EntwicklerCamp 2014

Wo speichert Notes was?

Bernhard Köhler

Vorstellung

- Quereinsteiger: Studium Physik / Astronomie
- Seit 1992 Beschäftigung mit Notes
- Bis 1999 Entwicklung und Administration
- Heute: Entwickler und „Aushilfs-Admin“
- CC Groupware, Bechtle Systemhaus Freiburg

Übersicht

- Wichtige Informationen des Clients
- Voreinstellungen – was steht wo?
- Mail-DB, NAMES.NSF, BOOKMARK.NSF und Co.
- Desktop, Workspace-Verzeichnis, NOTES.INI
- Designelemente und ihre Merkmale
- Und wo speichern wir?

NOTES.INI

- Einfacher Zugriff über

  @Environment und

  NotesSession.GetEnvironmentString / Value
- Über Formel nur eigene ($-)Variable
- LotusScript: Kompletter Zugriff

NOTES.INI

- Beispiel: Preferences
- Achtung: Wert ist Zahl, nur per LS erreichbar!

  0 <0> = Keep workspace in back when maximized (Enabled=1)

  01 <2> = Scan for unread

  02 <4> = <Unknown?>

  03 <8> = Large fonts

NOTES.INI

05 <32> = Make Internet URLs (http//:) into hotspots

06 <64> = <Unknown?>

07 <128> = Typewriter fonts only

08 <256> = Monochrome display

09 <512> = Scandinavian collation

10 <1024> = <Unknown?>

11 <2048> = <see 28>

NOTES.INI

12 <4096> = Sign sent mail (Enabled(1))

13 <8192> = Encrypt sent mail

14 <16384> = Metric(1)/Imperial(0) measurements

15 <32768> = Numbers last collation

16 <65536> = French casing

NOTES.INI

17 <131072> = empty trash folder (prompt during db close=0/always during db close=1(plus "EmptyTrash=1" in INI)/manual=1(plus "EmptyTrash=2" in INI)

18 <262144> = Check for new mail every x minutes (Enabled=0)

19 <524288> = Enable local background indexing

NOTES.INI

20 <1048576> = Encrypt saved mail

21 <2097152> = <Unknown?>

22 <4194304> = <Unknown?>

23 <8388608> = Right double-click closes window

24 <16777216> = Prompt for location

25 <33554432> = <Unknown?>

NOTES.INI

26 <67108864> = Mark documents read when opened in the preview pane

27 <134217728> = Enable local scheduled agents

28 <268435456> = Save sent mail (Always prompt=10/Don't keep a copy=00/Always keep a copy=01)

29 <536870912> = <see 30>

NOTES.INI

30 <1073741824> = New mail notification

(None=10/Audible=00/Visible=01)

31 <2147483648> = Textured workspace

NOTES.INI (Server)

- Nur via LotusScript erreichbar
- ReturnValue$ = NotesSession.SendConsoleCommand (server, command)
- Auslesen:

  show config INI-Variable oder

  show config *

NOTES.INI (Server)

- Setzen von Server-INI-Variablen:

  set config command=wert

NAMES.NSF (Client)

- Bekanntermassen die wichtigste DB des Clients
- Anlage nur bei Neukonfiguration, daher keine Experimente
- Zentraler Ort für essentielle Informationen des Clients

NAMES.NSF (Client)

- Instantiierung einfach
- Name und Ort de facto immer names.nsf,

  sonst

  Session.GetEnvironmentString („Directory“, True)

  (NOTES.INI wird vor NAMES.NSF gelesen)
- Diverse Dokumenttypen …

NAMES.NSF (Client)

- Arbeitsumgebung (Location)
- Erreichbar über Ansicht „Locations“ (Alias)
- Name und Typ der Location:

  Name

  LocationType

  (0 – LAN, 1 – Dialup, 2 – Custom, 3 – No connection,

  4 – Network dialup)

NAMES.NSF (Client)

- Verbindungsprobleme?

  In den Items $SavedAddresses und $SavedServers

  Items haben keine Entsprechungen in Feldern

  Items können geleert werden (Leerstring zuweisen oder Item löschen) oder

  Einträge können selektiv entfernt werden (Korrelation beachten!)

NAMES.NSF (Client)

- Adressierungsprobleme?

