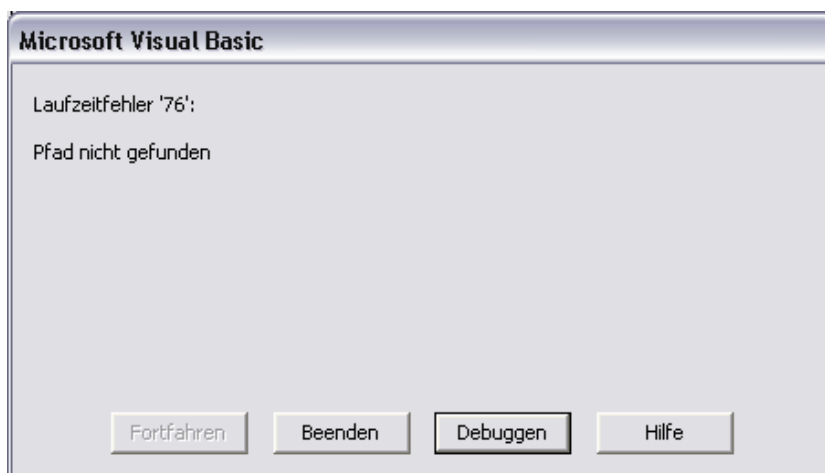


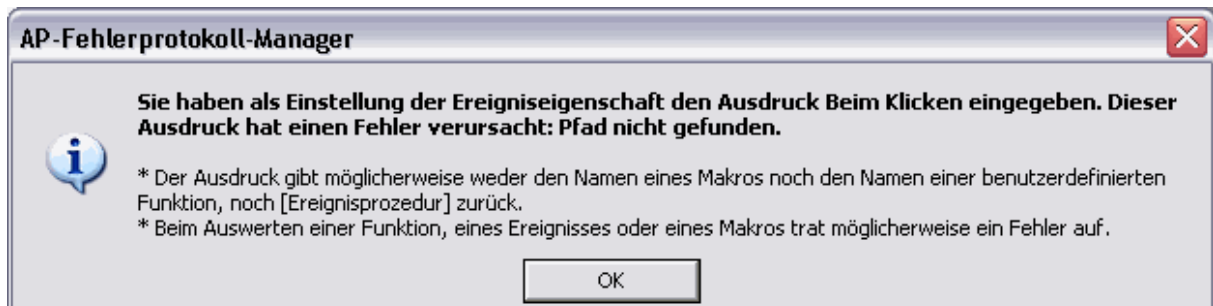
AP-SystemInfo-Manager

(Modul für eine professionelle Fehlerbehandlung und Ermittlung der Systemkonfiguration)

Wenn man mit Access eine Datenbankanwendung erstellt und dabei VBA verwendet, um die Anwendung mit bestimmten Funktionen zu erweitern, kann es unter Umständen bei der Ausführung zu Fehlern kommen. Die Gründe hierzu können die unterschiedlichsten Ursachen haben. Sei es, dass eine erwartete Datei nicht vorhanden ist oder das Rechenfeld 0 enthält und durch diese Variable dividiert werden soll, usw. Im schlimmsten Fall fängt der Entwickler die Fehler gar nicht ab und es erscheinen unprofessionelle Fehlermeldungen, mit welchen der Anwender selten etwas anfangen kann.



Fehlermeldung in einer offenen MDB-Datenbank ohne AP-SystemInfo-Manager



Fehlermeldung in einer gesperrten MDE-Datenbank ohne AP-Fehlerprotokoll-Manager bzw. AP-SystemInfo-Manager

Selbst wenn der Anwender dem Entwickler genau mitteilt, wo und bei welcher Aktion der Fehler aufgetreten ist, reichen diese Informationen meist nicht aus, um den Fehler sofort zu lokalisieren und zu entfernen.

Der AP-SystemInfo-Manager ist ein Access-Modul, welches mit seinen Funktionen eine professionelle Fehlerbehandlung zur Verfügung stellt. Innerhalb weniger Augenblicke hat man die dafür nötigen Objekte in die eigene Datenbank importiert und mit ein paar Codezeilen passt man jede Prozedur und Funktion so an, dass der Anwender eine vernünftige Fehlermeldung erhält. Er kann darüber hinaus den Entwickler über diesen Fehler informieren und diesem per Bildschirmausdruck, Fax oder Email die Details zum Fehler und zum Computersystem mitteilen.

Verweise:

Bevor Sie das Modul in Ihre Anwendung integrieren können, müssen bestimmte Verweise verfügbar sein. Prüfen Sie deshalb zunächst die verfügbaren Referenzen in Ihrer Anwendung.

Wie prüft man die vorhandenen Verweise?

Öffnen Sie Ihre Anwendung. Im Datenbankfenster klicken Sie auf den Karteireiter **Module**. Öffnen Sie im Entwurf ein vorhandenes Modul oder erstellen Sie ein neues Modul. Über das Menü **Extras / Verweise** öffnen Sie den Verweisdialog.

Folgende Verweise müssen aktiviert sein:

Für Access 97:

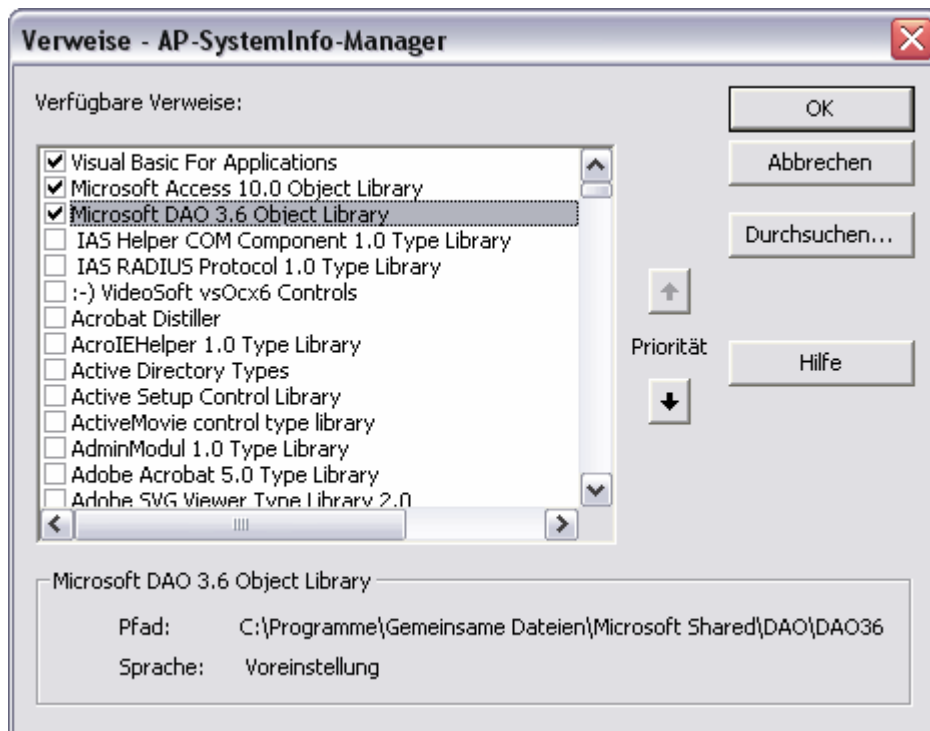
- Visual Basic For Applications
- Microsoft Access 8.0 Object Library oder
- Microsoft DAO 3.51 Object Library (oder höher)

Für Access 2000:

- Visual Basic For Applications
- Microsoft Access 9.0 Object Library (oder höher)
- Microsoft DAO 3.6 Object Library (oder höher)

Für Access 2002/XP:

- Visual Basic For Applications
- Microsoft Access 10.0 Object Library (oder höher)
- Microsoft DAO 3.6 Object Library (oder höher)



Beispiel für Access XP

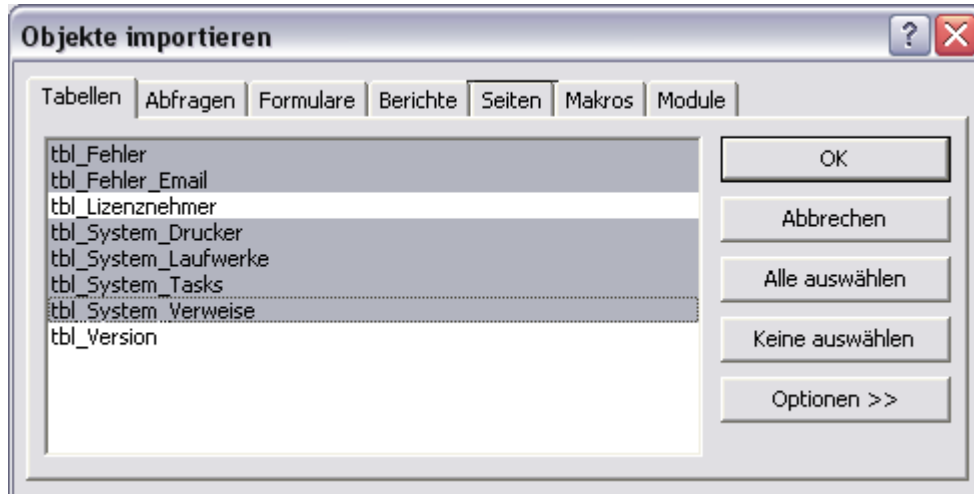
Es können noch weitere Verweise aktiviert sein, aber die o.g. Verweise werden vom AP-SystemInfo-Manager in jedem Fall benötigt.

