Lotus. software

IBM

Domino Objects for LotusScript/COM/OLE, Part 1

Supports Visual Basic, VBA, VBS, Delphi, and other COM- and OLE-compliant languages
IBM Lotus Notes® and Domino™ 6

Front-End Classes

Code Sample

```
Dim ws As New NotesUIWorkspace
Dim uidoc As NotesUIDocument
Dim lastname As String
Call ws.OpenDatabase("snapper", "names", _
  "People", "A")
Set uidoc = ws.EditDocument(False)
lastname = uidoc.FieldGetText("LastName")
```

NotesUIWorkspace
Not supported by COM

Properties
datevariable = CurrentCalendarDateTime
notesuidatabase = CurrentDatabase
notesuidocument = CurrentDocument
notesuiview = CurrentView

Methods
notesuiworkspace = New NotesUIWorkspace
AddDatabase(server, filename)
CheckAlarms
notesuidocument = ComposeDocument([server], [file], [form], [windowwidth], [windowheight], [returnnotesuidocument])
flag = DialogBox(form, [autohorzfit], [autovertfit], [nocancel], [nonewfields], [nofieldupdate], [readonly], [title], [notesdocument], [sizetotable], [nookcancel], [okcancelatbottom])
notesuidocument = EditDocument([editmode], [notesdocument], [notesdocumentreadonly], [documentanchor], [returnnotesuidocument], [newinstance])
notesuidocument = EditProfile(profilename, [uniqueKey])
EnableAlarms(enable)
variant = Folder([foldername], move)
notesuidatabase = GetCurrentDatabase
variant = GetListOfTunes
OpenDatabase(server, file, view, key, newinstance, temp)
stringarray = OpenFileDialog(multipleselection, [title], [filters], [initialdirectory], [initialfile])
OpenFrameSet(frameset)
OpenPage(pagename)
notesdocumentcollection = PickListCollection(type, [multipleselection], server, databasefilename, viewname, title, prompt, [singlecategory])
stringarray = PickListStrings(type, [multipleselection])
stringarray = PickListStrings(type, [multipleselection], server, databasefilename, viewname, title, prompt, column, [categoryname])
PlayTune(tunename)
variant = Prompt(type, title, prompt, [default], [values])
RefreshParentNote
ReloadWindow
stringarray = SaveFileDialog(directoriesonly, [title], [filters], [initialdirectory], [initialfile])
SetCurrentLocation([location])
SetTargetFrame(frame)
URLOpen(url, reload, urllist, charset, webusername, webpassword, proxywebusername, proxywebpassword, notesretriever)
UseLSX(lsxlibraryname)
ViewRebuild([viewrebuild], [loadpreferences])
ViewRefresh

Button

Events
Click(source)
ObjectExecute(source, data)

Field

Events
Entering(source)
Exiting(source)
OnBlur(source)
OnChange(source)
OnFocus(source)

Navigator

Events
Click(source)

NotesUIDocument

Properties
bool = AutoReload
AutoReload = bool
string = CurrentField
bool = DialogBoxCanceled
notesdocument = Document
bool = EditMode
EditMode = bool
bool = FieldHelp
FieldHelp = bool
bool = HiddenChars
HiddenChars = bool
bool = HorzScrollBar
HorzScrollBar = bool
bool = InPreviewPane
bool = IsNewDoc
bool = PreviewDocLink
PreviewDocLink = bool
bool = PreviewParentDoc
PreviewParentDoc = bool
bool = Ruler
Ruler = bool
string = WindowTitle

Methods
Categorize([categoryname])
Clear
Close([immediate])
CollapseAllSections
Copy
handlev = CreateObject([name], [type], [filepath])
Cut
DeleteDocument
DeselectAll
ExpandAllSections
FieldAppendText(fieldname, text)
FieldClear([fieldname])
bool = FieldContains(fieldname, textvalue)
textvalue = FieldGetText([fieldname])
FieldSetText(fieldname, textvalue)
FindFreeTimeDialog(reqpeopleitems, optpeopleitems, reqroomsitems, optroomsitems, reqresourcesitems, optresourcesitems, removedpeopleitems, startdateitem, enddateitem)
bool = FindFreeTimeDialogEx(reqpeopleitems, optpeopleitems, reqroomsitems, optroomsitems, reqresourcesitems, optresourcesitems, removedpeopleitems, startdateitem, enddateitem)
FindString(text, [wholeword], [backwards], [casesensitive], [wildcards], [accentsensitive], [searchdocument], [findnext], [wraparound], [ignorediacritics])
Forward
handlev = GetObject(name)
notesuischeduler = GetSchedulerObject(name)
text = GetSelectedText([fieldname])
GotoBottom
GotoField(fieldname)
GotoNextField
GotoPrevField
GotoTop
Import([filter, filename])
InsertText(textvalue)
NavBarSetText(urltext, [windowtitle])
NavBarSpinnerStart
NavBarSpinnerStop
Paste
Print
Print([numcopies], [frompage], [topage], [draft])
Refresh([includerichtextitems])
RefreshHideFormulas
Reload
Save
SaveNewVersion
SelectAll
Send
SpellCheck

