Notes.net

Iris Today

Common ground: COM access to Domino objects

by
Robert
Perron

Level: Advanced
Works with: Designer 5.0
Updated: 01/10/2000

Inside this article:

Related links:

QMR/QMU Incremental Installers

Documentation Library

Lotus Developer Network Technology and Toolkits Zone

Sidebar: Common paths to Domino objects

Get the PDF:

Visual Basic and other programmers working outside Domino and Notes have for many years accessed the Domino objects through OLE (Object Linking and Embedding). But OLE:

- Requires the presence of the Notes client on the accessing machine, and Notes must be running. If Notes is not running at the time of OLE access, it is launched for you as a separate process.
- Supports only late binding. Objects are generic and cannot be typed for compile-time checking and browser lookup.

Now -- as of release 5.0.2b -- the Domino objects are accessible through the Component Object Model (COM) with these benefits:

- COM requires the presence of Domino or Notes but the software can be Domino server, Domino Designer, or Notes client.
- The Domino or Notes software does not have to be running.
- Supported are both early (compile-time) and late (run-time) binding.

You can download the 5.0.2b QMU from the **QMR and QMU Incremental Installers** page on Notes.net.

The Domino objects consist of the back-end classes available through LotusScript and Java. For the front-end classes -- NotesTimer, NotesUIDatabase, NotesUIDocument, NotesUIView, and NotesUIWorkspace -- you still need OLE and a running Notes client.

The examples in this article are in Visual Basic (early binding) and VBScript (late binding). However, you can access the Domino objects from any language or application interface that supports COM, including C++, VBA in Word and Excel, and VBScript in ASP.

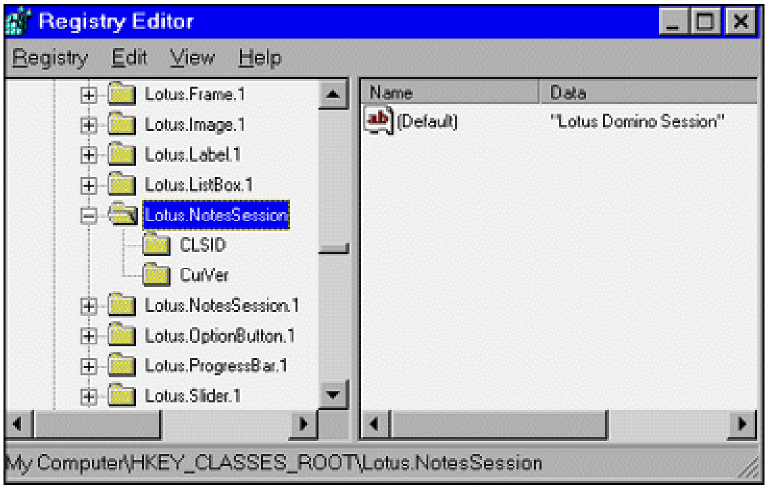
Getting additional information

The Domino Designer documentation will be updated for release 5.0.3 to include COM information. When it is available, see the topic "Accessing the Domino objects through COM" in the updated Domino 5 Designer Help database. In the index, look for "COM." In the contents, scroll through the LotusScript/COM/OLE topics. The new help database will be posted in the **Documentation Library** on Notes.net and will be available in the release 5.0.3 software kits.

The Lotus Domino and Notes Toolkit for COM is available for download from the **Lotus Developer Network Technology and Toolkits Zone**. The toolkit contains sample code, the COM support files, and documentation. The code examples are in C++, Visual Basic, VBA in Word and Excel, and VBScript in ASP.

Installation and registration

After installing release 5.0.2b, you will see in the operating system registry a class named Lotus.NotesSession, as shown below. This is the root of the Domino COM classes. Still present are Notes.NotesSession and Notes.NotesUIWorkspace which are the OLE roots.

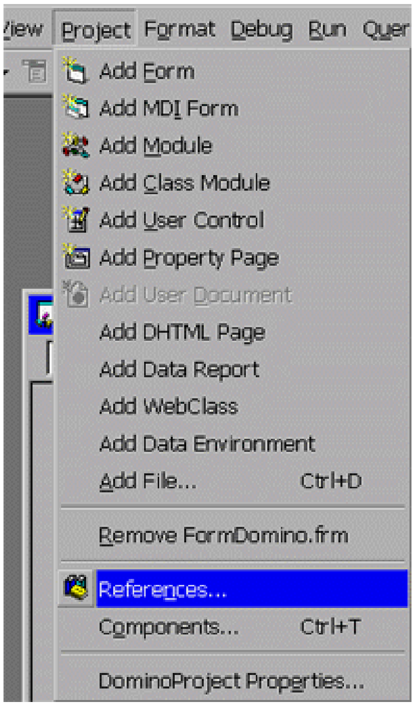


Name	Data
(Default)	"Lotus Domino Session"

