Session D-STO1-2

Der Vertrieb von VFP Anwendungen im Windows Store

Uwe Habermann
Uwe@hceood.eu

Einführung

Microsoft hat den Windows Store für VFP Anwendungen geöffnet! Besser gesagt wurde der künftige Microsoft Store für 32bit-Desktop-Anwendungen geöffnet, die gewisse Anforderungen erfüllen müssen.

In dieser Session wird gezeigt wie eine mit VFP erstellte Anwendung für den Store vorbereitet wird. Die Eröffnung eines Stores und die Hürden, die hierbei zu überwinden sind, werden am Beispiel einer Anwendung gezeigt, die erfolgreich im Windows Store platziert wurde.

Als diese Session-Notes geschrieben wurden, hatte Microsoft bereits angekündigt den „Windows Store“ in „Microsoft Store“ umzubenennen. Der Leser dieses Artikels kann den Store daher möglicherweise bereits unter dem Namen „Microsoft Store“ vorfinden.

UWP – die universelle Windows-Plattform

Microsoft öffnet den Windows Store nicht nur für uns VFP Entwickler. Die Idee von Microsoft ist jede Art von Windows-Desktop-Anwendungen über den Windows Store zu vertreiben. Das Interesse von Microsoft ist dabei natürlich über die Verkaufsprovision Geld zu verdienen.

Wichtige Voraussetzung für die Verwendung des Windows Store ist der Einsatz von Windows 10. Dies gilt sowohl für den Entwickler, als auch für die Kunden. Ältere Windows-Versionen werden nicht unterstützt.

Alle Anwendungen aus dem Windows Store laufen in einer eigenen, abgeschotteten Umgebung. Bereits für Windows 8 gab es die Windows Runtime, die eine abgeschottete Laufzeitumgebung für Windows 8 Apps bereitstellte. Für Windows 10 heißt diese Laufzeitumgebung „universelle Windows-Plattform“ oder abgekürzt UWP.

Die Entwicklung von Anwendungen für UWP ist mit Visual Studio 2013 SP 2 oder einer neueren Version von Visual Studio möglich. Als Programmiersprachen stehen C#, Visual Basic .NET, Visual C++ und Javascript zur Verfügung. Die erstellten Anwendungen heißen „Universal Apps“.

Universal Apps laufen grundsätzlich im Sandkasten der universellen Windows-Plattform. Damit soll eine hohe Sicherheit geboten werden, weil diese Apps nur einen eingeschränkten Zugriff auf den Rechner haben. Die Bedienung von Universal Apps soll für berührungsempfindliche Bildschirme optimiert sein, dies ist jedoch keine zwingende Anforderung. Universal Apps werden über den Windows Store vertrieben.

Projekt Centennial

Nun hieß es aber, dass alle Windows Desktop Anwendungen im Windows Store vertrieben werden sollen. Universal Apps sind aber etwas Neues und erfordern eine Neuentwicklung mit einer der oben aufgeführten Programmiersprachen.

Um bestehenden Windows Desktop Anwendungen den Zugang zum Windows Store zu ermöglichen hat Microsoft das Projekt Centennial entwickelt.

Centennial enthält einen Desktop App Konverter. Dieser Konverter liest bestehende MSI-Installationspakete und erstellt APPX-Installationspakete für den Windows Store. Die Voraussetzung ist hier also, dass man über ein MSI-Installationspaket verfügt. MSI-Installationspakete können mit Visual Studio oder mit Produkten von Drittanbietern erstellt werden. So ein Paket haben wir für VFP-Anwendungen in der Regel leider nicht.

Windows 10 SDK

Neben dem automatisierten Desktop App Konverter, den Centennial bietet, gibt es die Möglichkeit der manuellen Erstellung von APPX-Installationspaketen für den Windows Store. Das ist es, was wir als VFP Entwickler brauchen. Die manuelle Erstellung von APPX-Installationspaketen ist mit den Werkzeugen aus dem Windows 10 SDK möglich. Die aktuelle Version kann von dieser Webseite heruntergeladen werden:

https://developer.microsoft.com/de-de/windows/downloads/windows-10-sdk

Bei der manuellen Erstellung eines APPX-Installationspakets kann als Quelle ein Ordner mit fast beliebigen Dateien bereitgestellt werden. Nur Dateien mit der Namenserweiterung OCX sind nicht erlaubt. Die Dateien müssen alle erforderlichen Komponenten enthalten, um eine Windows-Desktop-Anwendung ausführen zu können.

