Dokumentation signoMobileCapture API Win

Software-Komponente für die Erfassung von Unterschriften von mobilen Geräten

signotec GmbH
Am Gierath 20 b
D-40885 Ratingen

+49 (0) 2102/53575-10
www.signotec.com
support@signotec.de

Version 1.4 vom 18.10.2018

Inhaltsverzeichnis

Impressum

1 Dokumentenhistorie

Version	Datum	Bearbeiter	Status/Bemerkung
1.0	13.04.2016	Thorsten Stelter	Dokument erstellt
1.1	02.05.2016	Thorsten Stelter	COM-Schnittstelle ergänzt
1.2	20.07.2016	Thorsten Stelter	Neue und geänderte Events
1.3	29.07.2016	Thorsten Stelter	Änderungen an „SignatureDataReceived“
1.4	18.10.2018	Thorsten Stelter	Eigenschaften „SignaturePenColor“ und „SignatureTransparency“ der MobileDevice-Klasse hinzugefügt

2 Funktionsübersicht

Das signoMobileCapture API für Windows besteht aus einer Assembly für .NET 4.0 Client Profile. Das Assembly bietet die notwendigen Klassen, welche die Kommunikation mit mobilen Geräten ermöglichen. Zusätzlich ist auch ein Windows Forms Control enthalten, welches die Echtzeitdarstellung der Unterschrift am Rechner ermöglicht.

Zusätzlich bietet das Assembly auch eine COM-Schnittstelle an. Die zwei wesentlichen Klassen der .NET-Schnittstelle wurden hier in 4 Interfaces abgebildet welche den gleichen Funktionsumfang anbieten. Statt dem Windows Forms Control steht hier ein ActiveX-Control mit der gleichen Funktion zur Verfügung.

Durch die Verwendung von „Bonjour" ist keinerlei Konfiguration notwendig. Das mobile Gerät muss sich lediglich im gleichen Netzwerk mit dem PC befinden und auf dem mobilen Gerät muss die passende App installiert und gestartet sein.

Zum Einbringen der mittels diesem API erfassten Unterschriften in PDF-Dokumente ist die Verwendung weiterer Komponenten des signoAPI (z.B. SignPDF3) erforderlich. Näheres zu der Verwendung entnehmen Sie bitte den Dokumentationen der entsprechenden Komponenten.

3 Systemvoraussetzungen und Integration

Für die Entwicklung von Anwendungen mit dem signoMobileCapture API ist folgendes notwendig:

- Windows-PC (ab XP mit SP 3)
- .NET 4.0 Client Profile
- VC++ Runtimes 10.0 (VS 2010)
- Bonjour

Achtung: Das „signoAPI" enthält ausschließlich x86 (32 Bit) Komponenten. Daher müssen Anwendungen, die dieses verwenden, auch für x86 (32 Bit) kompiliert werden. Bitte nehmen Sie entsprechende Einstellungen in ihrer Entwicklungsumgebung vor.

3.1 .NET

Das signoMobileCapture API besteht aus folgenden Dateien:

- signoMobileCapture.dll
- signoMobileCapture.xml
- signoSignatureUtils.dll

Die XML-Datei wird nur auf dem Entwicklungsrechner benötigt. Sie enthält Informationen für das IntelliSense von Visual Studio.

In den Projekt-Einstellungen muss lediglich eine Referenz zur „signoMobileCapture.dll" gesetzt werden. Die „signoSignatureUtils.dll" wird nur intern von diesem API verwendet. Sie ist deshalb nicht dokumentiert und es ist auch nicht erforderlich eine Referenz zu dieser DLL zu setzen.

3.2 COM

Das signoMobileCapture API besteht aus folgenden Dateien:

- signoMobileCapture.dll
- signoMobileCapture.tlb
- signoSignatureUtils.dll

Zwei Besonderheiten sind bei der Entwicklung gegen die COM-Schnittstelle zu berücksichtigen.

Zunächst handelt es sich bei der Komponente um ein .NET-Assembly. Zur Registrierung ist somit `regasm.exe` zu verwenden, nicht `regsvr32.exe`. Achten Sie bei der Verwendung von `regasm.exe` darauf, die richtige Version zu verwenden, nämlich die 32Bit-Variante des .NET 4 Frameworks. Auf dem Entwicklungsrechner wurde die Registrierung vom Setup des signoAPI übernommen und muss somit nicht von Hand durchgeführt werden.

Ferner werden in dieser Komponente zahlreiche Events verwendet, welche zwingend ausgewertet werden müssen. Unter C++ muss hierzu eine eigene EventSink implementiert werden. Eine Beispiel-Implementierung ist in der C++-Demo zu finden.

4 Klasse MobileCapture

Die Klasse `MobileCapture` ist der Einstiegspunkt in das signoMobileCapture API. Diese Klasse übernimmt die Bonjour-Kommunikation und stellt für die gefundenen mobilen Geräte passende Instanzen der `MobileDevice` Klasse zur Verfügung.

```
class MobileCapture : IDisposable
```

Anwendung:

```
MobileCapture mobileCapture = new MobileCapture();
```

Bei der Verwendung von COM stehen statt der Klasse die 2 Interfaces `IMobileCapture` und `IMobileCaptureEvents` zur Verfügung, welche einen identischen Funktionsumfang bieten.