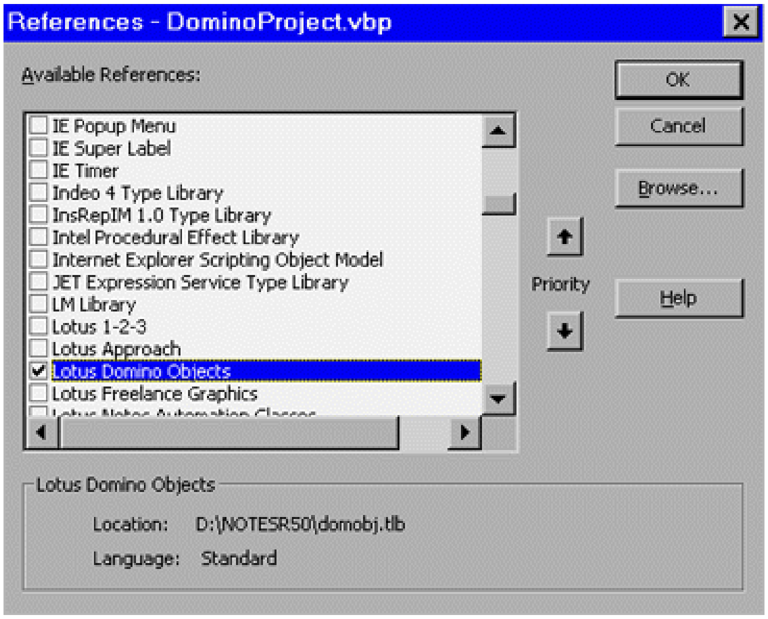
My Computer\HKEY_CLASSES_ROOT\Lotus.NotesSession

Getting started in Visual Basic

In Visual Basic, create or open a project and choose Project - References:



Scroll down and check "Lotus Domino Objects":



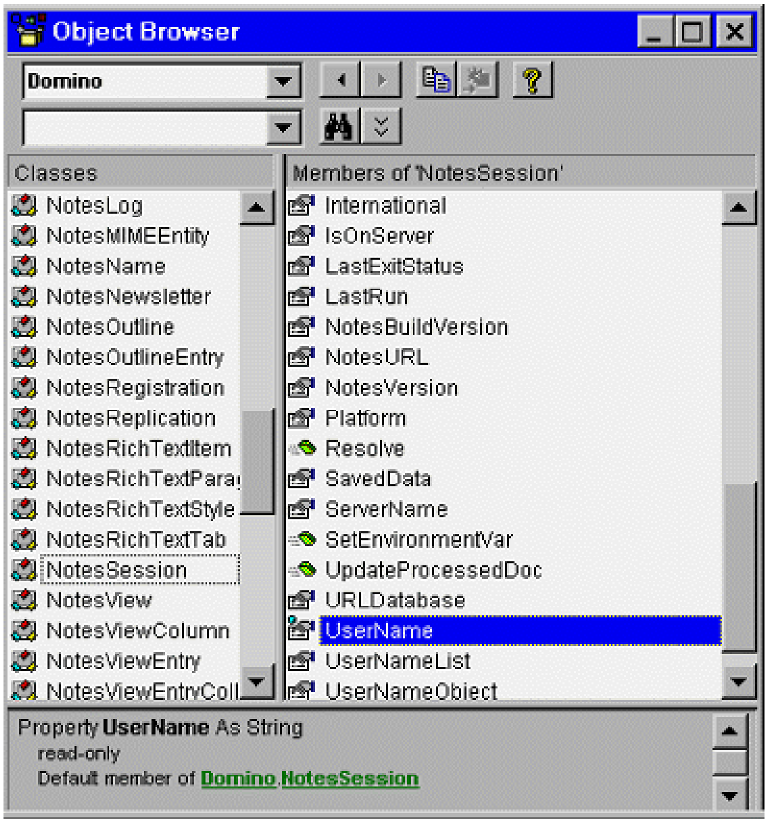
□ IE Popup Menu
□ IE Super Label
□ IE Timer
□ Indeo 4 Type Library
□ InsRepIM 1.0 Type Library
□ Intel Procedural Effect Library
□ Internet Explorer Scripting Object Model
□ JET Expression Service Type Library
□ LM Library
□ Lotus 1-2-3
□ Lotus Approach
☑ Lotus Domino Objects
□ Lotus Freelance Graphics

Lotus Domino Objects

Location: D:\NOTESR50\domobj.tlb

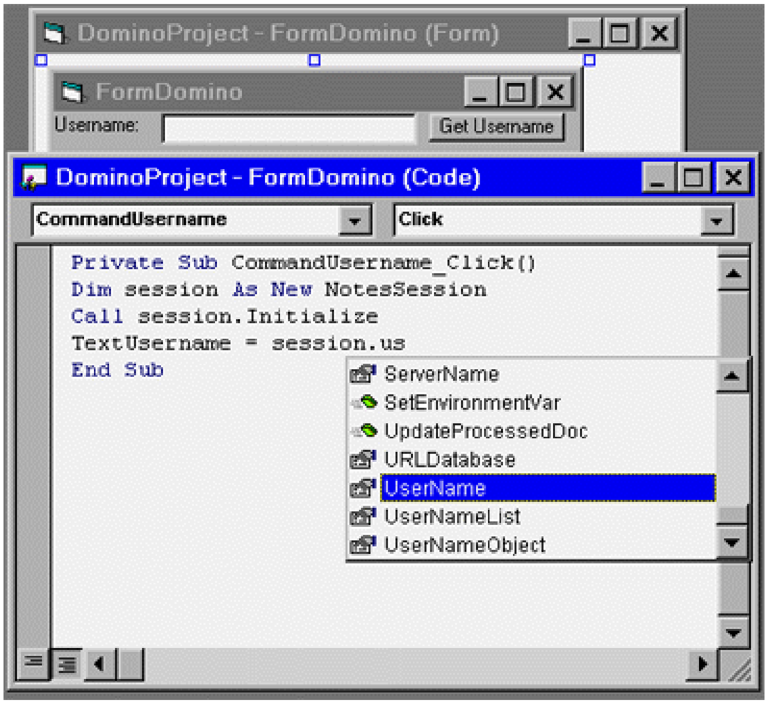
Language: Standard

The Domino objects then become available to the project. You can access them through the object browser.