Der große Vorteil von VFP Anwendungen ist, dass sie ohne Installation auf einem beliebigen Windows-Rechner ausgeführt werden können. In den Ordner der mit VFP erstellten Exe-Datei werden die Laufzeitumgebung von VFP sowie ggf. weitere Dateien kopiert und schon kann die Anwendung gestartet werden.

Was ist ein APPX-Installationspaket?

Ein APPX-Installationspaket ist eine Archivdatei. Wenn wir die Namenserweiterung von APPX in ZIP ändern, können wir das Archiv mit einem der üblichen Archivierungsprogramme öffnen. Im Archiv befinden sich zwei XML-Dateien:

[Content_Types].xml

AppxBlockMap.xml

Außerdem befinden sich in dem Archiv alle Dateien, die veröffentlicht werden sollen.

Die beiden XML-Dateien werden mit dem Werkzeug MakeAppx generiert. Ansonsten ist die Aufgabe von MakeAppx die Erstellung der Archivdatei.

Ferner können APPX-Installationspakete mit Produkten von Drittanbietern, wie zum Beispiel installShield erstellt werden.

AppxManifest.xml

Für die Erstellung eines APPX-Installationspakets muss eine Manifest-Datei bereitgestellt werden. In dieser Datei sind die Anwendung und der Hersteller beschrieben sowie aufgeführt was installiert werden soll.

Hier als Beispiel eine gültige Manifest-Datei, wie ich sie für dLaboft verwende:

```
<?xml version="1.0" encoding="utf-8"?>
<Package

xmlns="http://schemas.microsoft.com/appx/manifest/foundation/windo
ws10"

xmlns:uap="http://schemas.microsoft.com/appx/manifest/uap/windows1
0"

xmlns:rescap="http://schemas.microsoft.com/appx/manifest/foundatio
n/windows10/restrictedcapabilities">
      <Identity Name="dLabSoft"
        ProcessorArchitecture="x86"
        Publisher="CN=Habermann Consulting EOOD, O=HCEOOD,
L=Varna, S=Varna, C=Bulgaria"
        Version="6.8.5.0"/>
      <Properties>
        <DisplayName>dLabSoft</DisplayName>
        <PublisherDisplayName>Habermann Consulting
EOOD</PublisherDisplayName>
        <Description>Software fuer das Dentallabor</Description>
        <Logo>HC.jpg</Logo>
      </Properties>
      <Resources>
```

```
        <Resource Language="en-us" />
      </Resources>
      <Dependencies>
        <TargetDeviceFamily Name="Windows.Desktop"
MinVersion="10.0.14316.0" MaxVersionTested="10.0.14316.0" />
      </Dependencies>
      <Capabilities>
        <rescap:Capability Name="runFullTrust"/>
      </Capabilities>
      <Applications>
        <Application Id="dLabSoft" Executable="dLabSoft60.exe"
EntryPoint="Windows.FullTrustApplication">
          <uap:VisualElements
           BackgroundColor="#464646"
           DisplayName="dLabSoft"
           Square150x150Logo="HC150.jpg"
           Square44x44Logo="HC44.jpg"
           Description="Software fuer das Dentallabor" />
        </Application>
      </Applications>
</Package>
```

Alle Logos müssen pixelgenau die erwartete Größe haben und selbstverständlich mit dem angegebenen Namen vorhanden sein.

In der XML-Datei dürfen keine Umlaute verwendet werden.

Der Name des Publisher und der Name der Anwendung müssen exakt mit den im Windows Store verwendeten Namen übereinstimmen.

Die Versionsnummer muss vierstellig angegeben werden. Eine mit VFP erstellte Exe-Datei enthält aber nur eine dreistellige Versionsnummer. Es muss also in jedem Fall „.0“ ergänzt werden, um eine vierstellige Nummer zu erhalten.