Events
OnHelp(source)
OnLoad(source)
OnSubmit(source, continue)
OnUnload(source, continue)
PostModeChange(source)
PostOpen(source)
PostRecalc(source)
PostSave(source)
PostSend(source)
QueryClose(source, continue)
QueryModeChange(source, continue)
QueryOpen(source, mode, isnewdoc, continue)
QueryRecalc(source, continue)
QuerySave(source, continue)
QuerySend(source, continue)

NotesUIDatabase

Properties
notesdatabase = Database
notesdocumentcollection = Documents

Methods
Close
OpenNavigator(navigatorname, [fullwindow])
OpenView(viewname, [key], [newinstance], [replaceview])

Events
PostDocumentDelete(source)
PostDragDrop(source, entryname, continue)
PostDropToArchive(source, entryname, continue)
PostOpen(source)
QueryClose(source, continue)
QueryDocumentDelete(source, continue)
QueryDocumentUndelete(source, continue)
QueryDragDrop(source, entryname, continue)
QueryDropToArchive(source, entryname, continue)

NotesUIScheduler

Properties
bool = ColorLegend
ColorLegend = bool
bool = DisplayAlternateNames
DisplayAlternateNames = bool
long = DisplayHoursPerDay
DisplayHoursPerDay = long
bool = DisplayMeetingSuggestions
DisplayMeetingSuggestions = bool
bool = DisplayParticipantStatus
DisplayParticipantStatus = bool
bool = DisplayPeople
DisplayPeople = bool
bool = DisplayResources
DisplayResources = bool
bool = DisplayRooms
DisplayRooms = bool
bool = DisplayTwisties
DisplayTwisties = bool
notesdaterange = Interval
Interval = notesdaterange
bool = MeetingIndicator
MeetingIndicator = bool
notesdatetime = ScheduleGridStart
ScheduleGridStart = notesdatetime
string = SchedulerName
string = TimeZone
TimeZone = string

Methods
AddParticipant(name, role, status, [type], [alternatename], [inetName], [storagetype], [flags])
participants = GetParticipants(participanttype)
GetScheduleData([flag])
RemoveParticipants([participants])

Events
OnIntervalChange(source)

NotesUIView

Properties
datevariant = CalendarDateTime
datevariant = CalendarDateTimeEnd
variant = CaretCategory
string = CaretNoteID
notesdocumentcollection = Documents
notesview = View
string = ViewAlias
string = ViewInheritedFrom
string = ViewName

Methods
Close
DeselectAll
Print
Print(numcopies, [frompage], [topage], [draft], [pageseparator], [formoverride], [printview], [daterangebegin], [daterangeend])
SelectDocument(notesdocument)

Events
InViewEdit(source, requesttype, colprogname, columnvalue, continue)
PostDragDrop(source)
PostEntryResize(source)
PostOpen(source)
PostPaste(source)
QueryAddToFolder(source, target, continue)
QueryClose(source, continue)
QueryDragDrop(source, continue)
QueryEntryResize(source, continue)
QueryOpen(source, continue)
QueryOpenDocument(source, continue)
QueryPaste(source, continue)
QueryRecalc(source, continue)
RegionDoubleClick(source)

Back-End Classes

Code Sample

```
Dim session As New NotesSession
Dim db As NotesDatabase
Dim dc As NotesDocumentCollection
Dim doc As NotesDocument
Dim lastname As String
Set db = session.GetDatabase("snapper","names")
Set dc = db.AllDocuments
Set doc = dc.GetFirstDocument
lastname = doc.LastName(0)
```