Property **UserName** As String
read-only
Default member of Domino.NotesSession

You can access them from a drop-down menu that becomes available after you type the name of an object and a period.

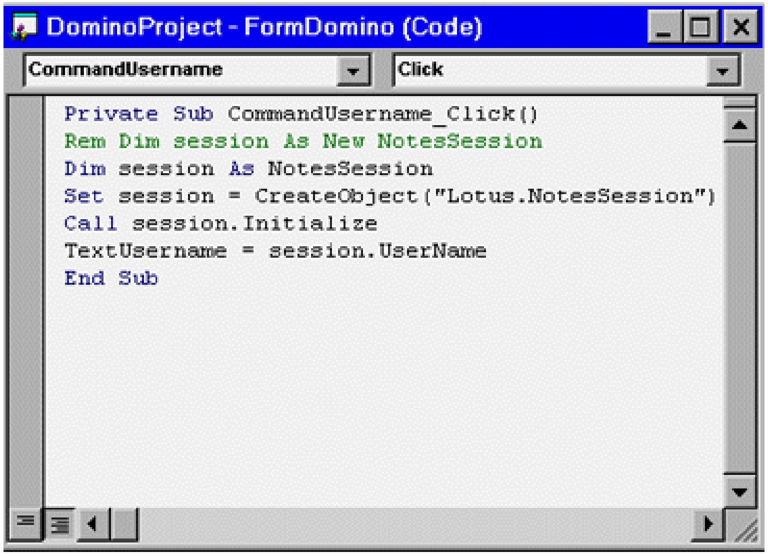


```
Private Sub CommandUsername_Click()
Dim session As New NotesSession
Call session.Initialize
TextUsername = session.us
End Sub
```

Initializing a Domino session

Create the root NotesSession object with Dim New or CreateObject. Then initialize the session with one of the COM-only NotesSession methods Initialize or InitializeUsingNotesUserName. A COM session requires explicit initialization, which is not the case in LotusScript.

The following Sub shows the minimum code needed to access the Domino objects. The code shows both techniques for creating the NotesSession object, the Dim New technique being commented out.



```
Private Sub CommandUsername_Click()
Rem Dim session As New NotesSession
Dim session As NotesSession
Set session = CreateObject("Lotus.NotesSession")
Call session.Initialize
TextUsername = session.UserName
End Sub
```

The Initialize method shown in the code bases the session on the current Domino or Notes user ID. If the user ID requires a password, a prompt appears at run-time. Optionally you can specify the password in the code:

```
Call session.Initialize("UserIDPassword")
```

If the machine contains both server and client software, the user ID depends on which software is current in the operating system registry.
If the code is running on a machine containing a Domino server, you can base the session on a user name in the local Domino Directory with:

```
Call session.InitializeUsingNotesUserName("UserName")
```

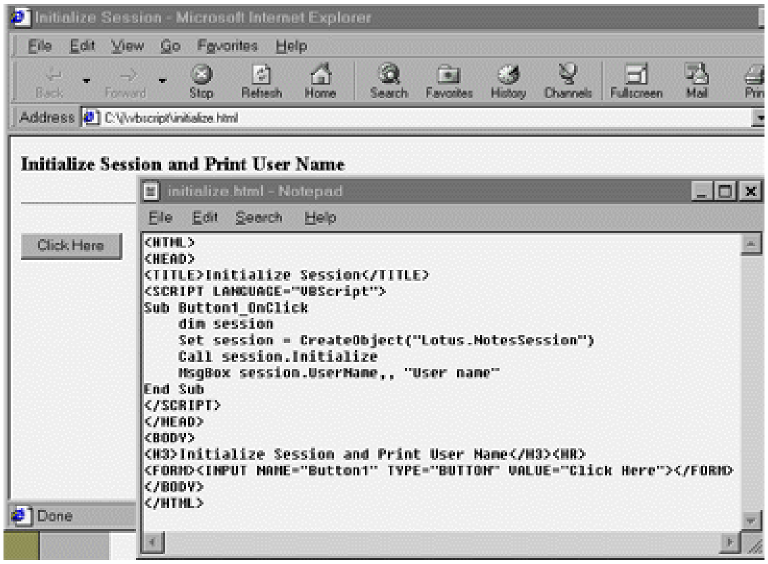
An empty string means "Anonymous." Access depends on the "Server Access" and "Java/COM Restrictions" settings in the user record in the Domino Directory.

In this case, the password is the Internet password in the user record in the Domino Directory. If a password is required and you do not want prompting or the software (for example, VBScript under ASP/IIS) does not support prompting, specify a second parameter:

```
Call session.InitializeUsingNotesUserName("UserName", "InternetPassword")
```

