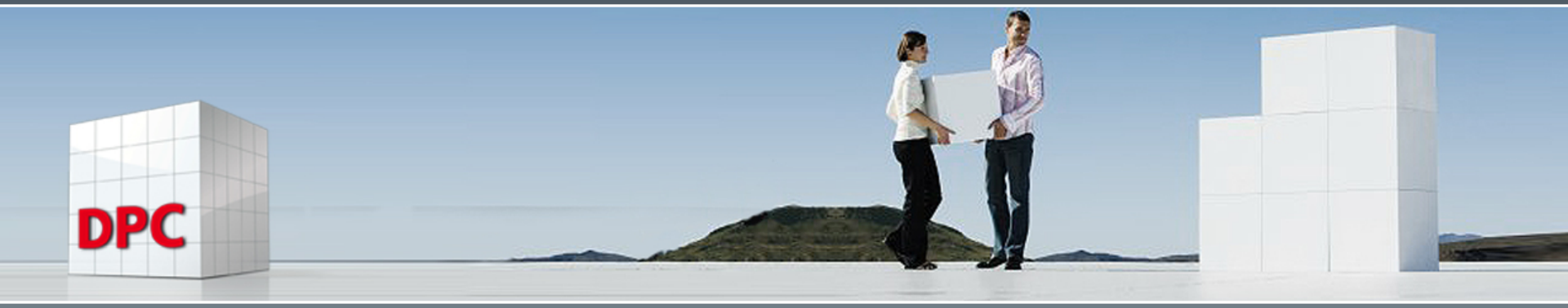


dyalog.WSEngine

Deployment of the Dyalog Interpreter as OLE-Server for interactive Use

(Microsoft Windows only)

Jürgen Wiedemann
DPC

Dyalog ´24
17. September 2024

Agenda

- Introducing DPC and myself
- Starting point and Requirements
- Implementation
- Demo

About DPC (Dittrich & Partner Consulting)

- Founded in 1981
- Acquired from HBA Consulting in 2023
- 10 employees, together with HBA 50
- Located in Germany, customers in Germany, Austria and Switzerland
- Providing consulting services
  mainly for life insurance companies
- Distributing APL Software licences

About me

- Working for DPC since 1998
- Consultant with focus on APL
- Done a lot of APL migrations
- Managing director since 2019
  (second job)

Starting point and Requirements

- Migration of several APL+Win applications
  (with all the typical challenges)
- Requirement:
  - special functionality referred to as
    “APL+Win as an ActiveX server”
- Possibility to
  - create an instance of APL
  - to use any workspace in it without special preparation

Use cases

- Regression test for lots of calculations in parallel on several local instances
- Integration of APL in other environments, e.g. in Excel (VBA)
- Operation in “Developer mode”:
  - fully functional IDE
  - visible
  - debugging and tracing

Solution

- OLEServer object
  - allows visibility and full access to the session
  - but deploys a specific namespace and only certain functions from it
  - an OLEServer object is generated from a namespace and the entry points to be provided are defined with its ExportedFns property
- Emulation for most of the APL+-Features

Solution - Template

- Overview “APL+Win as an ActiveX server”

Methods	Description
Call	Call a function and return its value
Exec	Execute an expression and return its value
SetOrphanTimeout	Sets the timeout period for the parent process
SysCall	Call a system-function and return its value
SysCommand	Execute a system-command
UExec	Execute an APL expression in the Unicode-aware client and return a (Variant) value
Properties	
SysVariable	Return or set the value of a system-variable
Variable	Return or set the value of a variable
Visible	Return or set the visibility of the APL session
Events	
onComAction	Triggered in the APL server when a command is initiated in the client session
onNotify	Triggered when the APL server's system object invokes the Notify method
onSysNotify	Triggered when a specified action occurs on the APL server

Solution - Template

- Overview “APL+Win as an ActiveX server”

Methods	Description
`Call`	Call a function and return its value
`Exec`	Execute an expression and return its value
`SysCommand`	Execute a system-command
Properties	
`SysVariable`	Return or set the value of a system-variable
`Variable`	Return or set the value of a variable
`Visible`	Return or set the visibility of the APL session

dyalog.WSEngine - Essentials

- Tiny namespace as an OLEServer
- Very few and general methods
- To overcome some limitations that apply to an OLEServer
- The OLEServer namespace only acts as an intermediate layer for dynamic use of any APL resources.