Zu importierende Objekte:

Folgende Objekte müssen aus der Originaldatei des AP-SystemInfo-Managers in Ihre Anwendung importiert werden:

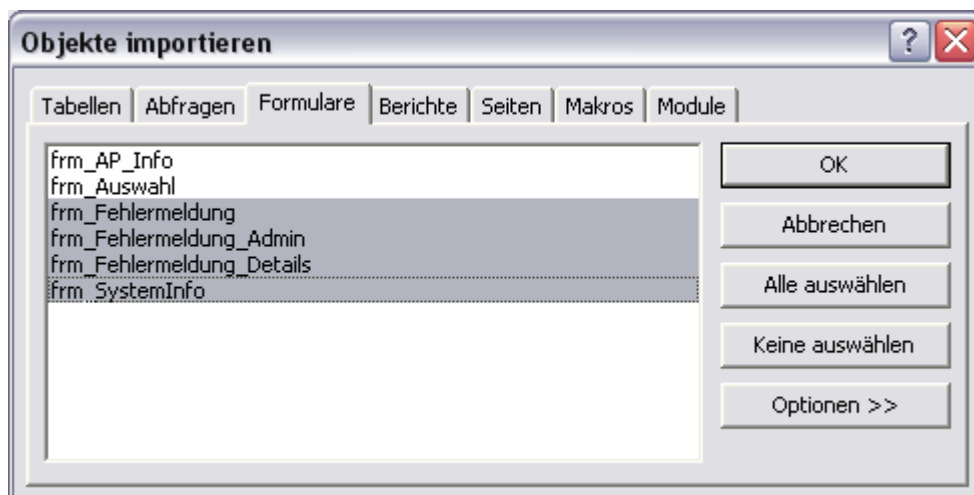
Tabellen

tbl_Fehler
tbl_Fehler_Email
tbl_System_Drucker
tbl_System_Laufwerke
tbl_System_Tasks
tbl_System_Verweise



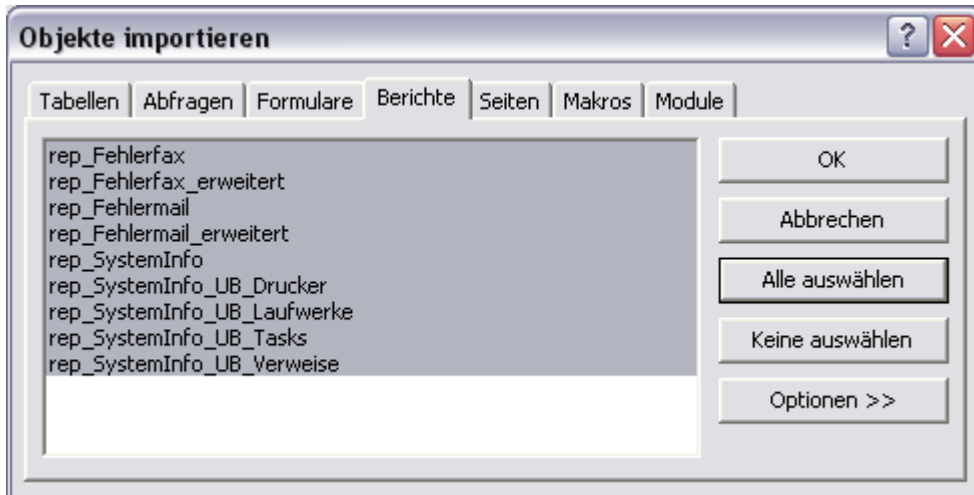
Formulare

frm_Fehlermeldung
frm_Fehlermeldung_Admin
frm_Fehlermeldung_Details
frm_SystemInfo



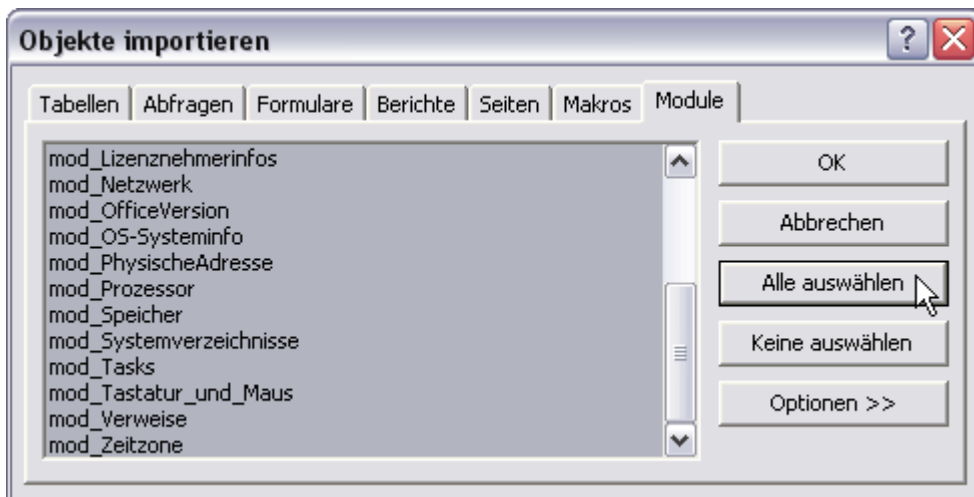
Berichte

rep_Fehlerfax
 rep_Fehlerfax_erweitert
 rep_Fehlermail
 rep_Fehlermail_erweitert
 rep_SystemInfo
 rep_SystemInfo_UB_Drucker
 rep_SystemInfo_UB_Laufwerke
 rep_SystemInfo_UB_Tasks
 rep_SystemInfo_UB_Verweise



Module

Klicken Sie hier auf die Schaltfläche „**Alle auswählen**“ und importieren Sie alle Module der Originaldatei.



Wie importiert man Objekte aus einer anderen Access-Datenbank?

Öffnen Sie Ihre Anwendung. Über das Menü **Datei / Externe Daten / Importieren** wird der Dateidialog geöffnet. Wählen Sie über diesen Dialog die Access-Datei aus, aus der die Objekte importiert werden sollen. Es öffnet sich der Importdialog, den Sie in den oben abgebildeten Ansichten sehen. Hier können Sie auswählen, welche Tabellen, Abfragen usw. Sie in die aktuelle Datenbank importieren möchten.

Erklärung der importierten Module

Tabellen

- **tbl_Fehler**
 - Hier werden alle Fehler und die dazugehörigen Details protokolliert.

- **tbl_Fehler_Email**
 - In dieser Tabelle werden die Informationen zum letzten Fehler gespeichert. Diese Tabelle dient auch als Basis für die Erstellung des Anhangs für das Fehlermail.

- **tbl_System_Drucker**
 - In dieser Tabelle werden alle ermittelten Drucker gespeichert

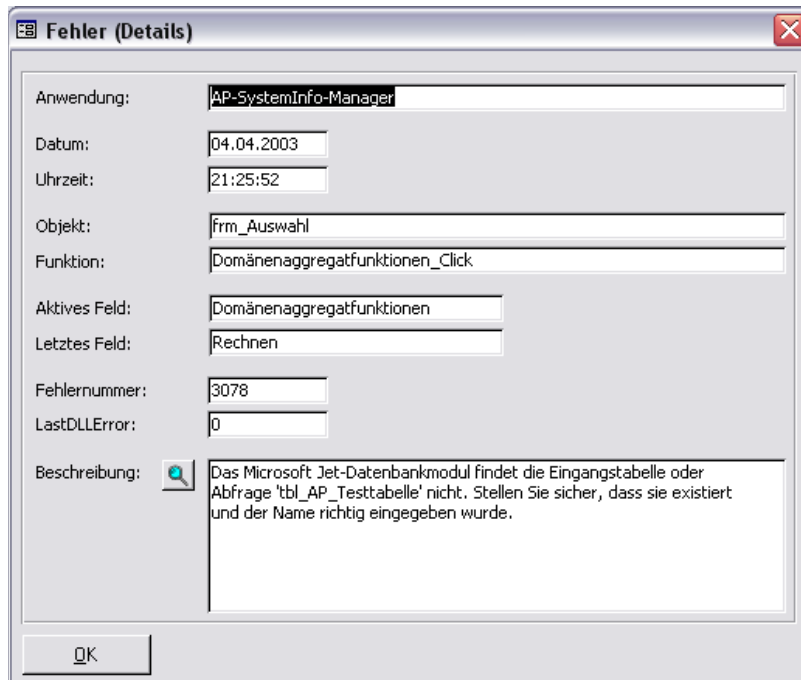
- **tbl_System_Laufwerke**
 - Alle Daten der Laufwerke werden hier zwischengespeichert