NotesSession

Properties
notesdatabasearray = AddressBooks
string = CommonUserName
bool = ConvertMIME
ConvertMIME = bool
notesdatabase = CurrentDatabase
notesdocument = DocumentContext
string = EffectiveUserName
string = HttpURL
bool = IsOnServer
integer = LastExitStatus
datevariant = LastRun
long = NotesBuildVersion
string = NotesURL
string = NotesVersion
string = OrgDirectoryPath
string = Platform
notesdocument = SavedData
string = ServerName
notesdatabase = URLDatabase
notesnamearray = UserGroupNameList
string = UserName
notesnamearray = UserNameList
notesname = UserNameObject

Methods
notessession = New()
notesdaterange = CreateDateRange
notesdatetime = CreateDateTime(datetime)
notesnewsletter = CreateNewsletter (notesdocumentcollection)
notesrichtextparagraphstyle = CreateRichTextParagraphStyle
notesrichtextstyle = CreateRichTextStyle
variant = Evaluate(formula, doc)
notesdaterange = FreeTimeSearch(window, duration, names, [firstfit])
notesdatabase = GetDatabase(server, dbfile, [createonfail])
notesdbdirectory = GetDbDirectory(servername)
value = GetEnvironmentString(name, [system])
value = GetEnvironmentValue(name, [system])
notesdocument = GetUserPolicySettings(server, name, [type], [explicitpolicy], [reserved])
hashedpassword = HashPassword(password)
Initialize([password])
InitializeUsingNotesUserName(name, [password])
unknown = Resolve(url)
consolereturn = SendConsoleCommand (servername, consolecommand)
SetEnvironmentVar(name, value, [issystemvar])
UpdateProcessedDoc(notesdocument)
flag = VerifyPassword(password, hashedpassword)

NotesDbDirectory

Properties
string = Name
notessession = Parent

Methods
notesdbdirectory = New(servername)
notesdatabase = CreateDatabase(dbfile, [open])
notesdatabase = GetFirstDatabase(filetype)
notesdatabase = GetNextDatabase
notesdatabase = OpenDatabase(dbfile, [failover])
notesdatabase = OpenDatabaseByReplicaID(rid)
notesdatabase = OpenDatabaseIfModified(dbfile, notesdatetime)
notesdatabase = OpenMailDatabase

NotesDatabase

Properties
stringarray = ACLActivityLog
notesdocumentcollection = AllDocuments
string = Categories
Categories = string
datevariant = Created
integer = CurrentAccessLevel
bool = DelayUpdates
DelayUpdates = bool
string = DesignTemplateName
integer = FileFormat
string = FileName
string = FilePath
bool = FolderReferencesEnabled
FolderReferencesEnabled = bool
notesformarray = Forms
integer = FTIndexFrequency
FTIndexFrequency = integer
string = HttpURL
bool = IsClusterReplication
IsClusterReplication = bool
bool = IsConfigurationDirectory
bool = IsCurrentAccessPublicReader
bool = IsCurrentAccessPublicWriter
bool = IsDesignLockingEnabled
IsDesignLockingEnabled = bool
bool = IsDirectoryCatalog
bool = IsDocumentLockingEnabled
IsDocumentLockingEnabled = bool
bool = IsFTIndexed
bool = IsInMultiDbIndexing
IsInMultiDbIndexing = bool
bool = IsInService
IsInService = bool
bool = IsLink
bool = IsMultiDbSearch
bool = IsOpen
bool = IsPendingDelete
bool = IsPrivateAddressBook
bool = IsPublicAddressBook
datevariant = LastFixup
datevariant = LastFTIndexed
datevariant = LastModified
double = LimitRevisions
LimitRevisions = double
double = LimitUpdatedBy
LimitUpdatedBy = double
bool = ListInDbCatalog
ListInDbCatalog = bool
stringarray = Managers
double = MaxSize
string = NotesURL
notessession = Parent
double = PercentUsed
string = ReplicaID
string = Server
double = Size
long = SizeQuota
SizeQuota = long
integer = SizeWarning
SizeWarning = integer
string = TemplateName
string = Title
Title = string
integer = Type
integer = UndeleteExpireTime
UndeleteExpireTime = integer
notesdocumentcollection = UnprocessedDocuments
notesviewarray = Views