dyalog.WSEngine - Details

SysCommand

```
⍝ ⎕CY into OLE-Session
objAPL.SysCommand⊂'LOAD ',ws
```

SysVariable

```
⍝ Get ⎕WA value
objAPL.SysVariable⊂'WA'
```

Variable

```
⍝ Return or set value of a variable
x←objAPL.Variable⊂'APLvar'
x←objAPL.Variable 'XX'(2 2⍴⍳4)
```

Call

```
⍝ 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

Exec

```
⍝ Execute expressions, return value
x←objAPL.Exec⊂'⎕NC⊂''myVar'''
```

Exec_async
(no result)

```
⍝ Execute asynchr. (add to event queue
objAPL.Exec_async⊂'⎕OFF'
objAPL.Exec_async⊂'longCalc'
```

setVisible

```
⍝ Hide/Show APL-Session
objAPL.setVisible 0
objAPL.setVisible 1
```

getObjects

```
⍝ Return names according to name class
vars←objAPL.getObjects 2
```

Update

```
⍝ Update WSEngine namespace
objAPL.Update⊂'E:\Fix.dws'
```

Deployment

- Registration
  - with the OLERegister method invoked in ⎕LX
  - called with a special switch /RegWSEngine
  - **`dyalog.exe WSEngine.dws -MAXWS=256M RegWSEngine=1 DYALOG_NOPOPUPS=1`**
- Invocation
  - VBA

```
Dim objAPL As Object
Set objAPL = CreateObject("dyalog.WSEngine")
```

  - Dyalog

```
'objAPL' ⎕WC 'OLEClient' 'dyalog.WSEngine'
```

Examples

From 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'
```

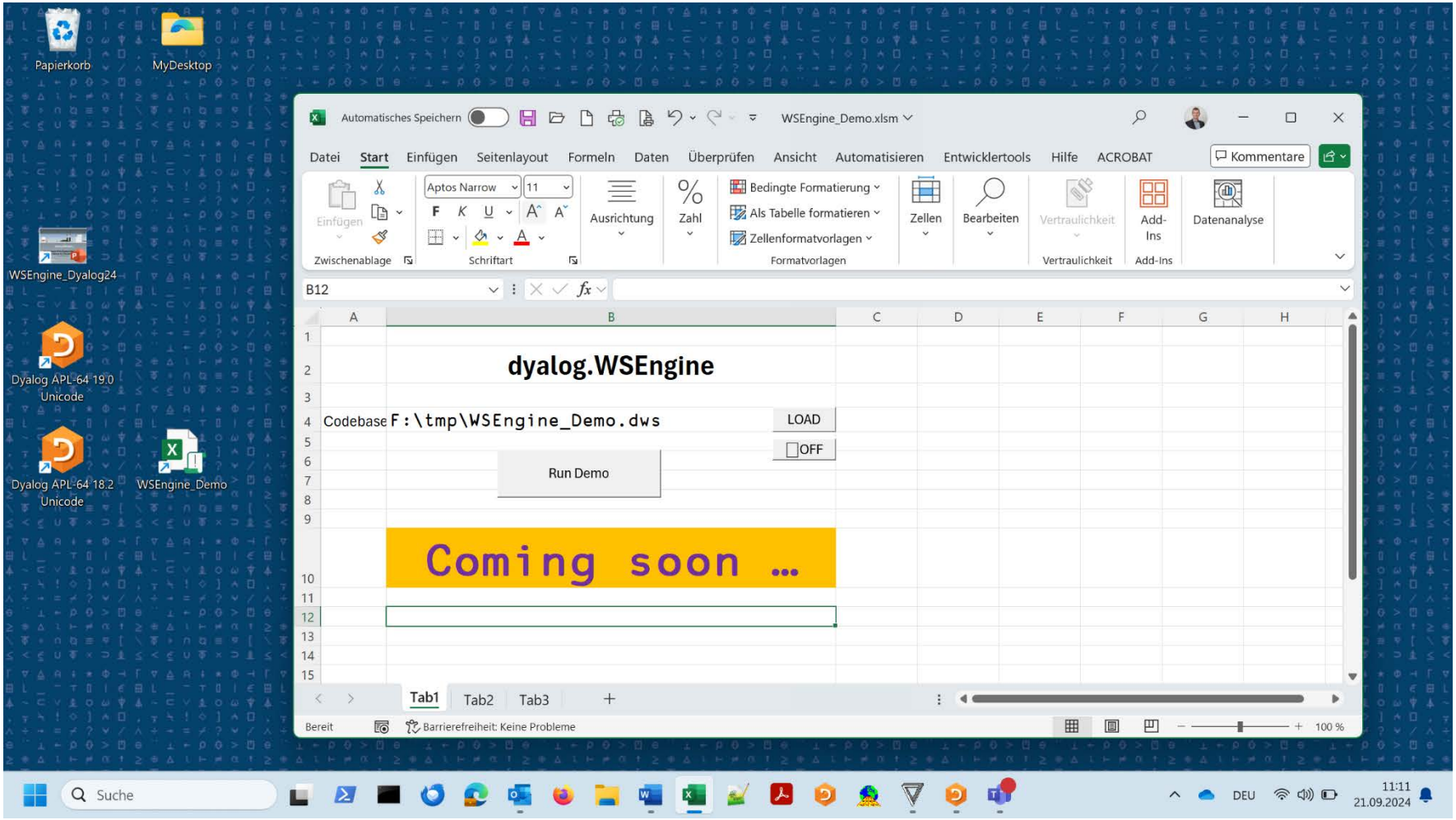
From Excel (VBA)

```
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
```