Getting started in VBScript

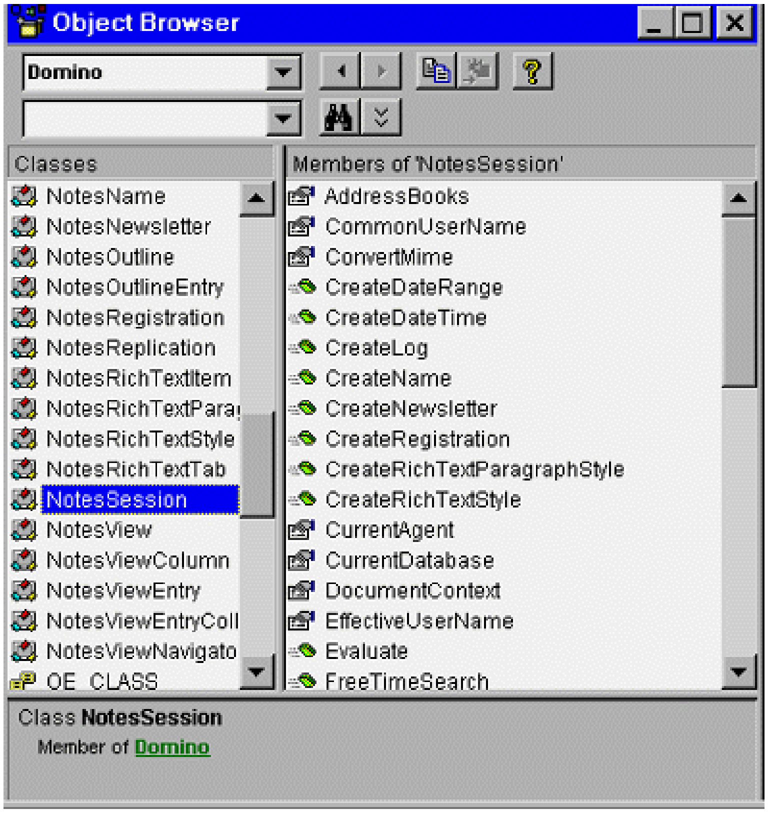
Only late binding is available in VBScript. The language precludes data typing and compile-time checks. You must use CreateObject to create the NotesSession object. The following example shows the source code from an HTML file being run by Internet Explorer. The example was created using a simple text editor.


Initialize Session - Microsoft Internet Explorer
File Edit View Go Favorites Help
Back Forward Stop Refresh Home Search Favorites History Channels Fullscreen Mail Prin
Address C:\\vbscript\initialize.html
Initialize Session and Print User Name
Click Here
Done
initialize.html - Notepad
File Edit Search Help

```
<HTML>
<HEAD>
<TITLE>Initialize Session</TITLE>
<SCRIPT LANGUAGE="VBScript">
Sub Button1_OnClick
    dim session
    Set session = CreateObject("Lotus.NotesSession")
    Call session.Initialize
    MsgBox session.UserName,, "User name"
End Sub
</SCRIPT>
</HEAD>
<BODY>
<H3>Initialize Session and Print User Name</H3><HR>
<FORM><INPUT NAME="Button1" TYPE="BUTTON" VALUE="Click Here"></FORM>
</BODY>
</HTML>
```

Working through the object hierarchy

NotesSession is the only Domino object you can create directly. You obtain other Domino objects by calling properties and methods in NotesSession that return them and working down through the hierarchy. For example, various methods in NotesSession beginning with "Create" return Domino objects, as shown below:



Classes:

NotesName
NotesNewsletter
NotesOutline
NotesOutlineEntry
NotesRegistration
NotesReplication
NotesRichTextItem
NotesRichTextPara
NotesRichTextStyle
NotesRichTextTab
NotesSession
NotesView
NotesViewColumn
NotesViewEntry
NotesViewEntryColl
NotesViewNavigato
OE_CLASS

Members of 'NotesSession':

AddressBooks
CommonUserName
ConvertMime
CreateDateRange
CreateDateTime
CreateLog
CreateName
CreateNewsletter
CreateRegistration
CreateRichTextParagraphStyle
CreateRichTextStyle
CurrentAgent
CurrentDatabase
DocumentContext
EffectiveUserName
Evaluate
FreeTimeSearch

Class **NotesSession**
Member of Domino

The CurrentAgent, CurrentDatabase, and DocumentContext properties you see in the object browser are not implemented. Neither are the UnprocessedDocuments property, UnprocessedFTSearch method, and UnprocessedSearch methods in NotesDatabase. They compile without error but fail at run-time. At this time, the COM objects do not have the concept of a current environment.

See the **Common paths sidebar** for common paths to Domino objects. Note that the information in the sidebar shows typically used paths and is not comprehensive.

Opening and creating databases

GetDatabase in NotesSession opens a database if it exists. You supply the server name (an empty string means the local Domino or Notes data directory) and file name of the database. Use IsOpen to make sure you have a database because GetDatabase does not return an error if the database does not exist. These procedures are the same as in LotusScript.

This example opens NAMES.NSF on the server Snapper and displays the title of the database:

```
Private Sub GetNames_Click()
Dim session As New NotesSession
session.Initialize
Dim db As NotesDatabase
Set db = session.GetDatabase("Snapper", "names")
If db.IsOpen Then
    MsgBox db.Title, , "Title of names.nsf"
Else
    MsgBox "File does not exist", , "Title of names.nsf"
End If
End Sub
```

You can access a Domino or Notes directory and its constituent databases through GetDbDirectory in NotesSession. The following example is functionally the same as the preceding.

```
Private Sub GetNames2_Click()
Dim session As New NotesSession
Dim dir As NotesDbDirectory
Dim db As NotesDatabase
session.Initialize
Set dir = session.GetDbDirectory("Snapper")
Set db = dir.OpenDatabase("names")
If db.IsOpen Then
        MsgBox db.Title, , "Title of names.nsf"
Else
        MsgBox "File does not exist", , "Title of names.nsf"
End If
End Sub
```

The methods that create and open databases differ in containment between COM and LotusScript. In COM, you perform the create or open function with a NotesDbDirectory method; in LotusScript, you create a NotesDatabase object with New then use a NotesDatabase object to perform the function. You must rewrite this code when porting between COM and LotusScript. The specific differences are as follows:

COM: NotesDbDirectory	**LotusScript: NotesDatabase**
CreateDatabase	Create
OpenDatabase	Open (with parameters)
OpenDatabaseByReplicaID	OpenByReplicaID
OpenDatabaseIfModified	OpenIfModified
OpenMailDatabase	OpenMail

NotesDatabase.Open without parameters is the same in COM and LotusScript.

