</p>	
<p>使用例 string strResult = object.LC_GetGroupNamePath()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LC_GetUserName	
捺印履歴のユーザ名を取得します。	
string LC_GetUserName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には選択されている捺印履歴のユーザ名の値が戻ります。 失敗した場合には空文字の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strResult = object.LC_GetUserName()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LC_GetUserID	
捺印履歴のユーザ CID を取得します。	
long LC_GetUserID(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には選択されている捺印履歴のユーザ CID の値が戻ります。 失敗した場合には以下の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考 ユーザ CID の履歴を取得するには LF_SetUserIDFlag メソッドなどでユーザ CID の履歴が収集されている必要があります。</p>	
<p>使用例 long nResult = object.LC_GetUserID()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LC_GetDocFileName	
捺印履歴の文書ファイル名を取得します。	
string LC_GetDocFileName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には選択されている捺印履歴の文書ファイル名の値が戻ります。 失敗した場合には空文字の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 文書ファイル名の履歴を取得するには LF_SetDocFileNameFlag メソッドなどで文書ファイル名の履歴が収集されている必要があります。</p>	
<p>使用例 string strResult = object.LC_GetDocFileName()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LC_GetDocTitle	
捺印履歴の文書タイトルを取得します。	
string LC_GetDocTitle(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には選択されている捺印履歴の文書タイトルの値が戻ります。 失敗した場合には空文字の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 文書タイトルの履歴を取得するには LF_SetDocTitleFlag メソッドなどで文書タイトルの履歴が収集されている必要があります。</p>	
<p>使用例 string strResult = object.LC_GetDocTitle()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LC_GetDocNumber	
捺印履歴の文書番号を取得します。	
string LC_GetDocNumber(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には選択されている捺印履歴の文書番号の値が戻ります。 失敗した場合には空文字の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 文書番号の履歴を取得するには LF_SetDocNumberFlag メソッドなどで文書番号の履歴が収集されている必要があります。</p>	
<p>使用例 string strResult = object.LC_GetDocNumber()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LC_GetDocItem	
捺印履歴の承認項目を取得します。	
string LC_GetDocItem(long nIndex)	
<p>パラメータ nIndex: 承認項目のインデックス (0～4)</p> <p>戻り値 string 成功した場合には選択されている捺印履歴の承認項目の値が戻ります。 失敗した場合には空文字の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 承認項目の履歴を取得するには LF_SetDocItemFlag メソッドなどで承認項目の履歴が収集されている必要があります。</p>	
<p>使用例 string strResult = object.LC_GetDocItem()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LC_GetDomainName	
捺印履歴のドメイン名を取得します。	
string LC_GetDomainName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には選択されている捺印履歴のドメイン名の値が戻ります。 失敗した場合には空文字の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 ドメイン名の履歴を取得するには LF_SetDomainNameFlag メソッドなどでドメイン名の履歴が収集されている必要があります。</p>	
<p>使用例 string strResult = object.LC_GetDomainName()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LC_GetComputerName	
捺印履歴のコンピュータ名を取得します。	
string LC_GetComputerName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には選択されている捺印履歴のコンピュータ名の値が戻ります。 失敗した場合には空文字の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 コンピュータ名の履歴を取得するには LF_SetComputerNameFlag メソッドなどでコンピュータ名の履歴が収集されている必要があります。</p>	
<p>使用例 string strResult = object.LC_GetComputerName()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LC_GetLoginName	
捺印履歴のログイン名を取得します。	
string LC_GetLoginName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には選択されている捺印履歴のログイン名の値が戻ります。 失敗した場合には空文字の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 ログイン名の履歴を取得するには LF_SetLoginNameFlag メソッドなどでログイン名の履歴が収集されている必要があります。取得されるログイン名はパソコン決裁を利用したコンピュータにログインしているアカウント名（Windows にログオンしているユーザ名）になります。</p>	
<p>使用例 string strResult = object.LC_GetLoginName()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_DsmOpen	
引数で指定した場所の捺印用印鑑データファイルを開きます。	
long LF_DsmOpen(string strFilePath, string strPassword)	
<p>パラメータ</p> <p>strFilePath:捺印用印鑑データファイルへのパス文字列 strPassword:開くパスワード</p> <p>戻り値</p> <p>long</p> <p>成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には以下の値が戻ります。 DSM_SUCCESS_READONLY(-21):読み取り専用で開かれています E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例</p> <p>long nResult = object.LF_DsmOpen("捺印用印鑑データファイルの場所", "開くパスワード")</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LC_GetImpressTime	
捺印履歴の捺印日時を取得します。	
string LC_GetImpressTime(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には選択されている捺印履歴の捺印日時の値が戻ります。 失敗した場合には空文字の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 string strResult = object.LC_GetImpressTime()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessage() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LC_GetAdditionDate	