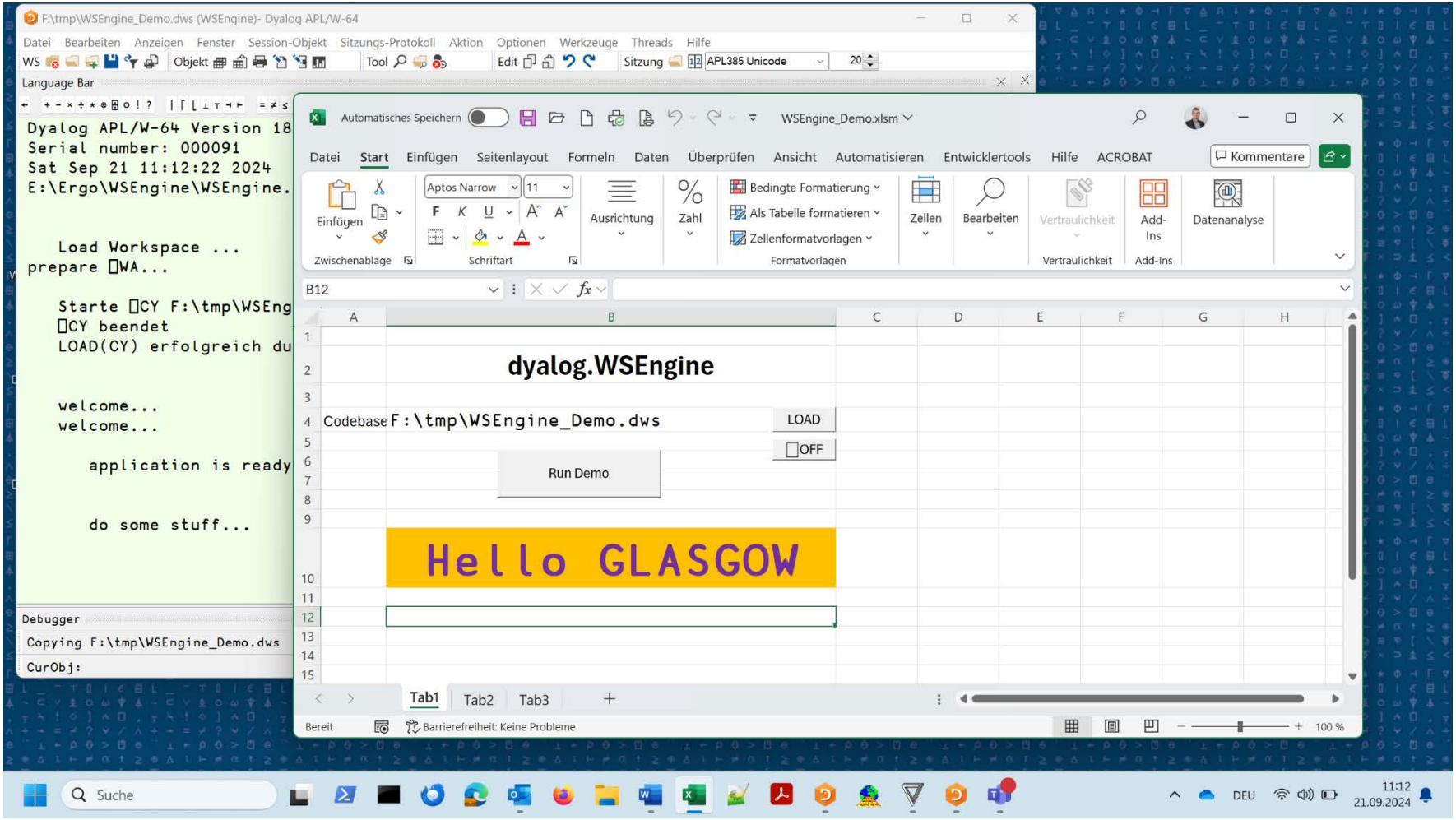
Demo

- Excel/VBA
  - `CreateObject("dyalog.WSEngine")`