- **tbl_System_Tasks**
 - Die aktiven Anwendungen, die ermittelt werden konnten finden in dieser Tabelle ihren Platz

- **tbl_System_Verweise**
 - Alle Verweise, die für diese Datenbank verfügbar sind, werden ermittelt und in dieser Tabelle gespeichert

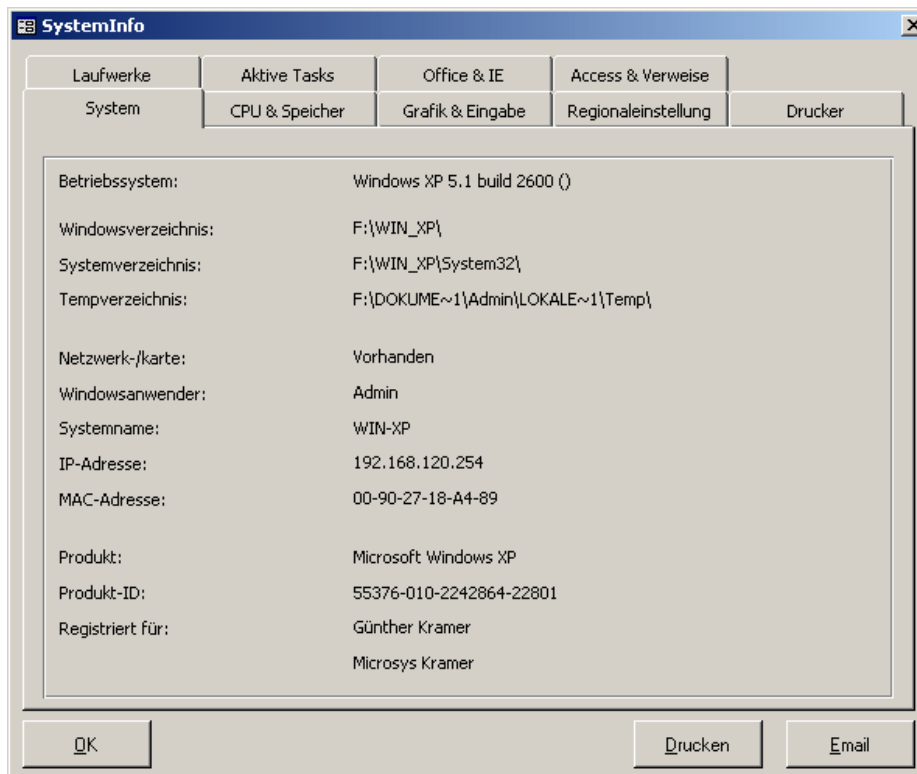
- **frm_Fehlermeldung_Details**

- Klickt der Anwender in der Liste der historisierten Fehler (Adminansicht) mit einem Doppelklick auf einen Datensatz, werden die Details des Fehlers angezeigt.



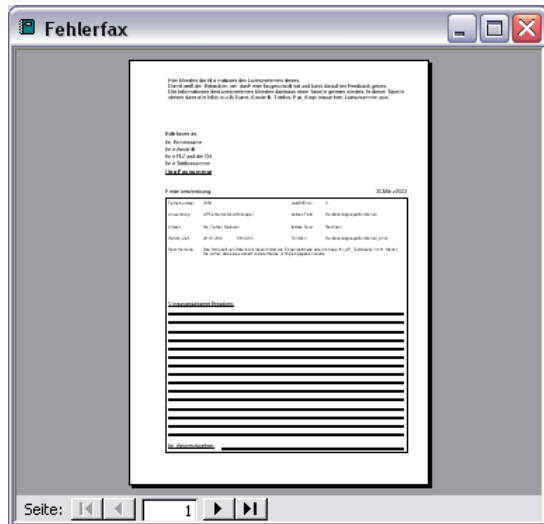
- **frm_SystemInfo**

- In dieser Anwendungsmaske werden die Details zu dem Computersystem, auf welchem der Fehler aufgetreten ist, angezeigt.



Berichte

- **rep_Fehlerfax**
 - Dieser Bericht wird ausgedruckt, wenn der Anwender im Fehlerformular auf **Fax** klickt. Hier hat er die Möglichkeit, in eigens dafür vorgesehene Zeilen, den Hergang des Fehlers zu schildern.



- **rep_Fehlerfax_erweitert**
 - Dieser Bericht beinhaltet neben den o.g. (rep_Fehlerfax) Fehlerdetails noch alle ermittelten Systeminformationen.
- **rep_Fehlermail**
 - Dieser Bericht wird dafür verwendet, bei einer Benachrichtigung per Mail, dem Entwickler die Infos zum Fehler in einem Snapshot-Format zu übermitteln. Den SnapShot-Viewer gibt es von Microsoft kostenlos und jeder Entwickler kann einen Bericht in diesem Format auch direkt aufrufen und ausdrucken.

Infos für die Version Access 97 findet man unter:

<http://office.microsoft.com/germany/downloads/9798/snpvw80.aspx>

(In Access 2000 und XP sind die Funktionen bereits enthalten.)

-
- **rep_Fehlermail_erweitert**
 - Wie der Bericht **rep_Fehlermail** nur zusätzlich mit allen ermittelten Systeminformationen
- **rep_SystemInfo**
 - Alle ermittelten Systeminformationen
Unterberichte hierzu:
 - rep_SystemInfo_UB_Drucker
 - rep_SystemInfo_UB_Laufwerke
 - rep_SystemInfo_UB_Tasks
 - rep_SystemInfo_UB_Verweise

AP-SystemInfo-Manager

(Modul für eine professionelle Fehlerbehandlung und Ermittlung der Systemkonfiguration)

Microsys Kramer • <http://www.access-paradies.de>

Module

- mod_AP-Fehlerprotokoll-Manager
In diesem Modul befindet sich die gesamte Logik der Fehlerermittlung und Protokollierung. In diesem Modul müssen die Werte der Konstanten für Ihre Angaben geändert und angepasst werden.

```
' Ändern Sie hier die Werte für Ihre Anwendung ab!
' -----
' Welche Bezeichnung hat Ihre Anwendung?
Global Const AppName = "AP-SystemInfo-Manager"

' Ihr Namen bzw. Firmenname und Anschrift
Global Const AP_SIM_Firma = "Ihr Firmenname"
Global Const AP_SIM_Anschrift = "Ihre Anschrift"
Global Const AP_SIM_PLZ_Ort = "Ihre PLZ und der Ort"
Global Const AP_SIM_Tel = "Ihre Telefonnummer"
Global Const AP_SIM_Fax = "Ihre Faxnummer"

' An welche Emailadresse soll das Fehlermail geschickt werden?
Global Const AP_SIM_Emailadresse = "support@software.de"

' Welcher Betreff soll in das Email eingefügt werden?
Global Const AP_SIM_Emailbetreff = "Programmfehler im AP-SystemInfo-Manager"

' Welcher Text soll für das Email verwendet werden?
Global Const AP_SIM_Emailtext = "In der Anwendung ist ein Fehler aufgetreten."
```

- mod_AccessGruppen
 - Ermittelt die Access-Gruppenzugehörigkeit des aktuellen Access-Anwenders

Funktion:

```
Function AccessGruppen_ermitteln()
```

- mod_Bildschirm
 - Ermittelt die aktuelle Bildschirmauflösung und Farbtiefe

Funktion:

```
Function Aufloesung_ermitteln(strWhat As String) As String
```

```
    - Aufruf: Aufloesung_ermitteln("resolution")
```

```
Function Farbpalette_ermitteln() As String
```

- mod_Comuter_und_Username
 - Ermittelt den aktuellen Anwender, der am Windowssystem angemeldet ist und den Namen des aktuellen PCs

Funktionen:

```
Function fOSUserName() As String
```

```
Function GetLocalComputerName() As String
```

- mod_Drucker
 - Schreibt alle Druckerbezeichnungen in die Tabelle „tbl_System_Drucker“

Prozedur:

```
Sub SystemDrucker_ermitteln()
```

- mod_IP-Adresse
 - Ermittelt die aktuelle IP-Adresse des Netzwerkadapters
 - Funktion:
Function LocalIPAddress() As String

- mod_JetEngine
 - Ermittelt die Version der JetEngine, mit der die Datenbank erstellt wurde
 - Funktion:
Function JetVersion_ermitteln() As String