Alle Werte sind Pflichtangaben. Bei vielen Werten muss mehr als ein Zeichen eingegeben werden.

Sollte eine dieser Anforderungen nicht erfüllt sein, erscheinen mehr oder weniger unverständliche Fehlermeldungen oder der XML Parser bleibt hängen und muss mit dem Windows Task-Manager beendet werden.

Erstellung eines Installationspakets für den Windows Store

Die erforderlichen Werkzeuge des Windows 10 SDK befinden sich im Ordner

C:\Program Files (x86)\Windows Kits\10

Die Erstellung eines APPX-Installationspakets geschieht mit dem Werkzeug MakeAppx, das von der DOS-Eingabeaufforderung mit Parametern gestartet wird. Die DOS-Eingabeaufforderung muss hierfür mit Administratorrechten gestartet werden. Der Befehl lautet:

> "C:\Program Files (x86)\Windows Kits\10\bin\x86\MakeAppx.exe" pack /d <Name des Ordners mit den bereitgestellten Dateien> /p <Name des zu erstellenden Installationspakets>.appx

Anmerkung: Es gibt auch eine 64bit-Version von MakeAppx. Wir müssen die 32bit-Version verwenden, weil wir ein Installationspaket für eine VFP Anwendung, also eine 32bit-Anwendung erstellen. Dies hat nichts mit der verwendeten Windows-Version zu tun.

In dem Ordner mit den bereitgestellten Dateien muss sich die Datei AppxManifest.xml befinden.

Seit der neuesten Version des Windows 10 SDK befinden sich die Werkzeuge in einem weiteren Unterordner, der die Buildnummer angibt. MakeAppx.exe aus der Befehlszeile von oben befindet sich dann in dem Ordner:

C:\Program Files (x86)\Windows Kits\10\bin\10.0.15063.0\x86\

Mit künftigen Versionen des Windows 10 SDK wird es neue Buildnummern geben.

Der erste Test

Es liegt nahe, das erstellte Installationspaket sofort zu testen, also auszuführen. Wir erhalten die Meldung:

> Fordern Sie vom App-Entwickler ein neues App-Paket an. Dieses Paket ist nicht mit einem vertrauenswürdigen Zertifikat signiert (0x800B0100)

Dem APPX-Installationspaket muss also noch ein Zertifikat hinzugefügt werden, bevor wir es testen können.

Der Upload eines APPX-Installationspakets in den Windows Store ist jedoch ohne Zertifikat möglich. Microsoft fügt selbst ein Zertifikat hinzu.

Zertifikat für den Entwicklungsrechner

Um diese Meldung zu umgehen, muss das Installationspaket signiert werden. Hierfür muss zuerst ein Zertifikat erstellt werden. Für die Erstellung eines Zertifikats wird das Werkzeug MakeCert verwendet. Die DOS-Eingabeaufforderung ist mit Administratorrechten zu starten und die folgenden Befehle sind einzugeben:

```
"C:\Program Files (x86)\Windows Kits\10\bin\x86\MakeCert" -r -h 0 -n "CN=Habermann Consulting EOOD" -eku 1.3.6.1.5.5.7.3.3 -pe -sv dlabsoft.pvk dlabsoft.cer

"C:\Program Files (x86)\Windows Kits\10\bin\x86\pvk2pfx" -pvk dlabsoft.pvk -spc dlabsoft.cer -pfx dlabsoft.pfx
```

Bei der Verwendung des aktuellen Windows 10 SDK ist auch in diesen Pfaden die Buildnummer des SDK einzufügen.

Die Parameter sind für jede mit VFP erstellte Anwendung anzuwenden. Im erscheinenden Dialog sollte kein Kennwort eingeben werden. Das Zertifikat ist nur auf dem Entwicklungsrechner gültig.