Methods
notesdatabase = New(server, dbfile)
sizedelta = Compact
sizedelta = CompactWithOptions(options)
sizedelta = CompactWithOptions(options, [spacethreshold])
Create(server, dbfile, openflag, [maxsize])
notesdatabasenew = CreateCopy(newserver, newdbfile, [maxsize])
notesdocument = CreateDocument
notesdatabasenew = CreateFromTemplate (newserver, newdbfile, inheritflag, [maxsize])
CreateFTIndex(options, recreate)
notesdatabasereplica = CreateReplica(newserver, newdbfile)
notesview = CreateView([viewname], [viewselectionformula], [templateview], [prohibitdsignrefreshmodifications])
EnableFolder(foldername)
Fixup([options])
notesdocument = FTDomainSearch(query, maxdocs, [sortoptions], [otheroptions], [start], [count], [entryform])
notesdocumentcollection = FTSearch(query, maxdocs, [sortoptions], [otheroptions])
notesdocumentcollection = FTSearchRange(query, maxdocs, [sortoptions], [otheroptions], [start])
notesdocument = GetDocumentByID(noteid)
notesdocument = GetDocumentByUNID(unid)
notesdocument = GetDocumentByURL(url, [reload], [urllist], [charset], [webusername], [webpassword], [proxywebusername], [proxywebpassword], [returnimmediately])
notesform = GetForm(name)
flag = GetOption(optionname)
notesdocumentcollection = GetProfileDocCollection (profilename)
notesdocument = GetProfileDocument(profilename, [uniquekey])
notesview = GetView(viewname)
GrantAccess(name, level)
MarkForDelete
Open
flag = Open(server, dbfile)
flag = OpenByReplicaID(server, replicaid)
flag = OpenIfModified(server, dbfile, notesdatetime)
OpenMail
flag = OpenURLDb
flag = OpenWithFailover(server, dbfile)
level = QueryAccess(name)
roles = QueryAccessRoles(name)
Remove
RemoveFTIndex
flag = Replicate(servername)
RevokeAccess(name)
notesdocumentcollection = Search(formula, notesdatetime, maxdocs)
SetOption(optionname, flag)
Sign([documenttype], [existingsigsonly], [namestr], [namestrisnoteid])
notesdocumentcollection = UnprocessedFTSearch (query, maxdocs, [sortoptions], [otheroptions])
notesdocumentcollection = UnprocessedFTSearchRange(query, maxdocs, [sortoptions], [otheroptions], [start])
notesdocumentcollection = UnprocessedSearch (formula, notesdatetime, maxdocs)
UpdateFTIndex(createflag)

NotesForm

Properties
stringarray = Aliases
stringarray = Fields
stringarray = FormUsers
FormUsers = stringarray
string = HttpURL
bool = IsSubForm
stringarray = LockHolders
string = Name
string = NotesURL
notesdatabase = Parent
bool = ProtectReaders
ProtectReaders = bool
bool = ProtectUsers
ProtectUsers = bool
stringarray = Readers
Readers = stringarray

Methods
integer = GetFieldType(name)
flag = Lock([name], [provisionalok])
flag = LockProvisional([name])
Remove
Unlock

NotesDocumentCollection

Properties
long = Count
bool = IsSorted
notesdatabase = Parent
string = Query

Methods
AddDocument(document, [checkdups])
DeleteDocument(document)
FTSearch(query, maxdocs)
notesDocument = GetDocument(document)
notesDocument = GetFirstDocument
notesDocument = GetLastDocument
notesDocument = GetNextDocument(document)
notesDocument = GetNthDocument(n)
notesDocument = GetPrevDocument(document)
PutAllInFolder(foldername, [createonfail])
RemoveAll(force)
RemoveAllFromFolder(foldername)
StampAll(itemname, value)
UpdateAll

NotesRichTextParagraphStyle

Properties
integer = Alignment
Alignment = integer
long = FirstLineLeftMargin
FirstLineLeftMargin = long
integer = InterLineSpacing
InterLineSpacing = integer
long = LeftMargin
LeftMargin = long
integer = Pagination
Pagination = integer
long = RightMargin
RightMargin = long
integer = SpacingAbove
SpacingAbove = integer
integer = SpacingBelow
SpacingBelow = integer
notesrichtexttabarray = Tabs

Methods
ClearAllTabs
SetTab(position, type)
SetTabs(number, startposition, interval, type)