Demo: Excel/VBA ↔ dyalog.WSEngine



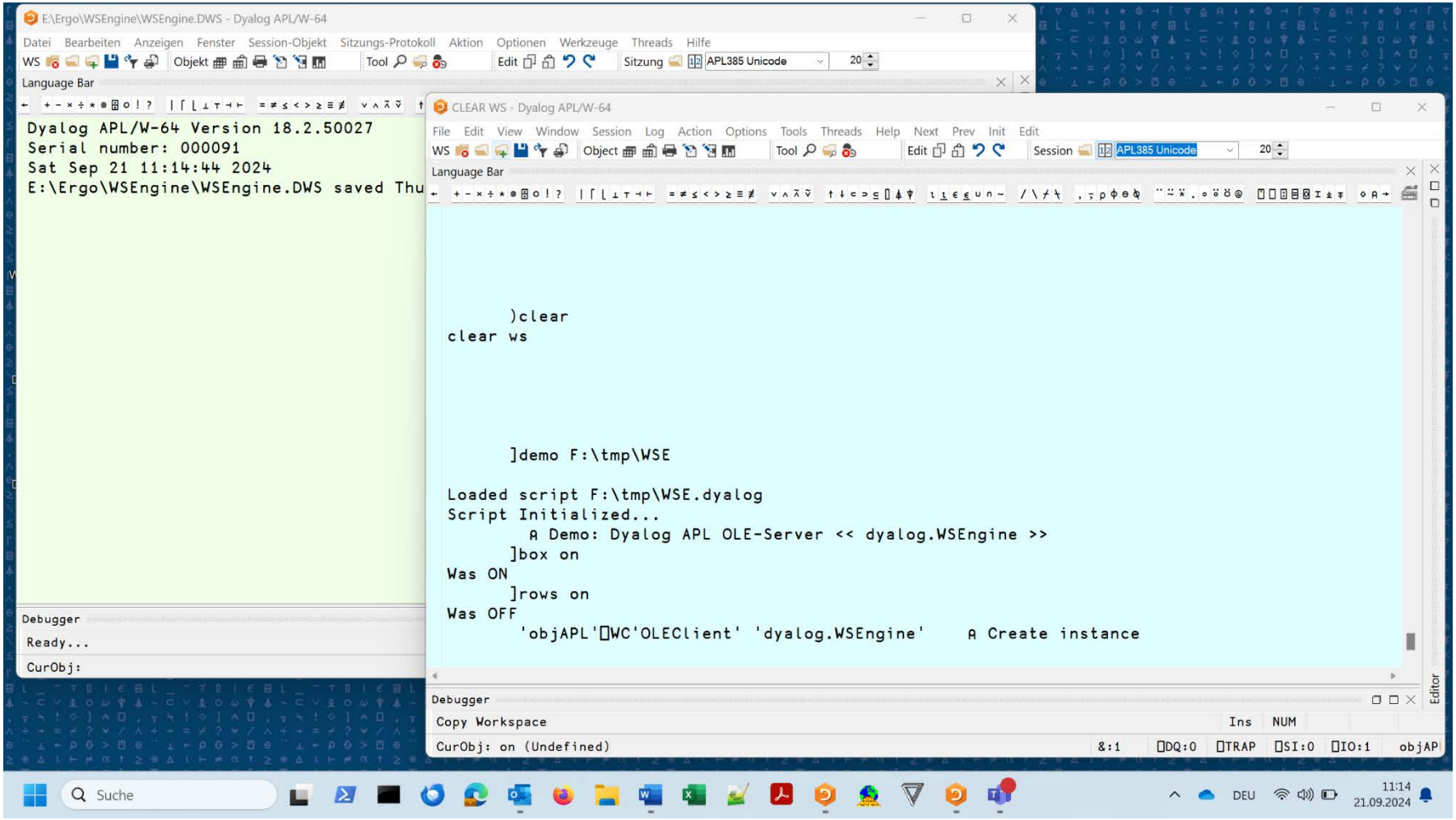
