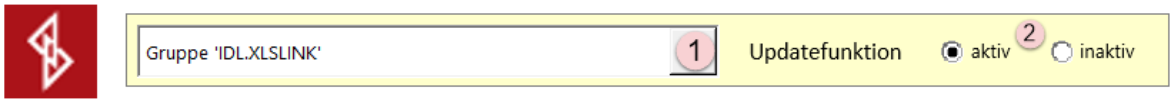
IDL.XLSLink – Makrofähigkeit

Funktionsaufruf per VBA-Makro

Tabellenblatt ‘ID Funktionen‘

Hier ist eine Routine hinterlegt, mit der die verschiedenen Funktionen ausgeführt werden können.



Click auf diese Schaltfläche startet die Prozedur.

1. Auswahl der gewünschten Funktion des XLSLink.
   - Der Funktionsaufruf startet nicht, solange bestimmte Bedingungen nicht erfüllt sind.
   - Bitte die entsprechenden Excel ‘DisplayAlerts‘ beachten.
2. Option für die Update-Funktion.
   - Updatefunktion aktiv = XLSLINK Update Funktion (Open Update, 201) wird ausgeführt.
   - Updatefunktion inaktiv = XLSLINK Update Funktion (Block Update, 200) wird ausgeführt.
   - Soll eine Arbeitsmappe z.B. aktualisiert werden, muss die Updatefunktion aktiv sein, da sonst die IDL-Formeln nicht korrekt rechnen und dargestellt werden.

Voraussetzungen für bestimmte Funktionen:

CallID	XLSLink - Funktion	Beschreibung	Bezeichnung XLSLink
	Gruppe 'Aktualisieren'		
2	selektierter Bereich	Bereich atualisieren	BereichAktualisieren
3	Tabellenblatt	Arbeitsblatt aktualisieren	ArbeitsblattAktualisieren
4	Arbeitsmappe	Gesamtes Dokument aktualisieren	DokumentAktualisieren
	Feld/Auswahlliste v		
38	selektierter Bereich	IDLCombo-Bereich atualisieren	IDLCombo_BereichAktualisieren
39	Tabellenblatt	IDLCombo-Arbeitsblatt aktualisieren	IDLCombo_ArbeitsblattAktualisieren
40	Arbeitsmappe	IDLCombo-Gesamtes Dokument aktualisieren	IDLCombo_DokumentAktualisieren
27	Protokoll	Fortschritt zeigen (wieviele offenen bzw beantwortete Anfragen)	ShowProgressInfo
	Gruppe 'Archiv'		
	Verpacken v		
19	selektierter Bereich	Alle markierten Formeln verpacken	PackAllFormulasCurrentSelection
20	Tabellenblatt	Alle Formeln im aktuellen Arbeitsblatt verpacken	PackAllFormulasCurrentWorksheet
21	Arbeitsmappe	Alle Formeln in Arbeitsmappe verpacken	PackAllFormulasCurrentWorkbook
	Entpacken v		
22	selektierter Bereich	Alle markierten Formeln entpacken	UnpackAllFormulasCurrentSelection
23	Tabellenblatt	Alle Formeln im aktuellen Arbeitsblatt entpacken	UnpackAllFormulasCurrentWorksheet
24	Arbeitsmappe	Alle Formeln in Arbeitsmappe entpacken	UnpackAllFormulasCurrentWorkbook
	Gruppe 'Export'		
7	selektierter Bereich	Bereich nach Konsis exportieren	ExportToKonsisCurrentSelection
8	Tabellenblatt	Tabellenblatt nach Konsis exportieren	ExportToKonsisCurrentWorksheet
9	Arbeitsmappe	Gesamte Arbeitsmappe nach Konsis exportieren	ExportToKonsisCurrentWorkbook
48	Export zurücksetzen	Alle IDLSetValue-Formeln zurücksetzen	ExportToKonsisResetFormula

- Für diese Funktionen (grün hinterlegt) muss / darf maximal eine Datei mit entsprechenden IDL - Formeln geöffnet sein.
- Damit z.B. die Funktion "Aktualisierung Bereich" oder "Aktualisierung Tabellenblatt" per Makro aufgerufen werden kann, muss in der zweiten Datei ein entsprechender Bereich oder ein Tabellenblatt ausgewählt werden.

