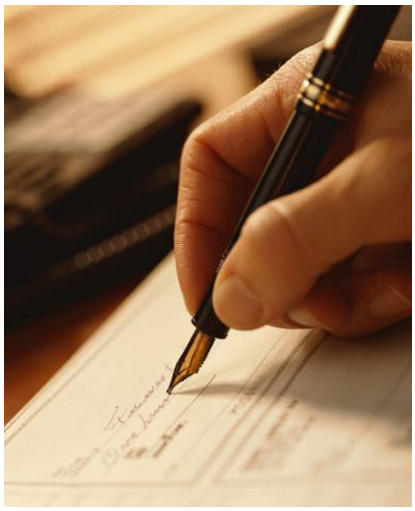
signotec

Dokumentation
Digital Signature Fields

(Erweiterung der signoAPI)

Erzeugen und Unterschreiben von Adobe Acrobat konformen digitalen Unterschriftsfeldern in PDF-Dokumenten.



Version *1.2.76*
Datum: *26.10.2018*

Vertraulich
Änderungen vorbehalten

Inhaltsverzeichnis

1. Komponente SignPDF3: DigSig-Felder in PDF-Dokumenten unterschreiben und prüfen/Weitere Funktionen für DigSig-Felder

Digitale Signaturfelder (DigSig-Felder) gehören zu dem von Adobe® definierten Mechanismus zum digitalen Unterschreiben von PDF-Dokumenten auf Basis von Standard Zertifikaten. Diese werden auch Adobe-konforme Unterschriftenfelder genannt. DigSig = Digital Signature.

Mit Hilfe dieser Komponente ist es möglich PDF-Dokumente digital zu signieren und wichtige biometrische Zusatzinformationen des jeweiligen Unterzeichners sicher im Dokument zu hinterlegen.

Beim Unterzeichnen wird jeweils eine neue Revision des PDF-Dokuments erstellt und digital signiert, so dass in jedem DigSig-kompatiblen PDF-Anzeigeprogramm die Unversehrtheit und Echtheit des Dokuments geprüft werden kann (z. B. mit Adobe Acrobat® oder Adobe® Reader).

Alle erfassten Daten der Unterschriftendigitalisierung (u.a. die biometrischen Daten der Unterschrift) werden im unsichtbaren Bereich des jeweiligen DigSig-Feldes RSA verschlüsselt und sicher gespeichert (siehe dazu die Erklärung der dazu verwendeten XML-Strukturen im Anhang).

Für einführende und weitergehende Informationen zum Thema Digitale Signaturen in PDF Dokumenten steht die separat erhältliche Dokumentation „signotec Verfahrensbeschreibung der e-Signatur und Verschlüsselung“ zur Verfügung. Diese kann bei der signotec GmbH angefordert werden.

1.1. Systemvoraussetzungen

Die Komponente SignPDF3 setzt auf die Microsoft CryptoAPI (CAPI) auf (http://msdn.microsoft.com/en-us/library/aa380256(VS.85).aspx).
Dabei werden unter anderem die zurzeit als sicher geltenden Hash-Algorithmen SHA256 und SHA512 verwendet. Aus diesem Grund ist die Benutzung eines aktuellen *Cryptographic Service Provider*s (CSP) von Microsoft erforderlich. Ein CSP der die geforderten Algorithmen unterstützt ist jedoch erst ab Windows XP Service Pack 3 und neueren Systemen vorhanden (SP3 enthält Updates für das Microsoft Kernel Mode Cryptographic Module).
Zur Benutzung der Komponente SignPDF3 ist also zwingend Windows XP mit Servicepack 3 oder ein neueres Betriebssystem erforderlich.
Für Windows 2000 ist kein solcher neuerer CSP verfügbar, daher funktioniert die Komponente unter Windows 2000 nicht und wird auf so einem System vom Setup auch nicht installiert.

Wichtige Information:
Die Komponente signPDF3.dll hat folgende Abhängigkeiten

1. Die Abhängigkeit ist eine Dll **STPdfLib13.dll**. Die muss im gleichen Ordner mit signPDF3.dll liegen.

2. Eine COM Abhängigkeit ist die Dll **signlib.dll**. Bevor der Instanziierung der Komponenten signPDF3.dll muss die Dll auf dem System registriert sein. In der Kommandozeile den Befehl *regsvr32 signlib.dll* ausführen. Die Dll **signlib.dll** wird nur in 32 Bit Version von signPDF3.dll und aus Kompatibilitätsgründen benötigt.
3. Die Methoden, die einen BSTR zurückgeben, benötigen die folgende Vorgehensweise.

```
//Get the actual PadID from the pad
virtual /* [helpstring][id][propget] */ HRESULT
STDMETHODCALLTYPE get_PadID(/* [retval][out] */
BSTR __RPC_FAR *pVal) = 0;

HRESULThr        = E_FAIL;
BSTR   bstrPadID = NULL;

hr = m_pSignPad->get_PadID(&bstrPadID);

if(bstrPadID != NULL)
{
  SysFreeString(bstrPadID);
  bstrPadID = NULL;
}
```

Der zurückgegebene BSTR muss unbedingt mit SysFreeString freigegeben werden.

1.2. Definition des Interface ISignDigSigPDF

Interface ISignDigSigPDF : IDispatch

1.3. Interface-IDs Zuordnung

Nachfolgend die IID für das Interface:

Interface	**IID**
ISignDigSigPDF	8EBAA8A2-E603-4C66-AAF9-716BB6AD14BA

Nachfolgend die CLSID für das Interface:

Interface	**CLSID**
ISignDigSigPDF	6CF797C0-F930-4AC9-8E39-2968282964B4

1.4. Hinweise zur Deklaration

Zur Instanziierung unter VB ist folgende Deklaration notwendig

Interface	Deklaration
ISignDigSigPDF	Signotec.signpdf3

1.5. Methoden und Eigenschaften

Methode SignPdfDocument

Diese Methode ist veraltet und nur noch aus Kompatibilitätsgründen enthalten. Bitte verwenden Sie stattdessen SignPdfDocumentMemory().

Methode SignPdfDocumentMemory

Elektronisches Unterschreiben eines DigSig-Feldes in einem PDF-Dokument mittels gegebenen Unterschriftendaten (SignData).
Der erfasste Unterschriftenzug und die Biometrie werden in das durch den Namen eindeutig identifizierte DigSig-Feld eingebracht.
Ist der Name des Feldes noch nicht vorhanden, so wird dieses neu erzeugt (Positionsangaben müssen dann gesetzt werden).

Hinweis: Zum Unterzeichnen eines DigSig Feldes müssen zwei Zertifikate an die Methoden übergeben werden. Das Erste muss ein PKCS12-Zertifikat mit dem privaten Schlüssel zum digitalen Signieren des Dokumentes nach den DigSig-Spezifikationen sein. Das Zweite muss ein öffentliches X.509-Zertifikat oder ein PKCS12-Zertifikat mit dem öffentlichen Schlüssel zum Verschlüsseln für die signotec Zusatzdaten (Biometrieverschlüsselung) sein. Wenn ein PKCS12-Zertifikat verwendet wird, muss das Passwort im Parameter „bstrCryptBioCertPassword“ übergeben werden.

Hinweis: Die Unterschriftendaten (SignData) müssen aus einer vorangegangen Unterschrifterfassung auf einem signotec LCD Signature Pad übernommen werden.

Hinweis: Das Unterschriftenbild (Image) muss aus einer vorangegangen Unterschrifterfassung auf einem signotec LCD Signature Pad übernommen werden. Für PDF/A-1b-Konformität darf das Unterschriftenbild keinen Alphakanal enthalten.

Hinweis: Die XML-Struktur für den Parameter „bstrXMLSignatureData“ ist im Anhang ausführlich beschrieben.

BSTR [in]	bstrPadID	Eindeutige ID des angeschlossenen Pads.
VARIANT* [in/out]	pvarPdfArray	PDF in einem Variant.
VARIANT (VT_ARRAY) [in]	varSignDataArray	Der einzubringende biometrische Datenblock. (VT_ARRAY mit SignData).
BSTR [in]	bstrSigfieldTitle	Eindeutiger Name des zu unterschreibenden DigSig Feldes.
VARIANT [in]	varSignCertArray	Das Daten Array des PKCS12-Zertifikats mit dem privaten Schlüssel

