4.2.1 COMtoTC

Siehe Funktions- Beschreibung -> *ObjCreateInterface.htm*

Definitionen

Klassenidentifizierer: ???

`CLSID` ist ein global eindeutiger Bezeichner, der ein COM-Klassenobjekt identifiziert**.**

`ProgID` ist ein Registrierungseintrag, der einer `CLSID` zugeordnet werden kann.

`IID` ist ein Akronym für `Interface Identifier`. Eine Zeichenfolge, die den eindeutigen Namen einer Schnittstelle bereitstellt. Eine `IID` ist eine Art `GUID` (`Globally Unique Identifier`; *global eindeutige Kennung*)

Lassen sich mit dem TLB-Viewer ermitteln – für die TC-*IMSIGX10.dll*:

```
CLSID = {6A481000-E531-11CF-A115-00A024158DAF}
IID = {6A481100-E531-11CF-A115-00A024158DAF}
```

Die Funktion wird noch nicht von AutoIt 3 unterstützt.

Alternativ-Funktionalität ggf. vom Quellcode des TLB-Viewers ...

In der aktuellen AutoIt 3.3.16 – neu installiert - läuft leider SciTE nicht …

Die aktuelle Bearbeitung dieses Kapitels wird solange unterbrochen, bis Hilfe aus dem AutoIt-Forum vorliegt.

In der nicht ganz aktuellen Version 3.3.10 wird zumindest die Funktion *ObjCreateInterface* nicht als “unbekannt“ verworfen und auch die integrierte SciTE 3.3.6 läuft noch unter WinXP.

Damit wurden weitere Versuche gemacht.
Im Testcode (-> *Grafik-Elemente_TC.au3*) sind nach der Vorlage:

```
$TCapp = ObjCreateInterface ( "CLSID" , "IID" [, "interface_description", [flag
= True]] )                                                    ; Vorlage
```

folgendes versucht worden:

```
$TCapp = ObjCreateInterface($CLSID, $IID)
$TCapp = ObjCreateInterface("TurboCAD", "Imsigx10.dll")
$TCapp = ObjCreateInterface("Imsigx10.dll", "TurboCAD")
```

alle Varianten funktionieren nicht!

Als nächstes soll der Beispielcode aus der Funktion *ObjCreateInterface* getestet werden.

Dazu wurde dieser etwas modifiziert. Zunächst sollte untersucht werden, ob mit der `ObjCreateInterface`-Funktion wirklich ein Objekt erzeugt werden kann, was mit der `IsObj`–Funktion überprüfbar ist.

```
Func Example()
; Testfunktion aus dem Funktions-Beispiel
; Deklariert die CLSID, IID und Interfacebezeichnung für ITaskbarList.
; Variablen-Definition
Local Const $sCLSID_TaskbarList = "{56FDF344-FD6D-11D0-958A-006097C9A090}"
Local Const $sIID_ITaskbarList = "{56FDF342-FD6D-11D0-958A-006097C9A090}"
Local Const $sTagITaskbarList = "HrInit hresult(); AddTab hresult(hwnd);
      DeleteTab hresult(hwnd); ActivateTab hresult(hwnd); SetActiveAlt
      hresult(hwnd);"
MsgBox(0, "Example-Anf", "Example-Anf")                ; nur zum Test
; Erstellt das Objekt.
Local $oTaskbarList = ObjCreateInterface($sCLSID_TaskbarList,
      $sIID_ITaskbarList, $sTagITaskbarList)
MsgBox(0, "Example-Objekt", $oTaskbarList)      ; nur zum Test -> Variable ohne Inhalt?
If IsObj($oTaskbarList) Then
      MsgBox(0, "Variablen-Abfrage", "Die Variable ist ein Objekt.")
Else
      MsgBox(0, "Variablen-Abfrage", "Die Variable ist kein Objekt.")
EndIf
EndFunc
```

Ergebnis: In diesem Code-Beispiel wird erfolgreich das Objekt `$oTaskbarList` erzeugt.
Die gleiche Verfahrensweise wurde auf die TC-Referenzierung angewendet.

```
Local $CLSID = "{6A481000-E531-11CF-A115-00A024158DAF}", $IID = "{6A481100-E531-
11CF-A115-00A024158DAF}"              ; aus dem TLB-Viewer mit geschweiften Klammern

; Local $CLSID = "6A481000-E531-11CF-A115-00A024158DAF", $IID = "6A481100-E531-
11CF-A115-00A024158DAF"        ; Variante ohne Klammern???

Local $CLSID = "(6A481000-E531-11CF-A115-00A024158DAF)", $IID = "(6A481100-E531-
11CF-A115-00A024158DAF)"              ; Variante mit normalen Klammern

; $TCapp = ObjCreateInterface ( "CLSID" , "IID" [, "interface_description",
            [flag = True]] )        ; Vorlage
```

```
; $TCapp = ObjCreateInterface($CLSID, $IID)     ; Parameter nur $CLSID, $IID
; $TCapp = ObjCreateInterface("TurboCAD", "Imsigx10.dll")
; $TCapp = ObjCreateInterface("Imsigx10.dll", "TurboCAD")
; Local $TCapp = ObjCreateInterface("Imsigx10.Application", "TurboCAD")
                                    ; Variante mit String-Variablen, Name oberster Ebene
$TCapp = ObjCreateInterface($CLSID, $IID, $appl)      ; mit Description-Parameter
```