Signierung

Mit dem so erstellten Zertifikat soll nun das APPX-Installationspaket signiert werden. Dies geschieht mit dem Befehl:

```
"C:\Program Files (x86)\Windows Kits\10\bin\x86\signtool" sign -f dlabsoft.pfx -fd SHA256 -v .\dlabsoft.appx
```

Nach der Ausführung dieses Befehls erscheinen die Meldungen:

> The following certificate was selected:

```
Issued to: Habermann Consulting EOOD
Issued by: Habermann Consulting EOOD
Expires:   Sun Jan 01 01:59:59 2040
SHA1 hash: BE6733C7B6AC9A5488166F20D7D70BDE99925924

Done Adding Additional Store
SignTool Error: An unexpected internal error has occurred.
Error information: "Error: SignerSign() failed." (-2147024885/0x8007000b)
```

Leider kann das erstellte Zertifikat einem APPX-Installationspaket nicht hinzugefügt werden. Das Zertifikat muss zunächst installiert werden. Dies geschieht durch einen Rechtklick auf die Datei mit der Namenserweiterung CER und die Auswahl „Zertifikat installieren“ aus dem Kontextmenü. Die Signierung ist nur für den eigenen Entwicklungsrechner gültig. Beim Versuch der Installation auf einem anderen Rechner wird das Zertifikat als ungültig zurückgewiesen und die Installation wird abgebrochen.

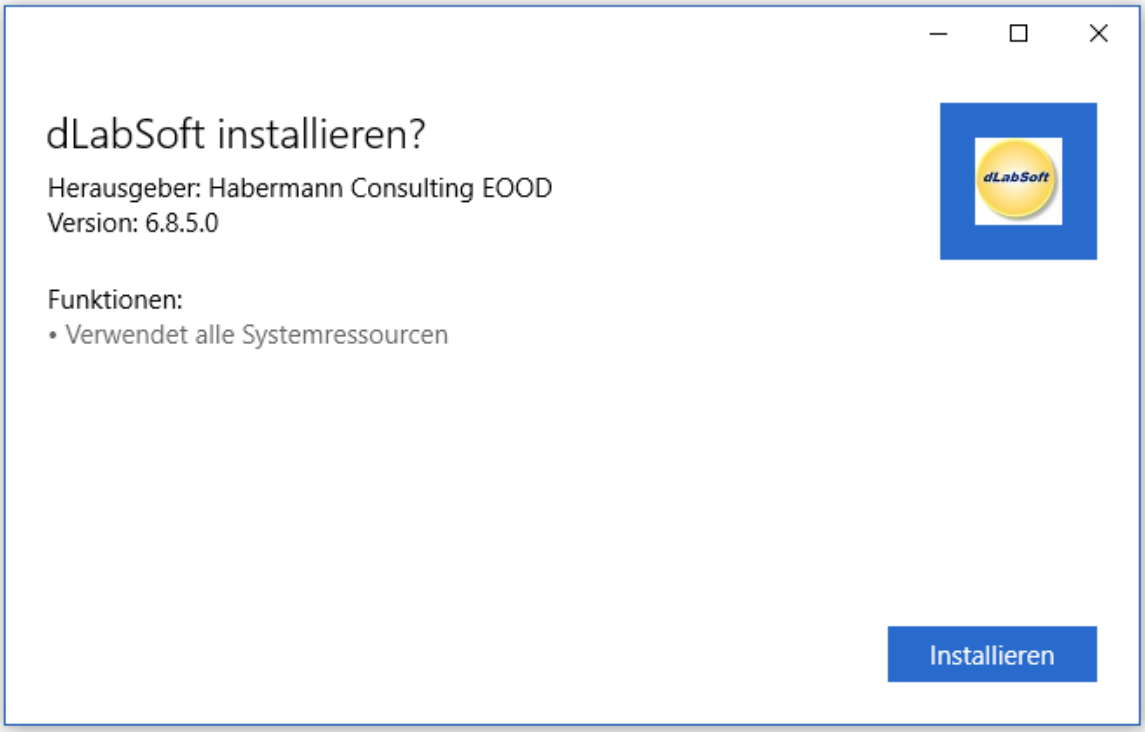
Anschließend kann das APPX-Installationspaket mit Befehl von oben erfolgreich signiert werden.

Wer seine App nicht testen möchte, braucht auch kein Zertifikat erstellen und dem APPX-Installationspaket hinzufügen.

Microsoft fügt den hochgeladenen APPX-Installationspaketen in jedem Fall selbst ein Zertifikat hinzu.