		zum digitalen Signieren (für DigSig Mechanismus).
BSTR [in]	bstrSignCertPassword	Passwort für das Daten Array von PKCS12-Zertifikat mit dem privaten Schlüssel zum digitalen Signieren (für DigSig Mechanismus).
VARIANT [in]	varCryptBioCertArray	Daten Array von einem öffentlichen X.509-Zertifikat oder einem PKCS12-Zertifikat mit dem öffentlichen Schlüssel zum Verschlüsseln der Biometrie.
BSTR [in]	bstrCryptBioCertPassword	Optionales Passwort falls ein Daten Array von einem PKCS12-Zertifikat mit dem öffentlichen Schlüssel zum Verschlüsseln der Biometrie verwendet werden soll.
BSTR [in]	bstrXMLSignatureData	XML-codierte Zeichenkette mit zusätzlichen einzubringenden Informationen (separat beschrieben) Siehe Kapitel 2.4. XML Struktur mit den benötigten Eingabeinformationen beim Unterschreiben
VARIANT (VT_ARRAY) [in]	varImageArray	Unterschriftenbild, das in PDF-Dokument eingebracht wird. Für PDF/A-1b-Konformität darf das Unterschriftenbild keinen Alphakanal enthalten.
long [in]	lOptions	Der Parameter lOptions ist eine Bitmaske. Folgende Bits sind erlaubt: - 0x00: Dokument nicht PDF/A konform signieren - 0x01: Dokument PDF/A-1b konform signieren - 0x02: Dokument PDF/A-2b konform signieren - 0x04: Alle Formularfelder sperren - 0x08: Nur ausgefüllte Formularfelder sperren Die Bits 0x01 und 0x02 dürfen nicht beide gesetzt sein. Die Bits 0x04 und 0x08 dürfen nicht beide gesetzt sein. Gültige Werte sind also 0, 1, 2, 4, 5, 6, 8, 9 und 10. In allen anderen Fällen wird E_INVALIDARG zurückgegeben.
long* [in/out]	plReserved	Reserviert, immer 0.

```
HRESULT SignPdfDocumentMemory( [in] BSTR      bstrPadID,
                               [in/out] VARIANT* pvarPdfArray,
                               [in] VARIANT      varSignDataArray,
                               [in] BSTR      bstrSigfieldTitle,
                               [in] VARIANT      varSignCertArray,
                               [in] BSTR      bstrSignCertPassword,
                               [in] VARIANT      varCryptBioCertArray,
                               [in] BSTR      bstrCryptBioCertPassword,
                               [in] BSTR      bstrXMLSignatureData,
                               [in] VARIANT      varImageArray,
                               [in] long      lOptions,
                               [in/out] long* plReserved);
```

Methode PreparePdfDocumentMemory

Vorbereitung des elektronischen Unterschreibens eines DigSig-Feldes in einem PDF-Dokument mittels gegebenen Unterschriftendaten (SignData).

Das PDF-Dokument wird zum Signieren vorbereitet. Der erfasste Unterschriftenzug und die Biometrie werden verschlüsselt und in das durch den Namen eindeutig identifizierte DigSig-Feld eingebracht. Das Verschlüsseln der Biometrie kann sowohl im Pad als auch bevor die Methode aufgerufen wird, erfolgen. Das Signieren kann genauso im Pad oder nach dem Aufruf der Methode erfolgen. Ist der Name des Feldes noch nicht vorhanden, so wird dieses neu erzeugt (Positionsangaben müssen dann gesetzt werden). Im PDF-Dokument wird ein Platzhalter fürs spätere Einbringen der digitalen Signatur vorbereitet und seine Position ermittelt. Das vorbereitete PDF-Dokument in einem Byte-Array und die Platzhalters Position werden zurückgegeben.

Nach dem Aufruf der Methode muss über den Byte-Array mit dem vorbereitenden PDF-Dokument ein SHA256 Hash gebildet werden. Der Hash muss dann signiert werden. Dadurch wird eine digitale Signatur erzeugt. Die erzeugte Signatur muss dann an der Stelle des Platzhalters ins PDF-Dokument geschrieben werden.

Hinweis: Zum Verschlüsseln von der signotec Zusatzdaten (Biometrieverschlüsselung) wird ein öffentliches X.509-Zertifikat oder ein PKCS12-Zertifikat mit dem öffentlichen Schlüssel verwendet. Wenn ein PKCS12-Zertifikat verwendet wird, muss das Passwort im Parameter „bstrCryptBioCertPassword“ übergeben werden.

Hinweis: Die Unterschriftendaten (SignData) müssen aus einer vorangegangen Unterschrifterfassung auf einem signotec LCD Signature Pad übernommen werden.

Hinweis: Das Unterschriftenbild (Image) muss aus einer vorangegangen Unterschrifterfassung auf einem signotec LCD Signature Pad übernommen werden. Für PDF/A-1b-Konformität darf das Unterschriftenbild keinen Alphakanal enthalten.

Hinweis: Die XML-Struktur für den Parameter „bstrXMLSignatureData“ ist im Anhang ausführlich beschrieben.

BSTR [in]	bstrPadID	Eindeutige ID des angeschlossenen Pads.
VARIANT* [in/out]	pvarPdfArray	PDF in einem Variant.
VARIANT (VT_ARRAY) [in]	varSignDataArray	Der einzubringende biometrische Datenblock. (VT_ARRAY mit SignData).
BSTR [in]	bstrSigfieldTitle	Eindeutiger Name des zu unterschreibenden DigSig Feldes.
VARIANT [in]	varSignCertArray	Daten Array von PKCS12-Zertifikat mit dem öffentlichen Schlüssel zum Ermitteln seiner Eigenschaften.
VARIANT [in]	varCryptBioCertArrayOrRefString	Daten Array von einem öffentliches X.509-Zertifikat oder einem PKCS12-Zertifikat mit dem öffentlichen Schlüssel zum Verschlüsseln der Biometrie. Wenn die biometrischen Daten im Pad verschlüsselt wurden, muss Zertifikate-ID des Pad-Zertifikates in einem BSTR-String übergeben werden.
BSTR [in]	bstrCryptBioCertPassword	Passwort falls ein Daten Array von PKCS12-Zertifikat mit dem öffentlichen Schlüssel zum Verschlüsseln der Biometrie verwendet werden soll.
BSTR [in]	bstrXMLSignatureData	XML-codierte Zeichenkette mit zusätzlichen einzubringenden Informationen (separat beschrieben)

		Siehe Kapitel 2.4. XML Struktur mit den benötigten Eingabeinformationen beim Unterschreiben
VARIANT (VT_ARRAY) [in]	varImageArray	Unterschriftenbild, das in PDF-Dokument eingebracht wird. Für PDF/A-1b -Konformität darf das Unterschriftenbild keinen Alphakanal enthalten.
VARIANT*(VT_ARRAY) [out]	pvarDocHashArray	Der Data Array mit dem PDF-Dokument, der zurückgegeben wird. Über die Daten muss eine SHA256 Hash gebildet werden. Der Hash muss dann signiert und ins Dokument geschrieben werden.
long* [out]	plDocSignaturePosition	Das Offset des Platzhalters der Signatur im PDF-Dokument. Der signierte Hash (Digitale Signatur) muss an der Stelle ins Dokument geschrieben werden, um das Unterzeichnen des DigSig Feldes abzuschließen.
long [in]	lOptions	Der Parameter lOptions ist eine Bitmaske. Folgende Bits sind erlaubt: - 0x00: Dokument nicht PDF/A konform signieren - 0x01: Dokument PDF/A-1b konform signieren - 0x02: Dokument PDF/A-2b konform signieren - 0x04: Alle Formularfelder sperren

		- 0x08: Nur ausgefüllte Formularfelder sperren Die Bits 0x01 und 0x02 dürfen nicht beide gesetzt sein. Die Bits 0x04 und 0x08 dürfen nicht beide gesetzt sein. Gültige Werte sind also 0, 1, 2, 4, 5, 6, 8, 9 und 10. In allen anderen Fällen wird E_INVALIDARG zurückgegeben.
long* [in/out]	plReserved	Reserviert, immer 0.

HRESULT **PreparePdfDocumentMemory**([in] BSTR bstrPadID,
[in/out] VARIANT* pvarPdfArray,
[in] VARIANT varSignDataArray,
[in] BSTR bstrSigfieldTitle,
[in] VARIANT varSignCertArray,
[in] VARIANT varCryptBioCertArrayOrRefString,
[in] BSTR bstrCryptBioCertPassword,
[in] BSTR bstrXMLSignatureData,
[in] VARIANT varImageArray,
[out] VARIANT* pvarDocHashArray,
[out] long* plDocSignaturePosition,
[in] long lOptions,
[in/out] long* plReserved);

Methode PreparePDFDocumentSigningMemory

Vorbereitung des elektronischen Unterschreibens eines DigSig-Feldes in einem PDF-Dokument.

Das PDF-Dokument wird zum Signieren vorbereitet. Im PDF-Dokument wird ein Platzhalter fürs spätere Einbringen der digitalen Signatur vorbereitet und seine Position ermittelt. Das vorbereitete PDF-Dokument in einem Byte-Array und die Platzhalters Position werden zurückgegeben.

Nach dem Aufruf der Methode muss über den Byte-Array mit dem vorbereitenden PDF-Dokument ein SHA256 Hash gebildet werden. Der Hash muss dann signiert werden. Dadurch wird eine digitale Signatur erzeugt. Die erzeugte Signatur muss dann an der Stelle des Platzhalters ins PDF-Dokument geschrieben werden.

Hinweis: Die XML-Struktur für den Parameter „bstrXMLSignatureData“ ist im Anhang ausführlich beschrieben.