Übersicht aller möglichen XLSLink - Funktionen und deren ID

CallID	XLSLink - Funktion	Beschreibung	Bezeichnung XLSLink
	Gruppe 'IDL.XLSLINK'		
1	Formeleditor aufrufen	Formel in AppConfig anzeigen	Edit_Bezug_Formel
5	Abfragen rücksetzen	Alle abfragen löschen	CancelAllRequests
17	Info-Anzeige	Excel Info-Anzeige aufrufen	ShowExcelLog
	Gruppe 'Aktualisieren'		
2	selektierter Bereich	Bereich atualisieren	BereichAktualisieren
3	Tabellenblatt	Arbeitsblatt aktualisieren	ArbeitsblattAktualisieren
4	Arbeitsmappe	Gesamtes Dokument aktualisieren	DokumentAktualisieren
	Feld/Auswahlliste v		
38	selektierter Bereich	IDLCombo-Bereich atualisieren	IDLCombo_BereichAktualisieren
39	Tabellenblatt	IDLCombo-Arbeitsblatt aktualisieren	IDLCombo_ArbeitsblattAktualisieren
40	Arbeitsmappe	IDLCombo-Gesamtes Dokument aktualisieren	IDLCombo_DokumentAktualisieren
27	Protokoll	Fortschritt zeigen (wieviele offenen bzw beantwortete Anfragen)	ShowProgressInfo
	Gruppe 'Archiv'		
	Verpacken v		
19	selektierter Bereich	Alle markierten Formeln verpacken	PackAllFormulasCurrentSelection
20	Tabellenblatt	Alle Formeln im aktuellen Arbeitsblatt verpacken	PackAllFormulasCurrentWorksheet
21	Arbeitsmappe	Alle Formeln in Arbeitsmappe verpacken	PackAllFormulasCurrentWorkbook
	Entpacken v		
22	selektierter Bereich	Alle markierten Formeln entpacken	UnpackAllFormulasCurrentSelection
23	Tabellenblatt	Alle Formeln im aktuellen Arbeitsblatt entpacken	UnpackAllFormulasCurrentWorkshee
24	Arbeitsmappe	Alle Formeln in Arbeitsmappe entpacken	UnpackAllFormulasCurrentWorkbook
	Gruppe 'Export'		
7	selektierter Bereich	Bereich nach Konsis exportieren	ExportToKonsisCurrentSelection
8	Tabellenblatt	Tabellenblatt nach Konsis exportieren	ExportToKonsisCurrentWorksheet
9	Arbeitsmappe	Gesamte Arbeitsmappe nach Konsis exportieren	ExportToKonsisCurrentWorkbook
48	Export zurücksetzen	Alle IDLSetValue-Formeln zurücksetzen	ExportToKonsisResetFormula
	Datei v		
6	Kopie ohne Formeln erstellen	Tabelle in Text-Datei mit Zeichencode 1252 exportieren	ExportToCleanFile
	Gruppe 'Werkzeuge'		
33	Nullzeichen ergänzen	Nullzeichen ergänzen	PadLeftCurrentSelection
47	Excel Password	Excel-Passwörter verwalten	EditExcelPassword
	Feld/Auswahlliste v		
28	Einfügen aktive Zelle	Auswahlliste (IDLCombo) in Zelle einfügen	AddIDLCombo
29	entfernen	Auswahlliste (IDLCombo) aus Zelle löschen	RemoveIDLCombo