- mod_Ländereinstellungen
 - Ermittelt die Regionaleinstellungen des Systems
 - Funktion:
Function Regionaleinstellungen_ermitteln(Info As String)
As String
 - Aufruf: Regionaleinstellungen_ermitteln(Parameter)
 - Parameter:
"Land"
"Währungssymbol"
"ISO-CODE"
"Dezimaltrennzeichen"
"Zifferngruppierung"
"Dezimalstellen"
"Datum_kurz"
"Datum_lang"
"Zeitformat"

- mod_Laufwerksinfos
 - Ermittelt die Informationen aller angeschlossenen Laufwerke und schreibt diese in die Tabelle „tbl_System_Laufwerke“
 - Prozedur:
Sub LaufwerksInfos_ermitteln()

- mod_Lizenznehmerinfos
 - Ermittelt die Lizenznehmerdaten von Windows
 - Funktion:
Function Lizenzdaten_ermitteln(Parameter) As String
 - Parameter:
"Lizenznehmer"
"Firma"
"Produkt"
"ProduktID"

- mod_Netzwerk
 - Ermittelt, ob eine Netzwerkkarte, ein Netzwerktreiber bzw. ein Netzwerk vorhanden ist.
Funktion:
Function IstNetzwerkVorhanden() As Boolean

- mod_OfficeVersion
 - Ermittelt die genaue Versionsnummer des jeweiligen Office-Programms bzw. vom IE.
Funktion:
Function OfficeVersion_ermitteln(Parameter) As String
Parameter: Der genaue Pfad der jeweiligen EXE-Datei des Programms
iexplore.exe
msaccess.exe
winword.exe
excel.exe
outlook.exe
powerpnt.exe
frontpg.exe

- mod_OS-Systeminfo
 - Ermittelt die Version des Betriebssystems inkl. Build- und SP-Nummer
Funktion:
Function OS_Version_ermitteln()

- mod_PhysischeAdresse
 - Ermittelt die physikalische Adresse (MAC-Adresse) der Netzwerkkarte
Funktion:
Function PhysischeAdresse()

- mod_Prozessor
 - Ermittelt die Prozessorgeschwindigkeit, die Anzahl der Prozessoren, sowie die Infos des Levels und der Revision des Prozessors
Funktionen:
Function GetCPUSpeed() As String
Function Prozessorinfos_ermitteln(Parameter) As String
 - Parameter:
"Anzahl"
"Typ"
"Level"
"Revision"

- **mod_Speicher**
 - Ermittelt die Informationen rund um den Arbeitsspeicher und der Auslagerungsdatei
Funktionen:
Function HolPhysSpeicherTotal() As String
Function HolPhysSpeicherFrei() As String
Function HolPhysSpeicherAusnutzung() As String
Function HolVirtSpeicherTotal() As String
Function HolVirtSpeicherFrei() As String
Function Auslagerungsdatei_gesamt() As String
Function Auslagerungsdatei_frei() As String

- **mod_Systemverzeichnisse**
 - Ermittelt das Verzeichnis für Windows sowie das System- und Temp-Verzeichnis
Funktionen:
Function GetWindowsDir() As String
Function GetSystemDir() As String
Function GetTempDir() As String

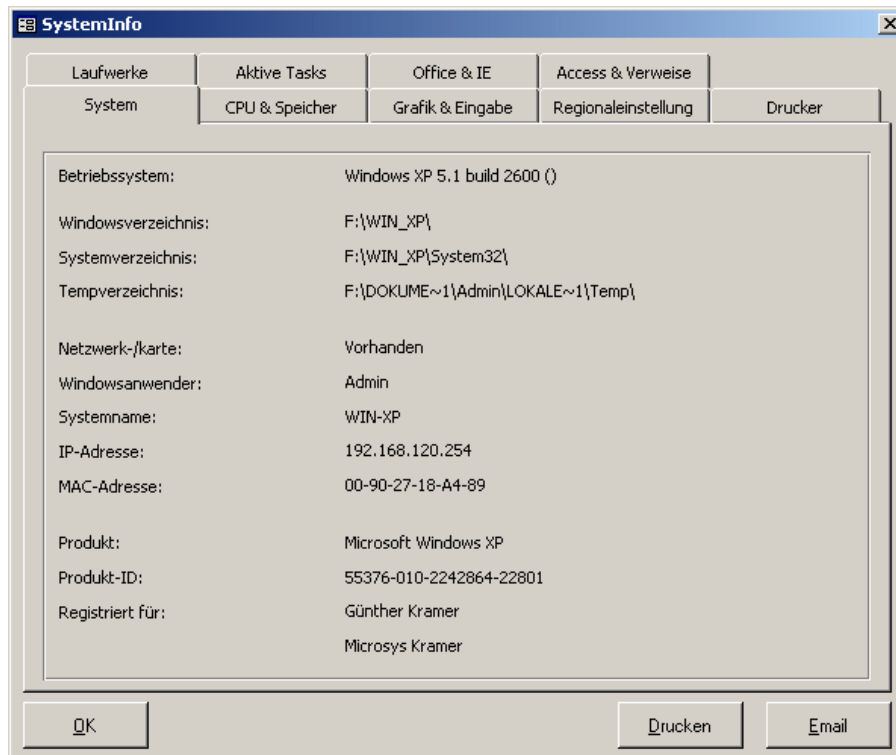
- **mod_Tasks**
 - Ermittelt alle sichtbaren, aktiven Anwendungen und schreibt deren Titel und Klassennamen in die Tabelle „tbl_System_Tasks“.
Prozedur:
Sub Tasks_ermitteln()

- **mod_Tastatur_und_Maus**
 - Ermittelt welche Tasten gedrückt bzw. aktiv sind und ob eine Maus angeschlossen ist. Falls eine Maus vorhanden ist, wird die Anzahl der Tasten ermittelt und geprüft, ob ein Mausrad vorhanden ist.
Funktionen:
Function IstUmschaltAn() As Boolean
Function IstNummernblockAn() As Boolean
Function IstRollenAn() As Boolean
Function HolAnzahlMausTasten() As Long
Function IstMausradVorhanden() As Boolean
Function ExistiertMaus() As Boolean

- **mod_Verweise**
 - Ermittelt alle in der aktuellen Datenbank verfügbaren Verweise und schreibt diese mit der Versionsnummer und dem Dateipfad in die Tabelle „tbl_System_Verweise“
Prozedur:
Sub Verweise_ermitteln()

- mod_Zeitzone
 - Ermittelt die aktuell eingestellte Zeitzone
- Funktion:**
Function GetCurrentTimeZone() As String

Den exakten Aufruf der Funktionen und Prozeduren ersehen Sie im Formular „frm_SystemInfo „. Öffnen Sie das Formular in der Entwurfsansicht. Klicken Sie im Menü **Ansicht** auf **Code**. Es öffnet sich die Codeansicht des Formulars. Bewegen Sie den Cursor an den Anfang des Moduls. In der Prozedur Form_Load finden Sie den VBA-Code, der die oben beschriebenen Funktionen und Prozeduren aufruft.



```
Dim Temp_AccessPfad
Dim Temp_Windowsverzeichnis

Temp_Windowsverzeichnis = GetWindowsDir()
' Ergebnis (Beispiel): C:\WINDOWS\
Temp_AccessPfad = AccessPfad()
' Ergebnis (Beispiel): C:\Programme\Microsoft Office\Office10\

Me![Betriebssystem] = OS_Version_ermitteln()
' Ergebnis (Beispiel): Windows XP 5.1 build 2600 (Service Pack 1)

Me![Windowsverzeichnis] = Temp_Windowsverzeichnis

Me![Systemverzeichnis] = GetSystemDir()
' Ergebnis (Beispiel): C:\WINDOWS\System32\

Me![Tempverzeichnis] = GetTempDir()
' Ergebnis (Beispiel): C:\DOKUME~1\Admin\LOKALE~1\Temp\

If IstNetzwerkVorhanden() Then
    Me![Netzwerk] = "Vorhanden"
Else
    Me![Netzwerk] = "Nicht vorhanden"
End If
```

```

Me![Windowsanwender] = fOSUserName()
' Ergebnis (Beispiel): Admin

Me![Systemname] = GetLocalComputerName()
' Ergebnis (Beispiel): WIN-XP

Me![IP-Adresse] = LocalIPAddress()
' Ergebnis (Beispiel): 192.168.120.254

Me![MAC-Adresse] = PhysischeAdresse()
' Ergebnis (Beispiel): 00-90-27-18-A4-89

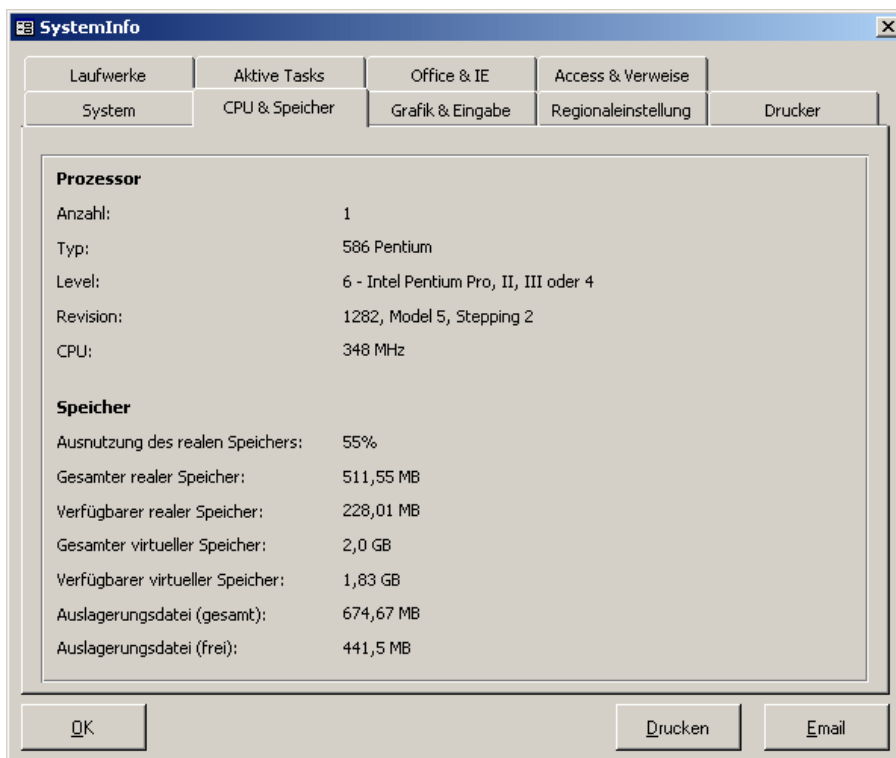
Me![Produkt] = Lizenzdaten_ermitteln("Produkt")
' Ergebnis (Beispiel): Microsoft Windows XP

Me![Produkt-ID] = Lizenzdaten_ermitteln("ProduktID")
' Ergebnis (Beispiel): 55372-OEM-0011903-00126

Me![Lizenznehmer] = Lizenzdaten_ermitteln("Lizenznehmer")
' Ergebnis (Beispiel): Günther Kramer

Me![Firma] = Lizenzdaten_ermitteln("Firma")
' Ergebnis (Beispiel): Microsys Kramer