NotesRichTextTab

Properties
long = Position
integer = Type

Methods
Clear

NotesView

Properties
stringarray = Aliases
Aliases = stringarray
bool = AutoUpdate
AutoUpdate = bool
integer = BackgroundColor
BackgroundColor = integer
integer = ColumnCount
variant = ColumnNames
datevariant = Created
integer = EntryCount
integer = HeaderCount
string = HttpURL
bool = IsCalendar
bool = IsCategorized
bool = IsConflict
bool = IsDefaultView
IsDefaultView = bool
bool = IsFolder
bool = IsHierarchical
bool = IsModified
bool = IsPrivate
bool = IsProhibitDesignRefresh
IsProhibitDesignRefresh = bool
datevariant = LastModified
stringarray = LockHolders
string = Name
Name = string
string = NotesURL
notesdatabase = Parent
bool = ProtectReaders
ProtectReaders = bool
stringarray = Readers
Readers = stringarray
integer = RowLines
string = SelectionFormula
SelectionFormula = string
integer = Spacing
Spacing = integer
integer = TopLevelEntryCount
string = UniversalID
string = ViewInheritedName

Methods
Clear
numdocs = FTSearch(query, maxdocs)
notesdocumentcollection = GetAllDocumentsByKey (keyarray, [exactmatch])
notesdocument = GetChild(document)
notesdocument = GetDocumentByKey(keyarray, [exactmatch])
notesdocument = GetFirstDocument
notesdocument = GetLastDocument
notesdocument = GetNextDocument(document)
notesdocument = GetNextSibling(document)
notesdocument = GetNthDocument(index)
notesdocument = GetParentDocument(document)
notesdocument = GetPrevDocument(document)
notesdocument = GetPrevSibling(document)
flag = Lock([name], [provisionalok])
flag = LockProvisional([name])
Refresh
Remove
RemoveColumn([columnnameorindex])
return = SetAliases(alias)
Unlock

NotesNewsletter

Properties
bool = DoScore
DoScore = bool
bool = DoSubject
DoSubject = bool
notessession = Parent
string = SubjectItemName
SubjectItemName = string

Methods
notesnewsletter = New(notesdocumentcollection)
notesdocument = FormatDocument (notesdatabase, n)
notesdocument = FormatMsgWithDoclinks (notesdatabase)

NotesDocument

Properties
stringarray = Authors
variantarray = ColumnValues
datevariant = Created
notesembeddedobjectarray = EmbeddedObjects
stringarray = EncryptionKeys
EncryptionKeys = stringarray
EncryptionKeys = string
bool = EncryptOnSend
EncryptOnSend = bool
stringarray = FolderReferences
integer = FTSearchScore
bool = HasEmbedded
string = HttpURL
bool = IsDeleted
bool = IsEncrypted
bool = IsNewNote
bool = IsProfile
bool = IsResponse
bool = IsSigned
bool = IsUIDocOpen
bool = IsValid
notesitemarray = Items
string = Key
datevariant = LastAccessed
datevariant = LastModified
stringarray = LockHolders
string = NameOfProfile
string = NoteID
string = NotesURL
notesdatabase = ParentDatabase
string = ParentDocumentUNID
notesview = ParentView
notesdocumentcollection = Responses
bool = SaveMessageOnSend
SaveMessageOnSend = bool
bool = SentByAgent
string = Signer
bool = SignOnSend
SignOnSend = bool
long = Size
string = UniversalID
UniversalID = string
string = Verifier

Methods
notesdocument = New(notesdatabase)
notesitem = AppendItemValue(itemname, value)
flag = CloseMIMEEntities([savechanges], [entityitemname])
flag = ComputeWithForm(dodatatypes, raiseerror)
CopyAllItems(notesdocument, [replace])
notesitem = CopyItem(item, newname)
CopyItem(item, newname)
newnotesdocument = CopyToDatabase (notesdatabase)
notesmimeentity = CreateMIMEEntity([itemname])
replynotesdocument = CreateReplyMessage(all)
notesrichtextitem = CreateRichTextItem(name)
Encrypt
notesembeddedobject = GetAttachment(filename)
notesitem = GetFirstItem(name)
valuearray = GetItemValue(itemname)
notesdatetimearray = GetItemValueDateTimeArray (itemname)
notesmimeentity = GetMIMEEntity([itemname])
stringarray = GetReceivedItemText
flag = HasItem(itemname)
flag = Lock([name], [provisionalok])
flag = LockProvisional([name])
MakeResponse(document)
PutInFolder(foldername, [createonfail])
flag = Remove(force)
RemoveFromFolder(foldername)
RemoveItem(itemname)
flag = RemovePermanently(force)
flag = RenderToRTItem(notesrichtextitem)
notesitem = ReplaceItemValue(itemname, value)
flag = Save(force, createresponse, [markread])
Send(attachForm, [recipients])
Sign
Unlock