  Das zuletzt verwendete Adreßbuch steht im Item

  MailaddressFile

  Item hat keine Feldentsprechung

  Item kann gelöscht / geleert werden

  Ablageformat: „server!!pathfile.nsf“

NAMES.NSF (Client)

- Verwendete DBs?

  $LastFileNames (löschbar)

  $StackedFileNames

  $LastReplicaIDs

- Angaben zu den DBs auf der Replicator Page?

  Kein lesbares Item, eigener Datentyp – unerreichbar

  Location-Dokument muss hierzu gelöscht werden

NAMES.NSF (Client)

- Angaben zum Mailfile:

  MailFile

  MailType (0 – on server, 1 – local)

  Domain

NAMES.NSF (Client)

- Dokumenttyp Connection

  Speichert alle erforderlichen Informationen zu Notes-Verbindungen

  Programmatisch erstellbar und änderbar

NAMES.NSF (Client)

- Recent Contacts:

  Zu finden in der Ansicht (Recent Contacts alias RecentCollaborators)

  Ansicht kann (gezielt) geleert werden, aber …

  NOTES.INI muss auch bereinigt werden

  (Variablennamen beginnen alle mit DPAB)

  Und:

  workspace\.metadata\.plugins\com.ibm.notes.dip

NAMES.NSF (Client)

- Recent Contacts:

  Und:

  workspace\.metadata\.plugins\com.ibm.notes.dip

NAMES.NSF (Client)

- Erhaltene Policies:

  Ansicht ($Policies)

  Ansicht kann geleert werden

  Neuziehen erfolgt nicht zwingend beim nächsten Clientstart!

NAMES.NSF (Client)

- Contact Synchronization:

  ProfileDocument „DirectoryProfile“

  Profile ist personengebunden

Desktop

- Desktop-File ist „im wesentlichen“ eine Notes-Datenbank
- Änderung der File Extension zu „NSF“ erlaubt das Öffnen im Designer
- Struktur aber „sehr speziell“ und real NICHT nutzbar!
- Werkzeuge einsetzen (Panagenda, BCC)

BOOKMARK.NSF

- Wenig Eingriffsmöglichkeiten
- Dokumente sind löschbar (die meisten)
- Symbolleisten: Outline „UserToolbar“
- Lesezeichen: Ebenfalls Outlines wie UserBookmarkOrder

Rules / Regeln

- Verwaltung über Regel-Dokumente
- Für die Ausführung der Regeln sind diese Dokumente aber völlig unnötig:
- Regeln sind in Items des CalendarProfiles der Mail-DB gespeichert (tokenisiert)

Regeln / Rules

ab	$DPLocked	1	219	0	429	
ab	$Event	1	219	0	374	
ab	$Filter_BlockAddressList	1	219	0	219	
@	$FilterFormula	1	219	0	62	@False
@	$FilterFormula_1	1	219	0	435	CondSubject_C := @LowerCase(Subject); @If((@Contains(CondSubject_C;"nagios@neoperl.de")) ; (@Do((@MailFilterAddToFolder("52A74F38911D...
@	$FilterFormula_2	1	219	0	436	CondSender_C := @LowerCase(@Name([Abbreviate]; @Unique(From:Principal:SMTPOriginator))) ; @If(((@Contains(CondSender_C ;"@schiedel"...
@	$FilterFormula_3	1	219	0	437	CondSubject_C := @LowerCase(Subject); @If((@Contains(CondSubject_C;"new account")) ; (@Do((@DeleteDocument)); "")
@	$FilterFormula_4	1	219	0	438	CondInternetDomain_C := @If(@IsAvailable(SMTPOriginator); @If(@ValidateInternetAddress([Address821]; SMTPOriginator) = "";@Right(SMTPOri...
ab	$FilterFormulaCount	1	219	0	48	5
ab	$LangOwner	1	219	0	60	
ab	$Meeting	1	219	0	370	

- Regeln müssen explizit deaktiviert werden.
- Das Löschen einer Regel entfernt diese nicht aus dem CalendarProfile!
- Das erklärt häufige Probleme bei Anwendern

Regeln / Rules

- Wie wird man Regeln wieder los?

