



# dyalog.WSEngine

## Interaktive Nutzung des Dyalog Interpreters als OLE-Server

*(Microsoft Windows)*

Jürgen Wiedemann  
DPC

APL Germany Frühjahrstagung 2025  
28. März 2025

# Agenda

- Ausgangssituation und Anforderungen
- Umsetzung
- Demo

# Ausgangssituation und Anforderungen

- Migration von APL+Win nach Dyalog APL  
(mit allen typischen Herausforderungen)
- Anforderung
  - Bereitstellung einer Funktionalität wie in APL+Win:  
“APL+Win as an ActiveX server”
- Möglichkeit
  - zur Erzeugung einer neuen APL-Instanz
  - zum Aufruf aus APL und Excel heraus
  - zur Nutzung beliebiger Workspaces  
ohne zusätzliche Vorbereitung oder Bereitstellung

# Einsatzgebiet / Fallbeispiele

- Regressionstest mit parallelen Berechnungen in mehreren lokalen APL-Sessions
- Integration von APL in andere Umgebungen, z.B. in Excel (VBA)
- Einsatz im “Entwicklermodus”:
  - komplette Funktionalität der IDE
  - sichtbar
  - Debugger

# Herausforderung

- Dyalog bietet keine “fertige” Lösung für diese Anforderungen.
- Man kann zwar
  - Workspaces als Runtime, In- und Out-of-Process Server oder .Net Assembly exportieren.
  - Namespaces via Jarvis bereitstellen.
  - Aber stets mit einer fest definierten, festen Codebasis.
- IDE mit Editor und Debugger ?!?  
Genau dies aber war die Haupt-Anforderung

# Lösung

- OLEServer Object (□WC)
  - Erlaubt Sichtbarkeit und vollen Zugriff auf die **Session** (überraschenderweise)
  - Stellt aber einen festen Namespace und nur bestimmte Funktionen daraus bereit
  - Exkurs:  
Ein OLEServer Object wird aus einem Namespace erzeugt und die bereitgestellten Einstiegspunkte werden über dessen Eigenschaft “ExportedFns” definiert.
- Emulation der meisten APL+-Features

# Lösungsvorlage: APL+Win

- Overview “APL+Win as an ActiveX server”

Methods	Description
<b>Methods</b>	<b>Description</b>
<u>Call</u>	Call a function and return its value
<u>Exec</u>	Execute an expression and return its value
<u>SysCommand</u>	Execute a system-command
<b>Properties</b>	
<u>SysVariable</u>	Return or set the value of a system-variable
<u>Variable</u>	Return or set the value of a variable
<u>Visible</u>	Return or set the visibility of the APL session
<u>onComAction</u>	Triggered in the APL server when a command is initiated in the client session
<u>onNotify</u>	Triggered when the APL server's system object invokes the <code>Notify</code> method
<u>onSysNotify</u>	Triggered when a specified action occurs on the APL server

# dyalog.WSEngine - Überblick

- Sehr kleiner Namespace als OLEServer
- Sehr wenige und allgemeine Methoden
- Zweck: Umgehung von Beschränkungen des OLEServer Objekts
- Der OLEServer Namespace fungiert nur als Zwischenschicht zur “dynamischen” Nutzung beliebiger APL Quellen.



# dyalog.WSEngine - Details

**SysCommand**    A `□CY` into OLE-Session  
`objAPL.SysCommand<'LOAD ',ws`

**SysVariable**    A Get `□WA` value  
`objAPL.SysVariable<'WA'`

**Variable**        A Return or set value of a variable  
`x←objAPL.Variable<'APLvar'`  
`x←objAPL.Variable 'XX'(2 2ρ14)`

**Call**            A Call niladic/monadic/dyadic fn  
`x←objAPL.Call<'fn_nil'`  
`x←objAPL.Call 'fn_mon' 123`  
`x←objAPL.Call 'fn_dya' 123 ^2`

