Module6: In-Process OLE Servers

§ 6.1 Creating an OLE Server in a DLL

§§ 6.1.1 The `OLEServer` Object

Object Linking and Embedding (OLE), released in 1991, was the evolution of Dynamic Data Exchange (DDE) that Microsoft developed for early versions of Windows. DDE was implemented in Dyalog APL through shared variable syntax.

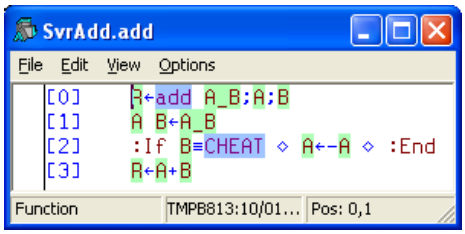
OLE evolved to become an architecture for software components known as the Component Object Model (COM), and later (Distributed COM) DCOM. OLE can handle compound documents containing text, graphics, video, and sound. COM has been called "the center of the Microsoft universe."

Component Object Model (COM) is a Microsoft platform for software components introduced by Microsoft in 1993. It is used to enable interprocess communication and dynamic object creation in any programming language that supports the technology. The term COM is often used in the software development world as an umbrella term that encompasses the OLE, OLE Automation, ActiveX, COM+ and DCOM technologies. Microsoft did not begin emphasizing the name COM until 1997.

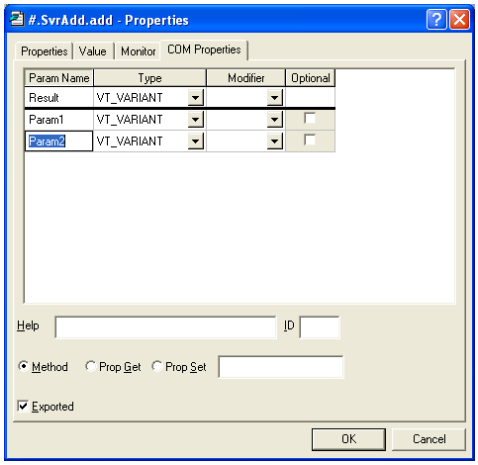
6.1.1.1 In a clear ws, create an object called `SvrAdd` of `Type OLEServer`. Rename the ws to `SvrAdd`.

§§ 6.1.2 Exporting Variables and Functions as Properties and Methods

6.1.2.1 In space `SvrAdd`, write a monadic, result-returning function, `add`, such as that below, or a more useful example.

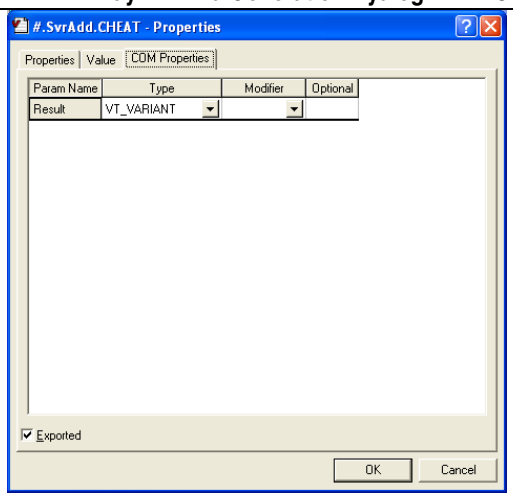


Right click on the function `add` and select [Properties][COM Properties]. Check the Exported check box then click on Param1 and hit the down arrow key. Set all types to VT_VARIANT, or any appropriate type. The properties box should then look like the adjacent property page:



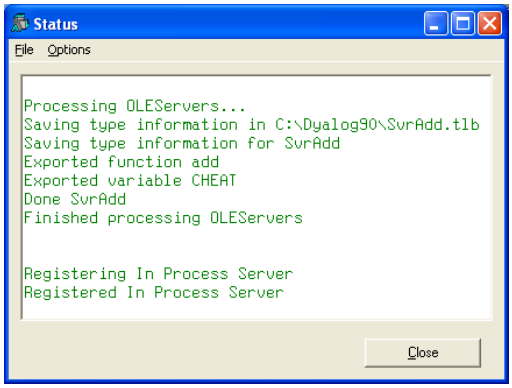
Select OK to set the function argument and result type information.