CallID	XLSLink - Funktion	Beschreibung	Bezeichnung XLSLink
	Einstellungen (Zellinhalt) v		
	Excel Berechnungsoption v		
25	Automatisch	Wechsel ExcelApplicationCalculationMode to xlCalculationAutomatic	ExcelCalculationModeAutomatic
26	Manuell	Wechsel ExcelApplicationCalculationMode to xlCalculationManual	ExcelCalculationModeManual
51	Semiautomatisch	Wechsel ExcelApplicationCalculationMode to xlCalculationSemiautomatic	ExcelCalculationModeSemiautomatic
35	Periode im Datumformat an/aus	Einstellung "Periode als Datumsformat" ändern	TogglePeriodAsDateFormat
46	Lange Fehlermeldungen	Fehlertexte in Excel Lang oder Kurz zeigen	ToggleShowLongErrorMessages
37	Nur IDL Formeln an/aus	Einstellung "Nur IDL Formeln" ändern	TogglePreciseFormulaEvaluation
	IDL Zwischenablage v		
13	Zelle merken	Zwischenablage für Zell-Adressen füllen	ClipboardAddCurrentSelection
45	anzeigen	Zwischenablage für Zell-Adressen zeigen	ClipboardShow
14	löschen	Zwischenablage für Zell-Adressen löschen	ClipboardClear
	Extras v		
	Externe Verknüpfungen v		
42	prüfen	Externe Verknüpfungen prüfen	ExternalLinksCollectAndShow
43	anzeigen	Externe Verknüpfungen anzeigen	ExternalLinksShow
44	Prüfen beim Laden an/aus	Einstellung "Check Externe Verknüpfungen" ändern	ToggleCheckExternalLinks
	IDL.XLSLINK Hauptanwendung		
16	Status	IDL.XLSLINK Hauptanwendung Statusinformation ausgeben	Status
15	Verbinden	IDL.XLSLINK Hauptanwendung verbinden (NamedPipe - AddIn nach IDLOffice)	Verbinden
49	Anmeldung: IDL.KONSIS	Anmeldung Datenbanken ausführen	PerformDBLogin
50	Informationen zu IDL.XLSLINK	IDL System Info anzeigen	ShowIDLSystemInfo
	Allgemeine Funktionen		
111	Get Current Status	Aktuellen Status abfragen (keine detailierte Beschreibung im Code)	GetCurrentStatus
200	Block Update	XLSLink Update Funktion deaktivieren (keine detailierte Beschreibung im Code)	BlockUpdate
201	Open Update	XLSLink Update Funktion aktivieren (keine detailierte Beschreibung im Code)	OpenUpdate
	Extras v		
	Konvertierung v		
18	(Connector) abschließen	Konvertierung (Alt-Connector -> IDLOffice) abschließen (Nacharbeiten)	FinishConversion
36	(Connector) abschließen AddIn	Konvertierung (Alt-Connector -> IDLOffice) abschließen (Nacharbeiten) - innerhalb AddIn	FinishConversionFromAddIn
41	Format zurücksetzen	Konvertierung Zellenformatierung zurücksetzen	ResetCellFormat
	Sonstige (nicht in Ribbon enthalten)		
10	Dokumentenvariable bearbeiten	Dokumentenvariable bearbeiten	DocVarEdit
11	Dokumentenvariable löschen	Dokumentenvariable löschen	DocVarDelete
12	Update Macros	Makros in aktueller Datei aktualisieren	UpdateMacros
31	RefreshRange Bezüge berücksichtigen	Beim RefreshRange Bezüge berücksichtigen	UpdateCellContentTypeBezug
32	RefreshRange IDLCombo berücksichtigen	Beim RefreshRange IDLCombo berücksichtigen	UpdateCellContentTypeCombo
34	Liste Konsis Exporte	Formular mit Konsis-Exporten zeigen	ShowExportJobList

Hinweise für den VBA - Entwickler/Nutzer

- Die Programmierung und Nutzung von Makros liegt generell in Ihrer Verantwortung.
- Wir können Ihnen nur die Möglichkeit der Nutzung von Makros zur Verfügung stellen.
- Nachfolgende VBA - Codezeilen sind Beispiele, wie die verschiedenen Funktionen des XLSLink per Makro aufgerufen werden können.
- Eine individuelle Anpassung an Ihre Umgebung / Datei ist gegebenenfalls / wahrscheinlich nötig.
- Den folgenden VBA - Code finden Sie am Ende dieses Dokuments als '.bas' – Dateien, ebenso wie eine kurze Anleitung über den Import in ein VBA - Projekt.
- Vorkenntnisse im Bereich der VBA - Programmierung sind erforderlich.