```



```

Me![Prozessor_Anzahl] = Prozessorinfos_ermitteln("Anzahl")
' Ergebnis (Beispiel): 1

Me![Prozessor_Typ] = Prozessorinfos_ermitteln("Typ")
' Ergebnis (Beispiel): 586 Pentium

Me![Prozessor_Level] = Prozessorinfos_ermitteln("Level")
' Ergebnis (Beispiel): 6 - Intel Pentium Pro, II, III oder 4

Me![Prozessor_Revision] = Prozessorinfos_ermitteln("Revision")
' Ergebnis (Beispiel): 1282, Model 5, Stepping 2

```

```

Me![Prozessor_CPU] = GetCPUSpeed()
' Ergebnis (Beispiel): 348 MHz oder 1,99 GHz

Me![Speicher_Ausnutzung] = HolPhysSpeicherAusnutzung()
' Ergebnis (Beispiel): 55%

Me![Speicher_realer_gesamt] = HolPhysSpeicherTotal()
' Ergebnis (Beispiel): 511,55 MB

Me![Speicher_realer_freie] = HolPhysSpeicherFrei()
' Ergebnis (Beispiel): 228,01 MB

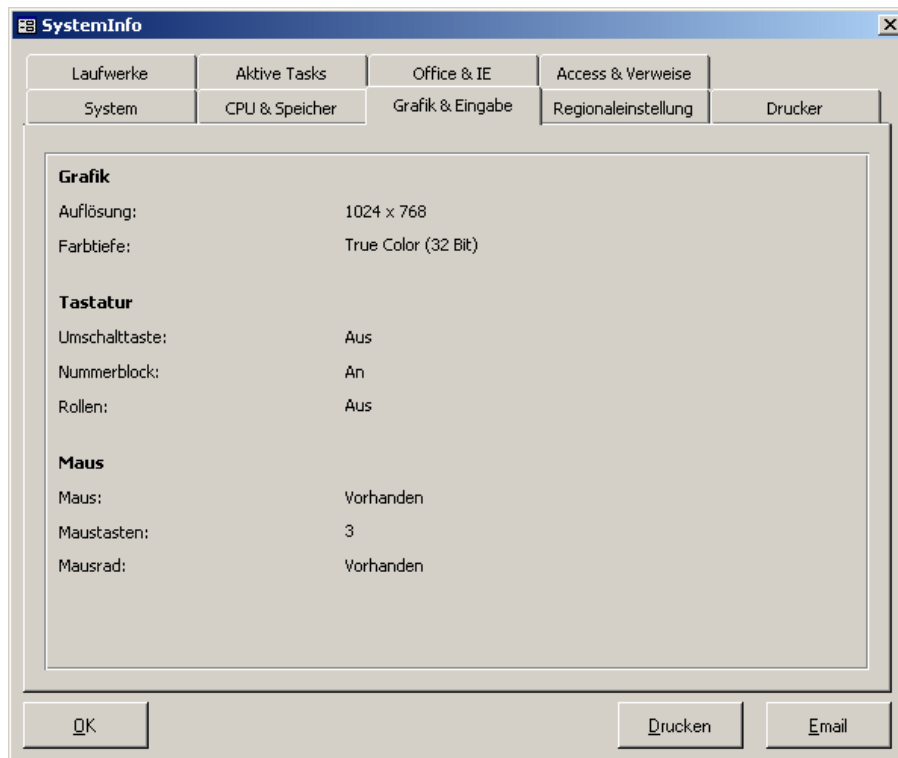
Me![Speicher_virtueller_gesamt] = HolVirtSpeicherTotal()
' Ergebnis (Beispiel): 2,0 GB

Me![Speicher_virtueller_freie] = HolVirtSpeicherFrei()
' Ergebnis (Beispiel): 1,83 GB

Me![Speicher_Auslagerungsdatei_gesamt] = Auslagerungsdatei_gesamt()
' Ergebnis (Beispiel): 674,67 MB

Me![Speicher_Auslagerungsdatei_freie] = Auslagerungsdatei_freie()
' Ergebnis (Beispiel): 441,5 MB

```



```

Me![Auflösung] = Aufloesung_ermitteln("resolution")
' Ergebnis (Beispiel): 1024 x 768

Me![Farbtiefe] = Farbpalette_ermitteln()
' Ergebnis (Beispiel): True Color (32 Bit)

If IstUmschaltAn() Then
    Me![Umschalttaste] = "An"
Else
    Me![Umschalttaste] = "Aus"
End If

```

```

If IstNummernblockAn() Then
    Me![Nummernblock] = "An"
Else
    Me![Nummernblock] = "Aus"
End If

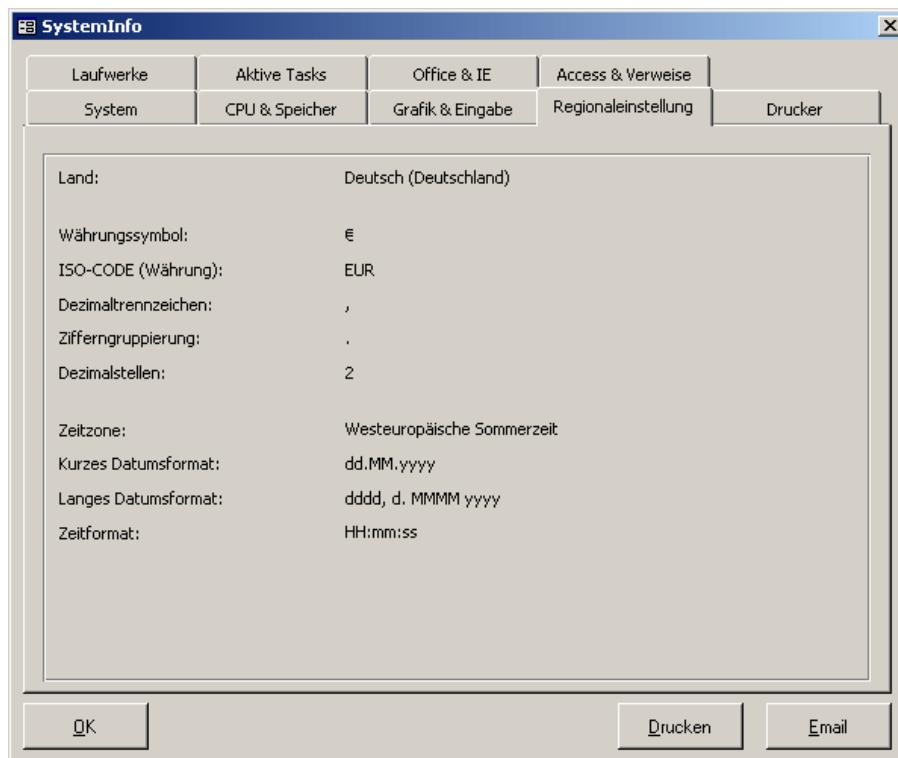
If IstRollenAn() Then
    Me![Rollen] = "An"
Else
    Me![Rollen] = "Aus"
End If

If ExistiertMaus() Then
    Me![Maus] = "Vorhanden"
Else
    Me![Maus] = "Nicht vorhanden"
End If

Me![Maustasten] = HolAnzahlMaustasten()
' Ergebnis (Beispiel): 3

If IstMausradVorhanden() Then
    Me![Mausrad] = "Vorhanden"
Else
    Me![Mausrad] = "Nicht vorhanden"
End If