6.1.2.2 Create a variable called `CHEAT` with value zilde (`⍬`). Right click and select [Properties] [COM Properties]. Check the Exported check box. Change the type of the result to VT_VARIANT and hit OK to set the variable type information.



§§ 6.1.3 Saving and registering your `OLEServer`

6.1.3.1 Check the menu item [File][Make OCX/DLL on)SAVE] (version 9) and save the workspace. This returns a status information box if [Tools][AutoStatus] is checked.



At this point, Dyalog APL automatically updates the Windows registry and type libraries with all of the information needed to make your object accessible by COM. SvrAdd.TLB should be stored in a suitable directory.

Note that [File][Make OCX/DLL on)SAVE] has been replaced with [File][Export…] in version 10. From version 10, in-process OLE Servers and ActiveX Controls may be bound with either the development or runtime Dyalog APL DLL.

§ 6.2 Variable type Information

§§ 6.2.1 Setting Method Information

Assuming `⎕WX↳1` , the above steps for exporting functions as methods could have been achieved under program control by

```
      ExportedFns←⊂'add'
      SetFnInfo(⊂'add'),⊂('' 'VT_VARIANT')↵
('Param1' 'VT_VARIANT')('Param2' 'VT_VARIANT')¶
```

Note that the names of optional parameters are surrounded by brackets, *eg* `('[Param2]'...)`

§§ 6.2.2 Setting Property Information

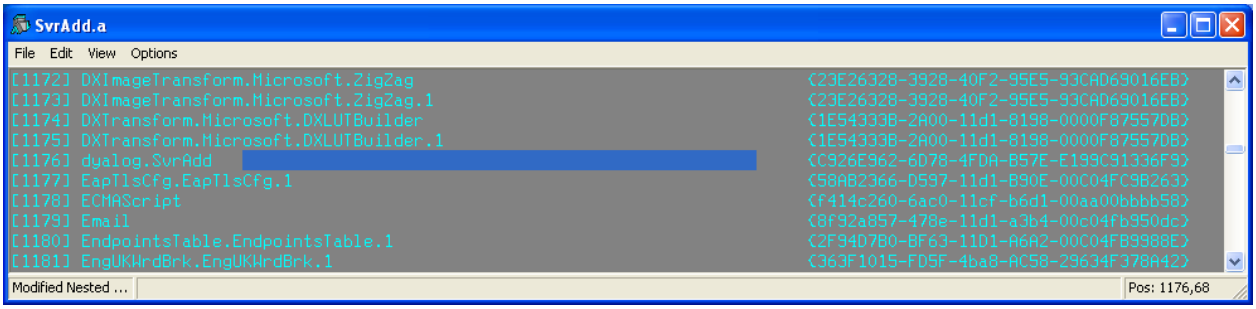
Similarly, variables can be exported as properties under program control.

```
      CHEAT←⍬
      ExportedVars←⊂'CHEAT'
      SetVarInfo'CHEAT' 'VT_VARIANT'
```

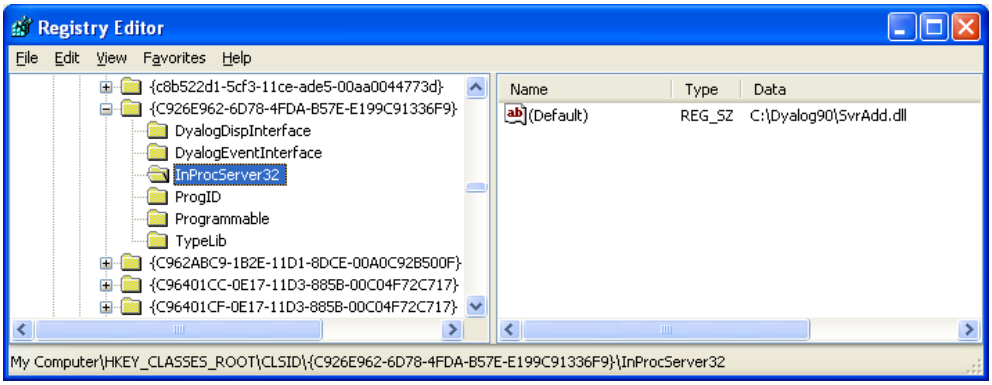
§§ 6.2.3 Exploring the Registry Entry

The Root has a property called OLEServers which lists the OLE Servers registered on your personal computer.