- Die **beiden** folgenden Codezeilen ersetzen den alten Funktionsaufruf des IDLConnectors.
  - Beide Codezeilen **müssen** immer zusammen verwendet werden.
  - CallID = entsprechende Nummer der Funktion lt. Übersicht.

```
Call Application.Run("IDLCall", CallID)
Call WaitForReady
```

- Folgende Prozedur wird **immer** in Verbindung mit dem Aufruf der Funktionen benötigt.
  - Sie **muss** korrekt einmalig in dem jeweiligen VBA - Projekt vorhanden sein.
  - Sie prüft den Status / Fortschritt des XLSLink und geht erst wieder in die Prozedur zurück, wenn die Funktion komplett ausgeführt wurde.
  - Die Sekunden können entsprechend angepasst werden, ist aber nicht erforderlich.

```
Sub WaitForReady()

Dim tValue As Boolean

tValue = True

While tValue <> False
   VBA.DoEvents
   tValue = Application.Run("IDLCall", 111)
   If tValue <> False Then
      Application.Wait (Now + TimeValue("00:00:05"))
   End If
Wend

End Sub
```

Die wichtigsten Prozeduren für fest definierte XLSLink - Funktionen.

- Sie können diese Prozeduren direkt in ein VBA - Projekt einkopieren.
- Prozedur **WaitForReady()** muss als Prozedur korrekt in dem VBA - Projekt vorhanden sein.
- Diese Prozeduren können entweder aus einer bestehenden Prozedur aufgerufen werden.
- Oder der Aufruf erfolgt als ‘zugewiesenes Makro‘ an einer Schaltfläche in Excel.

- Gruppe ‘**Aktualisieren**‘:

```
Sub Aktualisieren_SelBereich()

Call Application.Run("IDLCall", 2)
Call WaitForReady

End Sub

Sub Aktualisieren_Tabellenblatt()

Call Application.Run("IDLCall", 3)
Call WaitForReady

End Sub

Sub Aktualisieren_Arbeitsmappe()

Call Application.Run("IDLCall", 4)
Call WaitForReady

End Sub
```

- Gruppe ‘**Verpacken**‘:

```
Sub Verpacken_SelBereich()

Call Application.Run("IDLCall", 19)
Call WaitForReady

End Sub

Sub Verpacken_Tabellenblatt ()

Call Application.Run("IDLCall", 20)
Call WaitForReady

End Sub

Sub Verpacken_Arbeitsmappe()

Call Application.Run("IDLCall", 21)
Call WaitForReady

End Sub
```

- Gruppe ‘**Entpacken**‘:

```
Sub Entpacken_SelBereich()

Call Application.Run("IDLCall", 22)
Call WaitForReady

End Sub

Sub Entpacken_Tabellenblatt ()

Call Application.Run("IDLCall", 23)
Call WaitForReady

End Sub

Sub Entpacken_Arbeitsmappe()

Call Application.Run("IDLCall", 24)
Call WaitForReady

End Sub
```

- Gruppe ‘**Export**‘:
  - Mit diesen Prozeduren wird durch die Hauptanwendung ein Fenster mit eigener Steuerung aufgerufen.
  - Ein Beispiel finden Sie unter der Nummer 3) auf Seite 8.

```
Sub Export_SelBereich()

Call Application.Run("IDLCall", 7)
Call WaitForReady

End Sub

Sub Export_Tabellenblatt ()

Call Application.Run("IDLCall", 8)
Call WaitForReady

End Sub

Sub Export_Arbeitsmappe()

Call Application.Run("IDLCall", 9)
Call WaitForReady

End Sub
```

Weitere Prozeduren für fest definierte XLSLink - Funktionen.