The navigation methods of NotesDbDirectory are the same in COM and LotusScript with two caveats. In early-binding COM code, do not use the constant DATABASE to avoid a naming conflict; use NOTES_DATABASE instead. In late-binding COM code, the constants are not available and you must specify their integer values: 1245 (REPLICA_CANDIDATE), 1246 (TEMPLATE_CANDIDATE), 1247 (NOTES_DATABASE), and 1248 (TEMPLATE).

The following Visual Basic code loops through the local Domino or Notes directory:

```
Private Sub LoopDbDirectory_Click()
Dim session As New NotesSession
Dim dir As NotesDbDirectory
Dim db As NotesDatabase
session.Initialize
Set dir = session.GetDbDirectory("")
Set db = dir.GetFirstDatabase(NOTES_DATABASE)
While Not (db Is Nothing)
        MsgBox db.Title, , db.FilePath
        Set db = dir.GetNextDatabase
Wend
End Sub
```

In VBScript, the coding is as follows:

```
<HTML>
<HEAD>
<TITLE>Initialize Session</TITLE>
```

```
<SCRIPT LANGUAGE="VBScript">
Sub Button1_OnClick
	dim s, dir, db
	Set s = CreateObject("Lotus.NotesSession")
	Call s.Initialize
	Set dir = s.GetDbDirectory("")
	Set db = dir.GetFirstDatabase(1247)
	While Not (db Is Nothing)
		Msgbox db.Title,, db.Filepath
		Set db = dir.GetNextDatabase
	Wend
End Sub
</SCRIPT>
</HEAD>
<BODY>
<H3>Loop through local Notes directory</H3><HR>
<FORM><INPUT NAME="Button1" TYPE="BUTTON" VALUE="Click
Here"></FORM>
</BODY>
</HTML>
```

Getting item values

In COM, you cannot use the shorthand that permits specification of an item name as a NotesDocument property. The following code, for example, cannot be ported from LotusScript to COM:

```
Dim doc as NotesDocument
...
Msgbox doc.Subject(0),, "Contents of Subject item"
```

To access item values, you must use AppendItemValue, GetItemValue, or ReplaceItemValue in NotesDocument, or Values in NotesItem.

The following example gets the values of the Subject and Categories items in each document in a local database. As in LotusScript, text, numeric, and date values are returned in an array. To access a single-value item, such as Subject, access the first element. To access a multi-value item, access all the elements or all the desired elements.

```
Private Sub GetSubjects_Click()
Dim session As New NotesSession
Dim db As NotesDatabase
Dim dc As NotesDocumentCollection
Dim doc As NotesDocument
session.Initialize
Set db = session.GetDatabase("", "document examples")
If Not db.IsOpen Then
	MsgBox "File does not exist", , "document examples"
	Exit Sub
End If
Set dc = db.AllDocuments
Set doc = dc.GetFirstDocument
While Not (doc Is Nothing)
	catmsg = "Categories:"
	REM Get all values in Categories item
	For Each cat In doc.GetItemValue("Categories")
		catmsg = catmsg & Chr(10) & " " & cat
	Next
	REM Get first (and only) value in Subject item
	MsgBox catmsg, , doc.GetItemValue("Subject")(0)
	Set doc = dc.GetNextDocument(doc)
Wend
End Sub
```

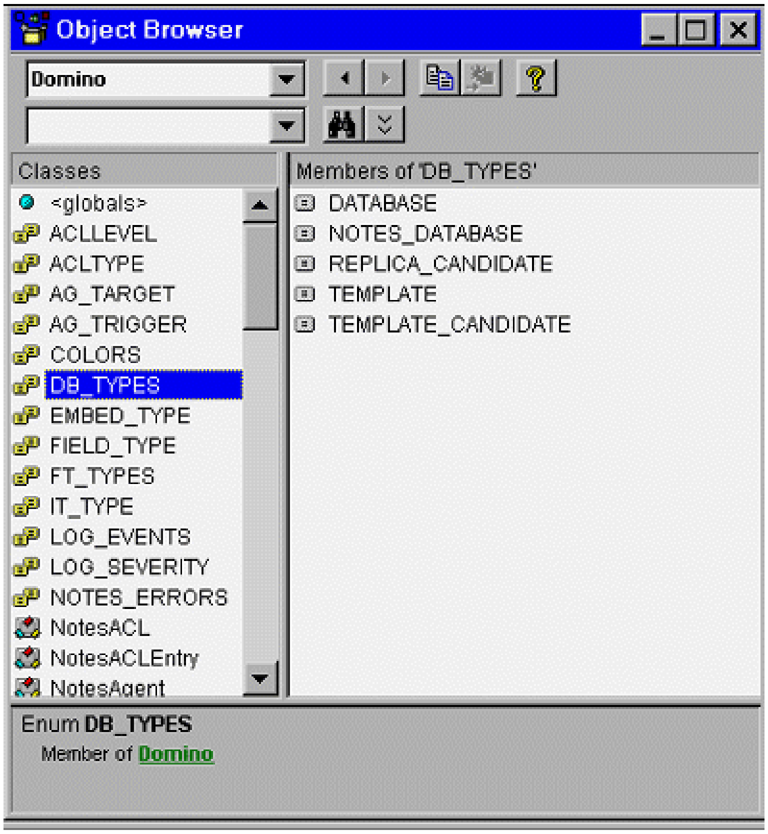
Notice that in COM the loop structure is "For Each" and "Next," while in LotusScript it is "ForAll" and "End ForAll." This syntax must be changed manually when porting.