  NotesDatabase.GetProfileDocument („CalendarProfile“)

  Forall NotesDocument.Items …

  If Instr (itm.Name, „$FilterFormula“) = 1 -> itm.Remove

- Nicht vergessen, $FilterFormulaCount anzupassen!

Regeln / Rules

- Es geht natürlich auch eleganter:
- Durch vorhandene $FilterFormula im CalendarProfile laufen
- Regeldokumente (haben auch $FilterFormula) durchlaufen und vergleichen.
- Dokument nicht gefunden / deaktiviert: Regel im CalendarProfile löschen

CalendarProfile

- Mail-Datenbank ist zentrales Repository für Mail, Calendaring & Scheduling
- Alle Items des Profils sind Read / Write
- Profil ist löschbar
- Immer an das Caching denken – solange der Benutzer MailDB geöffnet hat, sind externe Änderungen „flüchtig“

CalendarProfile

- Interessante Items:
- $Filter_BlockAddressList für blockende Absenderadressen kann überlaufen – Items muss dann gelöscht werden
- $Times Items: Verfügbarkeit für Planer / $BusyTime. Kann bei Problemen gelöscht werden, baut sich beim nächsten Öffnen im Client wieder auf

CalendarProfile

- $WorkDays ähnlich $Times, Zahlen entsprechen WeekDay-Werten (Sonntag = 1, Samstag = 7)
- AutoAnniverseryRepeat: Vorgabe für Wiederholung der Jahrestage (Standard: 25 Jahre)
- CalendarTimeSlotDuration (Standard = 60) für „Kalenderraster“ / Termindauervorabe
- CalendarTimeSlotStart und End: Terminvorgaben

CalendarProfile

- dspSender*-und Sender*-Items: Absenderspezifische Farbdarstellung für Ansichten (siehe auch ColorProfile)
- EnableBlankSubject („1“ = ja) regelt Warnmeldung bei fehlendem Subject
- CalEnableOverlay („1“ = ja) regelt überlappende Termine

CalendarProfile

- ReadCalendar / ReadMail und entspr. Write-Items: Kalender- und Maildelegierung, Vorgabe für Übernahme in ACL
- UserCategories: Vorgaben für Kalenderkategorien (evtl. nützlich für Ergänzungen)
- Owner: Wohl das wichtigste Item, der „Besitzer" der MailDB

Weitere Profile

- CalendarSettings: Kalendervorgaben

  Userspezfisches Profile!
- CalColorProfile: Userspezifische Farbeinstellungen für Kalender im normalen RGB-Speicherstandard
- ColorProfile: Weitere Farbeinstellungen der MailDB
- OutOfOfficeProfile: Alle Angaben zum OoO-Agent

  Wird vom Service nur noch teilweise verwendet!

OutOfOfficeProfile

- CurrentStatus = „1“ – OoO ist aktiv
- Wichtig: Prüfen auf Owner-Item. Gelöschte User können „ewig gültige“ OoO-Einstellungen im Profil haben
- DateFirstOut/Back und firstDayOut/Back legen Abwesenheitsperiode fest

OutOfOffice Profile

Folgender einfacher Code liest die Infos zum OoO aus:

```
Set docOoOProfile = dbMail.GetProfileDocument ("outofofficeprofile")
If docOoOProfile Is Nothing Then
   Call ReportWriteWarning (dbMail.FilePath & ": Profile document 'outofofficeprofile' not found!")
   GoTo GetNextSourceDatabase
End If

If docOoOProfile.CurrentStatus (0) = "1" And docOoOProfile.~$Owner (0) <> "" Then
   If IsDate (docOoOProfile.DateFirstDayOut (0)) = True And IsDate (docOoOProfile.DateFirstDayBack (0)) = True Then
      If docOoOProfile.DateFirstDayOut (0) <= Today And docOoOProfile.DateFirstDayBack (0) > Today Then

         Set docOoOStatus = dbCurrent.CreateDocument
         docOoOStatus.Form = "Absence"
         docOoOStatus.AbsentPerson = docOoOProfile.~$Owner
         docOoOStatus.DateFirstDayOut = DateNumber (Year (docOoOProfile.DateFirstDayOut (0)), Month (docOoOProfile.DateFirstDayOut (0)), Day (docOoOProfile.DateFirstDayOut (0)))
         docOoOStatus.DateFirstDayBack = DateNumber (Year (docOoOProfile.DateFirstDayBack (0)), Month (docOoOProfile.DateFirstDayBack (0)), Day (docOoOProfile.DateFirstDayBack (0)))

         Set namPerson = New NotesName (docOoOStatus.AbsentPerson (0))
         Call ReportWrite ("Created new document for " & namPerson.Common & " (" & dbMail.Title & " / " & CStr (docOoOStatus.DateFirstDayOut (0)) & " - " & CStr (docOoOStatus.DateFirstDayBack (0)))

         Call docOoOStatus.Save (True, False, True)


      End If 'of "If docOoOProfile.DateFirstDayOut (0) <= Today And docOoOProfile.DateFirstDayBack (0) > Today"
   End If 'of "If IsDate (docOoOProfile.DateFirstDayOut (0)) = True And IsDate (docOoOProfile.DateFirstDayBack (0)) = True"
End If 'of "If docOoOProfile.CurrentStatus (0) = "1" And docOoOProfile.~$Owner (0) <> """
```