- Diese Prozeduren werden nicht zwingend benötigt.
- Sollten über bestehende Makros größere Veränderungen an einem Dokument vorgenommen werden, kann es ratsam sein, alle automatisch durch XLSLink durchgeführten ‘Aktualisierungen‘ mit **Block_Update()** zu verhindern.
- Wird die Prozedur **Block_Update()** verwendet **muss** danach die Prozedur **Open_Update()** passend dazu ausgeführt werden.
- Ansonsten werden **keine** IDL - Formeln mehr korrekt berechnet.

```
Sub Block_Update()

Call Application.Run("IDLCall", 200)
Call WaitForReady

End Sub

Sub Open_Update()

Call Application.Run("IDLCall", 201)
Call WaitForReady

End Sub
```

- Manchmal ist es nötig die korrekte Ausführung einer Prozeduren mit einer WAIT-Funktionen entsprechend zu steuern.
  - Hier unterbrechen wir die Prozedur für 3 Sekunden

```
Sub WAIT_Funktion()

Application.Wait (Now + TimeValue("00:00:03"))

End Sub
```

Beispielprozeduren für den Aufruf von XLSLINK – Funktionen

- Hier einige Beispiele, wie mehrere Funktionen nacheinander ausgeführt werden können.
- Sie sollen eine Idee vermitteln, wie der Aufruf funktionieren kann.
- Anpassungen an Ihre bestehenden Dokumente müssen mit Sicherheit vorgenommen werden.
- Diese Prozeduren hier sind einmal als eigenständige Prozeduren geschrieben, sowie mit Aufruf von weiteren Prozeduren.
- Prozedur **WaitForReady()** muss als Prozedur korrekt in dem VBA - Projekt vorhanden sein.
- Diese und weitere Prozeduren finden Sie in der angefügten Datei.

1) Das erste Beispiel zeigt einen dynamischen Aufruf von Funktion, hier 'IDL Formel' - Editor.

```
Sub Dynamischer_IDLFunktionsaufruf()

Dim IDLCallID as Integer     'Variable definieren

IDLCallID = Tabelle1.Range("A1")     Wert für Variable initialisieren (z.B. Zellinhalt 'A1'=1)

IDLFunktion_aufrufen IDLCallID     Aufruf Subprozedur mit Übergabe der Variable

End Sub

Sub IDLFunktion_aufrufen (CallID As Integer)     'Funktionsaufruf mit übergebener Variablen

Call Application.Run("IDLCall", CallID)
Call WaitForReady

End Sub
```

2) Mit dieser Prozedur wird ein Tabellenblatt mit verpackten Formeln aktualisiert.
(Eigenständige Prozedur).

```
Sub Tabellenblatt_Update()

Call Application.Run("IDLCall", 23)     'erst Tabellenblatt entpacken
Call WaitForReady

Call Application.Run("IDLCall", 3)     'danach Tabellenblatt aktualisieren
Call WaitForReady

Call Application.Run("IDLCall", 20)     'zu Letzt Tabellenblatt wieder verpacken
Call WaitForReady

End Sub
```