Anwendung:

```
HRESULT hresult =
pMobileCapture.CreateInstance(signoMobileCapture::CLSID_MobileCaptureCOM);
if (SUCCEEDED(hresult))
{
    pMobileCapture->StartService();
}
```

4.1 (Statische) Methode SetLicenseKey

Mittels dieser Methode kann der erworbene Lizenzschlüssel für das „signoAPI" übergeben und das API somit freigeschaltet werden. Wird diese Methode nicht aufgerufen bzw. ein ungültiger Lizenzschlüssel übergeben enthält die Unterschrift sowohl im `MobileControl` als auch in den Eigenschaften des `SignatureConfirmed`-Events der `MobileDevice`-Klasse einen Demo-Stempel.

Parameter	Bedeutung
`string licenseKey`	Ein gültiger Lizenzschlüssel für das „signoAPI".
Rückgabewert	**Bedeutung**
-	-

4.1.1 .NET

```
static void SetLicenseKey(string licenseKey)
```

Anwendung:

```
MobileCapture.SetLicenseKey("abc123def456");
```

4.1.2 COM

```
void SetLicenseKey(BSTR licenseKey)
```

Anwendung:

```
pMobileCapture->SetLicenseKey(CComBSTR("abc123def456"));
```

4.2 Methode Dispose

Sofern die `MobileCapture`-Klasse nicht mittels eines using-Blocks (nur .NET) instanziiert wird muss die `Dispose`-Methode aufgerufen werden, sobald die Instanz nicht länger benötigt wird. Hierdurch werden der Bonjour-Dienst gestoppt (sofern dieser vorher gestartet wurde) und weitere Aufräumarbeiten verrichtet.

Parameter	Bedeutung
-	-
Rückgabewert	**Bedeutung**
-	-

4.2.1 .NET

```
void Dispose()
```

Anwendung:

```
mobileCapture.Dispose();
```

4.2.2 COM

```
void Dispose()
```

Anwendung:

```
pMobileCapture->Dispose();
```

4.3 Methode StartService

Diese Methode startet den Bonjour-Dienst. Im Anschluss werden für alle gefundenen mobilen Geräte und alle etwaigen auftretenden Fehler die jeweiligen Events ausgelöst.

Achtung: Es wird dringend empfohlen zuerst passende Handler für alle Events dieser Klasse zu registrieren und erst dann den Service zu starten!

Parameter	Bedeutung
-	-
Rückgabewert	**Bedeutung**
-	-

4.3.1 .NET

```
void StartService()
```

Anwendung:

```
mobileCapture.StartService();
```

4.3.2 COM

```
void StartService()
```

Anwendung:

```
pMobileCapture->StartService();
```

4.4 Event MobileDeviceFound

Dieses Event wird immer dann aufgerufen, wenn vom laufendem Service ein mobiles Gerät gefunden wurde.

4.4.1 .NET

```
event MobileDeviceFoundHandler MobileDeviceFound;
```

Die Klasse `MobileDeviceEventArgs` enthält die folgende Eigenschaft:

Eigenschaft	Bedeutung
`MobileDevice Device`	Instanz der `MobileDevice`-Klasse für das gefundene mobile Gerät.

Anwendung:

```
private void MobileCapture_MobileDeviceFound(Object sender,
    MobileDeviceEventArgs e)
{
    MessageBox.Show(String.Format("New mobile device found: {0}",
        e.Device.Name));
}
```

4.4.2 COM

Das Event hat die Dispatcher-ID 1.

```
void OnMobileDeviceFound(IDispatch* device)
```

Eigenschaft	Bedeutung
`IDispatch* device`	Instanz von `IMobileDevice` für das gefundene mobile Gerät.

Anwendung:

```
HRESULT CCaptureEventSink::Invoke(DISPID dispIdMember, ...)
{
    switch(dispIdMember)
    {
    case 1L: // Mobile Device Found
        {
            IDispatch* mobileDevice = pDispParams->rgvarg[0].pdispVal;

            OnMobileDeviceFound(mobileDevice);
            break;
        }
    }

    return S_OK;
}
```

4.5 Event MobileDeviceLost

Dieses Event wird immer dann aufgerufen, wenn vom laufendem Service ein zuvor gefundenes mobiles Gerät nicht länger erreichbar ist.

4.5.1 .NET

```
event MobileDeviceLostHandler MobileDeviceLost;
```

Die Klasse `MobileDeviceEventArgs` enthält die folgende Eigenschaft:

Eigenschaft	Bedeutung
`MobileDevice Device`	Instanz der `MobileDevice`-Klasse für das nicht mehr erreichbare mobile Gerät.

Anwendung:

```
private void MobileCapture_MobileDeviceLost(Object sender,
    MobileDeviceEventArgs e)
{
    MessageBox.Show(String.Format("Mobile device lost: {0}", e.Device.Name));
}
```

4.5.2 COM

Das Event hat die Dispatcher-ID 2.

```
void OnMobileDeviceLost(IDispatch* device)
```

Eigenschaft	Bedeutung
`IDispatch* device`	Instanz von `IMobileDevice` für das nicht mehr erreichbare mobile Gerät.