NotesItem

Properties
notesdatetime = DateTimeValue
DateTimeValue = notesdatetime
bool = IsAuthors
IsAuthors = bool
bool = IsEncrypted
IsEncrypted = bool
bool = IsNames
IsNames = bool
bool = IsProtected
IsProtected = bool
bool = IsReaders
IsReaders = bool
bool = IsSigned
IsSigned = bool
bool = IsSummary
IsSummary = bool
datevariant = LastModified
itemname = Name
notesdocument = Parent
bool = SaveToDisk
SaveToDisk = bool
string = Text
integer = Type
integer = ValueLength
valuearray = Values
Values = valuearray

Methods
notesitem = New(notesdocument, name, value, [specialtype])
abbreviation = Abstract(maxabstract, dropvowels, usedictionary)
AppendToTextList(newvalue)
flag = Contains(value)
notesitem = CopyItemToDocument(document, newname)
notesmimeentity = GetMIMEEntity
notesdatetimearray = GetValueDateTimeArray
Remove

NotesRichTextItem

Properties
notesembeddedobjectarray = EmbeddedObjects

Methods
notesrichtextitem = New(notesdocument, name)
AddNewLine(n, [forceparagraph])
AddPageBreak([rtpstyle])
AddTab(n)
AppendDocLink(linkto, comment, [hotspottext])
AppendParagraphStyle(rtpstyle)
AppendRTItem(notesrichtextitem)
AppendStyle(notesrichtextstyle)
AppendTable(rows, columns, [labels], [leftmargin], [rtpstylearray])
AppendText(text)
BeginInsert(element, [after])
BeginSection(title, [titlestyle], [barcolor], [expand])
Compact
notesrichtextnavigator = CreateNavigator
notesrichtextrange = CreateRange
notesembeddedobject = EmbedObject(type, class, source, [name])
EndInsert
EndSection
notesembeddedobject = GetEmbeddedObject(name)
plaintext = GetFormattedText(tabstrip, linelength)
notesfont = GetNotesFont(facename, addonfail)
plaintext = GetUnformattedText
Update

NotesRichTextNavigator

Methods
notesrichtextnavigator = Clone
flag = FindFirstElement(type)
flag = FindFirstString(target, [options])
flag = FindLastElement(type)
flag = FindNextElement([type], [occurrence])
flag = FindNextString(target, [options])
flag = FindNthElement(type, [occurrence])
element = GetElement
element = GetFirstElement(type)
element = GetLastElement(type)
element = GetNextElement([type], [occurrence])
element = GetNthElement(type, [occurrence])
SetCharOffset(offset)
SetPosition(element)
SetPositionAtEnd(element)

NotesRichTextTable

Properties
notescolorobject = AlternateColor
notescolorobject = Color
integer = ColumnCount
bool = RightToLeft
RightToLeft = bool
integer = RowCount
stringarray = RowLabels
RowLabels = stringarray
integer = Style
Style = integer

Methods
AddRow([count], [targetrow])
Remove
RemoveRow([count], [targetrow])
SetAlternateColor(color)
SetColor(color)

NotesDateRange

Properties
notesdatetime = EndDateTime
EndDateTime = notesdatetime
notessession = Parent
notesdatetime = StartDateTime
StartDateTime = notesdatetime
text = Text
Text = text

NotesDateTime

Properties
string = DateOnly
string = GMTTime
bool = IsDST
bool = IsValidDate
string = LocalTime
LocalTime = string
datevariant = LSGMTTime
datevariant = LSLocalTime
LSLocalTime = datevariant
notessession = Parent
string = TimeOnly
integer = TimeZone
string = ZoneTime