捺印履歴の印面日時を取得します。	
string LC_GetAdditionDate(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には選択されている捺印履歴の印面日時の値が戻ります。 失敗した場合には空文字の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考 印面日時の履歴を取得するには LF_SetAdditionDateFlag メソッドなどで印面日時の履歴が収集されている必要があります。</p>	
<p>使用例 string strResult = object.LC_GetAdditionDate()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_DeleteAllLog	
捺印履歴をすべて削除します。	
long LF_DeleteAllLog(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には次の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_DeleteAllLog()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_DeleteLog	
選択されている捺印履歴を削除します。	
long LF_DeleteLog(void)	
<p>パラメータ ありません</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には次の値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessgae メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_DeleteLog()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") nEdition = objDstmLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmLib.LF_GetLogCount() 'ログの総数 nResult = objDstmLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmLib.LC_GetImpressTime() '取得フラグを確認する If objDstmLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmLib.LC_GetStampSerialNumber() End If objDstmLib.LF_MoveNext() Loop 'objDstmLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmLib.GetLastErrorNumber() & " " & objDstmLib.GetLastErrorMessgae() End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_DsmOpen	
開いている捺印用印鑑データファイルを閉じます。	
void LF_DsmClose(void)	
<p>パラメータ ありません</p> <p>戻り値 ありません</p> <p>備考</p>	
<p>使用例 object.LF_DsmClose()</p>	
<p>サンプルコード</p> <pre> Set objDstmpLib = CreateObject("DstmpLib.StmpLog") nEdition = objDstmpLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmpLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmpLib.LF_GetLogCount() 'ログの総数 nResult = objDstmpLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmpLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmpLib.LC_GetImpressTime() '取得フラグを確認する If objDstmpLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmpLib.LC_GetStampSerialNumber() End If objDstmpLib.LF_MoveNext() Loop 'objDstmpLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmpLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmpLib.GetLastErrorNumber() & " " & objDstmpLib.GetLastErrorMessage() End If Set objDstmpLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
GetVersion	
モジュールのバージョン情報を文字列で取得します	
string GetVersion(void)	
<p>パラメータ ありません</p> <p>戻り値 string 成功した場合には 0.00.000.0000 形式で DstmpLib ライブラリのバージョン情報が文字列で戻されます。</p> <p>備考</p>	
<p>使用例 string strVersion = object.GetVersion()</p>	
<p>サンプルコード</p> <pre> Set objDstmpLib = CreateObject("DstmpLib.StmpLog") nEdition = objDstmpLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmpLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmpLib.LF_GetLogCount() 'ログの総数 nResult = objDstmpLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmpLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmpLib.LC_GetImpressTime() '取得フラグを確認する If objDstmpLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmpLib.LC_GetStampSerialNumber() End If objDstmpLib.LF_MoveNext() Loop 'objDstmpLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmpLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmpLib.GetLastErrorNumber() & " " & objDstmpLib.GetLastErrorMessgae() End If Set objDstmpLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
ResetError	
GetLastErrorNumber メソッドおよび GetLastErrorMessage メソッドで利用する内部エラー変数を初期化します。	
void ResetError(void)	
<p>パラメータ ありません</p> <p>戻り値 ありません</p> <p>備考 初期化後の GetLastErrorNumber メソッドは DSM_SUCCESS(1)を GetLastErrorMessage は"エラーはありません"を戻します。</p>	
<p>使用例 object.ResetError()</p>	
<p>サンプルコード</p> <pre> Set objDstmpLib = CreateObject("DstmpLib.StmpLog") nEdition = objDstmpLib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmpLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmpLib.LF_GetLogCount() 'ログの総数 nResult = objDstmpLib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmpLib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmpLib.LC_GetImpressTime() '取得フラグを確認する If objDstmpLib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmpLib.LC_GetStampSerialNumber() End If objDstmpLib.LF_MoveNext() Loop 'objDstmpLib.LF_DeleteAllLog() 'すべてのログの削除 objDstmpLib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmpLib.GetLastErrorNumber() & " " & objDstmpLib.GetLastErrorMessage() End If Set objDstmpLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