Anwendung:

```
HRESULT CCaptureEventSink::Invoke(DISPID dispIdMember, ...)
{
    switch(dispIdMember)
    {
    case 2L: // Mobile Device Lost
        {
            IDispatch* mobileDevice = pDispParams->rgvarg[0].pdispVal;

            OnMobileDeviceLost(mobileDevice);
            break;
        }
    }

    return S_OK;
}
```

4.6 Event BonjourError

Dieses Event wird immer dann aufgerufen, wenn eine der asynchronen Bonjour-Methoden einen entsprechenden Fehler meldet.

Die möglichen Fehler-Codes sind im Kapitel „Aufzählung BonjourErrorCode“ aufgeführt.

4.6.1 .NET

```
event BonjourErrorHandler BonjourError;
```

Die Klasse `BonjourErrorEventArgs` enthält die folgende Eigenschaft:

Eigenschaft	Bedeutung
`BonjourErrorCode ErrorCode`	Der von Bonjour gemeldete Fehlercode.

Anwendung:

```
private void MobileCapture_BonjourError(object sender,
    BonjourErrorEventArgs e)
{
    MessageBox.Show(String.Format("Bonjour error code: {0} ({1})",
        e.ErrorCode.ToString(), (int)e.ErrorCode));
}
```

4.6.2 COM

Das Event hat die Dispatcher-ID 3.

```
void OnBonjourError(int errorCode);
```

Eigenschaft	Bedeutung
`int errorCode`	Der von Bonjour gemeldete Fehlercode.

Anwendung:

```
HRESULT CCaptureEventSink::Invoke(DISPID dispIdMember, ...)
{
    switch(dispIdMember)
    {
    case 3L: // Bonjour error
        {
            int errorCode = pDispParams->rgvarg[0].intVal;

            // Do something with the error code...
            break;
        }
    }

    return S_OK;
}
```

4.7 Event MobileCaptureError

Dieses Event wird immer dann aufgerufen, wenn eine asynchrone interne Methode des laufenden Services eine Exception wirft.

4.7.1 .NET

```
event MobileCaptureErrorHandler MobileCaptureError;
```

Die Klasse `MobileCaptureErrorEventArgs` enthält die folgende Eigenschaft:

Eigenschaft	Bedeutung
`Exception Exception`	Die von der internen Methode geworfene Exception.

Anwendung:

```
private void MobileCapture_MobileCaptureError(object sender,
    MobileCaptureErrorEventArgs e)
{
    MessageBox.Show(e.Exception.Message);
}
```

4.7.2 COM

Das Event hat die Dispatcher-ID 4.

```
void OnMobileCaptureError(BSTR errorMessage)
```

Eigenschaft	Bedeutung
`BSTR errorMessage`	Die Fehlermeldung der von der internen Methode geworfenen Exception.

Anwendung:

```
HRESULT CCaptureEventSink::Invoke(DISPID dispIdMember, ...)
{
    switch(dispIdMember)
    {
    case 4L: // Mobile Capture error
        {
            BSTR errorMessage = pDispParams->rgvarg[0].bstrVal;

            // Do something with the error message...
            break;
        }
    }

    return S_OK;
}
```

4.8 Aufzählung BonjourErrorCode

Die Aufzählung `BonjourErrorCode` enthält die folgenden Werte:

Code	Wert
PollingMode	-65567
NoRouter	-65566
NATPortMappingDisabled	-65565
NATPortMappingUnsupported	-65564
ServiceNotRunning	-65563
Transient	-65562
BadKey	-65561
BadSig	-65560
BadTime	-65559
DoubleNAT	-65558
NATTraversal	-65557
NoSuchKey	-65556
NoAuth	-65555
NoSuchRecord	-65554
Refused	-65553
BadInterfaceIndex	-65552
Incompatible	-65551
Firewall	-65550
Invalid	-65549
NameConflict	-65548
AlreadyRegistered	-65547
NotInitialized	-65545
Unsupported	-65544
BadFlags	-65543
BadState	-65542
BadReference	-65541
BadParam	-65540
NoMemory	-65539
NoSuchName	-65538
Unknown	-65537
NoError	0

5 Klasse MobileDevice

Die Klasse `MobileDevice` repräsentiert ein mobiles Gerät. Sie enthält Methoden zum Herstellen und Trennen einer Verbindung zu diesem Gerät sowie zum Starten und Steuern des Unterschriftenvorgangs. Ferner bietet Sie eine Reihe von Eigenschaften und Events.

Bei der Verwendung von COM stehen statt der Klasse die 2 Interfaces `IMobileDevice` und `IMobileDeviceEvents` zur Verfügung, welche einen identischen Funktionsumfang bieten.

Hinweis: Eine Instanz dieser Klasse kann nicht direkt erzeugt werden. Entsprechende Instanzen für mobile Geräte werden über das `MobileDeviceFound`-Event der `MobileCapture`-Klasse bereitgestellt.