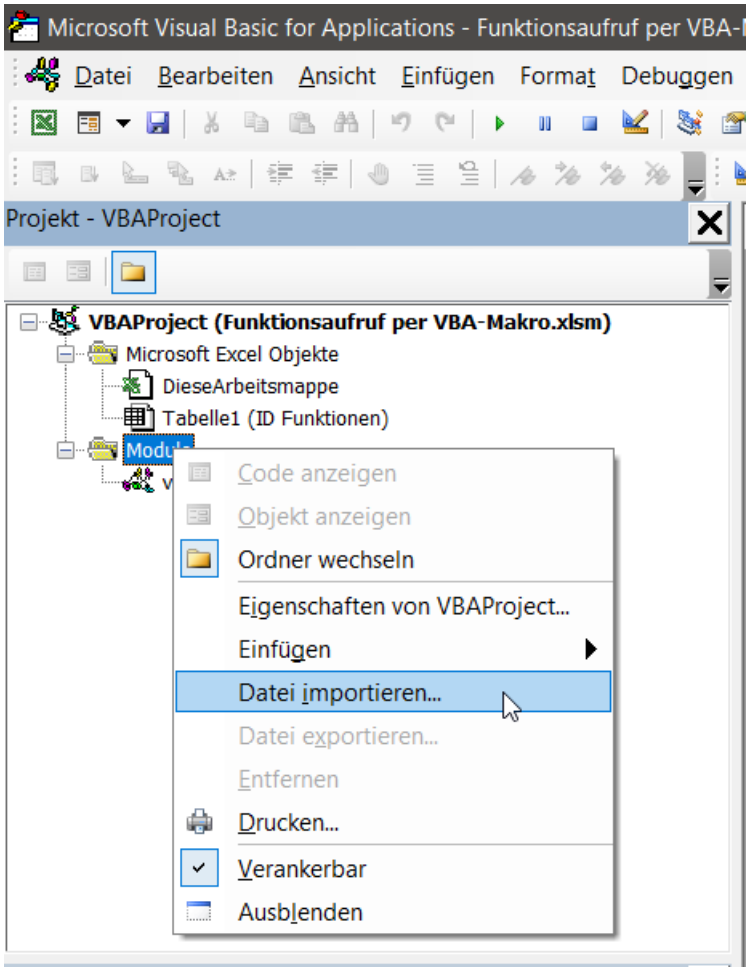
3) Diese Prozedur exportiert Werte eines selektierten Bereichs mit verpackten Formeln (Prozedur mit Aufruf von weiteren Prozeduren).

- Mit der Export - Prozedur ein eigenes Fenster aufgerufen, das eine gewisse Zeit für die Initialisierung benötigt.
- In der weiteren Bearbeitung müssen noch Auswahlen getroffen und der Export aktiv angestoßen werden.
- Dies kann **nicht** durch Makros gesteuert werden.
- Durch Einbau einer Messagebox sollte sichergestellt werden, ob der Exportprozess beendet und das Fenster geschlossen wurde.

```
Sub SelBereich_Export_SP()    ' Voraussetzung ist ein selektierter Bereich

Entpacken_SelBereich    'erst Aufruf Subprozedur selektierten Bereich entpacken
Export_SelBereich    'danach Aufruf Subprozedur Werte aus selektiertem Bereich exportieren
MsgBox "Export beendet und Dialog geschlossen?" & Chr(13) & Chr(13) & "Wenn nicht, bitte warten und Dialog parallel mit 'Ende' schließen.", vbOKOnly, IDLFunktion
Verpacken_SelBereich    'zuletzt Aufruf Subprozedur selektierten Bereich wieder verpacken

End Sub

Sub Entpacken_SelBereich()

Call Application.Run("IDLCall", 22)
Call WaitForReady

End Sub

Sub Export_SelBereich()

Call Application.Run("IDLCall", 7)
Call WaitForReady

End Sub

Sub Verpacken_SelBereich()

Call Application.Run("IDLCall", 19)
Call WaitForReady

End Sub
```

VBA - Code Datei in VBA - Projekt importieren.

- Nachfolgende Code - Dateien markieren, kopieren und z.B. auf dem Desktop einfügen.
- Rechtsclick auf Module und über ‘Datei importieren‘ diese in das aktuelle VBA - Projekt aufnehmen.



Gesamter VBA-Code.

- Die Dateien können auch mit einem geeigneten Editor geöffnet werden.
- ‘Notepad ++‘ bietet sich hier an.

vbEINZELAKTION.bas vbPROZEDUREN.bas