# dyalog.WSEngine - Details

<b>Exec</b>	<b>A Execute expressions, return value</b> <code>x←objAPL.Exec⊢'⊞NC⊢'myVar'''</code>
<b>Exec_async</b> <i>(no result)</i>	<b>A Execute asynchr. (add to event queue)</b> <code>objAPL.Exec_async⊢'⊞OFF'</code> <code>objAPL.Exec_async⊢'longCalc'</code>
<b>setVisible</b>	<b>A Hide/Show APL-Session</b> <code>objAPL.setVisible 0</code> <code>objAPL.setVisible 1</code>
<b>getObjects</b>	<b>A Return names according to name class</b> <code>vars←objAPL.getObjects 2</code>
<b>Update</b>	<b>A Update WSEngine namespace</b> <code>objAPL.Update⊢'E:\Fix.dws'</code>

# Deployment

## ■ Registrierung

- Mit der OLERegister Methode, in □L X
- Aufruf mit Option /RegWSEngine
- `dyalog.exe WSEngine.dws -MAXWS=256M  
RegWSEngine=1 DYALOG_NOPOPUIS=1`

## ■ Aufruf

- VBA  
`Dim objAPL As Object  
Set objAPL = CreateObject("dyalog.WSEngine")`
- Dyalog  
`'objAPL' □WC 'OLEClient' 'dyalog.WSEngine'`

# Beispiele

## Aus APL

```
'APL' □ WC 'OLEClient' 'dyalog.WSEngine'  
APL.setVisible 1  
x ← APL.SysCommand 'LOAD ', ws  
x ← APL.Call 'init_appl'  
x ← APL.Call 'do_calc' 123  
x ← APL.Exec_async '□OFF'  
□EX 'APL'
```

## Aus Excel (VBA)

```
Set APL = CreateObject("dyalog.WSEngine")  
rc = APL.SysCommand("LOAD " & DWSfile)  
rc = APL.SysVariable("PATH", "#.UTILS")  
rc = APL.Call("init_appl")  
rc = APL.Call("do_calc", 123)  
APL.Exec_async (apl_quad & "OFF")  
Set APL = Nothing
```

# Demo

- Dyalog v19 using WSEngine v18.2:
  - ]demo F:\tmp\WSE

# Demo: APL ↔ dyalog.WSEngine

The screenshot displays the Dyalog APL/W-64 environment. The main editor window shows the following code:

```
)clear
clear ws

]demo F:\tmp\WSE

Loaded script F:\tmp\WSE.dyalog
Script Initialized...
  A Demo: Dyalog APL OLE-Server << dyalog.WSEngine >>
]box on
Was ON
]rows on
Was OFF
'objAPL' 'WC'OLEClient' 'dyalog.WSEngine'  A Create instance
```

The left-hand pane shows the output of the demo:

```
Dyalog APL/W-64 Version 18.2.50027
Serial number: 000091
Sat Sep 21 11:14:44 2024
E:\Ergo\WSEngine\WSEngine.DWS saved Thu
```

The bottom status bar indicates the current object is 'objAPL' and the workspace is 'on (Undefined)'. The system tray shows the date and time as 11:14 on 21.09.2024.

# Demo: APL ↔ dyalog.WSEngine

F:\tmp\WSEngine\_Demo.dws (WSEngine)- Dyalog APL/W-64  
 Datei Bearbeiten Anzeigen Fenster Session-Objekt Sitzungs-Protokoll Aktion Optionen Werkzeuge Threads Hilfe  
 WS [Icons] Objekt [Icons] Tool [Icons] Edit [Icons] Sitzung [Icons] APL385 Unicode 20

```

Dyalog APL/W-64 Version 18.2.50027
Serial number: 000091
Sat Sep 21 11:14:44 2024
E:\Ergo\WSEngine\WSEngine.DWS saved Thu
prepare [WA...

Starte [CY F:\tmp\WSEngine_Demo.dws
[CY beendet
LOAD(CY) erfolgreich durchgeführt
    
```