Anwendung:

```
private void MobileCapture_MobileDeviceFound(Object sender,
    MobileDeviceEventArgs e)
{
    MobileDevice mobileDevice = e.Device;
}
```

In COM:

```
void CCaptureEventSink::OnMobileDeviceFound(IDispatch* mobileDevice)
{
    signoMobileCapture::IMobileDevicePtr pMobileDevice = mobileDevice;
}
```

5.1 Eigenschaft Name

Diese Eigenschaft stellt den Namen des mobilen Geräts zur Verfügung.

```
string Name { get; }
```

Anwendung:

```
MessageBox.Show(mobileDevice.Name);
```

In COM:

```
BSTR* name;
pMobileDevice->get_Name(name);
```

5.2 Eigenschaft Width

Diese Eigenschaft stellt die Breite (in Pixeln) des Displays des mobilen Geräts zur Verfügung.

Achtung: Diese Eigenschaft steht nur zur Verfügung, wenn erfolgreich eine Verbindung mit dem Gerät aufgebaut wurde. Wird die Eigenschaft ohne Verbindung zum Gerät abgefragt wird eine Exception geworfen.

```
int Width { get; }
```

Anwendung:

```
Control.Width = mobileDevice.Width;
```

In COM:

```
long* width;
pMobileDevice->get_Width(width);
```

5.3 Eigenschaft Height

Diese Eigenschaft stellt die Höhe (in Pixeln) des Displays des mobilen Geräts zur Verfügung.

Achtung: Diese Eigenschaft steht nur zur Verfügung, wenn erfolgreich eine Verbindung mit dem Gerät aufgebaut wurde. Wird die Eigenschaft ohne Verbindung zum Gerät abgefragt wird eine Exception geworfen.

```
int Height { get; }
```

Anwendung:

```
Control.Height = mobileDevice.Height;
```

In COM:

```
long* height;
pMobileDevice->get_Height(height);
```

5.4 Eigenschaft SignatureResolution

Diese Eigenschaft erlaubt das Auslesen und Setzen der gewünschten Auflösung des Signaturbildes.

Hinweis: Das Signaturbild wird vom `SignatureConfirmed`-Event geliefert.

```
double SignatureResolution { get; set; }
```

Anwendung:

```
SignatureResolution = 250.0;
```

In COM:

```
pMobileDevice->put_SignatureResolution(250.0);
```

5.5 Eigenschaft SignatureWidth

Diese Eigenschaft erlaubt das Auslesen und Setzen der gewünschten Breite des Signaturbildes.

Hinweis: Das Signaturbild wird vom `SignatureConfirmed`-Event geliefert.

```
int SignatureWidth { get; set; }
```

Anwendung:

```
SignatureWidth = 300;
```

In COM:

```
pMobileDevice->put_SignatureWidth(300);
```

5.6 Eigenschaft SignatureHeight

Diese Eigenschaft erlaubt das Auslesen und Setzen der gewünschten Höhe des Signaturbildes.

Hinweis: Das Signaturbild wird vom `SignatureConfirmed`-Event geliefert.

```
int SignatureHeight { get; set; }
```

Anwendung:

```
SignatureHeight = 100;
```

In COM:

```
pMobileDevice->put_SignatureHeight(100);
```

5.7 Eigenschaft SignaturePenColor

Diese Eigenschaft erlaubt das Auslesen und Setzen der gewünschten Farbe der Signatur. Die hier gewählte Farbe wird sowohl von der Echtzeitansicht der Apps (Android und iOS ab v2.2.0) als auch dem Control und dem gerenderten Signaturbild berücksichtigt.

Verfügbar seit: v1.2.6.0

Hinweis: Die Eigenschaft muss vor dem Aufruf der Methode `CaptureSignature` gesetzt werden, da bei diesem Aufruf die Farbinformation mit der App ausgetauscht wird.

```
System.Drawing.Color SignaturePenColor { get; set; }
```

Anwendung:

```
SignaturePenColor = Color.FromKnownColor(KnownColor.Blue);
```

In COM:

```
pMobileDevice->put_SignaturePenColor(0xFF0000);
```

5.8 Eigenschaft SignatureTransparency

Diese Eigenschaft erlaubt die Auswahl, ob der Hintergrund des gerenderten Signaturbilds transparent oder weiß seien soll. Ist der Wert true (Default) gesetzt wird das Bild mit transparentem Hintergrund gerendert.

Verfügbar seit: v1.2.6.0

Hinweis: Der Wert muss vor dem Aufruf der Methode `ConfirmSignature` gesetzt werden, da bei diesem Aufruf das Bild der Unterschrift gerendert und als Event-Argument übergeben wird.

```
bool SignatureTransparency { get; set; }
```

Anwendung:

```
SignatureTransparency = false;
```

In COM:

```
pMobileDevice->put_SignatureTransparency(FALSE);
```

5.9 Methode Dispose

Die `Dispose`-Methode sollte aufgerufen werden, sobald die Instanz nicht länger benötigt wird. Hierdurch werden eine gegebenenfalls noch aufgebaute Verbindung zu diesem Gerät beendet und weitere Aufräumarbeiten verrichtet.