Der erfolgreiche Test

Endlich kann das APPX-Installationspaket ausgeführt und installiert werden.



Startbildschirm des APPX-Installationspakets

Die Anwendung kann unmittelbar nach der Installation gestartet werden. Es wird eine Verknüpfung im Windows-Startmenü erstellt.

Der Windows Store

Der Windows Store eröffnet uns neue Möglichkeiten der Kundengewinnung. Er bietet uns einen neuen Vertriebsweg für unsere VFP Anwendungen. Unsere Produkte können von allen Windows-Benutzern im Windows Store gefunden werden. Dadurch können wir potenziell neue Kunden gewinnen, die wir auf anderen Wegen nicht erreichen könnten.

Ein Entwicklerkonto eröffnen

Wer Anwendungen im Windows Store veröffentlichen will, muss ein Entwicklerkonto eröffnen. Voraussetzung für die Eröffnung ist ein Microsoft-Konto, das hier eingerichtet werden kann:

https://support.microsoft.com/de-de/help/954124

Der Windows Store kann hier eröffnet werden:

https://developer.microsoft.com/de-de/store/register

Zu Beginn der Registrierung muss man sich entscheiden, ob man ein persönliches Konto oder ein Unternehmenskonto eröffnen will. Der Typ des Kontos kann später nicht geändert werden.

Für beide Kontotypen werden die eingegebenen Daten überprüft.

Der Name für den Windows Store

Der Name für den Windows Store ist von großer Wichtigkeit, weil er später an verschiedenen Stellen in exakt gleicher Schreibweise angegeben werden muss. Selbstverständlich muss der Name des Windows Store weltweit eindeutig sein.

Bisher ist es nicht möglich Umlaute in AppxManifest-Dateien zu verwenden. Es macht daher aus heutiger Sicht keinen Sinn im Namen für den Windows Store Umlaute zu verwenden.

Bei Firmenkonten sollte der Name des Windows Store in der Regel dem Namen des Unternehmens entsprechen. Der Name ist in den Installationspaketen sichtbar.

Kosten

Die Registrierungsgebühr für ein persönliches Konto beträgt 14,00 €. Für ein Unternehmenskonto werden 75,00 € fällig. Es handelt sich hierbei um einmalige Gebühren. Es ist keine Verlängerung erforderlich und es fallen keine laufenden Kosten an.

Wenn man über ein Unternehmenskonto Anwendungen verkaufen will, werden die Unternehmensdaten aufwendig verifiziert. In der Regel wird ein Mitarbeiter von Microsoft anrufen, um die Existenz der Firmentelefonnummer zu prüfen.

Wenn man bei Eröffnung des Stores angibt, dass man Anwendungen nur gratis vertreiben will, fällt die Unternehmensprüfung zunächst weg. So kann bei der Eröffnung viel Zeit gespart werden. Zu einem späteren Zeitpunkt kann man dennoch Anwendungen zum Verkauf anbieten. Die Unternehmensprüfung wird dann nachgeholt.

Mit dem Windows Store Geld verdienen

Wer seine Anwendungen über den Windows Store verkaufen möchte, muss eine Auszahlungsmethode eingeben. In der Regel ist hierfür ein Bankkonto erforderlich.

Von den Nettoeinnahmen aus den App-Verkäufen über den Windows Store behält Microsoft 30% als Provision ein. Dies entspricht etwa der Provision, die auch Apple und Google aus ihren Store-Verkäufen einbehalten.

Microsoft verdient am Verkauf unserer Anwendungen also gut mit. Wir müssen aber bedenken, dass wir diese Kunden ohne den Windows Store in der Regel nicht erreicht hätten.

Vereinbarungen

Bevor eine Anwendung veröffentlicht werden kann, müssen einige Vereinbarungen akzeptiert werden. Da ist zunächst die „App-Entwicklervereinbarung – Windows Store“, die für jede Art von App akzeptiert werden muss. Von dieser Vereinbarung können von Zeit zu Zeit Aktualisierungen erscheinen, die dann ebenfalls akzeptiert werden müssen.