Designelemente

- Designelemente sind auch „nur“ Notes Documents
- Wie kann der Zugriff erfolgen?

  - „Brutal“ über NoteID. Da es für Designelemente keine fixen IDs gibt, können das 16 hoch 8 sein!
  - Einfach bei Views und Folders über Database.Views

    $Flags (F = Folder) für Folder, sonst View

    Interessant: Private Folder hat Flag mit „V“

Designelements

- Neben Views / Folders gibt es auch NotesDatabase.Forms und Agents
- Rückgabewert ist ein Array of NotesAgent oder Form Object
- Jedes Object hat wieder eine NoteID
- Hierüber ist mit NotesDatabase.GetDocumentByID das NotesDocument erreichbar

Designelemente

- Über API (z.B. DBDesign Class von Damian Katz)

  Schneller als ein Loop über alle NoteIDs, aber nach wie vor ein Loop über alle Designelemente

- State of the Art: NoteNoteCollection

Designelements

Vorgehen:

Set nnc = NotesDatabaseCreateNoteCollection (flag)

nnc.SelectAgents = True

Call nnc.BuildCollection

Die NoteCollection beinhaltet jetzt alle Agents

Designelements

- Die Elemente unserer NotesNoteCollection haben wir eine NoteID Property
- NotesDatabase.GetDocumentByID gibt das zugehörige Dokument zurück
- Uns stehen nun alle Items des Documents zur Verfügung:

Designelements

- Neben spezifischen Items finden wir
- $Title (Elementname, mit Alias(en) ein Array)
- $Flags (elementspezifisch bei Views z.B. private, public, SPOFU usw., bei Agents der Agenttyp)
- $Class (Abhängigkeit von Template)
- Einheitlich (fast): Das ProtectionFlag „P" zeigt an, das Designelement vor Designupdate geschützt ist.

Designelements

- Ausnahme: Das Database Icon. Hier lautet das ProtectionFlag „R“ – und darf NICHT enthalten sein

Designelements

- Spezielle Design Items haben ganz eigene Datentypen
- Sie sind dadurch für uns nicht direkt manipulierbar.
- Beispiel Views mit $Formula.
- Hier kann man mit Formel-Feldern arbeiten oder
- greift auf eine vorhandene Property der NotesView zu SelectionFormula

Designelements

- Das klingt verlockend, aber …
- … hierfür sind mindestens Designer-Rechte nötig
- Alternativ:

  - Serverbased Agent (entsprechend signiert) und RunOnServer

  - danach Index neu aufbauen lassen!