The next example creates a document and uses AppendItemValue to add Subject and Categories items.

```
Private Sub CreateDocument_Click()
Dim session As New NotesSession
session.Initialize
Dim db As NotesDatabase
Set db = session.GetDatabase("", "document examples")
If Not db.IsOpen Then
    MsgBox "File does not exist", , "document examples"
    Exit Sub
End If
Dim doc As NotesDocument
Set doc = db.CreateDocument()
doc.AppendItemValue "Subject", "New Document"
Dim cats(2) As String
cats(0) = "Category One"
cats(1) = "Category Two"
doc.AppendItemValue "Categories", cats
doc.Save False, False
End Sub
```

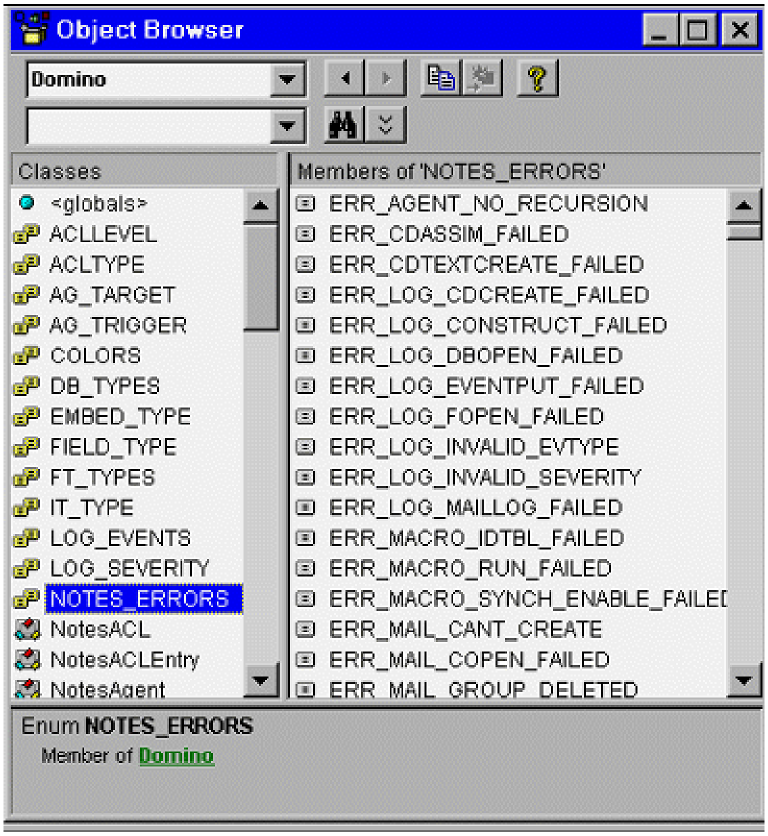
Working with constants and error codes

In Visual Basic, constants are available through the object browser. You can access them through Enum structures listed in all uppercase in the left-hand pane or through <globals> in the left-hand pane. Demonstrated below is the Enum structure for DB_TYPES. As noted previously, use NOTES_DATABASE, not DATABASE.



Enum **DB_TYPES**
Member of **Domino**

The error constants are listed under NOTES_ERRORS.



Classes

<globals>
ACLLEVEL
ACLTYPE
AG_TARGET
AG_TRIGGER
COLORS
DB_TYPES
EMBED_TYPE
FIELD_TYPE
FT_TYPES
IT_TYPE
LOG_EVENTS
LOG_SEVERITY
NOTES_ERRORS
NotesACL
NotesACLEntry
NotesAgent

Members of 'NOTES_ERRORS'

ERR_AGENT_NO_RECURSION
ERR_CDASSIM_FAILED
ERR_CDTEXTCREATE_FAILED
ERR_LOG_CDCREATE_FAILED
ERR_LOG_CONSTRUCT_FAILED
ERR_LOG_DBOPEN_FAILED
ERR_LOG_EVENTPUT_FAILED
ERR_LOG_FOPEN_FAILED
ERR_LOG_INVALID_EVTYPE
ERR_LOG_INVALID_SEVERITY
ERR_LOG_MAILLOG_FAILED
ERR_MACRO_IDTBL_FAILED
ERR_MACRO_RUN_FAILED
ERR_MACRO_SYNCH_ENABLE_FAILED
ERR_MAIL_CANT_CREATE
ERR_MAIL_COPEN_FAILED
ERR_MAIL_GROUP_DELETED

Enum **NOTES_ERRORS**
Member of **Domino**

The following code tests for the specific Domino error ERR_SYS_FILE_NOT_FOUND. If this is not the error, the code derives the Domino error code from the error number returned by the Err() intrinsic function. In COM, the error number is four bytes: the first two bytes are hexadecimal 8004; the last two bytes are the Domino error code. The sample code takes the last four hexadecimal digits of the error number and converts them to an integer.

```
Private Sub ErrorTest_Click()
On Error GoTo ErrorHandler
Dim session As New NotesSession
session.Initialize
Dim db As NotesDatabase
Set db = session.GetDatabase("", "fooey")
db.Open
MsgBox db.Title, , "Title of fooey"
Exit Sub
ErrorHandler:
If Err() = ERR_SYS_FILE_NOT_FOUND Then
        MsgBox "File not found", , "Domino Error"
Else
        MsgBox CInt("&H" & Right(Hex(Err()), 4)), , "Domino Error"
End If
End Sub
```

LotusScript does not currently support the Err intrinsic object. In COM, however, you could use Err.Number instead of Err(), for example:

```
If Err.Number = ERR_SYS_FILE_NOT_FOUND Then
```

In VBScript, you must use Err.Number because VBScript does not support the Err() intrinsic function.