Für die Veröffentlichung von UWP-Anwendungen muss außerdem der „Nachtrag zum Centennial-Programm“ akzeptiert werden.

Bemerkenswert ist, dass in den Vereinbarungen an verschiedenen Stellen Spiele erwähnt werden. Geschäftsanwendungen werden in den Vereinbarungen jedoch nicht erwähnt.

Dashboard

Nach der erfolgreichen Eröffnung eines Windows Store kann man sich beim Dashboard anmelden:

https://developer.microsoft.com/de-de/windows

Es erscheint das Dashboard, von dem alle Funktionen des Stores erreichbar sind.

In der Übersicht ist eine Liste der veröffentlichten Anwendungen zu sehen. Von hier kann auch eine neue App erstellt werden.

Der Name der Anwendung für den Windows Store

Der Anwendungsname muss weltweit eindeutig sein. Der Name muss mit der exakt gleichen Schreibweise in der Datei AppxManifest.xml angegeben werden.

Der persönliche Betreuer

Nach dem Hochladen des APPX-Installationspakets erfolgen bei Microsoft zunächst automatische Tests und es findet auch eine Überprüfung auf mögliche Schadsoftware statt.

Anschließend wird die Anwendung von einem Microsoft-Mitarbeiter persönlich getestet. Ich hatte das Glück als persönlichen Tester und somit auch Betreuer und Ansprechpartner Matteo Pagani zu haben. Matteo ist durch sein Blog bekannt:

http://www.qmatteoq.com/

Der persönliche Betreuer testet die Anwendung. Es dürfen keine Laufzeitfehler auftreten. Der Betreuer darf keine unerlaubte Funktionalität finden, wie zum Beispiel eine Produktaktivierung. Die Anwendung muss eine für den Betreuer plausible Funktionalität erfüllen.

Links zum Store

Nach der erfolgreichen Veröffentlichung im Windows Store stellt Microsoft einen Link bereit, zum Beispiel:

https://www.microsoft.com/store/apps/9P9HGVTW29KV

Wird dieser Link auf einem Windows 10 Rechner angeklickt, wird die Windows Store Anwendung mit der Seite der entsprechenden Anwendung geöffnet. Wenn dieser Link auf einem anderen Betriebssystem oder Gerät angeklickt wird, erscheint die Windows Store Seite der Anwendung im Browser.

Preise

Eine App kann kostenlos angeboten werden oder zu einem Preis zwischen 0,99 € und 994,99 € vertrieben werden. Innerhalb dieses Preisrahmens kann aus einer Auswahl von vordefinierten Preisen gewählt werden. Ein höherer Preis als der angegebene Maximalpreis kann nicht verlangt werden.

Eine App kann in einer beliebigen Auswahl aus zurzeit 242 Märkten angeboten werden. Für jeden Markt wird der Verkaufspreis in die jeweilige Landeswährung gerundet. Der Preis kann jedoch für jeden Markt individuell angepasst werden.

https://docs.microsoft.com/de-de/windows/uwp/publish/set-app-pricing-and-availability

---

Vorbereiten der VFP Anwendung

Eine Anwendung, die im Windows Store vertrieben werden soll, muss bestimmte Anforderungen erfüllen. Die Details sind in den Vereinbarungen für den Store nachzulesen.

Die wichtigsten Anforderungen sollen hier aufgeführt werden.

Eine Anwendung, die über den Windows Store vertrieben wurde, darf auch nur über den Windows Store aktualisiert werden. Die Bereitstellung von neuen Versionen, auch wenn es sich nur um Bugfixes handelt, darf nicht mittels anderer Medien oder zum Beispiel per Fernwartung auf einen Kundenrechner übertragen werden. Aktualisierte Programmversionen müssen in jedem Fall im Windows Store bereitgestellt werden.

In einer Anwendung darf keine Produktaktivierung oder Registrierung enthalten sein. Benutzer der Anwendung bleiben anonym, auch wenn sie die Anwendung gekauft haben. Mit der Installation aus dem Windows Store hat der Kunde die Nutzungsrechte erworben, sodass eine Produktaktivierung oder Registrierung nicht erlaubt sind.