Debugger

```

Copying F:\tmp\WSEngine_Demo.dws
CurObj:
    
```

CLEAR WS - Dyalog APL/W-64

```

File Edit View Window Session Log Action Options Tools Threads Help Next Prev Init Edit
WS [Icons] Objekt [Icons] Tool [Icons] Edit [Icons] Session [Icons] APL385 Unicode 20
Language Bar
    
```

objAPL.[-1] 'WC'OLEClient 'dyalog.WSEngine' A Create instance										
objAPL.[-1]NL [-2] A Properties										
AutoBrowse	ChildList	ClassID	ClassName	Data	Describe	Event	EventList	Handle	HelpFile	Instanc

objAPL.[-1]NL [-3] A Methods										
Call	Exec	Exec_async	SysCommand	SysVariable	Update	Variable	getObjects	getVariable	setVariab	

```

objAPL.SysVariable='WA'
1000933024
objAPL.setVisible 0 A Hide
objAPL.setVisible 1 A Show
objAPL.Update='F:\tmp\Update_2024_1_3.dws' A Backdoor-Update

[-2] Fehler beim Update. Workspace konnte nicht importiert werden

objAPL.SysCommand='LOAD F:\tmp\WSEngine_Demo.dws' A Load/Copy code

0 SysCommand: LOAD Parameter: F:\tmp\WSEngine_Demo.dws
    
```

Debugger

```

Ready...
CurObj: code (Undefined)
    
```

INS NUM  
8:1 [Icons] [Icons] [Icons] [Icons] objAP

11:16  
21.09.2024

# Demo: APL ↔ dyalog.WSEngine

The screenshot shows the Dyalog APL/W-64 environment. The main editor window displays the following APL code and its output:

```
0 SysCommand: LOAD Parameter: F:\tmp\WSEngine_Demo.dws  
  
objAPL.getObjects 2 A #.⎵NL 2  
┌ Describe aa bb cc  
└───
```

The output shows a table with columns 'Describe', 'aa', 'bb', and 'cc'. Below this, the code continues:

```
objAPL.getObjects 3 A #.⎵NL 3  
┌ demo_WSEngine func_with_error plus test_dy test_mon test_nil  
└───
```

Below the table, the code shows variable assignments and their values:

```
⎵x←objAPL.Variable←'aa' A get variable "aa"  
ABC123  
⎵x←objAPL.Variable←'cc' A get variable "cc"
```

The output shows a table with columns 'wert1', 'wert2', and 'wert3':

wert1	2222
wert2	1 2 3 4
wert3	111 2.22 ~3.333

The debugger window shows 'Ready...' and 'CurObj: code (Undefined)'. The status bar at the bottom indicates '8:1' and various flags like 'DQ:0', 'TRAP', 'SI:0', 'IO:1', and 'x+o'.



# Demo: APL ↔ dyalog.WSEngine

The screenshot displays the Dyalog APL/W-64 environment. The main window shows the following text:

```
Dyalog APL/W-64 Version 18.2.50027
Serial number: 000091
Sat Sep 21 11:14:44 2024
E:\Ergo\WSEngine\WSEngine.DWS saved Thu
prepare WA...

Starte CY F:\tmp\WSEngine_Demo.dws
CY beendet
LOAD(CY) erfolgreich durchgeführt
```

The debugger window shows the following execution log:

```
-2 Fehler beim Lesen der Variablen NotExistingVar:
  VALUE ERROR Variable[38] NotExistingVar

  [+x←objAPL.Call'exec' 'test_nil'           A Execute and get result: niladic function
test_nil called
  [+x←objAPL.Call'exec' 'test_mon' 123      A Execute and get result: monadic function
test_mon called
  argument: 123

  [+x←objAPL.Call'exec' 'test_dy' 123 -2   A Execute and get result: dyadic function
test_dy called
  argumenst: -2 123

  [+x←objAPL.Call'exec' 'plus' 123 0.456
123.456
  [+x←objAPL.Call'exec' 'plus' (13)(0.1×13)
1.1 2.2 3.3
```

The debugger status bar at the bottom indicates: CurObj: function (Undefined). The Windows taskbar at the bottom shows the time as 11:18 on 21.09.2024.