Parameter	Bedeutung
-	-
Rückgabewert	**Bedeutung**
-	-

5.9.1 .NET

```
void Dispose()
```

Anwendung:

```
mobileDevice.Dispose();
```

5.9.2 COM

```
void Dispose()
```

Anwendung:

```
pMobileDevice->Dispose();
```

5.10 Methode Connect

Die `Connect`-Methode stellt eine Verbindung zu dem mobilen Gerät her und führt den Handshake durch. Da diese Vorgänge asynchron ablaufen, liefert diese Methode keinen Rückgabewert und blockiert auch nicht den Programmlauf (bzw. Thread). Ob und wann eine Verbindung aufgebaut werden konnte, kann im Anschluss durch das `DeviceConnected`-Event geprüft werden.

Parameter	Bedeutung
`string pairingCode` `BSTR pairingCode`	Der in der App des mobilen Geräts angezeigte Pairing Code.
Rückgabewert	**Bedeutung**
-	-

5.10.1 .NET

```
void Connect(string pairingCode)
```

Anwendung:

```
mobileDevice.Connect("ABC123");
```

5.10.2 COM

```
void Connect(BSTR pairingCode)
```

Anwendung:

```
pMobileDevice->Connect(CComBSTR("ABC123"));
```

5.11 Methode Disconnect

Die `Disconnect`-Methode beendet die Verbindung zu dem mobilen Gerät.

Parameter	Bedeutung
-	-
Rückgabewert	**Bedeutung**
-	-

5.11.1 .NET

```
void Disconnect()
```

Anwendung:

```
mobileDevice.Disconnect();
```

5.11.2 COM

```
void Disconnect()
```

Anwendung:

```
pMobileDevice->Disconnect();
```

5.12 Methode CaptureSignature

Die `CaptureSignature`-Methode startet den Signaturvorgang auf dem mobilen Gerät. Wenn ein Bestätigungstext übergeben wurde, wird zunächst dieser auf dem mobilen Gerät angezeigt. Der eigentliche Signaturvorgang startet in diesem Falle nur, wenn der Text akzeptiert wurde.

Wird diese Methode aufgerufen, ohne dass eine Verbindung zum Gerät aufgebaut wurde, wird eine Exception geworfen.

Hinweis: Das `SigningStarted`-Event wird immer genau dann aufgerufen, wenn der Signaturvorgang im mobilen Gerät beginnt, im Falle von einem vorgeschalteten Bestätigungstext also genau dann (und auch nur dann), wenn dieser akzeptiert wurde. Es ist also unter Umständen ratsam, entsprechende weitere Programmschritte im entsprechenden Event-Handler vorzunehmen und nicht unmittelbar nach dem Aufruf dieser Methode.

Parameter	Bedeutung
`string confirmationText` `BSTR confirmationText`	Der Bestätigungstext, der im mobilen Gerät angezeigt werden soll.
Rückgabewert	**Bedeutung**
-	-

5.12.1 .NET

```
void CaptureSignature()
void CaptureSignature(string confirmationText)
```

Anwendung:

```
mobileDevice.CaptureSignature();
```

Oder:

```
mobileDevice.CaptureSignature("Your confirmation text...");
```

5.12.2 COM

```
void CaptureSignature(BSTR confirmationText)
```

Anwendung:

```
pMobileDevice->CaptureSignature(CComBSTR("Your confirmation text..."));
```

5.13 Methode RetrySignature

Die `RetrySignature`-Methode verwirft alle bisher erfassten Unterschriftsdaten und löscht die Unterschrift vom Display des mobilen Geräts. Das mobile Gerät verbleibt aber im Signaturvorgang.

Wird diese Methode aufgerufen, ohne dass eine Verbindung zum Gerät aufgebaut wurde, wird eine Exception geworfen.

Parameter	Bedeutung
-	-
Rückgabewert	**Bedeutung**
-	-

5.13.1 .NET

```
void RetrySignature()
```

Anwendung:

```
mobileDevice.RetrySignature();
```

5.13.2 COM

```
void RetrySignature()
```

Anwendung:

```
pMobileDevice->RetrySignature();
```

5.14 Methode CancelSignature

Die `CancelSignature`-Methode verwirft alle bisher erfassten Unterschriftsdaten. Das mobile Gerät verlässt den Signaturvorgang.

Wird diese Methode aufgerufen, ohne dass eine Verbindung zum Gerät aufgebaut wurde, wird eine Exception geworfen.

Hinweis: Das `SigningCanceled`-Event wird sowohl im Anschluss an diesen Methoden-Aufruf aufgerufen, als auch dann, wenn im mobilen Gerät der Abbrechen-Button gedrückt wird. Es wird somit dringend empfohlen entsprechende Programmabläufe im entsprechenden Event-Handler vorzunehmen, da diese dann in beiden Fällen abgearbeitet werden.