Methods
notesdatetime = New NotesDateTime(datetime)
AdjustDay(n, [preservelocaltime])
AdjustHour(n, [preservelocaltime])
AdjustMinute(n, [preservelocaltime])
AdjustMonth(n, [preservelocaltime])
AdjustSecond(n, [preservelocaltime])
AdjustYear(n, [preservelocaltime])
ConvertToZone(newzone, dst)
SetAnyDate
SetAnyTime
SetNow
difference = TimeDifference(notesdatetime)
difference = TimeDifferenceDouble(notesdatetime)

NotesEmbeddedObject

Properties
string = Class
long = FileSize
bool = FitBelowFields
FitBelowFields = bool
bool = FitToWindow
FitToWindow = bool
string = Name
variant = Object
notesrichtextitem = Parent
bool = RunReadOnly
RunReadOnly = bool
string = Source
integer = Type
stringarray = Verbs

Methods
variant = Activate(show)
variant = DoVerb(verb)
ExtractFile(path)
Remove

NotesRichTextDocLink

Properties
string = DbReplicaID
DbReplicaID = string
string = DisplayComment
DisplayComment = string
string = DocUNID
DocUNID = string
string = HotSpotText
HotSpotText = string
notesrichtextstyle = HotSpotTextStyle
string = ServerHint
ServerHint = string
string = ViewUNID
ViewUNID = string

Methods
Remove
SetHotSpotTextStyle(notesrichtextstyle)

NotesRichTextSection

Properties
notescolorobject = BarColor
bool = IsExpanded
IsExpanded = bool
string = Title
Title = string
notesrichtextstyle = TitleStyle

Methods
Remove
SetBarColor(notescolorobject)
SetTitleStyle(notesrichtextstyle)

NotesMIMEEntity

Properties
string = BoundaryEnd
string = BoundaryStart
string = Charset
string = ContentAsText
string = ContentSubType
string = ContentType
integer = Encoding
headerobjectarray = HeaderObjects
string = Headers
string = Preamble
Preamble = string

Methods
notesmimeentity = CreateChildEntity([nextsibling])
notesmimeheader = CreateHeader(headername)
notesmimeentity = CreateParentEntity
DecodeContent
EncodeContent(encoding)
GetContentAsBytes(stream, [decoded])
GetContentAsText(stream, [decoded])
GetEntityAsText(stream, [headerfilters], [inclusive])
notesmimeentity = GetFirstChildEntity
notesmimeentity = GetNextEntity([search])
notesmimeentity = GetNextSibling
notesmimeheader = GetNthHeader(headername, [instance])
notesmimeentity = GetParentEntity
notesmimeentity = GetPrevEntity([search])
notesmimeentity = GetPrevSibling
headers = GetSomeHeaders([headerfilters], [inclusive])
Remove
SetContentFromBytes(stream, contenttype, encoding)
SetContentFromText(stream, contenttype, encoding)

NotesMIMEHeader

Properties
string = HeaderName

Methods
flag = AddValText(valuetext, [charset])
headervalue = GetHeaderVal([folded], [decoded])
headerandparams = GetHeaderValAndParams ([folded], [decoded])
parametervalue = GetParamVal(paramname, [folded])
Remove
flag = SetHeaderVal(headervalue)
flag = SetHeaderValAndParams(headerparamvalue)
flag = SetParamVal(parametername, parametervalue)

NotesRichTextRange

Properties
notesrichtextnavigator = Navigator
notesrichtextstyle = Style
string = TextParagraph
string = TextRun
long = Type

Methods
notesrichtextrange = Clone
bool = FindAndReplace(target, replacement, [options])
Remove
Reset([begin], [end])
SetBegin(element)
SetEnd(element)
SetStyle(style)

NotesRichTextStyle

Properties
integer = Bold
Bold = integer
long = Bold
Bold = long
integer = Effects
Effects = integer
integer = FontSize
FontSize = integer
bool = IsDefault
integer = Italic
Italic = integer
long = Italic
Italic = long
integer = NotesColor
NotesColor = integer
integer = NotesFont
NotesFont = integer
notessession = Parent
integer = PassThruHTML
PassThruHTML = integer
long = PassThruHTML
PassThruHTML = long
integer = StrikeThrough
StrikeThrough = integer
long = StrikeThrough
StrikeThrough = long
integer = Underline
Underline = integer
long = Underline
Underline = long

Not supported by COM

COM only

www.lotus.com/ldd/doc

All trademarks are the property of their respective owners
G210-1442-00

Domino, IBM, the IBM logo, Lotus, and Lotus Notes are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. Other company, product or service names may be trademarks or service marks of others.