```



```

Me![Region_Land] = Regionaleinstellungen_ermitteln("Land")
' Ergebnis (Beispiel): Deutsch (Deutschland)

```

```

Me![Währungssymbol] = Regionaleinstellungen_ermitteln("Währungssymbol")
' Ergebnis (Beispiel): €

```

```

Me![ISO-CODE] = Regionaleinstellungen_ermitteln("ISO-CODE")
' Ergebnis (Beispiel): EUR

```

```

Me![Dezimaltrennzeichen] = Regionaleinstellungen_ermitteln("Dezimaltrennzeichen")
' Ergebnis (Beispiel): ,

```

```
Me![Zifferngruppierung] = Regionaleinstellungen_ermitteln("Zifferngruppierung")
' Ergebnis (Beispiel): .
```

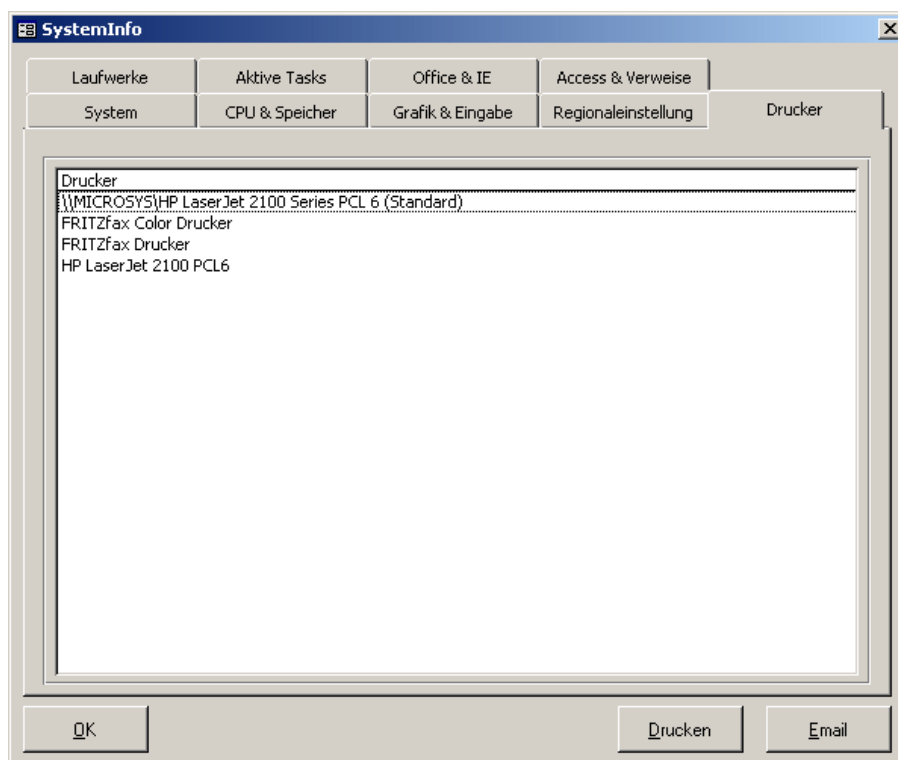
```
Me![Dezimalstellen] = Regionaleinstellungen_ermitteln("Dezimalstellen")
' Ergebnis (Beispiel): 2
```

```
Me![Zeitzone] = GetCurrentTimeZone
' Ergebnis (Beispiel): Westeuropäische Normalzeit
```

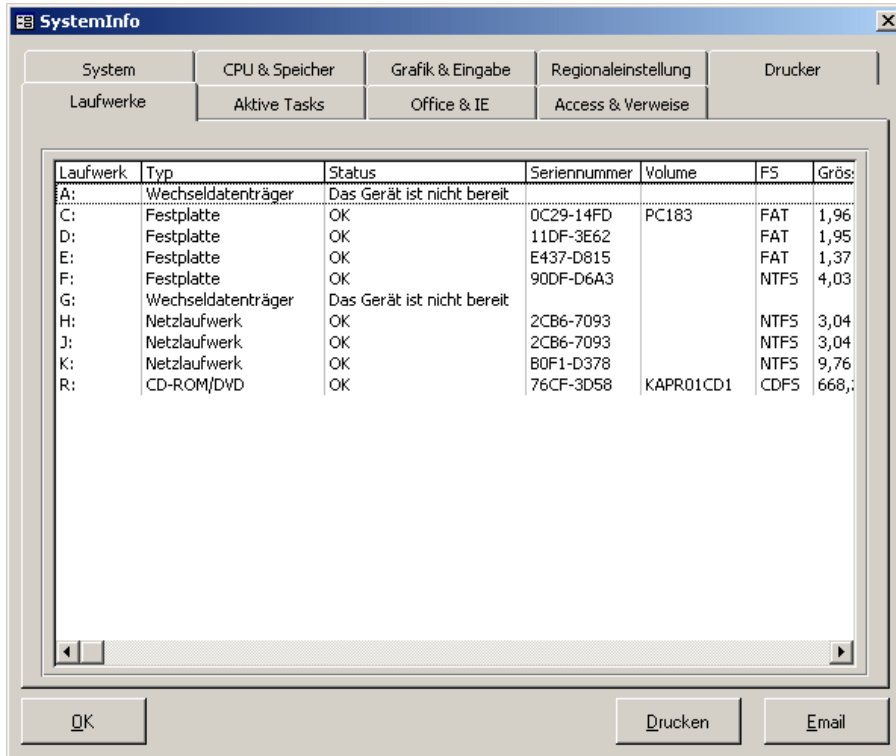
```
Me![Datum_kurz] = Regionaleinstellungen_ermitteln("Datum_kurz")
' Ergebnis (Beispiel): dd.MM.yyyy
```

```
Me![Datum_lang] = Regionaleinstellungen_ermitteln("Datum_lang")
' Ergebnis (Beispiel): dddd, d. MMMM yyyy
```

```
Me![Zeitformat] = Regionaleinstellungen_ermitteln("Zeitformat")
' Ergebnis (Beispiel): HH:mm:ss
```

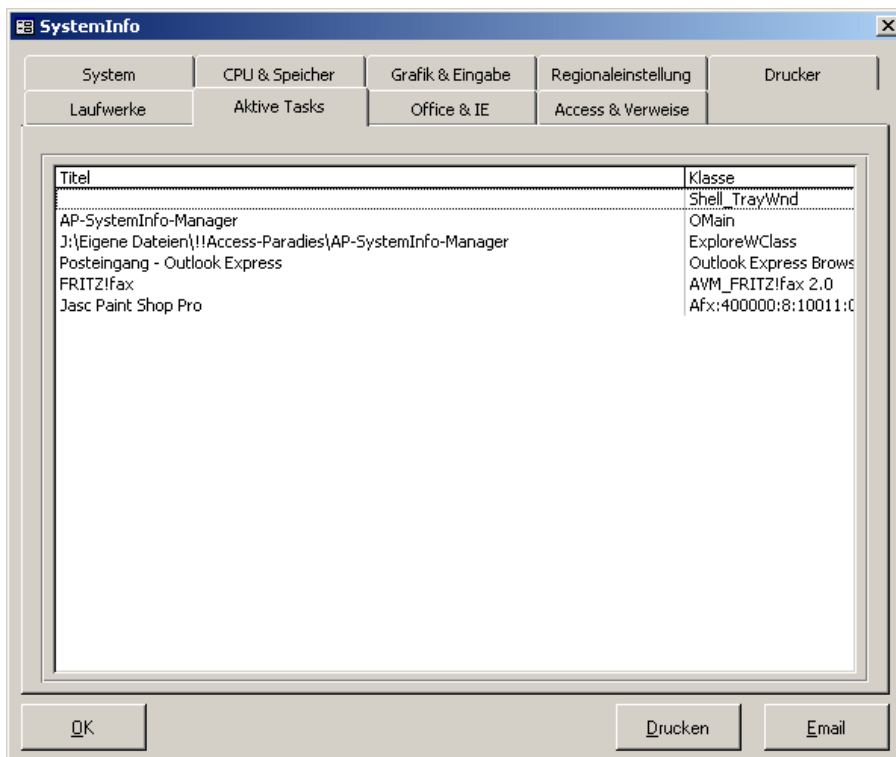


```
SystemDrucker_ermitteln
' Ergebnis (Beispiel): Speichert die Infos in eine Tabelle
```