GetLastErrorNumber	
IStmpLog インタフェイスで発生した最終エラー番号を取得します。	
long GetLastErrorNumber(void)	
<p>パラメータ ありません</p> <p>戻り値 long 最終エラー番号が戻ります。</p> <p>備考</p>	
<p>使用例 long nResult = object.GetLastErrorNumber()</p>	
<p>サンプルコード</p> <pre> Set objDstmplib = CreateObject("Dstmplib.StmpLog") nEdition = objDstmplib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmplib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmplib.LF_GetLogCount() 'ログの総数 nResult = objDstmplib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmplib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmplib.LC_GetImpressTime() '取得フラグを確認する If objDstmplib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmplib.LC_GetStampSerialNumber() End If objDstmplib.LF_MoveNext() Loop 'objDstmplib.LF_DeleteAllLog() 'すべてのログの削除 objDstmplib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmplib.GetLastErrorNumber() & " " & objDstmplib.GetLastErrorMessgae() End If Set objDstmplib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
GetLastErrorMessgae	
IStmpLog インタフェイスで発生した最終エラーメッセージを取得します。	
long GetLastErrorMessgae(void)	
<p>パラメータ ありません</p> <p>戻り値 string 最終エラーメッセージが戻ります。</p> <p>備考</p>	
<p>使用例</p> <pre>string strResult = object.GetLastErrorMessgae()</pre>	
<p>サンプルコード</p> <pre>Set objDstmplib = CreateObject("Dstmplib.StmpLog") nEdition = objDstmplib.GetEdition() '捺印用印鑑データファイルを開く ResetError() 'エラー情報を初期化 '捺印用印鑑データファイルを開く nResult = objDstmplib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>%STMPDAT.DSM", "<開くパスワード>") msgbox "LF_DsmOpen::" & nResult If nResult = 1 Then nLogCount = objDstmplib.LF_GetLogCount() 'ログの総数 nResult = objDstmplib.LF_MoveFirst() 'カレントを先頭レコードへ Do while objDstmplib.LF_IsEOF = False 'カレントレコードを先頭に移動 '捺印ログの保存値の取得 dtImpressDate = objDstmplib.LC_GetImpressTime() '取得フラグを確認する If objDstmplib.LF_GetStampSerialNumberFlag() = 1 Then 'ログの保存値を取得 strStampSerialNumber = objDstmplib.LC_GetStampSerialNumber() End If objDstmplib.LF_MoveNext() Loop 'objDstmplib.LF_DeleteAllLog() 'すべてのログの削除 objDstmplib.LF_DsmClose() '捺印用印鑑データファイルを閉じる Else 'エラー情報の取得 msgbox objDstmplib.GetLastErrorNumber() & " " & objDstmplib.GetLastErrorMessgae() End If Set objDstmplib = Nothing</pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_GetCurrentLogFileName	
現在、選択されている捺印履歴ファイル（拡張子.LGI）の名前を取得します。	
string LF_GetCurrentLogFileName(void)	
<p>パラメータ ありません</p> <p>戻り値 string 捺印履歴ファイルのファイル名</p> <p>備考 管理ツールなどで、捺印履歴の保存期間を変更していない場合には既定値（STMPDAT.LGI）が戻されます。</p>	
<p>使用例 string strResult = object.LF_GetCurrentLogFileName()</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") If nResult = 1 Then strCurrentLogFileName = objDstmLib.LF_GetCurrentLogFileName() '現在の履歴ファイルの名前 nLogFileCount = objDstmLib.LF_GetLogFileCount() 'ファイル数を取得 If nLogFileCount > 1 Then For nIndex = 0 To nLogFileCount strLogFileName = objDstmLib.LF_GetLogFileName(nIndex) objDstmLib.LF_SetCurrentLogFile(nIndex) Next End If End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_GetLogFileCount	
現在、保存されている捺印履歴ファイルの数を取得します。	
long LF_GetLogFileCount(void)	
<p>パラメータ ありません</p> <p>戻り値 long 捺印履歴ファイルのファイル数</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_GetLogFileCount()</p>	
<p>サンプルコード</p> <pre> Set objDstmplib = CreateObject("Dstmplib.StmpLog") '捺印用印鑑データファイルを開く nResult = objDstmplib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") If nResult = 1 Then strCurrentLogFileName = objDstmplib.LF_GetCurrentLogFileName() '現在の履歴ファイルの名前 nLogFileCount = objDstmplib.LF_GetLogFileCount() 'ファイル数を取得 If nLogFileCount > 1 Then For nIndex = 0 To nLogFileCount strLogFileName = objDstmplib.LF_GetLogFileName(nIndex) objDstmplib.LF_SetCurrentLogFile(nIndex) Next End If End If Set objDstmplib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_GetLogFileName	
引数で指定されたインデックスの捺印履歴ファイルの名前を取得します。	
string LF_GetLogFileName(long nIndex)	
<p>パラメータ nIndex:取得する捺印履歴ファイルのインデックス</p> <p>戻り値 string 捺印履歴ファイルのファイル名</p> <p>備考</p>	
<p>使用例 string strResult = object.LF_GetLogFileName(0)</p>	
<p>サンプルコード</p> <pre> Set objDstmLib = CreateObject("DstmLib.StmpLog") '捺印用印鑑データファイルを開く nResult = objDstmLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") If nResult = 1 Then strCurrentLogFileName = objDstmLib.LF_GetCurrentLogFileName() '現在の履歴ファイルの名前 nLogFileCount = objDstmLib.LF_GetLogFileCount() 'ファイル数を取得 If nLogFileCount > 1 Then For nIndex = 0 To nLogFileCount strLogFileName = objDstmLib.LF_GetLogFileName(nIndex) objDstmLib.LF_SetCurrentLogFile(nIndex) Next End If End If Set objDstmLib = Nothing </pre>	