VARIANT* [in/out]	pvarPdfArray	PDF in einem Variant.
BSTR [in]	bstrSigfieldTitle	Eindeutiger Name des zu unterschreibenden DigSig Feldes.
BSTR [in]	bstrXMLSignatureData	XML-codierte Zeichenkette mit zusätzlichen einzubringenden Informationen (separat beschrieben) Siehe Kapitel 2.4. XML Struktur mit den benötigten Eingabeinformationen beim Unterschreiben
VARIANT (VT_ARRAY) [in]	varImageArray	Unterschriftenbild, das in PDF-Dokument eingebracht wird. Für PDF/A-1b Konformität darf das Unterschriftenbild keinen Alphakanal enthalten.
VARIANT*(VT_ARRAY) [out]	pvarDocHashArray	Der Data Array mit dem PDF-Dokument, der zurückgegeben wird. Über die Daten muss eine SHA256 Hash gebildet werden. Der Hash muss dann signiert und ins Dokument geschrieben werden.
long* [out]	plDocSignaturePosition	Das Offset des Platzhalters

		der Signatur im PDF-Dokument. Der signierte Hash (Digitale Signatur) muss an der Stelle ins Dokument geschrieben werden, um das Unterzeichnen des DigSig Feldes abzuschließen.
long [in]	lOptions	Der Parameter lOptions ist eine Bitmaske. Folgende Bits sind erlaubt: - 0x00: Dokument nicht PDF/A konform signieren - 0x01: Dokument PDF/A-1b konform signieren - 0x02: Dokument PDF/A-2b konform signieren - 0x04: Alle Formularfelder sperren - 0x08: Nur ausgefüllte Formularfelder sperren Die Bits 0x01 und 0x02 dürfen nicht beide gesetzt sein. Die Bits 0x04 und 0x08 dürfen nicht beide gesetzt sein. Gültige Werte sind also 0, 1, 2, 4, 5, 6, 8, 9 und 10. In allen anderen Fällen wird E_INVALIDARG zurückgegeben.
long* [in/out]	plReserved	Reserviert, immer 0.

```
HRESULT PreparePDFDocumentSigningMemory (
                    [in/out] VARIANT* pvarPdfArray,
                    [in] BSTR      bstrSigfieldTitle,
                    [in] BSTR      bstrXMLSignatureData,
                    [in] VARIANT       varImageArray,
                    [out] VARIANT*     pvarDocHashArray,
                    [out] long*    plDocSignaturePosition,
                    [in] long      lOptions,
                    [in/out] long* plReserved);
```

Methode GetReferenceCount

Diese Methode ist veraltet und nur noch aus Kompatibilitätsgründen enthalten. Bitte verwenden Sie stattdessen GetReferenceCountMemory().

Methode GetReferenceCountMemory

Ermittelt die Anzahl der vorhandenen DigSig Felder (alle oder nur gefüllte) auf der angegebenen Seite im angegebenen PDF-Dokument.

VARIANT [in]	varPdfArray	PDF in einem Variant.
int [in]	nPage	Angabe auf welcher Seite gesucht werden soll. -1 liefert alle Felder im gesamten Dokument.
long [out]	*plCount	[out] Pointer auf einen long-Wert. Anzahl der gefundenen DigSig Felder.
int [in]	nOption	0 = nur gefüllte Felder, 1 = alle Felder zählen.

HRESULT **GetReferenceCountMemory**([in] VARIANT varPdfArray,
[in] int nPage,
[out] long* plCount,
[in] int nOption);

Methode GetReference

Diese Methode ist veraltet und nur noch aus Kompatibilitätsgründen enthalten. Bitte verwenden Sie stattdessen GetReferenceMemory().

Methode GetReferenceMemory

Entnimmt und dekodiert die Zusatzdaten aus einem unterschriebenen DigSig Feld aus dem angegebenen PDF-Dokument. Das DigSig Feld wird durch seinen angegebenen eindeutigen Namen identifiziert.

Hinweis: Zur Dekodierung der Zusatzdaten (SignData und Zusatzinformationen) muss eine PKCS12-Zertifikatsdatei mit dem privaten Schlüssel zum Entschlüsseln der Biometrie angegeben werden. Dieser muss zu dem verwendeten öffentlichen Schlüssel der zum Verschlüsseln verwendet wurde passen.
Hinweis: Die XML-Struktur für den Parameter „pbstrXMLResultData“ ist im Anhang ausführlich beschrieben.

VARIANT [in]	varPdfArray	PDF in einem Variant.
int [in]	nReserved	Reserviert. Immer 0.
BSTR [in]	bstrSigfieldTitle	Eindeutiger Name des auszulesenden DigSig Feldes.
VARIANT [in]	varPFXCertArray	Dateiname und Pfad zur PKCS12-Zertifikatsdatei mit dem privaten Schlüssel zum Enschlüsseln der biometrischen Daten.
BSTR [in]	bstrPFXPassword	Passwort für die PKCS12-Zertifikatsdatei mit dem privaten Schlüssel zum Entschlüsseln der Biometrie.
VARIANT (VT_ARRAY)	*pvarSigndata	[out] Pointer auf einen leeren VARIANT. Pointer auf einen VARIANT vom Typ

[out]		VT_ARRAY indem die Daten der SignData-Struktur enthalten sind.
BSTR [out]	*pbstrXMLResultData	Ausgabe einer XML-codierten Zeichenkette mit Informationen zum Signaturprozess.
int [in]	nOption	0 = Default, 1 = erweiterte Signatur/Integritäts Hash Überprüfung (siehe XML Parameter).

```
HRESULT GetReferenceMemory(     [in] VARIANT varPdfArray,
                                [in]    int         nReserved,
                                [in]    BSTR        bstrSigfieldTitle,
                                [in]    VARIANT     varPFXCertArray,
                                [in]    BSTR        bstrPFXPassword,
                                [out]   VARIANT*    pvarSigndata,
                                [out]   BSTR*       pbstrXMLResultData,
                                [in]    int         nOption);
```

Methode VerifyPdfDocument

Diese Methode ist veraltet und nur noch aus Kompatibilitätsgründen enthalten. Bitte verwenden Sie stattdessen VerifyPdfDocumentMemory().

Methode VerifyPdfDocumentMemory

Entnimmt und dekodiert die Daten der Digitalen Signaturen aus den unterschriebenen DigSig Feldern aus dem angegebenen PDF-Dokument und überprüft diese auf Ihre Gültigkeit und Unversehrtheit.

Hinweis: Die XML-Struktur für den Parameter „pbstrXMLResultData“ ist im Anhang ausführlich beschrieben.
Hinweis: Diese Methode gibt nicht die verschlüsselten biometrischen Daten von bereits unterschriebenen DigSig Feldern zurück. Diese verschlüsselten Informationen sind nur mittels der Methode GetReferenceMemory() und dem entsprechenden Zertifikat zu erhalten.

VARIANT [in]	varPdfArray	PDF in einem Variant
BSTR* [in]	pbstrXMLResultData	Ausgabe einer XML-codierten Zeichenkette mit Informationen zu den unterschriebenen DigSig Feldern.
long* [out]	plStatus	[out] Pointer auf einen long - Statuswert. (zur Zeit immer 0)

```
HRESULT VerifyPdfDocumentMemory(    [in]    VARIANT varPdfArray,
                                    [out]   BSTR* pbstrXMLResultData,
                                    [out]   long* plStatus);
```

Methode GetDSFieldInfo

Diese Methode ist veraltet und nur noch aus Kompatibilitätsgründen enthalten. Bitte verwenden Sie stattdessen GetDSFieldInfoMemory().

Methode GetDSFieldInfoMemory

Entnimmt die Daten aller DigSig Felder aus dem angegebenen PDF-Dokument und zeigt die Details zu diesen an.

Hinweis: Die XML-Struktur für den Parameter „pbstrXMLResultData“ ist im Anhang ausführlich beschrieben.
Hinweis: Diese Methode gibt nicht die verschlüsselten biometrischen Daten von bereits unterschriebenen DigSig Feldern zurück. Diese Informationen sind nur mittels der Methode GetReferenceMemory() zu erhalten.

VARIANT [in]	varPdfArray	PDF in einem Variant.
BSTR [out]	*pbstrXMLResultData	Ausgabe einer XML-codierten Zeichenkette mit Informationen zum Signaturprozess
int [in]	nReserved	Reserviert, immer 0

HRESULT **GetDSFieldInfoMemory**([in] VARIANT varPdfArray,
[out] BSTR* pbstrXMLResultData,
[in] int nReserved);

Methode SetLicenseKey

Setzt einen hardwareunabhängigen Lizenzschlüssel zur Verwendung dieser Schnittstelle. Es kann ein zeitlich begrenzter Schlüssel zu Testzwecken angefordert werden oder eine „Full License“ erworben werden.

Hinweis: Ohne gültigen Schlüssel kann diese Schnittstelle nicht verwendet werden.

BSTR [in]	bstrLicenseKey	Lizenzschlüssel als Zeichenkette (BSTR).

HRESULT **SetLicenseKey**([in] BSTR bstrLicenseKey);

Methode VerifyCertificate

Diese Methode ist veraltet und nur noch aus Kompatibilitätsgründen enthalten. Bitte verwenden Sie stattdessen VerifyCertificateMemory().

Methode VerifyCertificateMemory

Prüft das Zertifikat und liefert seinen Status zurück.

VARIANT [in]	varCertArray	Zertifikat in einem Variant.

BSTR [in]	bstrOptionalPFXPassword	Kennwort für das Zertifikat, der Parameter ist optional
int [in]	nReserved	Reserviert, immer 0
long* [out]	plCertStatus	Zertifikatsstatus. Siehe Kapitel signotec Zertifikatsstatus-Tags

HRESULT **VerifyCertificateMemoy**([in] VARIANT varCertArray,
[in] BSTR bstrOptionalPFXPassword,
[in] int nReserved,
[out] long* plCertStatus);

Methode CreatePKCS12CertificateMemory

Erzeugt ein pfx-Zertifikat und speichert es in eine Datei.