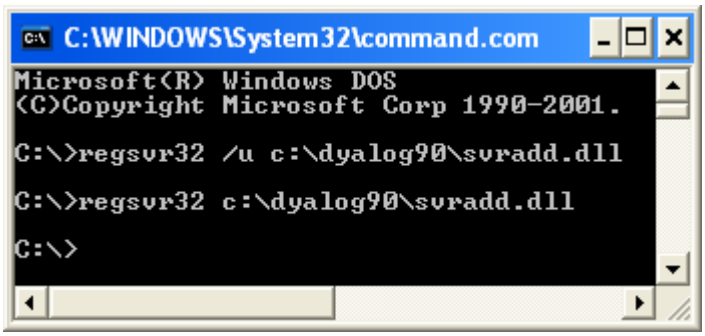
```
a←↑#.OLEServers
```



This list gives the unique class ID of each server. Looking in the registry at the classes under key HKEY_CLASSES_ROOT\CLSID reveals, amongst other things, the name of the DLL in which our server is stored.



The server may be unregistered and reregistered in a DOS box as required.



Note that Svradd.dll calls the dynamic link library version of Dyalog APL when it starts and Dyalog.dll must be visible to REGSVR32.EXE when the server is registered. In Dyalog version 10 DYALOG.DLL may be distributed free of charge as part of an application.

§ 6.3 Using your OLE Server

§§ 6.3.1 The *OLEClient* Object

In a clear workspace, create an *OLEClient* object with the *ClassName* property set to dyalog.SvrAdd.

```
⎕CS 'SVR'⎕WC'OLEClient' 'dyalog.SvrAdd'
```

The object contains no functions or variables but `add` is reported as a method and `CHEAT` is on the list of properties.

```
      )fns
      )vars
      )methods
add
```

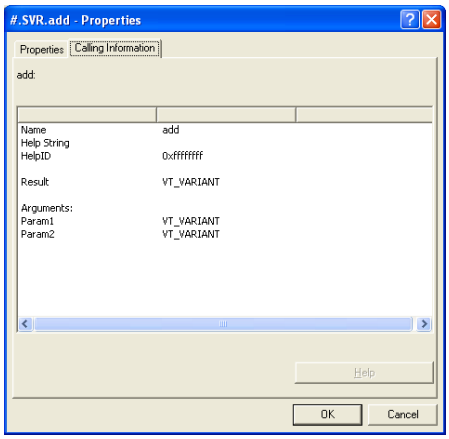
6.3.1.1 Try out your exported server method using various arguments. Set `CHEAT` to `5` and try `add` again.

6.3.1.2 Check the results of `GetMethodInfo'add'` and `GetPropertyInfo'CHEAT'`.

6.3.1.3 Investigate the workspaces CFILES.DWS and LOAN.DWS in conjunction with the explanations in Dyalog APL *Interface Guide* and related sections of www.dyalog.com

§§ 6.3.2 Examining Type Libraries

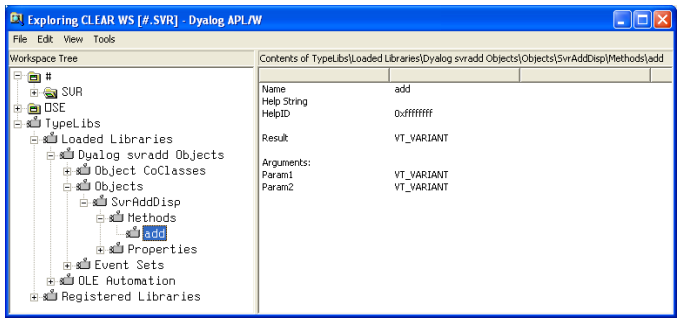
Calling information regarding exported methods may be found by right-clicking on `add` and selecting [Properties][Calling Information].