Die Freischaltung von Zusatzfunktionen durch einen Verkauf außerhalb des Windows Stores ist nicht erlaubt.

Jegliche Form von Kopierschutz ist nicht erlaubt. Wer eine App aus dem Windows Store heruntergeladen hat, darf diese Anwendung auch auf weiteren Geräten benutzen. Microsoft kümmert sich um die Einhaltung der selbst aufgestellten Regeln.

Anwendungen aus dem Windows Store laufen in einem bereitgestellten Windows 10 Sandkasten. Der Zugriff auf Dateien oder Komponenten außerhalb des Sandkastens ist nicht möglich. Dies betrifft insbesondere das Dateisystem sowie Netzwerkzugriffe.

Die Verwendung von ActiveX-Steuerelementen ist nicht möglich. ActiveX-Steuerelemente können schon nicht in APPX-Installationspaketen enthalten sein. Dagegen ist der Zugriff auf Funktionen in DLL-Dateien möglich, wenn diese DLL-Dateien in das APPX-Installationspaket eingeschlossen werden. Diese DLLs haben ebenfalls keinen Zugriff auf außerhalb des Sandkastens.

Vorbereiten der Anwendung

Die Anwendung hat zur Laufzeit nur auf einen einzigen Ordner Vollzugriff:

C:\Users\<Windows-Anmeldename>\AppData\Roaming

Diesen Ordner können wir aus einer VFP-Anwendung mit dem Befehl

GETENV(“APPDATA“)

ermitteln.

Eine VFP-Datenbank bringt man am Besten in einem ZIP-Archiv in einem Memofeld einer in die Exe-Datei eingeschlossenen Tabelle mit. Zur Laufzeit können die Tabellen in den Ordner AppData entpackt werden.

Änderungen im Projekt

Zunächst müssen wir feststellen, ob unsere Anwendung über den Windows Store installiert wurde.

Nach den Konventionen von Microsoft soll unter dem Ordner AppData ein Ordner mit dem Firmennamen angelegt werden. In meinem Fall heißt dieser Ordner HCEOOD. Unterhalb dieses Ordners soll ein Ordner mit dem Namen der Anwendung angelegt werden, in diesem Fall heißt der Ordner dLabSoft. Unterhalb dieses Ordners wird schließlich ein Ordner Demo angelegt, in dem die Demodatenbank erstellt wird.

```
llFehler = .F.
TRY
	USE vfxpath
	REPLACE importpath WITH vfxpath.importpath
	USE
CATCH
	* Es wird aus APPS vom Windows Store installiert.
	glWindowsStore = .T.
ENDTRY
IF glWindowsStore
	* Demodatenbank usw. aus dlabuwe auspacken
	LOCAL lcAppdata, loCreateArchive
	SET LIBRARY TO VFX.fll ADDITIVE
	SET CLASSLIB TO lib\Vfxappl, lib\Vfxctrl, lib\Vfxform ADDITIVE

	* CD nach appdata
	lcAppdata = GETENV("APPDATA")
	CD (lcAppdata)
	IF NOT DIRECTORY("HCEOOD")
		MD ("HCEOOD")
	ENDIF
	CD ("HCEOOD")
	IF NOT DIRECTORY("dLabSoft")
		MD ("dLabSoft")
		CD ("dLabSoft")
		USE dlabuwe
		LOCATE FOR bezeichnun = "dlabdemo.zip"
		COPY MEMO code TO dlabdemo.zip
		USE
		loCreateArchive = CREATEOBJECT("cArchive")
		loCreateArchive.ExtractFromArchive("dlabdemo.zip","*.*", ".")
		RELEASE loCreateArchive
		MD ("demo")
		CD ("demo")
		USE dlabuwe
		LOCATE FOR bezeichnun = "demodata.zip"
		COPY MEMO code TO demodata.zip
		USE
		loCreateArchive = CREATEOBJECT("cArchive")
		loCreateArchive.ExtractFromArchive("demodata.zip","*.*", ".")
		RELEASE loCreateArchive
		CD .. && jetzt sind wir im dLabSoft-Installationsordner

	ELSE
		CD ("dLabSoft")
	ENDIF
ENDIF
```