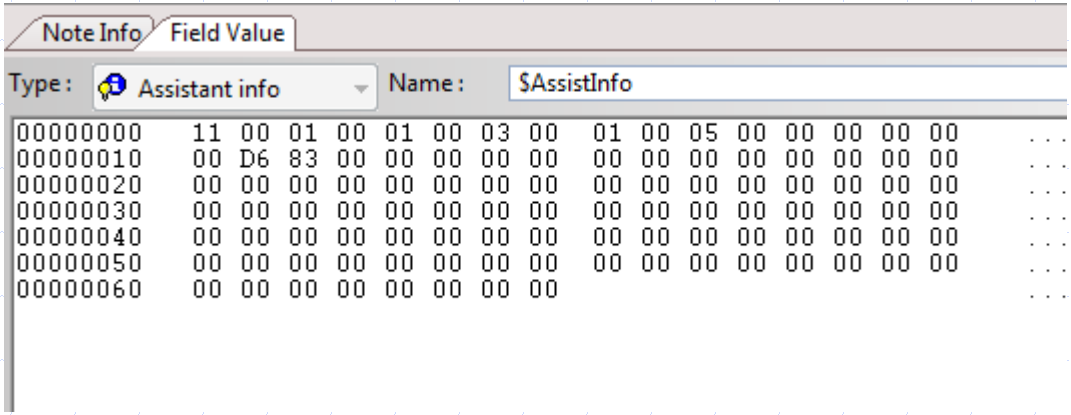
  Konkurrierende Anforderungen beachten!

Designelements

- Andere spezielle Datentypen lassen sich nur via API lesen und schreiben
  - API-Referenz
  - Tipps und Tricks aus dem Web.
  - Oft am besten: OpenNTF
- Code trotzdem verstehen!
- Fehlerhafter API-Code kann auch DB zerstören!

Designelements

- Beispiel: Auslesen der Agentlaufzeit
- Startwerte etc. stehen im speziellen Datentyp $AssistInfo



$AssistInfo auslesen

```
Dim b As Long
Dim dt As Integer
Dim np As String
Dim nt As Integer
Dim nv As Long
Dim p As Long
Dim pt As Long
Dim t As String
Dim ti As String
Dim v As Long

Public Type AgentProperties
   Comment As String
   IsEnabled As Integer
   LastRun As Variant
   Owner As String
   ServerName As String
End Type

Const wAPIModule = "NNOTES"  ' Windows/32
Const NOTE_CLASS_FILTER = &H0200

Type AssistInfo
   Version As Integer
   TriggerType As Integer  ' 0 none, 1 schedule, 2 new mail, 3 paste, 4 manual, 5 update, 6 router
   SearchType As Integer  ' 0 none, 1 all, 2 new, 3 new/mod, 4 selected, 5 view, 6 unread, 7 prompt, 8 UI
   IntervalType As Integer  ' 0 none, 1 minutes, 2 days, 3 weeks, 4 months
   Interval As Integer
   Time1 As Variant  ' Start time (ms since midnight)
   Time2 As Variant  ' Long (weekday or day of month) or end time (ms since midnight)
   StartTime As Variant  ' Time/Date
   EndTime As Variant  ' Time/Date
   Flags As Long  ' 1 hidden, 2 no weekends, 4 store highlights, 8 mail/paste, 16 choose server
   Spare(15) As Long
End Type
```

$AssistInfo auslesen

```
Type BlockID
   hPool As Long
   Block As Integer
End Type

Declare Function ConvertTIMEDATEToText Lib wAPIModule Alias "ConvertTIMEDATEToText" _
( Byval zI As Long, Byval zT As Long, Byval T As Long, Byval S As String, Byval nS As Integer, nT As Integer) As Integer

Declare Function ConvertTextToTIMEDATE Lib wAPIModule Alias "ConvertTextToTIMEDATE" _
( Byval zI As Long, Byval zT As Long, pS As Long, Byval nS As Integer, Byval T As Long) As Integer

Declare Private Function NIFFindDesignNote Lib wAPIModule Alias "NIFFindDesignNote" _
( Byval hDB As Long, Byval S As String, Byval C As Integer, N As Long) As Integer

Declare Private Function NSFDbOpen Lib wAPIModule Alias "NSFDbOpen" _
( Byval PathName As String, DbHandle As Long) As Integer

Declare Private Function NSFDbClose Lib wAPIModule Alias "NSFDbClose" _
( Byval DbHandle As Long) As Integer

Declare Private Function NSFNoteOpen Lib wAPIModule Alias "NSFNoteOpen" _
( Byval hDB As Long, Byval NoteID As Long, Byval F As Integer, hNT As Long) As Integer

Declare Private Function NSFNoteClose Lib wAPIModule Alias "NSFNoteClose" _
( Byval hNT As Long) As Integer

Declare Function NSFItemInfo Lib wAPIModule Alias "NSFItemInfo" _
( Byval hNT As Long, Byval N As String, Byval nN As Integer _
, iB As BlockID, D As Integer, vB As BlockID, nV As Long) As Integer

Declare Private Function NSFNoteUpdate Lib wAPIModule Alias "NSFNoteUpdate" _
( Byval hNT As Long, Byval F As Integer) As Integer

Declare Function NSFItemDelete Lib wAPIModule Alias "NSFItemDelete" _
( Byval hNT As Long, Byval N As String, Byval nN As Integer) As Integer

Declare Private Function OSMemAlloc Lib wAPIModule Alias "OSMemAlloc" _
( Byval T As Integer, Byval N As Long, hM As Long) As Long

Declare Private Function OSMemFree Lib wAPIModule Alias "OSMemFree"
```