電子印鑑 Extension > IStmpLog	メソッド
LF_SetCurrentLogFile	
引数で指定されたインデックスの捺印履歴ファイルの名前を取得します。	
long LF_SetCurrentLogFile(long nIndex)	
<p>パラメータ nIndex:設定する捺印履歴ファイルのインデックス</p> <p>戻り値 long 成功した場合には DSM_SUCCESS(1)が戻ります。 失敗した場合には次のいずれかの値が戻ります。 E_DSM_NOT_OPEN(-1):捺印用印鑑データファイルが開かれていません E_STFV3LIB (-4) :ライブラリエラー 詳細なエラー情報を取得するには、GetLastErrorNumber または GetLastErrorMessage メソッドを使用します。</p> <p>備考</p>	
<p>使用例 long nResult = object.LF_SetCurrentLogFile(0)</p>	
<p>サンプルコード</p> <pre> Set objDstmplLib = CreateObject("DstmplLib.StmpLog") '捺印用印鑑データファイルを開く nResult = objDstmplLib.LF_DsmOpen("<捺印用印鑑データファイルへのパス>¥STMPDAT.DSM", "<開くパスワード>") If nResult = 1 Then strCurrentLogFileName = objDstmplLib.LF_GetCurrentLogFileName() '現在の履歴ファイルの名前 nLogFileCount = objDstmplLib.LF_GetLogFileCount() 'ファイル数を取得 If nLogFileCount > 1 Then For nIndex = 0 To nLogFileCount strLogFileName = objDstmplLib.LF_GetLogFileName(nIndex) objDstmplLib.LF_SetCurrentLogFile(nIndex) Next End If End If Set objDstmplLib = Nothing </pre>	