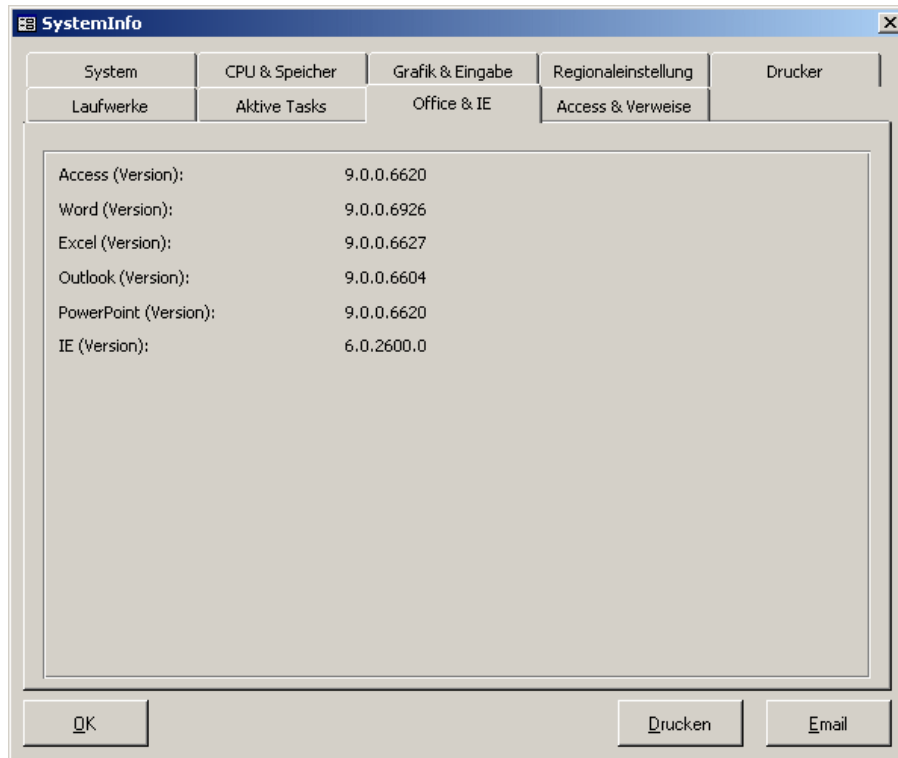
LaufwerksInfos_ermitteln

' Ergebnis (Beispiel): Speichert die Infos in eine Tabelle



Tasks_ermitteln

' Ergebnis (Beispiel): Speichert die Infos in eine Tabelle



```
Me![Version_Access] = OfficeVersion_ermitteln(Temp_AccessPfad & "msaccess.exe")
' Ergebnis (Beispiel): 9.0.0.6620
```

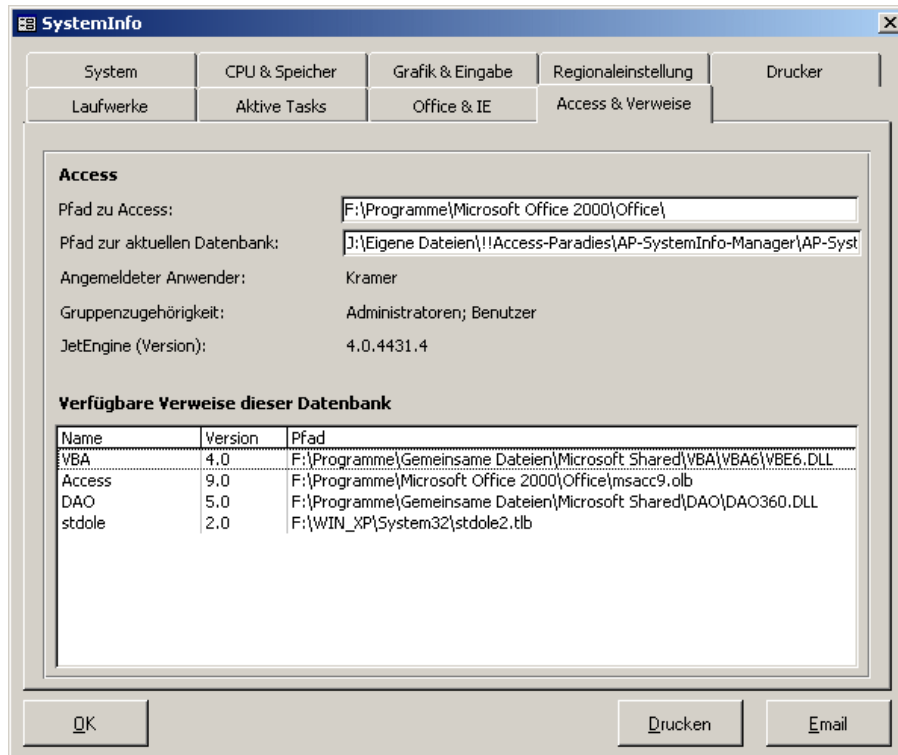
```
Me![Version_Word] = OfficeVersion_ermitteln(Temp_AccessPfad & "winword.exe")
' Ergebnis (Beispiel): 9.0.0.6926
```

```
Me![Version_Excel] = OfficeVersion_ermitteln(Temp_AccessPfad & "excel.exe")
' Ergebnis (Beispiel): 9.0.0.6627
```

```
Me![Version_Outlook] = OfficeVersion_ermitteln(Temp_AccessPfad & "outlook.exe")
' Ergebnis (Beispiel): 9.0.0.6604
```

```
Me![Version_Powerpoint] = OfficeVersion_ermitteln(Temp_AccessPfad & "powerpnt.exe")
' Ergebnis (Beispiel): 9.0.0.6620
```

```
Me![Version_IE] = OfficeVersion_ermitteln(Left(Temp_Windowsverzeichnis, 2) & _
"\Programme\Internet Explorer\iexplore.exe")
' Ergebnis (Beispiel): 6.0.2800.1106
```



Me![Access_Pfad] = Temp_AccessPfad

Me![Datenbank_Pfad] = CurrentDb.Name

' Ergebnis (Beispiel): D:\AP-SystemInfo-Manager\AP-SystemInfo-Manager2002.mdb

Me![Access_Anwender] = CurrentUser

' Ergebnis (Beispiel): Kramer

Me![Access_Gruppen] = AccessGruppen_ermitteln()

' Ergebnis (Beispiel): Administratoren; Benutzer

Me![Version_JetEngine] = JetVersion_ermitteln()