Demo: Excel/VBA ↔ dyalog.WSEngine



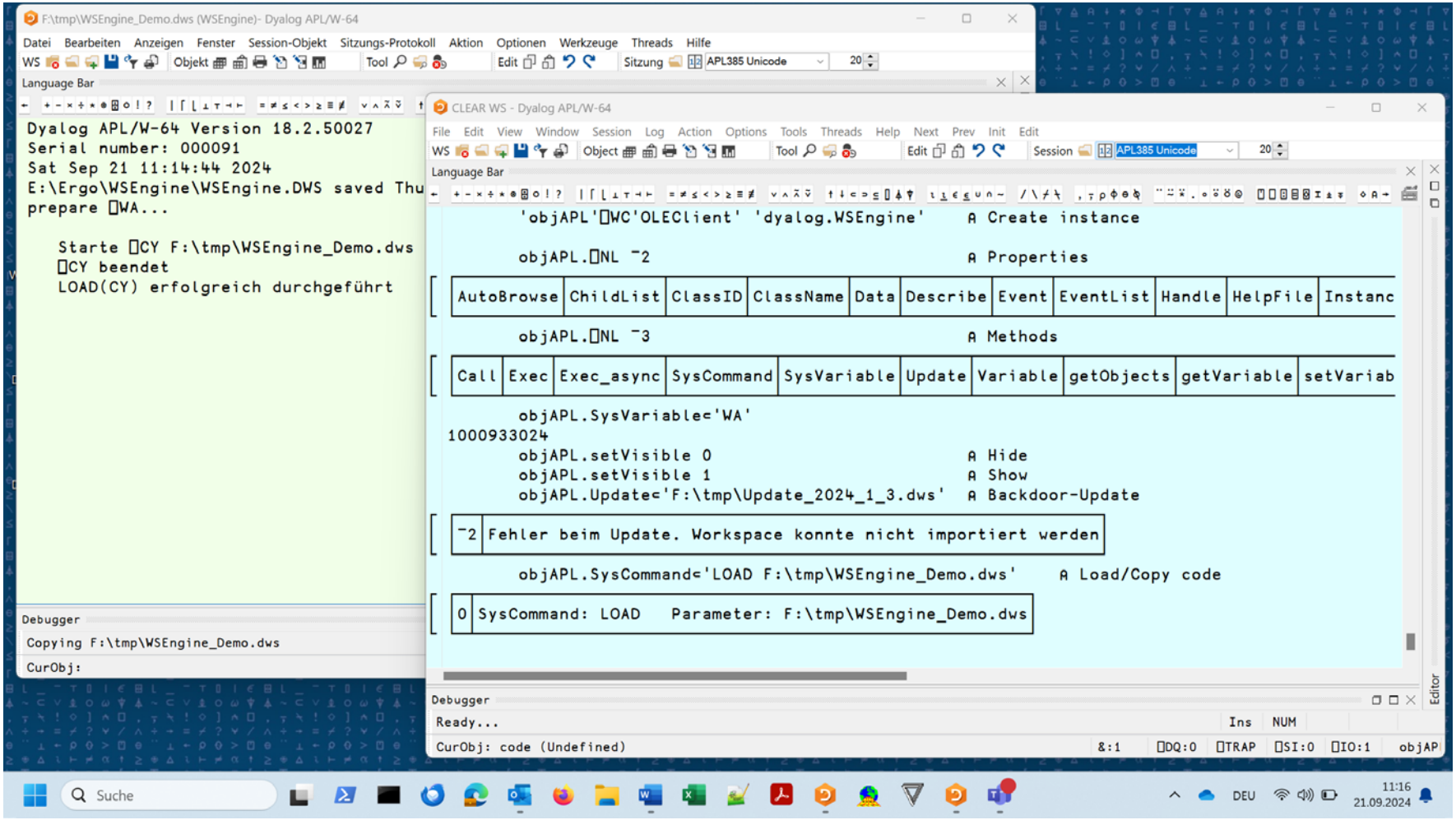
Demo