Or the same information may be obtained from the `OLEClient` method `GetMethodInfo` (see also `GetPropertyInfo`).

```
      DISPLAY GetMethodInfo'add'
.→-------------------------------------------------------------------------.
| .→-----------------. .→----------------------. .→----------------------. |
| | .⊖. .→----------. | | .→-----. .→----------. | | .→-----. .→----------. | |
| | | | |VT_VARIANT| | | |Param1| |VT_VARIANT| | | |Param2| |VT_VARIANT| | |
| | '-' '----------' | | '------' '----------' | | '------' '----------' | |
| '∈-----------------' '∈----------------------' '∈----------------------' |
'∈-------------------------------------------------------------------------'
```

Or the information may be obtained from the TypeLibs section of the workspace explorer.



Int32[] would be a more precise alternative to VT_VARIANT for these arguments and result.

§§ 6.3.3 Calling an OLE Server from VB

Given the files SvrAdd.DLL, Dyalog.DLL and SvrAdd.TLB, any language which can access OLE servers from DLLs may be used to run your server.

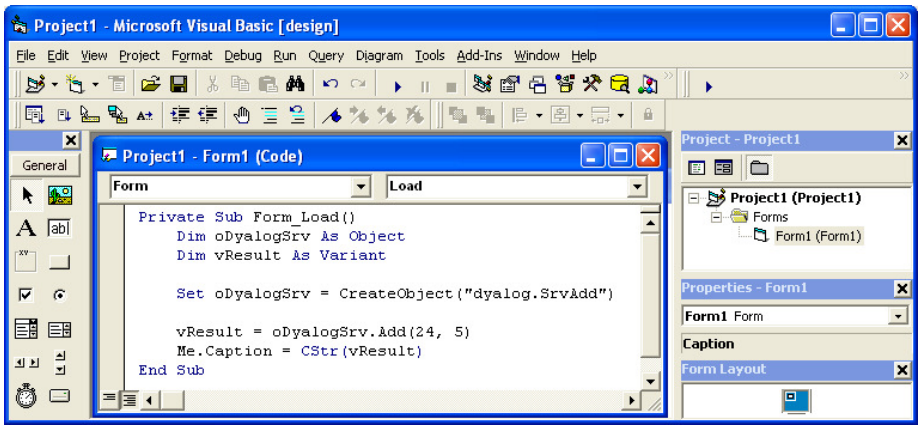
First the server has to be registered on the current machine, *eg* by the command line
C:\>regsvr32 c:\dyalog90\svradd.dll

Then an instance of the SvrAdd object must be created and the **add** function called.

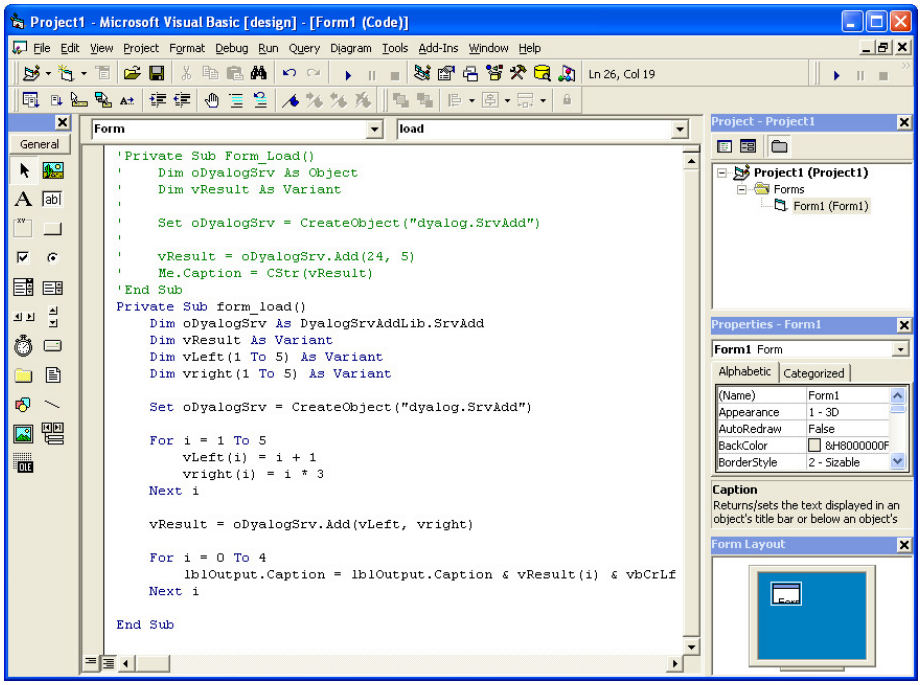
In VB a reference to the object is obtained using the CreateObject function
Set oDyalogSvr = CreateObject("dyalog.SvrAdd")

then the add method may be called using dot syntax
vResult = oDyalogSvr.add(24, 5)

In the following example, the result is placed on the caption of a form.

