Using Structures with the DCOM Wrapper

The DCOM Wrapper maps a structure definition to an object inside an automation object. The structure can be created and modified independently of program calls, e.g., a structure object can be created once and can be used in one or more programs several times - it is always the same object with the same information. It is also possible to retrieve a structure via a program call and use it in another program.

This chapter covers the following topics:

- Overview
- Software AG IDL File
- Using Structures

Overview

A structure definition has always a name like a program name. The difference is that a structure will be defined with the keyword STRUCT. Furthermore a structure cannot be called like a program. A structure represents only a data object to be used with the programs. A structure has more the character of a group but unlike a group a structure will not be created automatically by the DCOM Wrapper object. The user has to create a structure with the corresponding functions.

- If a structure has embedded structures, first create the parent structure and then the embedded structures.
- If a structure has embedded groups, the embedded groups are created automatically when the parent structure. is created.
- If a group has embedded structures, the embedded structures must be created by the user. The parent group self is created by DCOM Wrapper object.

Software AG IDL File

A Simple Structure Example

```
LIBRARY 'STRUCTLIB' is

  STRUCT 'User' Is
  DEFINE DATA PARAMETER
    1 UserId       (AV)
    1 Password     (AV)
  END-DEFINE

  PROGRAM 'Prog01' Is
  DEFINE DATA PARAMETER
    1 ID           ('User')
    1 something    (AV)
  END-DEFINE

  PROGRAM 'Prog02' Is
```

```
  DEFINE DATA PARAMETER
    1 ID           (’User’)
    1 somewhere    (AV)
  END-DEFINE
```

Example for Visual Basic Version 6

```
’first create an instance of the DCOM Wrapper Object
Dim obj As New STRUCTLIB
’Declares a variable for our User structure
Dim userobj As Object
’Declares a variable for the second parameter
Dim something as String


’create the ’User’ structure
Set userobj = obj.createStructure_User
’now you can access the structure items
userobj.UserId = "userid"
userobj.Password = "password"

something = "Hello"

’call the DCOM object
obj.Prog01 userobj, something
’the same userobj could now used by obj.Prog02. Please consider that
’obj.Prog01 may have changed the data if you reuse userobj.

’get the response from the server and show the message a message box
MsgBox "User ID = " & userobj.UserId
```

A Simple Structure Example

Structures can also be defined in a more complex manner. This example shows a definition of a structure in a structure. In this example the customer can have multiple addresses and could have bought multiple products. The following example may not make sense, it is only an example to show the handling of complex structures and embedded structures.

```
LIBRARY ’CUSTLIB’ is

STRUCT ’AddressStruct’ Is
DEFINE DATA PARAMETER
  1 Street      (A255)
  1 City        (A255)
  1 Postal_Code (A32)
END-DEFINE

STRUCT ’CustomerStruct’ Is
DEFINE DATA PARAMETER
  1 Name        (A255)
  1 Address     (’AddressStruct’/V)
END-DEFINE

STRUCT ’SalesStruct’ Is
DEFINE DATA PARAMETER
  1 ProductName (A255)
  1 ProductID   (PU20)
  1 SalesDate   (D)
  1 SalesVolume (N12.2)
END-DEFINE
```

```
PROGRAM 'CustomerInformation' is
DEFINE DATA PARAMETER
  1 Customer    ('CustomerStruct')
  1 SalesData   ('SalesStruct'/V)
END-DEFINE
```

Example for Visual Basic Version 6

```
'first create an instance of the DCOM Wrapper Object
Dim obj As New CUSTLIB
'Declares an array of variables for our sales structure
Dim salesobj() As Object
'Declares a variable for our customer structure
Dim custobj As Object
'Declares a variable for our address structure
Dim Addressobj As Object
'temp strings
Dim addressdata As String
Dim salesdata As String
Dim i As Integer

'create the 'Customer' structure
Set custobj = obj.createStructure_CustomerStruct
'redim the Customer structure to the size 2 in the first dimension.
custobj.redim_address 2, 1, 0, 0, 0
'set the name of the customer
custobj.Name = "Customer A"
'create a new address structure
Set Addressobj = obj.createStructure_AddressStruct
'fill the data items of this structure
Addressobj.street = "back street"
Addressobj.city = "back town"
Addressobj.postal_code = "12345"
'assign the address structure to the first array item of the
'Customer structure.
custobj.address_indexAccess(0) = Addressobj

'create the second 'Customer' structure
Set Addressobj = obj.createStructure_AddressStruct
'fill the data items of this structure, too
Addressobj.street = "front street"
Addressobj.city = "front town"
Addressobj.postal_code = "54321"
'assign the address structure to the second array item of the
'Customer structure.
custobj.address_indexAccess(1) = Addressobj

'Redim the sales array object, the salesobj has now one
'array item
ReDim salesobj(0)
'create a new sales structure
Set salesobj(0) = obj.createStructure_SalesStruct
'fill the data items of this structure
salesobj(0).ProductName = "phone"
salesobj(0).productid = "12547"
salesobj(0).salesdate = Now
salesobj(0).salesvolume = "99.91"


'call the DCOM object with the two parameters.
obj.CustomerInformation custobj, salesobj
```

```
’now show the received data
addressdata = ""
addressdata = custobj.Name & vbCrLf

’please note that you need to use the DCOM Wrapper special function
’to access the embedded structures.
For i = 0 To custobj.get_address_UpperBound(1, 0)
   ’get a pointer to the address data
   Set Addressobj = custobj.address_indexAccess(i)
   ’pick the data from the address structure
   addressdata = addressdata & Addressobj.street & ";" & _
   Addressobj.city & vbCrLf
 Next


’please note that you don’t have a special function to access
’a structure array for the data on the first level.
salesdata = addressdata & vbCrLf
’ show the other sales
For i = 0 To UBound(salesobj)
’pick the data from the address structure
   salesdata = salesdata & salesobj(i).ProductName & ";" & _
   CLng(salesobj(i).salesvolume) & vbCrLf
Next

’now show data in a message box
MsgBox (salesdata)
```

Using Structures

Creating New Instances of a Structure

The DCOM Wrapper creates a function for each structure definition. The function is always a member of the library object and has always the same naming template. It begins with the keyword *createStructure_* with a following structure name. This function returns a IDispatch object pointer to a new valid structure data object.

```
STDMETHODIMP createStructure_<structure name>( IDispatch ** );
```

The method `createStructure_<structure name>` creates a new instance of the named original structure, a pointer to this structure is returned.

Example

IDL File

```
LIBRARY 'EXAMPLELIB' is
STRUCT 'SData' Is
DEFINE DATA PARAMETER
  1 Name        (A255)
  1 Address     (A255)
END-DEFINE
.
.
.
.
```

Visual Basic Version 6 Source Code

```
Dim sdata As Object
Dim obj As New EXAMPLELIB
set sdata = obj.createStructure_SData
```

.NET C# Source Code

```
IStructure_SData sdata;
EXAMPLELIBClass obj = new EXAMPLELIBClass();
sdata = (IStructure_SData)obj.createStructure_SData();
```