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

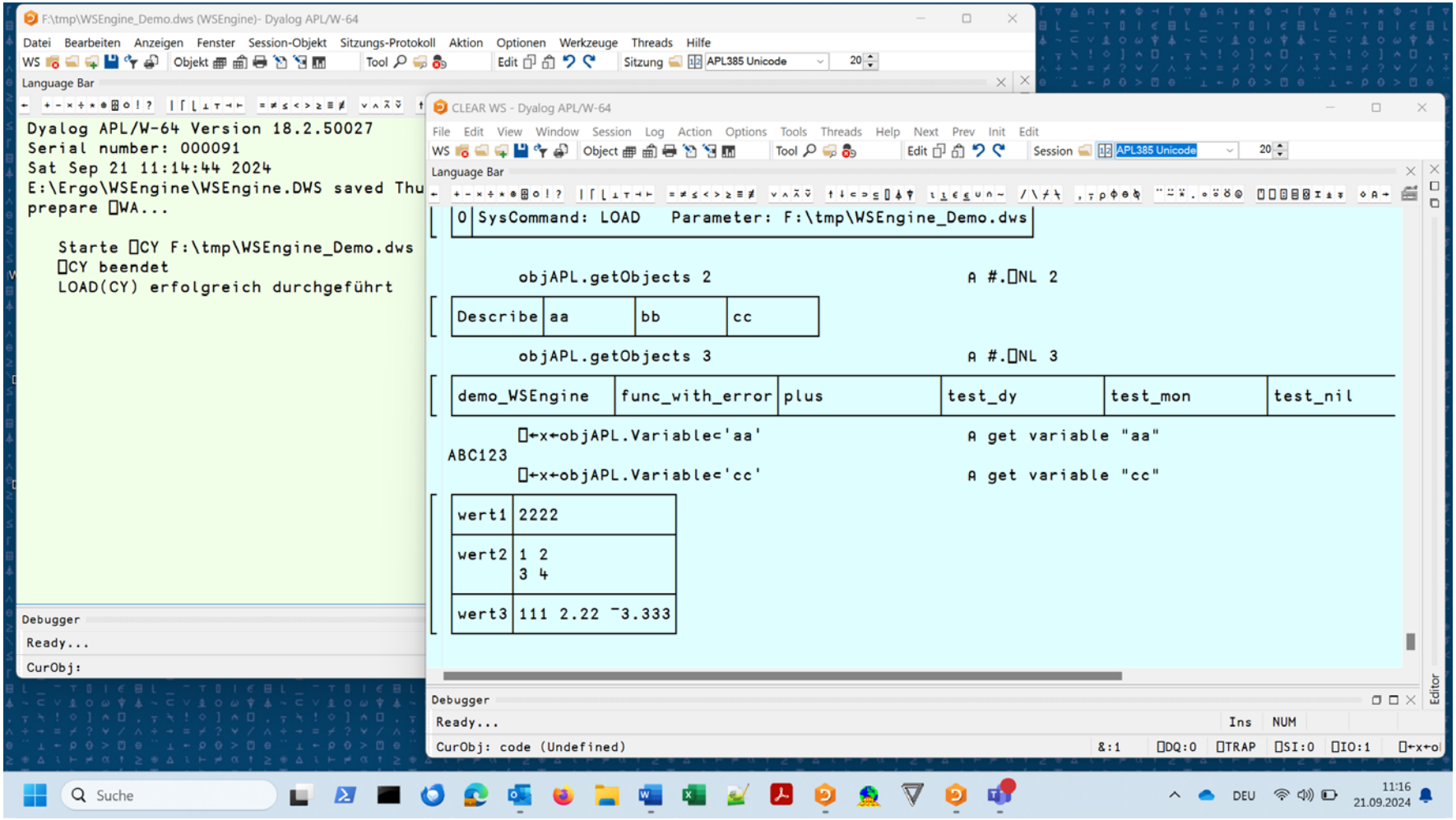