Parameter	Bedeutung
-	-
Rückgabewert	**Bedeutung**
-	-

5.14.1 .NET

```
void CancelSignature()
```

Anwendung:

```
mobileDevice.CancelSignature();
```

5.14.2 COM

```
void CancelSignature()
```

Anwendung:

```
pMobileDevice->CancelSignature();
```

5.15 Methode ConfirmSignature

Die `ConfirmSignature`-Methode übernimmt die erfassten Unterschriftsdaten. Das mobile Gerät verlässt den Signaturvorgang. Die erfassten Unterschriftsdaten sowie ein gerendertes Bild der Unterschrift werden im Anschluss über das `SignatureConfirmed`-Event bereitgestellt.

Wird diese Methode aufgerufen, ohne dass eine Verbindung zum Gerät aufgebaut wurde, wird eine Exception geworfen.

Hinweis: Das `SignatureConfirmed`-Event wird sowohl im Anschluss an diesen Methoden-Aufruf aufgerufen, als auch dann, wenn im mobilen Gerät der OK-Button gedrückt wird. Es wird somit dringend empfohlen entsprechende Programmabläufe im entsprechenden Event-Handler vorzunehmen, da diese dann in beiden Fällen abgearbeitet werden.

Parameter	Bedeutung
-	-
Rückgabewert	**Bedeutung**
-	-

5.15.1 .NET

```
void ConfirmSignature()
```

Anwendung:

```
mobileDevice.ConfirmSignature();
```

5.15.2 COM

```
void ConfirmSignature()
```

Anwendung:

```
pMobileDevice->ConfirmSignature();
```

5.16 Event DeviceConnected

Dieses Event wird immer dann aufgerufen, wenn der Versuch, eine Verbindung mit dem mobilen Gerät herzustellen, beendet wurde.

Die möglichen Result-Codes sind im Kapitel „Aufzählung DeviceConnectResult“ aufgeführt.

5.16.1 .NET

```
event DeviceConnectedHandler DeviceConnected;
```

Die Klasse `DeviceConnectedEventArgs` enthält die folgende Eigenschaft:

Eigenschaft	Bedeutung
`DeviceConnectResult Result`	Das Ergebnis des Versuchs, eine Verbindung mit dem mobilen Gerät herzustellen.

Anwendung:

```
private void MobileDevice_Connected(object sender,
    DeviceConnectedEventArgs e)
{
    if (e.Result != DeviceConnectResult.Successful)
    {
        MessageBox.Show(e.Result.ToString());
    }
}
```

5.16.2 COM

Das Event hat die Dispatcher-ID 1.

```
void OnDeviceConnected(int result)
```

Eigenschaft	Bedeutung
`int result`	Das Ergebnis des Versuchs, eine Verbindung mit dem mobilen Gerät herzustellen.

Anwendung:

```
HRESULT CDeviceEventSink::Invoke(DISPID dispIdMember, ...)
{
    switch(dispIdMember)
    {
    case 1L: // Device connected
        {
            int result = pDispParams->rgvarg[0].intVal;

            // Do something with the result code...
            break;
        }
    }

    return S_OK;
}
```

5.17 Event SigningStarted

Dieses Event wird immer dann aufgerufen, wenn der Signaturvorgang im mobilen Gerät gestartet wird. Im Falle eines vorgeschalteten Bestätigungstextes gibt es somit auch Aufschluss darüber, ob und wann dieser vom Bediener akzeptiert wurde.

5.17.1 .NET

```
event EventHandler SigningStarted;
```

Anwendung:

```
private void MobileDevice_SigningStarted(object sender, EventArgs e)
{
    MessageBox.Show("Signature capture on the mobile device has started.");
}
```

5.17.2 COM

Das Event hat die Dispatcher-ID 2.

```
void OnSigningStarted()
```

Anwendung:

```
HRESULT CDeviceEventSink::Invoke(DISPID dispIdMember, ...)
{
    switch(dispIdMember)
    {
    case 2L: // Signing started
        {
            // Do something...
            break;
        }
    }

    return S_OK;
}
```

5.18 Event SignatureRetry

Dieses Event wird immer dann aufgerufen, wenn der Signaturvorgang im mobilen Gerät wiederholt wird.

5.18.1 .NET

```
event EventHandler SigningStarted;
```

Anwendung:

```
private void MobileDevice_SignatureRetry(object sender, EventArgs e)
{
    MessageBox.Show("Signature capture on the mobile device was restarted.");
}
```

5.18.2 COM

Das Event hat die Dispatcher-ID 3.

```
void OnSignatureRetry()
```

Anwendung:

```
HRESULT CDeviceEventSink::Invoke(DISPID dispIdMember, ...)
{
    switch(dispIdMember)
    {
    case 3L: // SignatureRetry
        {
            // Do something...
            break;
        }
    }

    return S_OK;
}
```

5.19 Event SigningCanceled

Dieses Event wird immer dann aufgerufen, wenn der Signaturvorgang im mobilen Gerät abgebrochen wurde, egal ob dies durch den Button im mobilen Gerät oder durch den Aufruf der `CancelSignature`-Methode passierte.