Alle Varianten funktionieren nicht!

```
MsgBox(0, "TCapp-Anf", "TCapp-Anf")        ; nur zum Test (Haltepunkt)
;
MsgBox(0, "TCapp", $TCapp)                 ; nur zum Test -> Variable ohne Inhalt?

If IsObj($TCapp) Then
      MsgBox(0, "Variablen-Abfrage", "Die Variable ist ein Objekt.")
Else
      MsgBox(0, "Variablen-Abfrage", "Die Variable ist kein Objekt.")
EndIf
```

Mit keiner Variante wird ein Objekt erzeugt.
In der Funktionsbeschreibung gibt es auch keinen Hinweis, dass diese nicht auf eine DLL anwendbar iat.
Im AutoIt-Forum postet **eukalyptus**, dass man "...aus einem DLL-Return Pointer ein COM-Object machen" kann und dieses in der UDF Direct2D-UDF und DirectSound-UDF verwendet wurde.
Leider ist der diesbezügliche Quellcode der UDF Direct2D-UDF unkommentiert, so dass Verständnis für die Funktionalität nur sehr mühsam aufgebaut werden kann.

In der DirectSound-UDF ist es vielleicht etwas übersichtlicher ...

Nach dem bisherigen Erkenntnisstand sind zur Verwendung der ***ObjCreateInterface***-Funktion folgende Verfahrensschritte notwendig (adaptiert aus der UDF DirectSound-UDF)

1) Referenzierung der DLL mit `$DLL_Ref = DLLOpen("Imsigx10.dll")`

Mit der Abfrage

```
If IsObj($DLL_Ref) Then
      MsgBox(0, "Variablen-Abfrage", "Variable ist Objekt")
ElseIf IsHWnd($DLL_Ref) Then
      MsgBox(0, "Variablen-Abfrage", "Variable ist Hwnd")
ElseIf IsPtr($DLL_Ref) Then
MsgBox(0, "Variablen-Abfrage", "Variable ist ein Zeiger")
```

```
ElseIf IsArray($DLL_Ref) Then
MsgBox(0, "Variablen-Abfrage", "Variable ist ein Array")
ElseIf IsBinary($DLL_Ref) Then
MsgBox(0, "Variablen-Abfrage", "Variable ist binär")
ElseIf IsBool($DLL_Ref) Then
MsgBox(0, "Variablen-Abfrage", "Variable ist Boolsch")
ElseIf IsNumber($DLL_Ref) Then
MsgBox(0, "Variablen-Abfrage", "Variable ist numerisch")
ElseIf IsInt($DLL_Ref) Then
MsgBox(0, "Variablen-Abfrage", "Variable ist Ganzzahl")
ElseIf IsFloat($DLL_Ref) Then
      MsgBox(0, "Variablen-Abfrage", "Variable ist Gleitkommazahl")
ElseIf IsString($DLL_Ref) Then
      MsgBox(0, "Variablen-Abfrage", "Variable ist ein String")
Else
      MsgBox(0, "Variablen-Abfrage", "Variable ist unbekannt")
EndIf

DllClose($DLL_Ref)
```

Rückgabewert `$DLL-Ref` = kein Objekt, kein Array -> ein numerischer Wert: 0
entgegen der Angaben in der Hilfe, wo -1 bei Fehler angegeben ist

In TLB-Viewer wird `DLLOpen()` nur verwendet, um Windows-eigene DLLs zu referenzieren.

Möglicherweise wird die DLL nicht gefunden …?

Dazu die Versuche:

- die `Imsigx10.dll` in den Arbeitsordner *TCtoGerber-Converter* kopiert -> ohne Erfolg

- vom Versuchs-Skript mit *Built* eine .exe gemacht und diese in den *TC-Program*-Ordner kopiert

->	nach `DLLOpen` jetzt	0 1, d.h. kein Fehler!
	nach `DLLCall` jetzt	0 0, d.h. kein Fehler!
	Mit *SciTEnew*	3 0, d.h. Funktion in der DLL nicht gefunden

Damit dann weiter mit der Funktion `ObjCreateInterface` -> siehe dort (3)

2) Pointer zur DLL mit `$dllPointer = DLLCall($DLL-Ref,` diverse weitere Parameter`)`

3) Objektbildung mit
`$TCapp = ObjCreateInterface ($dllPointer, - 2 weitere Parameter - )`

Warum die Referenzierung mit den Parametern `$CLSID` und `$IID` - wie im Funktions-Beispiel angegeben - nicht funktioniert, ist damit noch nicht erklärt.

Mit der unter (1) erfolgten Referenzierung ergibt

-> `$TCapp`: 3 1, d.h. Fehler und wieder kein Objekt!

Ein letzter Versuch mit
`$TCapp = ObjCreateInterface($CLSID, $IID)`
wobei zunächst (1) die Variablen `$CLSID, $IID` direkt so eingesetzt wurden, wie sie im TLB-Viewer stehen (mit geschweiften Klammern)
und anschließender exe-Erzeugung und diese in den *TC-Program*-Ordner kopiert

-> `$TCapp`: 1 1, d.h. wieder Fehler und kein Objekt

Mit (2) Variable
`$IID = _WinAPI_GUIDFromString("{6A481100-E531-11CF-A115-00A024158DAF}")`

weil in der Hilfe zu steht: *Stringrepräsentation des Interface-Identifiers*.

-> `$TCapp`: 1 1, kein Objekt

auch mit der umgekehrten Variante `_WinAPI_StringFromGUID` ist das gleiche Ergebnis.

Letzter Versuch: beide Variablen mit `_WinAPI_GUIDFromString`

-> `$Tcapp`: 3 1, kein Objekt

Leider steht in der Fehlerbeschreibung von `ObjCreateInterface` nur:
Setzt das @error Flag auf ungleich null, was 1, bzw. 3 bedeutet ist unklar.