$AssistInfo auslesen

```
np$ = Space (1024)
OSPathNetConstruct 0, dbCurrent.Server, dbCurrent.FilePath, np$

Dim hDB As Long
NSFDbOpen np$, hDB
If hDB = 0 Then
   Messagebox "Die Datenbank kann nicht geöffnet werden !", MB_ICONEXCLAMATION, "Fehler"
   Exit Sub
End If

pt& = Instr (szAgentName, "|")
If pt& = 0 Then ti$ = Trim$(szAgentName) Else ti$ = Trim$(Left$(szAgentName, pt& - 1))

Dim nID As Long
NIFFindDesignNote hDB, ti$, NOTE_CLASS_FILTER, nID
If nID = 0 Then
   Messagebox "Zugriff auf den Agent '" & szAgentName & "' ist nicht möglich !", MB_ICONEXCLAMATION, "Fehler"
   Exit Sub
End If

If Instr (dbCurrent.GetDocumentByID (Hex$ (nID)).~$Flags (0), "f") = 0 Then
   Messagebox "'" & szAgentName & "' ist kein Agent !", MB_ICONEXCLAMATION, "Fehler"
   Exit Sub
End If

Dim hNT As Long
NSFNoteOpen hDB, nID, 0, hNT

Dim iB As BlockID, vB As BlockID
NSFItemInfo hNT&, "$AssistInfo", 11, iB, dt%, vB, nv&
If Not vB.hPool = 0 Then
   p& = OSLockObject (vB.hPool) + vB.Block
   Peek info.Version, p& + 2, 2
   Peek info.TriggerType, p& + 4, 2
   Peek info.SearchType, p& + 6, 2
   Peek info.IntervalType, p& + 8, 2
   Peek info.Interval, p& + 10, 2
   Peek v&, p& + 12, 4
   If v& <= 31 Then info.Time1 = v& Else info.Time1 = Cdat (v&/100/60/60/24)
   Peek v&, p& + 16, 4
   If v& <= 31 Then info.Time2 = v& Else info.Time2 = Cdat (v&/100/60/60/24)
   t$ = Space (81)
   ConvertTIMEDATEToText 0, 0, p& + 20, t$, 80, nt%
   If Not nt% = 0 Then info.StartTime = Cdat (Left$ (t$, nt%))
   t$ = Space (81)
   ConvertTIMEDATEToText 0, 0, p& + 28, t$, 80, nt%
   If Not nt% = 0 Then info.EndTime = Cdat (Left$ (t$, nt%))
   Peek info.Flags, p& + 36, 4
   OSUnlockObject vB.hPool
End If

NSFNoteClose hNT
NSFDbClose hDB
```

Designelements

- Hinweise zum Auslesen / Setzen von Items in Designelementen (gilt für alle Dokumente / Items)
- Problemlos: GetItemValue / ReplaceItemValue
- Direkte Notation (NotesDocument.Itemname =) scheitert, wenn Itemname mit $ beginnt
  -> hier muß eine Tilde vorangestellt werden

Agentlaufzeit

- Wie kann man verhindern, daß Agents auf Grund der konfigurierterten maximalen Laufzeit vom Agentmanager zwangsweise beendet werden?

Agentlaufzeit

- Wie kann man verhindern, daß Agents auf Grund der konfigurierterten maximalen Laufzeit vom Agentmanager zwangsweise beendet werden?
- Eine Strategie – niemals schädlich:

  Logischer Aufbau – ein Abbruch würde nie schaden anrichten.