5.19.1 .NET

```
event EventHandler SigningCanceled;
```

Anwendung:

```
private void MobileDevice_SigningCanceled(object sender, EventArgs e)
{
    MessageBox.Show("Signature capture on the mobile device was canceled.");
}
```

5.19.2 COM

Das Event hat die Dispatcher-ID 4.

```
void OnSigningCanceled()
```

Anwendung:

```
HRESULT CDeviceEventSink::Invoke(DISPID dispIdMember, ...)
{
    switch(dispIdMember)
    {
    case 4L: // Signing canceled
        {
            // Do something...
            break;
        }
    }

    return S_OK;
}
```

5.20 Event SignatureConfirmed

Dieses Event wird immer dann aufgerufen, wenn die im mobilen Gerät erfasste Unterschrift bestätigt wurde, egal ob dies durch den Button im mobilen Gerät oder durch den Aufruf der `ConfirmSignature`-Methode passierte.

Hinweis: Die Auflösung, Höhe und Breite des Signaturbildes, welches durch dieses Event geliefert wird, können durch die Eigenschaften `SignatureResolution`, `SignatureWidth` und `SignatureHeight` dieser Klasse beeinflusst werden.

5.20.1 .NET

```
event SignatureConfirmedHandler SignatureConfirmed;
```

Die Klasse `SignatureConfirmedEventArgs` enthält die folgenden Eigenschaften:

Eigenschaft	Bedeutung
byte[] SignData	Die erfassten Daten der Signatur.
int NumSignaturePoints	Die Anzahl der erfassten Punkte der Signatur.
Image Image	Ein gerendertes Bild der Signatur.

Anwendung:

```
private void MobileDevice_SignatureConfirmed(object sender,
    SignatureConfirmedEventArgs e)
{
    MessageBox.Show(e.NumSignaturePoints + " points have been captured.");
    File.WriteAllBytes(Path.Combine(Environment.GetFolderPath(
        Environment.SpecialFolder.Desktop), "SignData.sdb"), e.SignData);
    e.Image.Save(Path.Combine(Environment.GetFolderPath(
        Environment.SpecialFolder.Desktop), "Signature.png"),
ImageFormat.Png);
}
```

5.20.2 COM

Das Event hat die Dispatcher-ID 5.

```
void OnSignatureConfirmed(byte[] signData, int numSignaturePoints,
    byte[] imageData)
```

Eigenschaft	Bedeutung
byte[] signData	Die erfassten Daten der Signatur.
int numSignaturePoints	Die Anzahl der erfassten Punkte der Signatur.
byte[] image	Ein gerendertes Bild der Signatur.

Anwendung:

```
HRESULT CDeviceEventSink::Invoke(DISPID dispIdMember, ...)
{
    switch(dispIdMember)
    {
    case 5L: // Signature confirmed
        {
            // Do something...
            break;
        }
    }

    return S_OK;
}
```

5.21 Event SignatureDataReceived

Dieses Event wird immer dann aufgerufen, wenn im Rahmen der Unterschriftenerfassung neue Punkte im mobilen Gerät erfasst und an den PC übermittelt wurden. Die neu erfassten Punkte werden in einer Liste (bzw. einem Array bei COM) in den Event-Parametern geliefert.

Hinweis: Aus Gründen der Sicherheit enthalten die durch dieses Event bereitgestellten Punkte keine Zeitwerte. Eine Rekonstruktion der biometrischen Daten durch den Mitschnitt dieser unverschlüsselten Informationen ist somit nicht möglich.

5.21.1 .NET

```
event SignatureDataReceivedHandler SignatureDataReceived;
```

Die Klasse `SignatureDataReceivedEventArgs` enthält die folgende Eigenschaft:

Eigenschaft	Bedeutung
`List<SigPoint> SigPoints`	Liste der neu erfassten Signaturpunkte.

Die Punkte werden durch die Klasse `SigPoint` repräsentiert, welche die folgenden Eigenschaften aufweist:

Eigenschaft	Bedeutung
`double X`	Horizontale Koordinate des Signaturpunktes.
`double Y`	Vertikale Koordinate des Signaturpunktes.
`double P`	Druck des Signaturpunktes.

Anwendung:

```
private void MobileDevice_ SignatureDataReceived (object sender,
    SignatureDataReceivedEventArgs e)
{
    MessageBox.Show(String.Format("{0} new signature points captured.",
        e.SigPoints.Count);
}
```

5.21.2 COM

Das Event hat die Dispatcher-ID 6.

```
void OnSignatureDataReceived(byte[] newPoints, int newPointsCount);
```

Eigenschaft	Bedeutung
`byte[] newPoints`	Array der neu erfassten Signaturpunkte.
`int newPointsCount`	Anzahl der neu erfassten Signaturpunkte.

Das Byte-Array enthält pro Signaturpunkt 3 Double-Werte und umfasst somit 24 Byte pro Punkt. Die Werte sind immer in der Reihenfolge X (horizontale Koordinate), Y (vertikale Koordinate) und P (Druck) abgelegt. Ein Beispiel, wie alle Werte aus dem Byte-Array extrahiert werden können, ist in der COM-Demo zu finden.

Anwendung:

```
HRESULT CDeviceEventSink::Invoke(DISPID dispIdMember, ...)
{
    switch(dispIdMember)
    {
    case 6L: // SignatureDataReceived
        {
            CComVariant newPoints = pDispParams->rgvarg[1];
            int newPointsCount = pDispParams->rgvarg[0].intVal;

            // Do something...
            break;
        }
    }

    return S_OK;
}
```

5.22 Event BonjourError

Dieses Event wird immer dann aufgerufen, wenn eine der asynchronen Bonjour-Methoden einen entsprechenden Fehler meldet.