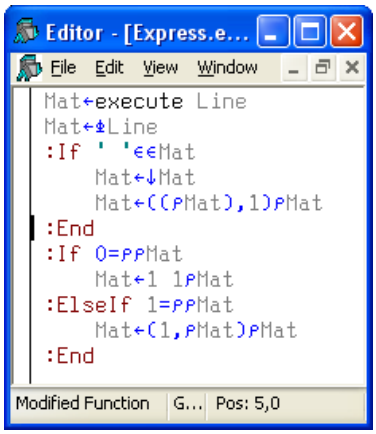


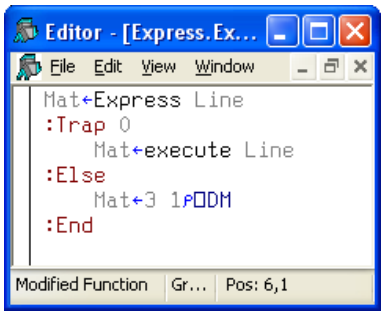
The next example sets the caption of a form in a loop.



A final example is based on two simple APL functions in a namespace called `#.Express`, and a third to turn the namespace into an OLEServer.

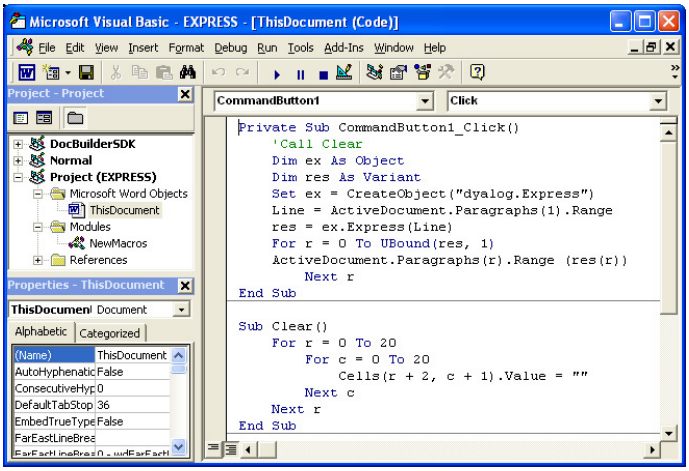
This example allows one to type a line of APL code into the first cell of an Excel spreadsheet and have a button on the spreadsheet execute the line. The result is placed on the ensuing lines of the spreadsheet.

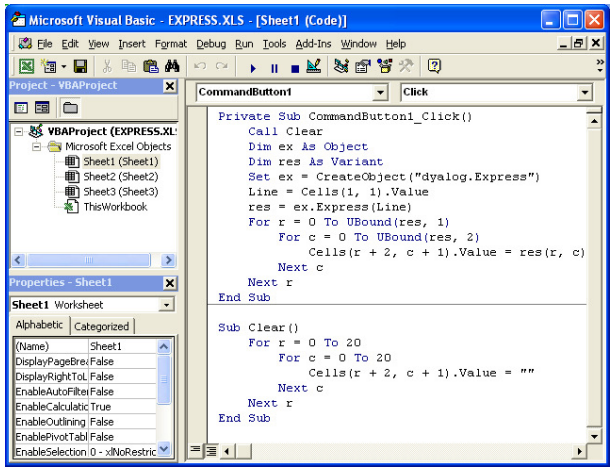




The exported functions need to have their Result and Param1 types set to VT_VARIANT before)SAVE.

The function `#.Express.Express` above is called by Visual Basic code behind Excel and the result is written to the cells of Excel in two different ways (below).





Note that later versions of Excel use Value2 in place of Value in the VB code above.

6.3.3.1 Ask for the next module on **OLE Clients** ☺.