In late binding languages such as VBScript, you do not have use of the named constants. You must specify the actual integer values. Use the following resources to find these values:

- In Domino 5.0.3 Designer Help, reference the topic "Accessing the Domino objects through COM" and "Handling run-time errors" in the contents under LotusScript/COM/OLE, or look up "COM" and "Error handling" in the index.
- Look up the error constants in the LSXBEERR.LSS file in the Domino or Notes program directory.
- Interrogate DOMOBJ.TLB in the Domino or Notes program directory with a tool such as Visual InterDev (VID).
- If you have Domino Designer or Visual Basic, display the value of a constant (for example, with MsgBox) or look it up in the object browser.

The following code is a VBScript implementation of the preceding example. The code defines a variable with the name and value of the error to be checked. The error value is constructed by appending the hexadecimal form of the Domino error code to hexadecimal 80040 (if the error code is three digits) or 8004 (if the error code is four digits), and converting the resulting 8-digit hexadecimal value to a long value. (Since the error code is a constant, the conditional logic is unnecessary. It is included for demonstration purposes.)

```
<HTML>
<HEAD>
<TITLE>Initialize Session</TITLE>
<SCRIPT LANGUAGE="VBScript">
Sub Button1_OnClick
    On Error Resume Next
    dim s, db, ERR_SYS_FILE_NOT_FOUND
    If Len(Hex(4003)) = 3 Then
        ERR_SYS_FILE_NOT_FOUND = Clng("&H80040" & Hex(4003))
    Else
        ERR_SYS_FILE_NOT_FOUND = Clng("&H8004" & Hex(4003))
    End If
    Set s = CreateObject("Lotus.NotesSession")
    Call s.Initialize
    Set db = s.GetDatabase("", "fooey")
    db.Open
    If Err.Number = 0 Then
        MsgBox db.Title,, "Title"
    Else
        If Err.Number = ERR_SYS_FILE_NOT_FOUND Then
            MsgBox "File not found",, _
            "Domino Error"
        Else
            MsgBox CInt("&H" & Right(Hex(Err.Number), 4)),, "Domino
            Error"
        End If
    End If
End Sub
</SCRIPT>
</HEAD>
<BODY>
<H3>Get fooey.nsf</H3><HR>
<FORM>
<INPUT NAME="Button1" TYPE="BUTTON" VALUE="Click Here">
</FORM>
</BODY>
</HTML>
```

Porting between Visual Basic and LotusScript

Many of the porting adjustments have been mentioned already. To reiterate:

- Only the back-end classes are available to COM. The NotesUI classes and NotesTimer are not.
- You must initialize a COM session. The NotesSession methods to do this are

Initialize("UserIDPasswordOptional") and InitializeUsingNotesUserName("UserName", "InternetPasswordOptional").

- You cannot create a Domino object directly except for NotesSession. You must work down through the Domino Objects hierarchy using methods and properties that return those objects.
- COM has no notion of a current environment. Not implemented are UnprocessedDocuments, UnprocessedFTSearch, and UnprocessedSearch in NotesDatabase, and CurrentAgent, CurrentDatabase, and DocumentContext in NotesSession.
- COM uses the NotesDbDirectory methods CreateDatabase, OpenDatabase, OpenDatabaseByReplicaID, OpenDatabaseIfModified, and OpenMailDatabase to create and open databases. In LotusScript, you create new NotesDatabase objects and use the NotesDatabase methods Create, Open, OpenByReplicaID, OpenIfModified, and OpenMail.
- In COM, you cannot treat an item name as a NotesDatabase property. Access items with AppendItemValue, GetItemValue, and ReplaceItemValue in NotesDatabase, and Values in NotesItem.
- Visual Basic uses For Each ... Next loops. LotusScript uses ForAll ... End ForAll loops.
- Constants port both ways as long the Enum structure is not specified in COM. For example, the constant NOTES_DATABASE (value of 1247) can be specified as DATA_TYPES.NOTES_DATABASE in COM but won't port back to LotusScript that way. Be sure to use NOTES_DATABASE and not DATABASE.
- LotusScript does not support the Err intrinsic object. Both LotusScript and Visual Basic support the Err() intrinsic function. (VBScript does not support the Err() intrinsic function.)

The COM classes contain additional properties and methods for purposes of utility. These properties and methods will probably be incorporated in LotusScript in future releases but are not planned for release 5.0.3. They include:

- HttpURL and NotesURL for returning the accessing URL in NotesAgent, NotesDatabase, NotesDocument, NotesForm, NotesSession, and NotesView.
- Parent for getting a handle to the containing object in NotesDateRange, NotesDateTime, NotesDbDirectory, NotesForm, NotesInternational, NotesLog, NotesName, NotesNewsletter, NotesOutlineEntry, NotesRichTextStyle, and NotesViewColumn. Also ParentDatabase in NotesOutline.
- In NotesSession: ServerName, URLDatabase, UserNameObject, CreateRegistration, Evaluate, and Resolve.
- In NotesView, CreateViewNavMaxLevel and GetColumn.
- In NotesViewNavigator: GetCurrent, GotoChild, GotoEntry, GotoFirst, GotoFirstDocument, GotoLast, GotoLastDocument, GotoNext, GotoNextCategory, GotoNextDocument, GotoNextSibling, GotoParent, GotoPos, GotoPrev, GotoPrevCategory, GotoPrevDocument, and GotoPrevSibling.