Die möglichen Fehler-Codes sind im Kapitel „Aufzählung BonjourErrorCode“ aufgeführt.

5.22.1 .NET

```
event BonjourErrorHandler BonjourError;
```

Die Klasse `BonjourErrorEventArgs` enthält die folgende Eigenschaft:

Eigenschaft	Bedeutung
`BonjourErrorCode ErrorCode`	Der von Bonjour gemeldete Fehlercode.

Anwendung:

```
private void MobileDevice_BonjourError(object sender,
    BonjourErrorEventArgs e)
{
    MessageBox.Show(String.Format("Bonjour error code: {0} ({1})",
        e.ErrorCode.ToString(), (int)e.ErrorCode));
}
```

5.22.2 COM

Das Event hat die Dispatcher-ID 7.

```
void OnBonjourError(int errorCode);
```

Eigenschaft	Bedeutung
`int errorCode`	Der von Bonjour gemeldete Fehlercode.

Anwendung:

```
HRESULT CDeviceEventSink::Invoke(DISPID dispIdMember, ...)
{
    switch(dispIdMember)
    {
    case 7L: // BonjourError
        {
            int errorCode = pDispParams->rgvarg[0].intVal;

            // Do something with the error code...
            break;
        }
    }

    return S_OK;
}
```

5.23 Event MobileCaptureError

Dieses Event wird immer dann aufgerufen, wenn eine asynchrone interne Methode des laufenden Services eine Exception wirft.

5.23.1 .NET

```
event MobileCaptureErrorHandler MobileCaptureError;
```

Die Klasse `MobileCaptureErrorEventArgs` enthält die folgende Eigenschaft:

Eigenschaft	Bedeutung
`Exception Exception`	Die von der internen Methode geworfene Exception.

Anwendung:

```
private void MobileDevice_MobileCaptureError(object sender,
    MobileCaptureErrorEventArgs e)
{
    MessageBox.Show(e.Exception.Message);
}
```

5.23.2 COM

Das Event hat die Dispatcher-ID 8.

```
void OnMobileCaptureError(BSTR errorMessage)
```

Eigenschaft	Bedeutung
`BSTR errorMessage`	Die Fehlermeldung der von der internen Methode geworfenen Exception.

Anwendung:

```
HRESULT CDeviceEventSink::Invoke(DISPID dispIdMember, ...)
{
    switch(dispIdMember)
    {
    case 8L: // Mobile Capture error
        {
            BSTR errorMessage = pDispParams->rgvarg[0].bstrVal;

            // Do something with the error message...
            break;
        }
    }

    return S_OK;
}
```

5.24 Event SignaturePointsChanged

Dieses Event wird immer dann aufgerufen, wenn zu der internen Liste der Signaturdaten neue Punkte hinzugefügt wurden. Dieses Event wird zur API-Internen Kommunikation mit dem Control benötigt, um dieses zu informieren, dass die Anzeige aktualisiert werden muss. Eine anderweitige Verwendung ist in aller Regel nicht notwendig.

Hinweis: Dieses Event ist in der COM-Schnittstelle nicht enthalten.

```
event EventHandler SignaturePointsChanged;
```

5.25 Aufzählung DeviceConnectResult

Die Aufzählung `DeviceConnectResult` enthält die folgenden Werte:

Code	Wert	Bedeutung
`Successful`	1	Der Handshake mit dem mobilen Gerät war erfolgreich. Die Verbindung wurde somit hergestellt.
`WrongPairingCode`	2	Der an die Connect-Methode übergebene Pairing Code entsprach nicht dem des mobilen Geräts.
`AnotherDeviceConnected`	3	Das mobile Gerät ist bereits mit einem anderen Gerät (PC) verbunden.
`UnknownResultCode`	99	Der vom mobilen Gerät gemeldete Result-Code ist unbekannt. Sollten Sie diesen Wert erhalten, prüfen Sie bitte zunächst, ob es eine aktuellere Version von diesem API gibt.
`None`	0	Der initiale Wert dieser Aufzählung.

6 Klasse MobileControl

Die Klasse `MobileControl` ist ein Windows Forms Control welches die auf dem mobilen Gerät erfasste Unterschrift in Echtzeit darstellt. Sie kann wie gewohnt über den Visual Studio Designer auf dem eigenen Programm-Fenster platziert werden.

Wird COM verwendet kann auf das ActiveX-Control zurückgegriffen werden, welches die gleiche Funktion aufweist.

6.1 Eigenschaft MobileDevice

Diese Eigenschaft erlaubt das Auslesen und Setzen des mobilen Geräts, von dem dieses Control die Unterschriften abbilden soll.

```
MobileDevice MobileDevice { get; set; }
```

Anwendung:

```
mobileControl.MobileDevice = mobileDevice;
```

In COM:

```
pMobileControl->put_MobileDevice(pMobileDevice);
```