BSTR [in]	bstrSubject	X.500 – String. Der String beinhaltet eine Definition von verschiedenen Atributen des Zertifikats (RFC 2253). Ein Beispiel von X.500 – String: "CN=Siggi Pinsel, L=Ratingen, O=signotec GmbH, OU=signosign, Email=sigcert@signotec.de, C=DE, ST=NRW, STREET=Am Gierath, Title=Doctor of Science, GivenName=Siggi Genius, Initials=S.P.G, SN=Pinsel, DC=signotec"
BSTR [in]	bstrPassword	Kennwort für das neue Zertifikat
short [in]	nRSABits	Schlüssellänge, mögliche Werte 1024, 2048
short [in]	nValidYears	Die Gültigkeitsperiode in Jahren
VARIANT* [out]	pvarCertArray	Ein Variant mit neu erzeugtem Zertifikat als Bytearray

HRESULT **CreatePKCS12CertificateMemory** ([in] BSTR bstrSubject,
[in] BSTR bstrPassword,
[in] short nRSABits,
[in] short nValidYears,
[out] VARIANT* pvarCertArray);

Methode CreateDSFieldMemory

Erzeugt ein leeres nicht unterschriebenes Signaturfeld.

VARIANT* [in/out]	pvarPdfArray	PDF in einem Variant
BSTR [in]	bstrTitle	Eindeutiger Name des Signaturfeldes im PDF-Dokument.

int [in]	nPage	Die Seite des PDFs, an der das neue Signaturfeldes erzeugt wird.
double [in]	dX	X-Koordinate der linken oberen Ecke des neuen Signaturfeldes. Der Ursprung des koordinatensystems ist in der linken oberen Ecke des PDFs.
double [in]	dY	Y-Koordinate der linken oberen Ecke des neuen Signaturfeldes. Der Ursprung des koordinatensystems ist in der linken oberen Ecke des PDFs.
double [in]	dWidth	Die Breite des neuen Signaturfeldes
double [in]	dHeight	Die Höhe des neuen Signaturfeldes
int nFlags [in]	nFlags	Für spätere Verwendung vorgesehen. Jetzt wird es nicht verwendet.

HRESULT **CreateDSFieldMemory** ([in] VARIANT* pvarPdfArray,
[in] BSTR bstrTitle,
[in] int nPage,
[in] double dX,
[in] double dY,
[in] double dWidth,
[in] double dHeight,
[in] int nFlags);

Methode DrawTextBoxPdfDocumentMemory

Fügt einen frei wählbaren Text in ein bestehendes PDF-Dokument ein.
Neben der gewünschten Seite muss die Position des Textes durch die Angabe der Eckpunkte eines Rechtecks (der eigentlichen Textbox) bestimmt werden.
Schriftart, Schriftgröße, Schriftfarbe und Schriftauszeichnung können gewählt werden.
Der eingebrachte Text ist nicht umrahmt und wird transparent eingebracht, sodass andere Texte nicht überdeckt werden. Der Text wird als Zeichenkette eingebracht, nicht als Bild.

Hinweis: Die Methode darf nicht auf schon unterschriebene Dokumente angewendet werden, da sonst die bisher eingebrachten Unterschriften ungültig werden und ein weiteres Unterschreiben unmöglich wird.
Hinweis: Es erfolgt ein automatischer Textumbruch, wenn die Breite der angegebenen Textbox zu klein für die Länge des Textes ist.
Hinweis: Es erfolgt keine automatisch Skalierung der Schriftgröße an die Größe der Textbox. Der Entwickler muss selber dafür Sorge tragen, dass der Platzbedarf der Textes, die Größe der Textbox nicht übersteigt. Ansonsten wird der nicht darstellbare Teil des Textes abgeschnitten.

Hinweis: Wenn bstrFontName leer ist, bestimmt der Parameter nStandardFont welcher Standartfont verwendet wird.

nStandardFont	Standartfont
0	Courier
1	CourierBold

2	CourierBoldOblique
3	CourierOblique
4	Helvetica
5	HelveticaBold
6	HelveticaBoldOblique
7	HelveticaOblique
8	TimesRoman
9	TimesBold
10	TimesItalic
11	TimesBoldItalic
12	Symbol
13	ZapfDingbats

VARIANT* (VT_ARRAY) [in/out]	pvarPdfArray	Inhalt eines PDF-Dokumentes in einem VARIANT als VT_ARRAY gespeichert
BSTR [in]	bstrText	Einzubringender Text
short [in]	xPos1	Obere linke Ecke der gewünschten Textbox in Points (X-Koordinate)
short [in]	yPos1	Obere linke Ecke der gewünschten Textbox in Points (Y-Koordinate)
short [in]	xPos2	Untere rechte Ecke der gewünschten Textbox in Points (X-Koordinate)
short [in]	yPos2	Untere rechte Ecke der gewünschten Textbox in Points (Y-Koordinate)
int [in]	nPage	Seite des Dokumentes, auf der der Text eingebracht werden soll
BSTR [in]	bstrFontName	Name der zu verwendenden Schriftart als Wide-Char Zeichenkette. Wenn die Zeichenkette leer ist, wird einer der Standartfonts verwendet. Der letzter Parameter bestimmt dann welcher.
double [in]	dbFontSize	Schriftgröße als double-Wert
VARIANT_BOOL [in]	bFontBold	Schriftauszeichnung Fett verwenden. Kann VARIANT_TRUE oder VARIANT_FALSE sein
VARIANT_BOOL [in]	bFontItalic	Schriftauszeichnung Kursiv verwenden. Kann VARIANT_TRUE oder VARIANT_FALSE sein
VARIANT_BOOL [in]	bUnderline	Schriftauszeichnung Unterstreichen verwenden. Kann VARIANT_TRUE oder VARIANT_FALSE sein
OLE_COLOR [in]	oleColor	Farbe des einzubringenden Textes als OLE_COLOR Struktur (Reihenfolge der Textfarbe BGR beachten)
int [in]	nStandardFont	Wenn bstrFontName leer ist, bestimmt welcher Standartfont verwendet werden muss.

HRESULT **DrawTextBoxPdfDocumentMemory**

```
(
    [in/out] VARIANT* pvarPdfArray,
    [in] BSTR bstrText,
    [in] short xPos1,
    [in] short yPos1,
    [in] short xPos2,
    [in] short yPos2,
    [in] int nPage,
    [in] BSTR bstrFontName,
    [in] double dbFontSize,
    [in] VARIANT_BOOL bFontBold,
    [in] VARIANT_BOOL bFontItalic,
    [in] VARIANT_BOOL bUnderline,
    [in] OLE_COLOR oleColor,
    [in] int nStandardFont
);
```

Eigenschaft get_SignaturePlaceholderLength

gibt die Länge von dem Platzhalter in Bytes in dem PDF-Dokument, der später die Signatur als hexadezimale Zeichenfolge enthalten wird, zurück. Das bedeutet, dass das Byte-Array der Signatur nicht mehr als die Hälfte dieses Wertes sein muss. Der Standardwert ist 8192, die Eigenschaft muss festgelegt werden, bevor die Methoden PreparePdfDocumentMemory oder PreparePDFDocumentSigningMemory aufgerufen werden, wenn eine andere Größe benötigt wird.

long* [out]	pVal	die Länge von dem Platzhalter in Bytes.

HRESULT **get_SignaturePlaceholderLength** ([out] long *pVal);

Eigenschaft set_SignaturePlaceholderLength

setzt die Länge von dem Platzhalter in Bytes in dem PDF-Dokument, der später die Signatur als hexadezimale Zeichenfolge enthalten wird. Das bedeutet, dass das Byte-Array der Signatur nicht mehr als die Hälfte dieses Wertes sein muss. Der Standardwert ist 8192, die Eigenschaft muss festgelegt werden, bevor die Methoden PreparePdfDocumentMemory oder PreparePDFDocumentSigningMemory aufgerufen werden, wenn eine andere Größe benötigt wird.

Long [in]	newVal	die Länge von dem Platzhalter in Bytes.

HRESULT **set_SignaturePlaceholderLength** ([in] long newVal);

2. Verwendete XML-Strukturen in der Komponente SignPDF3

2.1. XML-Struktur über die Gültigkeitsoptionen beim Verifizieren eines DigSig Feldes

XML-Struktur und Liste der darin enthaltenen Elemente die als Ergebnis eines Aufrufs der Methode **VerifyPdfDocument()** oder **VerifyPdfDocumentMemory()** zurückgegeben wird.