' Ergebnis (Beispiel): 4.0.6218.0

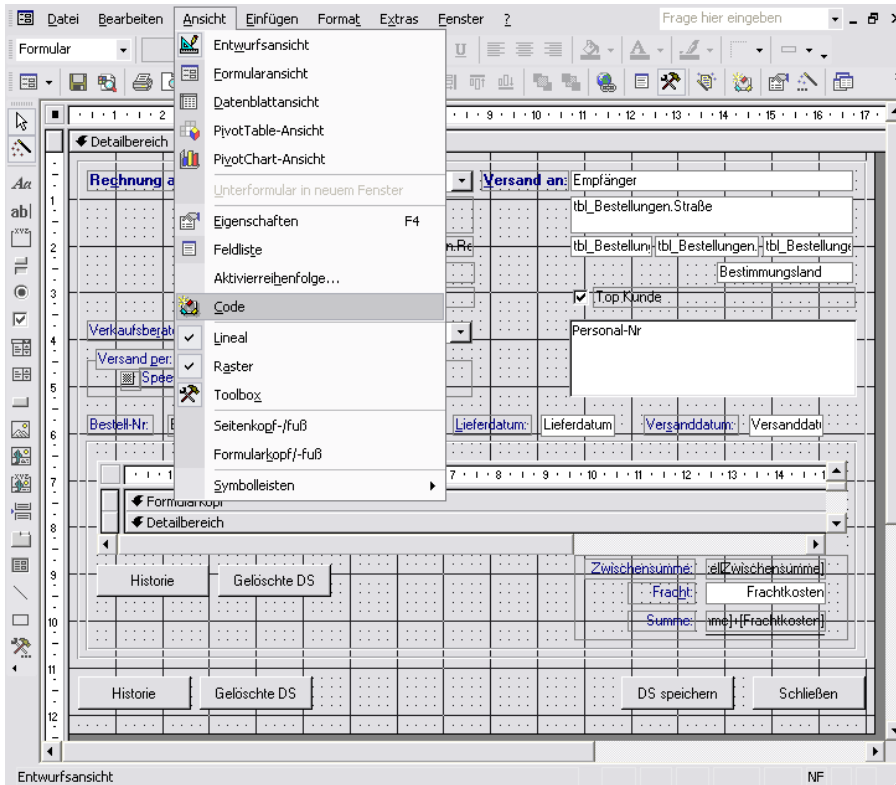
Verweise_ermitteln

' Ergebnis (Beispiel): Speichert die Infos in eine Tabelle

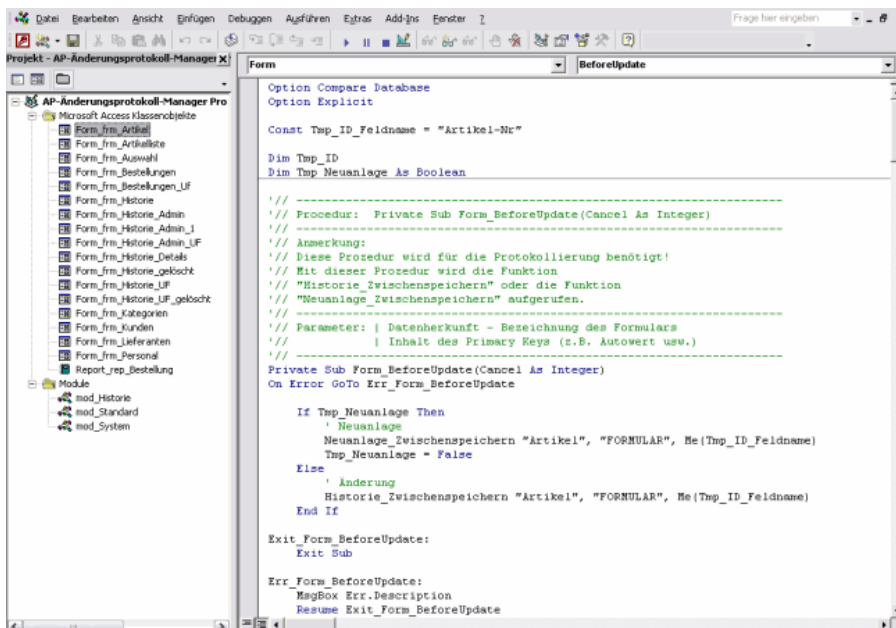
- Anpassung Ihrer Prozeduren und Funktionen

In diesem Teil der Integrationsdokumentation erfahren Sie, wie Sie den VBA-Code in Ihrer Datenbank ergänzen müssen, damit zukünftig Fehler professionell durch die Funktionen des AP-SystemInfo-Managers abgefangen werden.

Öffnen Sie ein Formular, in welchem sich ein VBA-Code befindet. Dieser VBA-Code kann auch durch einen Assistenten (z.B. Befehlsschaltflächen-Assistent) erzeugt worden sein. Wählen Sie aus dem Menü **Ansicht** den Punkt **Code** aus. Es öffnet sich die Codeansicht des Formulars.



Entwurfsansicht eines Formulars



Bewegen Sie den Cursor an den Anfang des Moduls und durchsuchen Sie das gesamte Modul nach Prozeduren und Funktionen.

Prozeduren

Sub Prozedurname()

Es kann vor dem Text Sub noch ein anderer Text stehen (z.B. Private, Public).

Innerhalb der Klammer können sich noch Variablendeklarationen für Übergabeparameter befinden.

Beispiel: Public Sub FehlerAufnahme(Maske, strFunktion)

Funktionen

Function Funktionsname()

Es kann vor dem Text Sub noch ein anderer Text stehen (z.B. Private, Public).

Innerhalb der Klammer können sich noch Variablendeklarationen für Übergabeparameter befinden.

Beispiel: Public Function Historie_Zwischenspeichern(Datenherkunft As String, DatenherkunftArt As String, Datensatz_ID)

Damit ein Fehler abgefangen werden kann, muss die Anweisung **On Error** eingetragen werden. Der Aufruf aktiviert eine Fehlerbehandlungsroutine und gibt deren Position innerhalb einer Prozedur bzw. Funktion an. Diese Routine befindet sich am Ende der jeweiligen Prozedur/Funktion. Im nachfolgenden Beispiel sehen Sie zunächst die Deklaration (1) der Prozedur. In der zweiten Zeile wird die Fehlerbehandlung aktiviert. Danach (4) kommt der VBA-Code Ihrer Prozedur/Funktion, an der nichts zu ändern ist. Bei einem normalen Ablauf des VBA-Codes wird die Prozedur/Funktion bei Zeile 7 wieder verlassen. Sollte jedoch ein Fehler auftreten, wird durch die Aktivierung in Zeile 2 in die Fehlerbehandlungsroutine (Zeile 9) verzweigt. Im Normalfall gibt man hier die Fehlermeldung durch eine Bildschirmmeldung (MsgBox) aus.



Im Anschluss wird die Prozedur/Funktion sauber verlassen und zur Endezeile (6) gesprungen. Hier wird die Prozedur/Funktion geschlossen und verlassen.

Codebeispiel:

```
Private Sub OK_Click()           ' 1
On Error GoTo Err_OK_Click      ' 2
                                  ' 3
    DoCmd.Close                 ' 4
                                  ' 5
Exit_OK_Click:                  ' 6
    Exit Sub                     ' 7
                                  ' 8
Err_OK_Click:                   ' 9
    MsgBox Err.Description      ' 10
    Resume Exit_OK_Click        ' 11
                                  ' 12
End Sub                          ' 13
```

Das eben gezeigte Beispiel stellt dar, wie man programmieren sollte, wenn man keinen AP-SystemInfo-Manager verwendet. Innerhalb einer Prozedur/Funktion können verschiedene Fehlerbehandlungsroutinen vorhanden sein. Die Aktivierung der einzelnen Abschnitte kann innerhalb des VBA-Codes geschehen und muss nicht unbedingt am Anfang der jeweiligen Prozedur gemacht werden. Durch Anweisungen wie **GoTo** und **Resume** kann innerhalb der Prozedur hin und her gesprungen werden, wovon wir jedoch abraten. Im Allgemeinen sagt man dazu Spagetti-Programmierung und deutet auf eine unstrukturierte und nicht saubere Codierung hin.

Damit Sie die Funktionen des AP-SystemInfo-Managers nutzen können, muss jede Prozedur und Funktion angepasst werden. Sofern noch nicht geschehen, passen Sie den Code Ihrer Anwendung wie oben beschrieben an, damit Fehler in der Anwendung abgefangen werden können.

Statt der Anweisung **Resume Exit_OK_Click** fügen Sie die Codezeile **FehlerAufnahme Me.Name, "Datei_kopieren_Click"** ein.

Die Anweisung **FehlerAufnahme** ruft die Prozedur zur Fehlerprotokollierung auf. Mit **Me.Name** wird der Name des aktuellen Objekts (Formular- bzw. Berichtname) übergeben, jedoch funktioniert das nur in Formularen und Berichten. Innerhalb eines Moduls muss hier der Name des Moduls übergeben werden (z.B. "mod_Standard"). Im zweiten Parameter übergeben Sie der Fehlerprozedur den Namen der aktuellen Prozedur. Diese Parameter werden durch den Bildschirmausdruck, per Fax oder Mail an den Entwickler weitergegeben. Damit weiß dieser sofort, in welchem Bereich der Anwendung hat sich der Fehler zugetragen.

Ändern Sie auf die zuvor genannte Weise alle Prozeduren und Funktionen innerhalb Ihrer Datenbank. Rufen Sie dazu jedes Formular, jeden Bericht und jedes Modul auf und passen Sie den VBA-Code an. Es kann jedoch in Modulen auch Funktionen und Prozeduren geben, die statt einer Anzeige und Protokollierung eines Fehlers ohne Aktion den weiteren VBA-Code ausführen. Mehr Informationen hierzu finden Sie in der Access-Onlinehilfe unter dem Begriff **Resume**. Lesen Sie hier insbesondere die Anweisung **Resume Next**. Funktionen und Prozeduren mit dieser Anweisung sollten nicht für den AP-SystemInfo-Manager angepasst werden.

Wie der VBA-Code innerhalb Formularen angepasst wird, sieht man auch im Beispiel **frm_Auswahl**. Öffnen Sie dieses Formular in der Entwurfsansicht und lassen Sie sich wie bereits beschrieben die Codeansicht anzeigen. Hier sehen Sie bei den Prozeduren **Datei_kopieren_Click**, **Domänenaggregatfunktionen_Click** und **Rechnen_Click** wie der VBA-Code angepasst wurde.

Schlußwort:

Wir waren bei der Erstellung dieser Beschreibung bemüht, die Integration des AP-SystemInfo-Managers in Ihre Anwendungen so gut wie möglich zu beschreiben. Sollten Sie dennoch Fragen zum Einbau des Moduls haben, so schicken Sie diese per Mail an support@access-paradies.de. Teilen Sie uns in dem Mail bitte Ihre Kundennummer, Rechnungsnummer und die von Ihnen verwendete Accessversion mit.

Wir wünschen Ihnen viel Freude an dem AP-SystemInfo-Manager

Microsys Kramer

Access-Paradies

<http://www.access-paradies.de>

<http://www.ms-office-forum.net>