LotusScript does not support the Boolean data type. The Domino LotusScript classes type boolean-style parameters and return values as Integer or Variant. This is not a problem porting from LotusScript to Visual Basic. However, data typed as Boolean in Visual Basic cannot be ported without modification to LotusScript.

Other more minor differences exist between the COM and LotusScript classes, and between the Visual Basic and LotusScript languages. These are enumerated in the updated Domino 5 Designer Help for release 5.0.3.

In conclusion

You can now access the back-end Domino objects from any software that supports COM. The only requirement is installation of Domino server, Domino Designer, or Notes client release 5.0.2b (or later) on the machine using COM. The Domino or Notes software need not be running.

This article provided examples in the early-binding Visual Basic and late-binding VBScript. For examples in C++, Visual Basic, VBA in Word and Excel, and VBScript in ASP, plus documentation, you can download the Lotus Domino and Notes Toolkit for COM from the **Lotus Developer Network Technology and Toolkits Zone**.

ABOUT THE AUTHOR

Robert Perron is a Documentation Architect with Lotus in Westford, Massachusetts. He has developed documentation for Lotus Notes and Domino for over five years with a primary concentration on programmability. He developed the documentation for the LotusScript and Java Notes classes and coauthored the book *60 Minute Guide to LotusScript 3 - Programming for Notes 4*.

What do you think about this article?

Notes.net

Iris Today

Home | Download | Iris Today | Iris Cafe | All About Domino | Iris Sandbox | Doc Library

[back to Common ground: COM access to Domino objects]

Common paths to Domino objects

The following table shows the common paths to Domino objects. The table is not comprehensive.

Domino object	Common path	
NotesACL	NotesDatabase	ACL
NotesACLEntry	NotesACL	GetEntry
		GetFirstEntry
		GetNextEntry
NotesAgent	NotesDatabase	Agents
		GetAgent
NotesDatabase	NotesDatabase	CreateFromTemplate
		CreateReplica
	NotesDbDirectory	CreateDatabase
		GetFirstDatabase
		GetNextDatabase
		OpenDatabase
		OpenDatabaseByReplicaID
		OpenDatabaseIfModified
		OpenMailDatabase
	NotesSession	AddressBooks
		GetDatabase
		URLDatabase
NotesDateRange	NotesSession	CreateDateRange
NotesDateTime	NotesDateRange	EndDateTime
		StartDateTime
	NotesSession	CreateDateTime
NotesDbDirectory	NotesSession	GetDbDirectory
NotesDocument	NotesDatabase	CreateDocument
		FTDomainSearch

		GetDocumentByID
		GetDocumentByUNID
		GetDocumentByURL
		GetProfileDocument
	NotesDocument	CopyToDatabase
		CreateReplyMessage
	NotesDocumentCollection	AddDocument
		GetDocument
		GetFirstDocument
		GetLastDocument, and so on
	NotesView	GetDocumentByKey
		GetFirstDocument
		GetLastDocument, and so on
	NotesViewNavigator	GetFirstDocument
		GetLastDocument, and so on
NotesDocumentCollection	NotesDatabase	AllEntries
		FTSearch
		GetProfileDocCollection
		UnprocessedDocuments
		UnprocessedFTSearch
		UnprocessedSearch
	NotesDocument	Responses
	NotesView	GetAllDocumentsByKey
NotesEmbeddedObject	NotesDocument	EmbeddedObjects
		GetAttachment
NotesForm	NotesDatabase	Forms
		GetForm
NotesInternational	NotesSession	International
NotesItem	NotesDocument	AppendItemValue
		CopyItem
		GetFirstItem
	NotesItem	CopyItemToDocument
NotesLog	NotesSession	CreateLog
NotesMIMEEntity	NotesItem	GetMIMEEntity

NotesName	NotesSession	CreateName
		UserNameObject
NotesNewsletter	NotesSession	CreateNewsletter
NotesOutline	NotesDatabase	CreateOutline
		GetOutline
NotesOutlineEntry	NotesOutline	CreateEntry
		CreateEntryFrom
		GetFirst, GetLast, and so on
NotesRegistration	NotesSession	CreateRegistration
NotesReplication	NotesDatabase	ReplicationInfo
NotesRichTextItem	NotesDocument	CreateRichTextItem
NotesRichTextParagraphStyle	NotesRichTextItem	AppendRichTextParagraphStyle
	NotesSession	CreateRichTextParagraphStyle
NotesRichTextStyle	NotesSession	CreateRichTextStyle
NotesRichTextTab	NotesRichTextParagraphStyle	Tabs
NotesView	NotesDatabase	GetView
		Views
NotesViewColumn	NotesView	Columns
NotesViewEntry	NotesView	GetEntryByKey
	NotesViewEntryCollection	GetEntry
		GetFirstEntry
		GetLastEntry
		GetNextEntry
		GetNthEntry
		GetPrevEntry
	NotesViewNavigator	GetChild
		GetCurrent
		GetEntry
		GetFirst, and do on
		GotoChild
		GotoEntry
		GotoFirst, and so on
NotesViewEntryCollection	NotesView	AllEntries
	NotesViewNavigator	GetAllEntriesByKey
NotesViewNavigator	NotesView	CreateViewNav

		CreateViewNavFrom, and so on

Register Here!

About this Site | **Feedback**
Lotus Home | **IBM Home** | **Iris Home**
Copyright 1999 Iris Associates Inc.