Beispiel:

```
<?xml version="1.0" encoding="utf-8" ?>
<digsignatures>
<digsignature Name="sgnsignature_1">
<Name>Max Mustermann</Name>
<Reason>Ich bin mit dem Inhalt einverstanden.</Reason>
<Location>Wien< /Location>
<ContactInfo>signotec GmbH</ContactInfo>
<Time>D:20090211141611</Time>
<Page>1</Page>
<Mandatory>false</Mandatory>
<Subfilter>adbe.x509.rsa_sha1</Subfilter>
<Filter>Adobe.PPKLite</Filter>
<HashAlgorithm>SHA256</HashAlgorithm>
<Status>1</Status>
<CertExpired>0</CertExpired>
<Rect>
        <Left>323</Left>
        <Right>497</Right>
        <Top>271</Top>
        <Bottom>325</Bottom>
</Rect>
<Certificate>
        <Issuer>C=DE, CN=DEMO , O=DEMO AG</Issuer>
        <Serial>48a9fe5043ea1e854ecf1caa89cf5b60</Serial>
        <PublicKeysize>2048</PublicKeysize>
        <ValidTo>2012/01/07 20:23:44</ValidTo>
        <ValidFrom>2008/01/07 20:23:44</ValidFrom>
        <CertErrorStatus>The certificate or certificate chain is based on an
   untrusted root;</CertErrorStatus>
        <CertErrorStatusCode>32</CertErrorStatusCode>
</Certificate>
<SIGNATURE_INFO>
        <COMPANY>signotec GmbH</COMPANY>
        <VERSION>8.0.0.88</VERSION>
        <SIGN_TIME>11.02.2009 14:16:11</SIGN_TIME>
        <USERID>ws</USERID>
```

```
        <MACHINE>OFFICE_1169</MACHINE>
        <MACADDRESS>0017F991241A</MACADDRESS>
        <PADID>1000013325</PADID>
        <PADMODEL>Sigma HID</PADMODEL>
        <PADTYPE>101</PADTYPE>
        <ADDREFERENCE>TRUE</ADDREFERENCE>
        <CERT>
            <CIPHERENC_FILENAME>Demo1_Cert.p12</CIPHERENC_FILENA
        ME>
            <BIOENC_FILENAME>Demo2_BIO.pem</BIOENC_FILENAME>
        </CERT>
</SIGNATURE_INFO>
</digsignature>
<digsignature Name="sgnsignature_2">
        <Weiteres Signaturfeld ähnlich wie sgnsignature_1>
</digsignatures>
```

Elemente <digsignatures> und <digsignature>

Das XML-Dokument enthält als Wurzelelement <digsignatures>. Unterhalb dieses Elementes existiert für jedes Signaturfeld ein Element <digsignature>. Dieses Element enthält die folgenden Elemente:

Element <Name> (optional)

Das Element Name enthält den Namen des Unterzeichners oder automatisch den Namen auf den das Zertifikat ausgestellt wurde. Standard Anzeige auch im Adobe Acrobat® oder Adobe® Reader.

Element <Reason> (optional)

Das Element Grund (Reason) enthält die Kurzbeschreibung über den Grund des Unterschreibens. Standard Anzeige auch im Adobe Acrobat® oder Adobe® Reader.

Element <Location> (optional)

Das Element Ort (Location) enthält eine Kurzbeschreibung über den Aufenthaltsort. Standard Anzeige auch im Adobe Acrobat® oder Adobe® Reader.

Element <ContactInfo> (optional)

Das Element ContactInfo enthält den optional einen Namen oder automatisch die Kontaktinformation aus dem Zertifikat. Standard Anzeige auch im Adobe Acrobat® oder Adobe® Reader unter Zertifikat Anzeige.

Element <Time>

Dieses Element enthält den Zeitstempel an dem das Signaturfeld signiert wurde im Adobe® Format.

Element <Page>

Dieses Element enthält die Seite im PDF auf der sich das Signaturfeld befindet.

Element <Mandatory>

Dieses Element beschreibt, ob es sich um ein Signatur Pflichtfeld handelt.

Element <SubFilter>

Dieses Element beschreibt mit welcher Methode die Signatur eingebracht wurde. Standard ist hier das Format „**adbe.x509.rsa_sha1**“.

Hinweis: Obwohl im Namen des Standardfilters SHA1 steht, werden je nach PDF-Dateiversion auch höhere SH-Algorithmen unterstützt (z.B. SHA256).

Element <Filter>

Dieses Element beschreibt mit welcher Standard-Überprüfungsmethode eine Überprüfung stattfinden kann. Standard ist hier die Methode „**Adobe.PPKLite**“.

Element <HashAlgorithm>

Dieses Element beschreibt den verwendeten Hash-Algorithmus der bei der Digitalen Signatur verwendet wurde (unter anderem abhängig von der PDF-Dateiversion).

Element <Status>

Dieses Element beschreibt den Status des Signaturfeldes bezogen auf die Dokumenten Revision. Folgende Werte können zurückgegeben werden:

STATUS_VALID	0	Signatur ist gültig.
STATUS_CHANGEDAFTERSIGNING	1	Signatur ist gültig (mit Änderung danach).
STATUS_INVALID	2	Signatur ist ungültig.
STATUS_EMPTY	3	Signaturfeld ist leer (nicht unterschrieben).
STATUS_UNKNOWN	4	Kann nicht überprüft werden(unbek. Format).

Element <CertExpired>

Dieses Element beschreibt den Status des Zertifikates. Es wird das Ablaufdatum des eingebetteten öffentlichen Teils des Zertifikates überprüft.

0 = OK.
1 = zeitlich abgelaufenes Zertifikat.

Element <Rect/Left> (optional, nur wenn das Signaturfeld noch nicht vorhanden ist)

Dieses Element enthält die Koordinate der linken Seite des Signaturfeldes in Pixel bezogen auf die obere linke Ecke der PDF Dokumenten Seite.

Element <Rect/Right> (optional, nur wenn das Signaturfeld noch nicht vorhanden ist)

Dieses Element enthält die Koordinate der rechten Seite des Signaturfeldes in Pixel bezogen auf die obere linke Ecke der PDF Dokumenten Seite.

Element <Rect/Top> (optional, nur wenn das Signaturfeld noch nicht vorhanden ist)

Dieses Element enthält die Koordinate der oberen Seite des Signaturfeldes in Pixel bezogen auf die obere linke Ecke der PDF Dokumenten Seite.

Element <Rect/Bottom> (optional, nur wenn das Signaturfeld noch nicht vorhanden ist)

Dieses Element enthält die Koordinate der unteren Seite des Signaturfeldes in Pixel bezogen auf die obere linke Ecke der PDF Dokumenten Seite.

Element <Certificate/Issuer>

Dieses Element enthält Bezeichnungen des eingebetteten Zertifikates.
Folgende genormten Bezeichnungen können enthalten sein (englisch):

CN The certificate owner's common name
E The certificate owner's e-mail address
T The certificate owner's locality
ST The certificate owner's state of residence
O The organization to which the certificate owner belongs
OU The name of the organizational unit to which the certificate owner belongs
C The certificate owner's country of residence
STREET The certificate owner's street address
ALL The certificate owner's complete distinguished name

Element <Certificate/ Serial>

Dieses Element enthält die Seriennummer des verwendeten Zertifikates.

Element <Certificate/PublicKeySize>

Dieses Element enthält die RSA Verschlüsselungsqualität. Zum Beispiel 2048.

Element <Certificate/ValidTo>

Dieses Element enthält das Gültigkeitsende des verwendeten Zertifikates.

Element <Certificate/ValidFrom>

Dieses Element enthält den Gültigkeitsbegin des verwendeten Zertifikates.

Element <Certificate/CertErrorStatus>

Dieses Element enthält den Fehlerstatus des verwendeten Zertifikates. Der Fehlerstatus ist eine Zeichenkette. Die Zeichenkette beinhaltet die durch Semikolons getrennten Fehlerbeschreibungen des Feldes *DWORD dwErrorStatus;* von der *CERT_TRUST_STATUS Struktur*.

Element <Certificate/CertErrorStatusCode>

Dieses Element enthält den Fehler-Status-Code des verwendeten Zertifikates. Der Fehlercode ist der Wert des Feldes *DWORD dwErrorStatus;* von der *CERT_TRUST_STATUS Struktur*.

Element <SIGNATURE_INFO/COMPANY>

Dieses Element beschreibt den Hersteller der API. Immer „signotec GmbH“

Element <SIGNATURE_INFO/VERSION>

Dieses Element beschreibt die Version des API mit der unterschrieben wurde.

Element <SIGNATURE_INFO/SIGN_TIME>

Dieses Element beschreibt den Zeitpunkt wann unterschrieben wurde.

Element <SIGNATURE_INFO/USERID>

Dieses Element enthält den Namen des angemeldeten Benutzers.

Element <SIGNATURE_INFO/ADDREFERENCE>

Dieses optionale Element ist nur für den internen Gebrauch.

Element <SIGNATURE_INFO/MACHINE>

Dieses Element enthält den Namen des PC an dem unterzeichnet wurde.

Element <SIGNATURE_INFO/PADID>

Dieses Element enthält die eindeutige Geräte ID des angeschlossenen PADs.

Element <SIGNATURE_INFO/PADMODEL>

Dieses Element enthält die Modellbezeichnung des Geräts mit dem unterzeichnet wurde.

Element <SIGNATURE_INFO/ PADTYPE>

Dieses Element enthält die Geräte Typ Nummer mit dem unterzeichnet wurde.

Element <SIGNATURE_INFO/MACADDRESS>

Dieses Element enthält die eindeutige MAC Adresse des PC an dem unterzeichnet wurde.

Element <CERT/CIPHERENC_FILENAME>

Dieses Element enthält den Namen des P12 Zertifikats das zur Signierung des Dokumentes verwendet wurde.

Element <CERT/BIOENC_FILENAME>

Dieses Element enthält den Namen des Zertifikats das zur Verschlüsselung der Biometrie verwendet wurde.

2.2. XML-Struktur mit den Zusatzdaten eines unterschriebenen DigSig Feldes nach der RSA Entschlüsselung

XML-Struktur und Liste der darin enthaltenen Elemente die als Ergebnis eines Aufrufs der Methode **GetReference()** oder **GetReferenceMemory()** zurückgegeben wird.