# Demo: APL ↔ dyalog.WSEngine

The screenshot displays the Dyalog APL IDE interface. The main window shows a code editor with the following APL code:

```
erg+larg plus arg
```

The debugger window is open, showing the execution flow and variable values:

- `test_mon called` with `argument: 123`
- `test_dy called` with `argumenst: -2 123`
- `plus` function execution: `123 0.456`
- `plus` function execution: `1.1 2.2 3.3`
- `plus` function execution: `1.1 2.2 3.3` (force an internal error --> Test debugge)

The debugger window also shows the error message: `VALUE ERROR Variable[38] NotExistingVar`.

# Demo: APL ↔ dyalog.WSEngine

The image shows a Windows desktop with two Dyalog APL/W-64 windows. The left window is a debugger window titled "Dyalog APL/W-64 Version 18.2.50027" with a green background. It displays the following text:

```
Dyalog APL/W-64 Version 18.2.50027
Serial number: 000091
Sat Sep 21 11:14:44 2024
E:\Ergo\WSEngine\WSEngine.DWS saved Thu
prepare WA...

  Starte CY F:\tmp\WSEngine_Demo.dws
  CY beendet
  LOAD(CY) erfolgreich durchgeführt
LENGTH ERROR: Mismatched left and right
plus[2] erg+larg+arg
      ^
      arg+10000
      →lc
2024 9 21 11 19 46 703
2024 9 21 11 19 50 930
```

The right window is an editor window titled "CLEAR WS - Dyalog APL/W-64" with a light blue background. It contains the following APL code and comments:

```
123.456
  [+x←objAPL.Call'plus'(13)(0.1×13)
1.1 2.2 3.3

  [+x←objAPL.Call'plus'(15)(0.1×13)      A force an internal error --> Test debugge
10000.1 10000.2 10000.3
  [+x←objAPL.Exec'⎕SE.Link.Import'#.Utils' 'F:\tmp\Utils'      A Import Code
Imported: #.Utils ← F:\tmp\Utils
objAPL.getObjects 9

  [+x←objAPL.Call'#.Utils.Δuserid'      A Execute and get result: niladic function
jwied
  [+x←objAPL.Exec'⎕dl 3 ⋄ ''done''      A Execute synchronous, wait for result
done
  [+x←objAPL.Exec_async'⎕TS ⋄ ⎕dl 4 ⋄ ⎕TS '      A Execute asynchronous, don't wait

  0 Kommando in Event-Queue gestellt

  [+x←objAPL.Call'plus'(13) 0.01      A next command, don't wait
1.01 2.01 3.01
```

The Windows taskbar at the bottom shows the search bar, taskbar icons, and system tray with the time 11:19 and date 21.09.2024.

# Demo: APL ↔ dyalog.WSEngine

CLEAR WS - Dyalog APL/W-64

File Edit View Window Session Log Action Options Tools Threads Help Next Prev Init Edit

WS Object Tool Edit Session APL385 Unicode 20

Language Bar

```
⎕←x←objAPL.Exec'⎕SE.Link.Import' '#.Utils' 'F:\tmp\Utils' A Import Code
Imported: #.Utils ← F:\tmp\Utils
objAPL.getObjects 9
```

Utils	WSEngine
⎕←x←objAPL.Call'#.Utils.Δuserid'	A Execute and get result: niladic function
jwied	
⎕←x←objAPL.Exec'⎕dl 3 ⋄ ''done''	A Execute synchronous, wait for result
done	
⎕←x←objAPL.Exec_async'⎕TS ⋄ ⎕dl 4 ⋄ ⎕TS '	A Execute asynchronous, don't wait
0 Kommando in Event-Queue gestellt	
⎕←x←objAPL.Call'plus'(ι3) 0.01	A next command, don't wait
1.01 2.01 3.01	
⎕←x←objAPL.Exec_async'⎕OFF'	A close remote APL session
0 Kommando in Event-Queue gestellt	
⎕EX'objAPL'	A expunge object

Debugger Ready... Ins NUM CurObj: object (Undefined) 8:1 ⎕DQ:0 ⎕TRAP ⎕SI:0 ⎕IO:1

11:20 21.09.2024



# dyalog.WSEngine

## Fragen... ?

Jürgen Wiedemann  
DPC

APL Germany Frühjahrstagung 2025  
28. März 2025