11 戻り値定数一覧

定数	値	内容と戻されるメソッド
DSM_SUCCESS	1	成功 エラーなし SelectStamp/Login など
DSM_SUCCESS_FALSE	0	失敗 エラーあり SF_DsmOpen など
E_DSM_NOT_OPEN	-1	捺印用印鑑データファイルが開かれていません SelectStamp など
E_GROUP_NOT_FOUND	-2	グループが見つかりません GF_AppendGroup のみ
E_DSM_CANNOT_OPEN	-3	捺印用印鑑データファイルを開くことができません Login など
E_STFV3LIB	-4	ライブラリ内部でエラーが発生しました SelectStamp など
E_USER_NOT_FOUND	-5	ユーザが見つかりません Login など
E_UNMATCH_PASSWORD	-6	パスワードが一致しません Login/SF_DsmOpen など
E_NOT_CERTIFICATION	-7	ログイン認証が行われていません GetStampCount など
E_DSM_PATH	-8	指定された捺印用印鑑データファイルのパスに問題があります Login メソッドなど
E_SAVE_JPG	-9	JPEG 画像ファイルの出力に失敗しました SaveJpg/GetJpgData のみ
E_SAVE_PNG	-10	PNG 画像ファイルの出力に失敗しました SavePng など
E_DRAWMODE_VALUE	-11	指定された描画設定の値に問題があります DrawStamp/SD_GetDrawPattern のみ
E_GENSTAMPDATA	-12	ESD 出力に失敗しました
E_SAVE_BMP	-13	BMP 画像ファイルの出力に失敗しました SaveBmp/GetBmpData のみ
E_ENCRYPTBMP	-14	暗号化された BMP 画像ファイルの出力に失敗しました EncryptBmp のみ
E_LOG_NOT_FOUND	-15	ログファイルが見つかりません LF_AppendLog のみ
E_SEEK_ERROR	-16	ユーザの検索に失敗しました SF_RenewGroup/LF_GetLogSeekUser など
E_STAMP_ERROR	-17	印影の描画に失敗しました SD_DrawStamp のみ
E_DSM_USING	-18	他のアプリケーションでユーザが利用中です Login/SF_SeekUser など
E_E9_VALIDATION	-19	インプレットのデバイス種類が一致しません
E_EXPIRATION_TRIAL	-20	試用版の期限が切れました(試用版のみ)
DSM_SUCCESS_READONLY	-21	捺印用印鑑データファイルを開いていますが取得専用です Login/SF_AppendStamp など
E_UNMATCH_VALUE	-22	指定されたパラメータが不正です SU_SetTimeServerMode など
E_DSM_ALREADY_OPEN	-23	既に捺印用印鑑データファイルは開かれています SF_DsmOpen/LF_DsmOpen など
E_IDX_NOT_OPEN	-25	印鑑セットアップ元ファイルが開いていません
E_IDX_CANNOT_OPEN	-26	印鑑セットアップ元ファイルが開けません
E_IDX_ALREADY_OPEN	-27	印鑑セットアップ元ファイルは開かれています
E_DSM_FILEHEADER	-101	捺印用印鑑データファイルを開きましたが、ヘッダー情報が異なるため、処理を中止しました。
E_DSM_FILE_ACCESS	-102	捺印用印鑑データファイルにアクセスできません。
E_DSM_FILE_READONLY	-103	捺印用印鑑データファイルが読み取り専用を設定されています
E_USERCACHEFILE_NOT_FOUND	-104	ユーザキャッシュファイルが見つかりません。
E_USERCACHEFILE_CANNOT_OPEN	-105	ユーザキャッシュファイルを開くことができませんでした

E_ESD_PATH	-1101	指定された ESD ファイルのパスに問題があります
E_ESD_DATA	-1102	指定された ESD ファイルに問題があります
E_TABLET_NOT_FOUND	-2101	タブレットが見つかりません
E_DEVICE_NOT_FOUND	-2102	デバイスが見つかりません
E_DEVICE_ID	-2103	デバイスから ID 情報が取得できませんでした
E_USER_DISABLED	-201	ユーザが無効になっています
E_DSM_FILE_ACCESS	-102	ファイルへのアクセスが拒否されました

著作権情報

© 2010 Shachihata Inc. All rights reserved.

本マニュアルの内容は、著作権法により保護されています。

本マニュアルが使用許諾契約を含むソフトウェアと共に提供される場合、本マニュアルおよびその中に記載されているソフトウェアは、使用許諾契約にもとづいて提供されるものであり、当該使用許諾契約の契約条件に従ってのみ使用または複製することが可能となるものです。本マニュアルのいかなる部分も、発行者（シヤチハタ株式会社）への書面による許可なしに、いかなる形式・手段でも、複製、検索システムへの保存、または伝送を行うことはできません。

本マニュアルに記載される内容は、可能な限り正確であることを考慮しますが予告なしに変更されることを条件として提供されるものであります。従って、本マニュアルの情報がシヤチハタ株式会社による確約として解釈されるものではありません。シヤチハタ株式会社は、本マニュアルにおけるいかなる誤りまたは不正確な記述に対しても、いかなる義務や責任を負うものではありません。

本マニュアルに記載されている会社名、製品名などは各社の商標または商標登録です。

電子印鑑 Extension サービスリリース 1
リファレンスマニュアル

2010/06 発行

発行者

シヤチハタ株式会社

<http://www.shachihata.co.jp/>

パソコン決裁についての情報

<http://www.shachihata.co.jp/interweb/>