Beispiel für das Signieren und Verschlüsselt in der signPDF3 Komponente:

```
<?xml version="1.0" encoding="utf-8" ?>
<SIGNATURE_INFO>
    <COMPANY>signotec GmbH</COMPANY>
    <VERSION>8.00.00.031</VERSION>
    <SIGN_TIME>11.02.2009 14:16:11</SIGN_TIME>
    <USERID>Max Mustermann</USERID>
    <MACHINE>OFFICE_1169</MACHINE>
    <MACADDRESS>0017F991241A</MACADDRESS>
    <PADID>1000013325</PADMODEL>
    <PADMODEL>Sigma HID</PADMODEL>
    <PADTYPE>101</PADTYPE>
    <ADDREFERENCE>TRUE</ADDREFERENCE>
    <CERT>
    <CIPHERENC_FILENAME>DEMO1_CERT.p12</CIPHERENC_FILENAME>
    <BIOENC_FILENAME>DEMO2_BIO.cer</BIOENC_FILENAME>
    </CERT>
    <BIOMETRIC_INTEGRITY>
        <DOC-HASH_VALUE>
        45BF6B0822BA17A2E16E1AD222B1073E721731F3E9097F4E0F285DAF
        2D206D63</DOC-HASH_VALUE>
        <DOC-HASH_RECALCEDVALUE>
        45BF6B0822BA17A2E16E1AD222B1073E721731F3E9097F4E0F285DAF
        2D206D63</DOC-HASH_RECALCEDVALUE>
        <DOC-HASH_ALGO>SHA256</DOC-HASH_ALGO>
        <TIMESTAMP>D:20090211141611</TIMESTAMP>
        <MACHINE>OFFICE_1169</MACHINE>
        <MACADDRESS>0017F991241A</MACADDRESS>
        <PADID>1000013325</PADID>
        <PADMODEL>101</PADMODEL>
    </BIOMETRIC_INTEGRITY>
</SIGNATURE_INFO>
```

Beispiel für das Signieren und Verschlüsselt im Pad:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SIGNATURE_INFO>
    <COMPANY>signotec GmbH</COMPANY>
    <VERSION>8.1.2.6</VERSION>
    <SIGN_TIME>D:20180927140151+02'00'</SIGN_TIME>
    <USERID>Max Mustermann</USERID>
    <MACHINE>MAX_MUSTERMANN-PC </MACHINE>
    <MACADDRESS>FCAA1F07B0B1</MACADDRESS>
```

```
    <PADID>1990092846</PADID>
    <PADTYPE>101</PADTYPE>
    <PADMODEL>Sigma HID</PADMODEL>
    <BIOMETRIC_INTEGRITY>
        <DOC-
        HASH_VALUE>1230DAE8D50FDA3452E5536D4262A43595290CBBDE8
        23C9A6E57CE5F0A5A7ECC</DOC-HASH_VALUE>
        <DOC-
        HASH_RECALCEDVALUE>1230DAE8D50FDA3452E5536D4262A43595
        290CBBDE823C9A6E57CE5F0A5A7ECC</DOC-
        HASH_RECALCEDVALUE>
        <DOC-HASH_ALGO>SHA256</DOC-HASH_ALGO>
        <BIO-
        HASH_VALUE>C146F79C211E5676AE13BBF41E950E1A8D95C418282
        C188AA3E5CA6F96355C18</BIO-HASH_VALUE>
        <BIO-HASH_ALGO>SHA256</BIO-HASH_ALGO>
        <TIMESTAMP>20180927140151+02'00'</TIMESTAMP>
        <MACHINE>MAX_MUSTERMANN-PC </MACHINE>
        <USERNAME>Max Mustermann</USERNAME>
        <PADID>1990092846</PADID>
        <PADMODEL>Sigma HID</PADMODEL>
        <CONTENTLENGTH>8781</CONTENTLENGTH>
        <HASHTYPE>COMBINATION</HASHTYPE>
        <RSA-SCHEME>PSS</RSA-SCHEME>
        <RSA-
        SIGNATURE>EaLT2MVA+ez9apYZuARZNglq9pL+OBLGRQa4AZK3
        CjCRosoxq9pzRwARVPuA+XOQgE64vk2ll3ccyNSIYpM4gzHgT4pbN2z
        pC70+Q8X4CBP98AMh2EAh3aoseYJ5jLX+6jSrhUjPBUNYoZ03s5UlBY
        4LJ/WautWpXbFQPO7O8Xg=</RSA-SIGNATURE>
        <RSA-SIGNATURE_STATUS>0</RSA-SIGNATURE_STATUS>
    </BIOMETRIC_INTEGRITY>
</SIGNATURE_INFO>
```

Element <SIGNATURE_INFO>

Das XML-Dokument enthält als Wurzelelement <SIGNATURE_INFO>. Unterhalb dieses Elements existieren folgende Elemente:

Element <COMPANY> (aus dem unverschlüsselten Bereich)

Dieses Element beschreibt den Hersteller der API. Immer „signotec GmbH“

Element <VERSION> (aus dem unverschlüsselten Bereich)

Dieses Element beschreibt die Version des API mit der unterschrieben wurde.

Element <SIGN_TIME> (aus dem unverschlüsselten Bereich)

Dieses Element beschreibt den Zeitpunkt wann unterschrieben wurde.

Element <USERID> (aus dem unverschlüsselten Bereich)

Dieses Element enthält den Namen des angemeldeten Benutzers.

Element <MACHINE> (aus dem unverschlüsselten Bereich)

Dieses Element enthält den Namen des PC an dem unterzeichnet wurde.

Element <MACADDRESS> (aus dem unverschlüsselten Bereich)

Dieses Element enthält die eindeutige MAC Adresse des PC an dem unterzeichnet wurde.

Element <PADID> (aus dem unverschlüsselten Bereich)

Dieses Element enthält die eindeutige Geräte ID des angeschlossenen PADs.

Element <PADMODEL> (aus dem unverschlüsselten Bereich)

Dieses Element enthält den Geräte Model mit dem unterzeichnet wurde.

Element <PADTYPE> (aus dem unverschlüsselten Bereich)

Dieses Element enthält den Geräte Typ mit dem unterzeichnet wurde.

Element <ADDREFERENCE> (optional, aus dem unverschlüsselten Bereich)

Dieses Element ist nur für den internen Gebrauch.

Element <CERT/CIPHERENC_FILENAME> (aus dem unverschlüsselten Bereich)

Dieses Element enthält den Namen des P12 Zertifikates das zur Signierung des Dokumentes verwendet wurde.

Element <CERT/BIOENC_FILENAME> (aus dem unverschlüsselten Bereich)

Dieses Element enthält den Namen des Zertifikates das zur Verschlüsselung der Biometrie verwendet wurde.

Die folgenden XML Daten werden nur beim Setzen des Optionsparameters mit dem Wert „1“ ausgegeben (bei der Verschlüsselung im Pad werden die XML Daten immer ausgegeben):

Element <BIOMETRIC_INTEGRITY/DOC-HASH_VALUE> (aus verschlüsseltem Bereich)

Dieses Element enthält den Hash des Dokumentes vor der Einbringung der digitalen Signatur und Biometrie. Dieser Hash liegt im verschlüsselten Bereich.

Element <BIOMETRIC_INTEGRITY/DOC-HASH_RECALCEDVALUE>

Dieses Element enthält den überprüften Hash des Dokumentes vor der Einbringung der digitalen Signatur und Biometrie. Der Wert muss den gleichen Inhalt haben, wie der „DOC-HASH-VALUE“ aus dem verschlüsselten Bereich.

Element <BIOMETRIC_INTEGRITY/DOC-HASH_ALGO> (aus verschlüsseltem Bereich)

Dieses Element enthält den verwendeten Hash Algorithmus. Interne Verwendung. Immer SHA256 im Moment.

Element <BIOMETRIC_INTEGRITY/BIO-HASH_VALUE> (aus verschlüsseltem Bereich)

Dieses Element enthält den Hash von den biometrischen Daten.

Element <BIOMETRIC_INTEGRITY/BIO-HASH_ALGO> (aus verschlüsseltem Bereich)

Dieses Element enthält den verwendeten Hash Algorithmus. Interne Verwendung. Immer SHA256 im Moment.

Element <BIOMETRIC_INTEGRITY/TIMESTAMP> (aus verschlüsseltem Bereich)

Dieses Element beschreibt den Zeitpunkt wann unterschrieben wurde.

Element <BIOMETRIC_INTEGRITY/MACHINE> (aus verschlüsseltem Bereich)

Dieses Element enthält den Namen des PC an dem unterzeichnet wurde. Der Wert muss den gleichen Inhalt haben, wie der aus dem unverschlüsselten Bereich.

Element <BIOMETRIC_INTEGRITY/USERNAME> (aus verschlüsseltem Bereich)

Dieses Element enthält den Namen des angemeldeten Benutzers.

Element <BIOMETRIC_INTEGRITY/MACADDRESS> (aus verschlüsseltem Bereich)

Dieses Element enthält die eindeutige MAC Adresse des PC an dem unterzeichnet wurde.
Der Wert muss den gleichen Inhalt haben, wie der aus dem unverschlüsselten Bereich.

Element <BIOMETRIC_INTEGRITY/PADID> (aus verschlüsseltem Bereich)

Dieses Element enthält die eindeutige Geräte ID des angeschlossenen PADs.
Der Wert muss den gleichen Inhalt haben, wie der aus dem unverschlüsselten Bereich.

Element <BIOMETRIC_INTEGRITY/PADMODEL> (aus verschlüsseltem Bereich)

Dieses Element enthält den Geräte Typ mit dem unterzeichnet wurde.
Der Wert muss den gleichen Inhalt haben, wie der aus dem unverschlüsselten Bereich.

Element <BIOMETRIC_INTEGRITY/CONTENTLENGTH> (aus verschlüsseltem Bereich)

Dieses Element enthält die Länge des Bereiches über welchen der Dokumenthash errechnet wurde.

Element <BIOMETRIC_INTEGRITY/HASHTYPE> (aus verschlüsseltem Bereich)

Dieses Element enthält den Hash-Typ mit dem die RSA-Signatur errechnet wurde. Es gibt drei Typen „COMBINATION“, „HASH1“ und „HASH2“, für weitere Details siehe bitte die Demoanwendung „SignoAPIDigSigMemDemoCSharp“.

Element <BIOMETRIC_INTEGRITY/RSA-SCHEME> (aus verschlüsseltem Bereich)

Dieses Element enthält die RSA-Schema.

Element <BIOMETRIC_INTEGRITY/RSA-SIGNATURE> (aus verschlüsseltem Bereich)

Dieses Element enthält die RSA-Signatur, für weitere Details siehe bitte die Demoanwendung „SignoAPIDigSigMemDemoCSharp“.

Element < BIOMETRIC_INTEGRITY/RSA-SIGNATURE_STATUS> (aus verschlüsseltem Bereich)

Dieses Element enthält den Status der RSA-Signatur. Es kann die folgenden Werte beinhalten:
„0“ - die RSA-Signatur ist intakt,
„2“ - die RSA-Signatur ist nicht intakt, die Daten wurden unerlaubt manipuliert.

2.3. XML-Struktur mit den allgemeinen Informationen eines leeren (nicht signierten) DigSig Feldes

XML-Struktur und Liste der darin enthaltenen Elemente die als Ergebnis eines Aufrufs der Methode **GetDSFieldInfo()** oder **GetDSFieldInfoMemory()** aus der Komponente SignPDF3 zurückgegeben wird.

Beispiel:

```
<?xml version="1.0" encoding="utf-8" ?>
<digsignatures>
<digsignature Name="sgnsignature_1">
<Name />
<Reason />
<Location />
<ContactInfo />
<Time />
<Page>1</Page>
<Mandatory>false</Mandatory>
<Subfilter />
<Filter />
<HashAlgorithm />
<Status>3</Status>
<CertExpired>0</CertExpired>
<Rect>
    <Left>323</Left>
    <Right>497</Right>
    <Top>271</Top>
    <Bottom>325</Bottom>
</Rect>
<Certificate>
    <Issuer />
    <Serial />
    <PublicKeysize />
    <ValidTo />
    <ValidFrom />
```

```
    <CertErrorStatus />
    <CertErrorStatusCode>0</CertErrorStatusCode>
</Certificate>
</digsignature>
<digsignature Name="sgnsignature_x">
  <Weitere Signaturfelder wie oben>
</digsignatures>
```

Elemente <digsignatures> und <digsignature>

Das XML-Dokument enthält als Wurzelelement <digsignatures>. Unterhalb dieses Elementes existiert für jedes Signaturfeld ein Element <digsignature>. Dieses Element enthält die folgenden Elemente:

Element <Name> (leer)

Das Element Name enthält den Namen des Unterzeichners oder automatisch den Namen auf den das Zertifikat ausgestellt wurde. Standard Anzeige auch im Adobe Acrobat® oder Adobe® Reader.

Element <Reason> (leer)

Der Grund (Reason) enthält die Kurzbeschreibung über den Grund des Unterschreibens. Standard Anzeige auch im Adobe Acrobat® oder Adobe® Reader.

Element <Location> (leer)

Der Ort (Location) enthält eine Kurzbeschreibung über den Aufenthaltsort. Standard Anzeige auch im Adobe Acrobat® oder Adobe® Reader.

Element <ContactInfo> (leer)

Das Element ContactInfo enthält den optional einen Namen oder automatisch die Kontaktinformation aus dem Zertifikat. Standard Anzeige auch im Adobe Acrobat® oder Adobe® Reader unter der Zertifikats-Anzeige.

Element <Time> (leer)

Dieses Element enthält den Zeitstempel an dem das Signaturfeld signiert wurde im Adobe® Format.

Element <Page>

Dieses Element enthält die Seite im PDF auf der sich das Signaturfeld befindet.

Element <Mandatory>

Dieses Element beschreibt, ob es sich um ein Signatur Pflichtfeld handelt.

Elemente <SubFilter> (leer)

Dieses Element beschreibt mit welcher Methode die Signatur eingebracht wurde. Standard ist hier das Format „**adbe.x509.rsa_sha1**“.

Element <Filter> (leer)

Dieses Element beschreibt mit welcher Standard Überprüfungsmethode eine Überprüfung stattfinden kann. Standard ist hier die Methode „**Adobe.PPKLite**“.

Element <HashAlgorithm> (leer)

Dieses Element beschreibt mit welchem Hash-Algorithmus der Hash von einem PDF-Dokument gebildet wurde.

Element <Status> (immer STATUS_EMPTY = 3)

Dieses Element beschreibt den Status des Signaturfeldes bezogen auf die Dokumenten Revision. Folgende Werte können zurückgegeben werden:

STATUS_VALID	0	Signatur ist gültig.
STATUS_CHANGEDAFTERSIGNING	1	Signatur ist gültig (mit Änderung danach).
STATUS_INVALID	2	Signatur ist ungültig.
STATUS_EMPTY	3	Signaturfeld ist leer (nicht unterschrieben).
STATUS_UNKNOWN	4	Kann nicht überprüft werden(unbek. Format).

Hinweis: Muss den Wert 3 haben für ein neues DigSig Feld.

Element <CertExpired> (leer)

Dieses Element beschreibt den Status des Zertifikates. Es wird das Ablaufdatum des eingebetteten öffentlichen Teils des Zertifikates überprüft.

0 = OK.
1 = zeitlich abgelaufenes Zertifikat.

Element <Rect/Left>

Dieses Element enthält die Koordinate der linken Seite des Signaturfeldes in Pixel bezogen auf die obere linke Ecke der PDF Dokumenten Seite.

Element <Rect/Right>

Dieses Element enthält die Koordinate der rechten Seite des Signaturfeldes in Pixel bezogen auf die obere linke Ecke der PDF Dokumenten Seite.

Element <Rect/Top>

Dieses Element enthält die Koordinate der oberen Seite des Signaturfeldes in Pixel bezogen auf die obere linke Ecke der PDF Dokumenten Seite.

Element <Rect/Bottom>

Dieses Element enthält die Koordinate der unteren Seite des Signaturfeldes in Pixel bezogen auf die obere linke Ecke der PDF Dokumenten Seite.

Element <Certificate/Issuer> (leer)

Dieses Element enthält Bezeichnungen des eingebetteten Zertifikates.
Folgende genormten Bezeichnungen können enthalten sein (englisch):

CN The certificate owner's common name

E The certificate owner's e-mail address
T The certificate owner's locality

ST The certificate owner's state of residence

O The organization to which the certificate owner belongs

OU The name of the organizational unit to which the certificate owner belongs
C The certificate owner's country of residence

STREET The certificate owner's street address

ALL The certificate owner's complete distinguished name

Element <Certificate/ Serial> (leer)

Dieses Element enthält die Seriennummer des verwendeten Zertifikates.

Element <Certificate/PublicKeySize> (leer)

Dieses Element enthält die RSA Verschlüsselungsqualität. Zum Beispiel „**2048**" Bits.

Element <Certificate/ValidTo> (leer)

Dieses Element enthält das Gültigkeitsende des verwendeten Zertifikates.

Element <Certificate/ValidFrom> (leer)

Dieses Element enthält den Gültigkeitsbegin des verwendeten Zertifikates.

Element <Certificate/ CertErrorStatus> (leer)

Dieses Element enthält den Fehlerstatus des verwendeten Zertifikates. Der Fehlerstatus ist eine Zeichenkette. Die Zeichenkette beinhaltet die durch Semikolons getrennten Fehlerbeschreibungen des Feldes *DWORD dwErrorStatus;* von der *CERT_TRUST_STATUS Struktur*.

Element <Certificate/ CertErrorStatusCode> (leer)

Dieses Element enthält den Fehler-Status-Code des verwendeten Zertifikates. Der Fehlercode ist der Wert des Feldes *DWORD dwErrorStatus;* von der *CERT_TRUST_STATUS Struktur*.

2.4. XML Struktur mit den benötigten Eingabeinformationen beim Unterschreiben

XML-Struktur und Liste der darin enthaltenen Elemente die als Parameter „bstrXMLSignatureData“ bei Aufruf der Methoden **SignPdfDocument()**, **SignPdfDocumentMemory()**, **PreparePdfDocumentMemory ()** oder **PreparePDFDocumentSigningMemory()** (befüllt) übergeben werden muss.

Beispiel:

```
<?xml version="1.0" encoding="utf-8" ?>
<SIGNATURE_INFO>
  <Name>Max Mustermann</Name>
  <Reason>Ich habe das Dokument gelesen</Reason>
  <Location>Wien</Location>
  <ContactInfo>signotec GmbH</ContactInfo>
  <TimeStamp>
    <Color>808080</Color>
  </TimeStamp>
  <CustomText>
    <Text>Max Mustermann</Text>
  </CustomText>
  <Rect>
    <Left>115</Left>
    <Right>468</Right>
    <Top>520</Top>
    <Bottom>646</Bottom>
  </Rect>
  <Signature>
   <Color>808080</Color>
   <Alignment>0</Alignment>
  </Signature>
  <Page>1</Page>
</SIGNATURE_INFO>
```

Element SIGNATURE_INFO

Das XML-Dokument enthält als Wurzelelement SIGNATURE_INFO.
Unterhalb dieses Elements befinden sich die folgenden Elemente:

Element <Name> (optional)

Das Element Name enthält den optional den Namen des Unterzeichners oder automatisch den Namen auf den das Zertifikat ausgestellt wurde. Standard Anzeige auch im Adobe Acrobat® oder Adobe® Reader.

Element <Reason> (optional)

Der Grund (Reason) enthält die Kurzbeschreibung über den Grund des Unterschreibens. Standard Anzeige auch im Adobe Acrobat® oder Adobe® Reader.

Element <Location> (optional)

Der Ort (Location) enthält eine Kurzbeschreibung über den Aufenthaltsort. Standard Anzeige auch im Adobe Acrobat® oder Adobe® Reader.

Element <ContactInfo> (optional)

Das Element Kontaktinformation (ContactInfo) enthält einen optionalen Namen oder automatisch die Kontaktinformation aus dem Zertifikat. Standard Anzeige auch im Adobe Acrobat® oder Adobe® Reader unter Zertifikats-Anzeige.

Element <Timestamp/Color>

Dieses Element enthält die Farbe für den sichtbaren Zeitstempel im unteren Bereich der Unterschrift. Das Beispiel enthält die RGB Werte „808080“. Dies entspricht einem Grauton.
Der Aufbau ist hexadezimal als RGB Wert.

FF0000	= Rot
00FF00	= Grün
0000FF	= Blau
808080	= Grau

Jede andere Farb-Kombination ist an dieser Stelle möglich.

Element <CustomText/Text> (optional)

Dieses Element enthält einen optionalen Text, der in der Farbe des Zeitstempels im unteren Bereich der Signatur optisch eingebettet wird. Zum Beispiel der Name des Unterzeichners.

Element <Rect/Left> (optional, notwendig wenn das Signaturfeld noch nicht vorhanden ist oder die Position des vorhandenen Signaturfeldes geändert werden soll)

Dieses Element enthält die Koordinate der linken Seite des Signaturfeldes in Punkten (pt) bezogen auf die obere linke Ecke der PDF Dokumenten Seite (siehe Zeichnung).

Element <Rect/Right> (optional, notwendig wenn das Signaturfeld noch nicht vorhanden ist oder die Position des vorhandenen Signaturfeldes geändert werden soll)

Dieses Element enthält die Koordinate der rechten Seite des Signaturfeldes in Punkten (pt) bezogen auf die obere linke Ecke der PDF Dokumenten Seite (siehe Zeichnung).

Element <Rect/Top> (optional, notwendig wenn das Signaturfeld noch nicht vorhanden ist oder die Position des vorhandenen Signaturfeldes geändert werden soll)

Dieses Element enthält die Koordinate der oberen Seite des Signaturfeldes in Punkten (pt) bezogen auf die obere linke Ecke der PDF Dokumenten Seite (siehe Zeichnung).

Element <Rect/Bottom> (optional, notwendig wenn das Signaturfeld noch nicht vorhanden ist oder die Position des vorhandenen Signaturfeldes geändert werden soll)

Dieses Element enthält die Koordinate der unteren Seite des Signaturfeldes in Punkten (pt) bezogen auf die obere linke Ecke der PDF Dokumenten Seite (siehe Zeichnung).

Element <Signature/Color> (optional)

Dieses Element enthält die Farbe für die Signaturlinie der Unterschrift. Das Beispiel enthält die RGB Werte „808080“. Dies entspricht einem Grauton.
Der Aufbau ist hexadezimal als RGB Wert.

FF0000 = Rot
00FF00 = Grün
0000FF = Blau
808080 = Grau

Jede andere Farb-Kombination ist an dieser Stelle möglich.

Element <Signature/Alignment> (optional)

Dieses Element gibt an, welche Ausrichtung die Unterschrift innerhalb des Signaturfeldes haben soll, falls sie dieses nicht vollständig ausfüllt.
Mögliche Werte:

0 = Rechtsbündig
1 = Linksbündig
2 = Zentriert

Element <Page> (optional, notwendig wenn das Signaturfeld noch nicht vorhanden ist)

Dieses Element enthält die Seitennummer im PDF auf der das Signaturfeld erstellt werden soll. Der Name des DigSig-Feldes muss eindeutig im Dokument sein und darf nicht mehrfach auf unterschiedlichen Seiten im Dokument definiert werden. Dies ist eine rechtliche Anforderung, denn Formularfelder können grundsätzlich mit dem gleichen Namen auf unterschiedlichen Seiten definiert werden.

Die zusätzlichen Elemente für **PreparePdfDocumentMemory()**, die beim Signieren und Verschlüsseln im Pad (befüllt) übergeben werden müssen.

Beispiel:

```
<RSAParams>
  <HashType>COMBINATION</HashType>
  <ContentLength>8781</ContentLength>
  <RSAScheme>PSS</RSAScheme>
  <DocAlgorithm>SHA256</DocAlgorithm>
  <BioAlgorithm>SHA256</BioAlgorithm>
 <RSASignature>IaaotszlPmu6OYSb5g0F6MyapmPeAHEeLIIoNDIy9q+FOhr
aLmzf+/z9D5oU1YXvjoOlLAMgLoxAuk957qVoRer/QMspbG2eM3OGmAo+2
pmPIebpzZg9oBTIsAtk4SYSzVdtcVq0bMgkcad7/MmPt/ETKVd/h5em4cKq
GXk+NYYaAEIMeMqITFK3/VVVpAXvLoO7VZtdqf1aw/J5V+ONr+xgGmAEu
Dmlp4r/dqMyZjCVlbdqYBXhThS9kFQHrNMQB8VOp8VgQqp1Ta7zukQ44nC/
Wwskd3i7+ZfMh7fSsjt2b397WhY/wQ0oPy0YvCBaI6V1B6lc4lnTZld1gtEEw
w==</RSASignature>
</RSAParams
```

Element < RSAParams>

Dieses Element enthält Unterelemente beim Signieren und Verschlüsseln im Pad.

Element < HashType>

Dieses Element enthält den Hashtyp.

Element < ContentLength>

Dieses Element enthält die Länge des Dokuments bevor dem Signiervorgang.

Element < RSAScheme>

Dieses Element enthält das RSA-Schema.

Element < DocAlgorithm>

Dieses Element enthält den Dokument Hash-Algorithmus.

Element < BioAlgorithm>

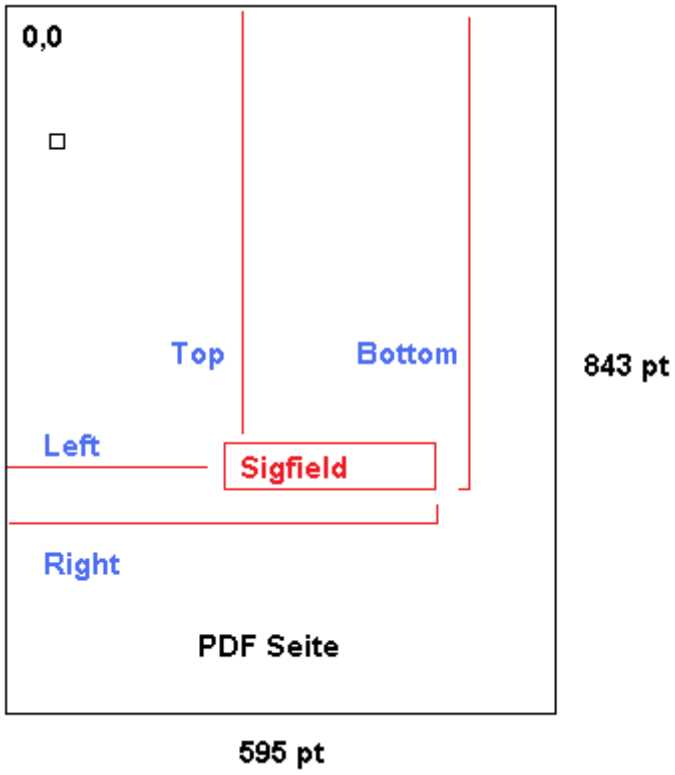
Dieses Element enthält den Biometrie Hash-Algorithmus

Element < RSASignature>

Dieses Element enthält die RSA-Signature.

2.5. Erklärung des Koordinatensystems für das DigSig Signaturfeld

Skizze zur Erklärung des in DigSig-Feldern verwendeten Koordinatensystems anhand einer Seite im PDF Dokument mit der Dimension 843*595 Punkten (pt). Der Ursprung liegt immer in der oberen linken Ecke (0,0). Alle Angaben in Punkten (Points):



Skizze des in DigSig-Feldern verwendeten Koordinatensystems