  Erneutes Bearbeiten wird verhindert.

Agentlaufzeit

- Wir verhindern, daß die max. Agentlaufzeit erreicht wird:
  - Aktuellen Server ermitteln (NotesDatabase.Server)
  - Domino Directory öffnen
  - Serverdokument instantiieren
  - MaxExecution auslesen
  - Laufzeit überwachen

Agentlaufzeit

- Weiterer Tipp – wir speichern selber:

  Merken Sie sich in einem Konfigurations- oder Protokolldokument, wann der Agent letztmals lief

  Logisches Problem lösen je nach Anforderung:

  Agentstart oder erfolgreiches Ende?

  Oder beides?

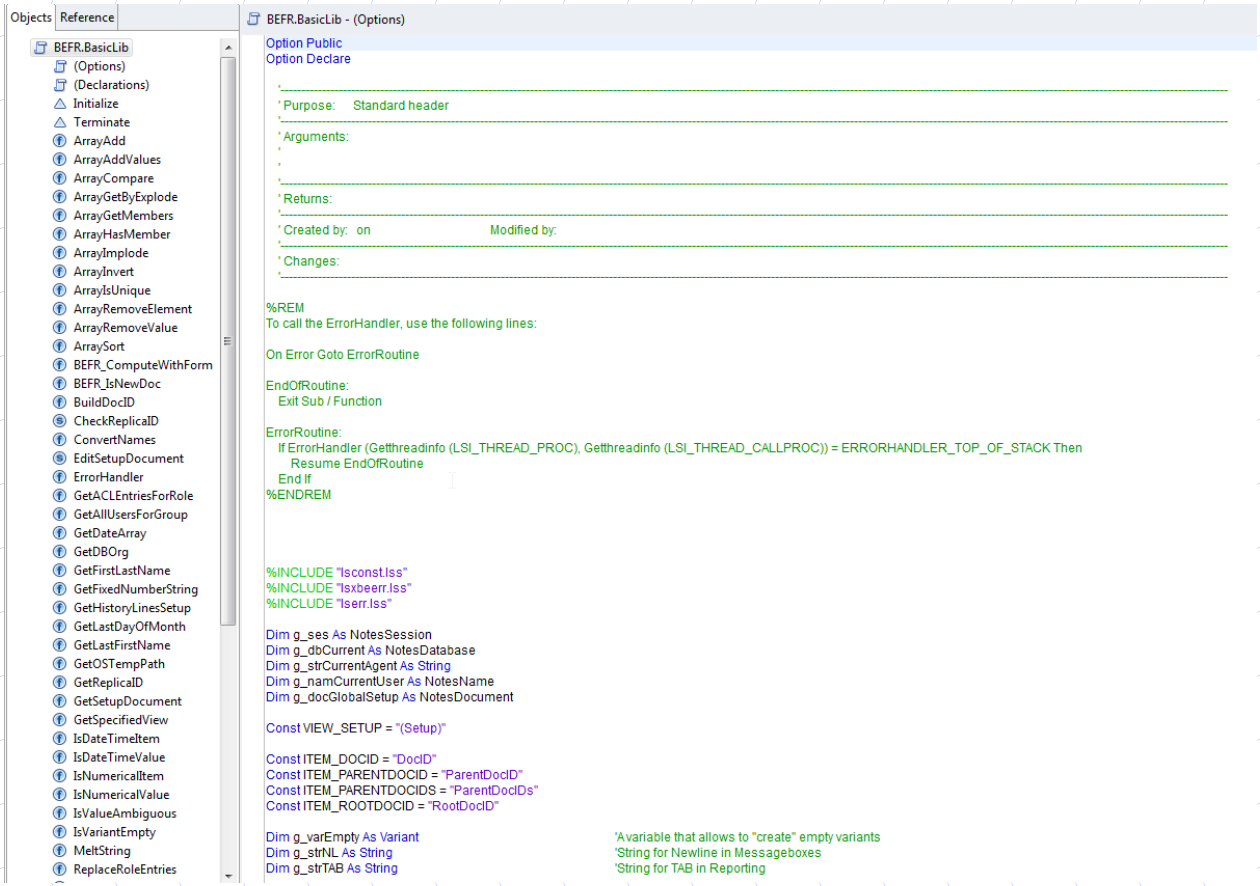
Designelements

- ScriptLibraries verfügen über ein oder mehrere Items namens $ScriptLib

Designelements

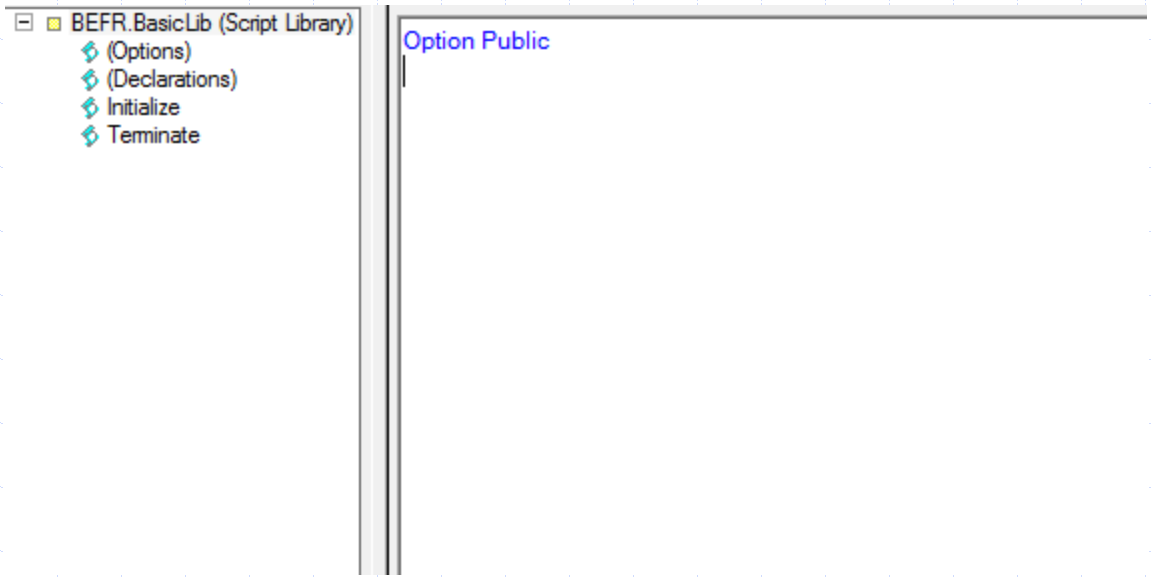
- ScriptLibraries verfügen über ein oder mehrere Items namens $ScriptLib
- Hierin ist der Quelltext der Bibliothek gespeichert
- daher sind es ggf. mehrere gleichnamige Items
- Der kompilierte Code steht aber in $ScriptLib_O
- $ScriptLib kann gelöscht werden …

Designelements



Designelements

wird so etwas:



Alle anderen Designelemente sind unverändert.

Und wo speichern wir?

- Datenbankweite Informationen ablegen ist nicht trivial
- Ungeeignet:
  - DatabaseScript
  - ScriptLibraries
  - Shared fields

Und wo speichern wir?

- Administrative Vorgaben:

  Konfigurationsdokumente

  Autorfeldgeschützt

  Replizierkonflikte überwachen!

- Profildokumente sind wegen Caching kritischer

Und wo speichern wir?

- Zwei prinzipielle Verfahren:

  - NOTES.INI

  - persönliche Profile

- Verfahren werden unterschiedlich bewertet, Entscheidung bleibt subjektiv

Pros und Kontras

- NOTES.INI:
  - an Client und nicht an ID gebunden
  - INI wird unübersichtlich / schwer pflegbar (das Löschen ist nicht 100% zuverlässig)
  - bei Roaming ist Mobilität gegeben (wenn Roaming zuverlässig funktioniert)
  - keine Rechteprobleme

Pros und Kontras

- Persönliches Profil:

  - an ID und nicht an Client gebunden
  - zentrale Pflege ist möglich
  - „Aufräumen“ ist nicht erforderlich
  - Rechteproblematik: Für Leser muss ein komplexeres Verfahren eingesetzt werden (RunOnServer)

Hilfreiche Werkzeuge

- MayFlower Docviewer (Freeware)
- NotesPeek (Freeware)
- Ytria ScanEZ (mächtiges Tool, aber teuer)