Demo: APL ↔ dyalog.WSEngine



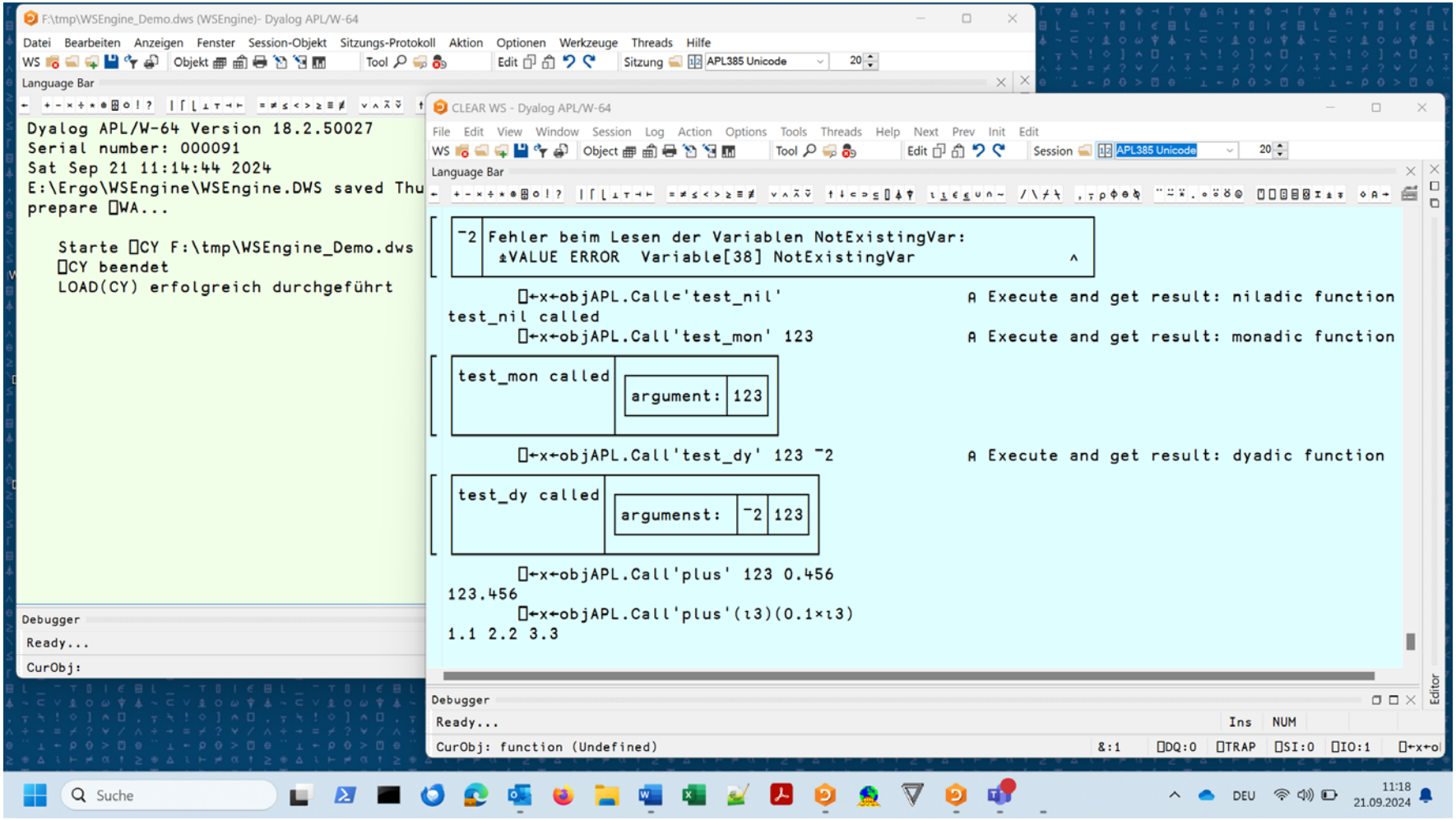
Demo: APL ↔ dyalog.WSEngine



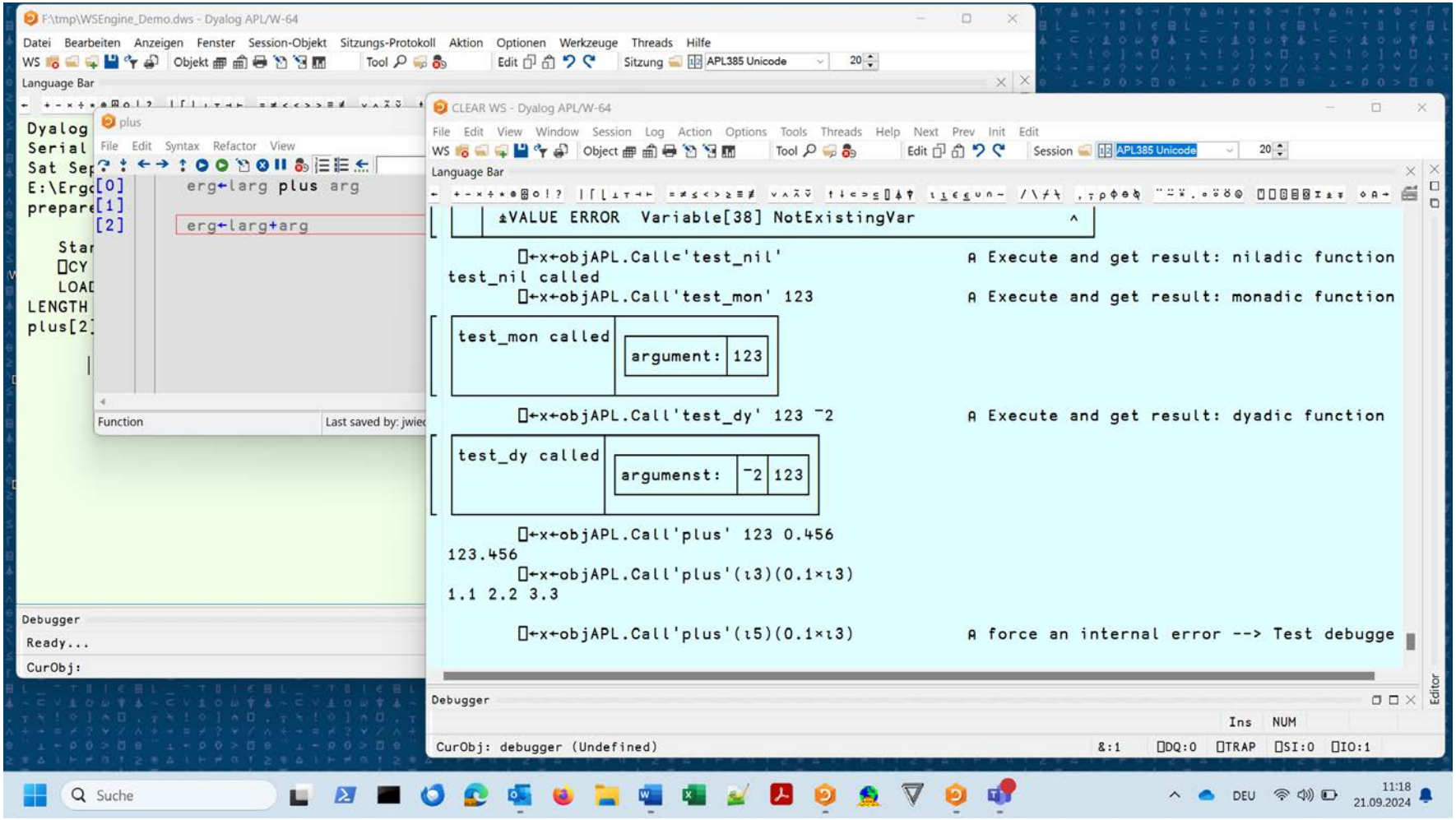
Demo: APL ↔ dyalog.WSEngine



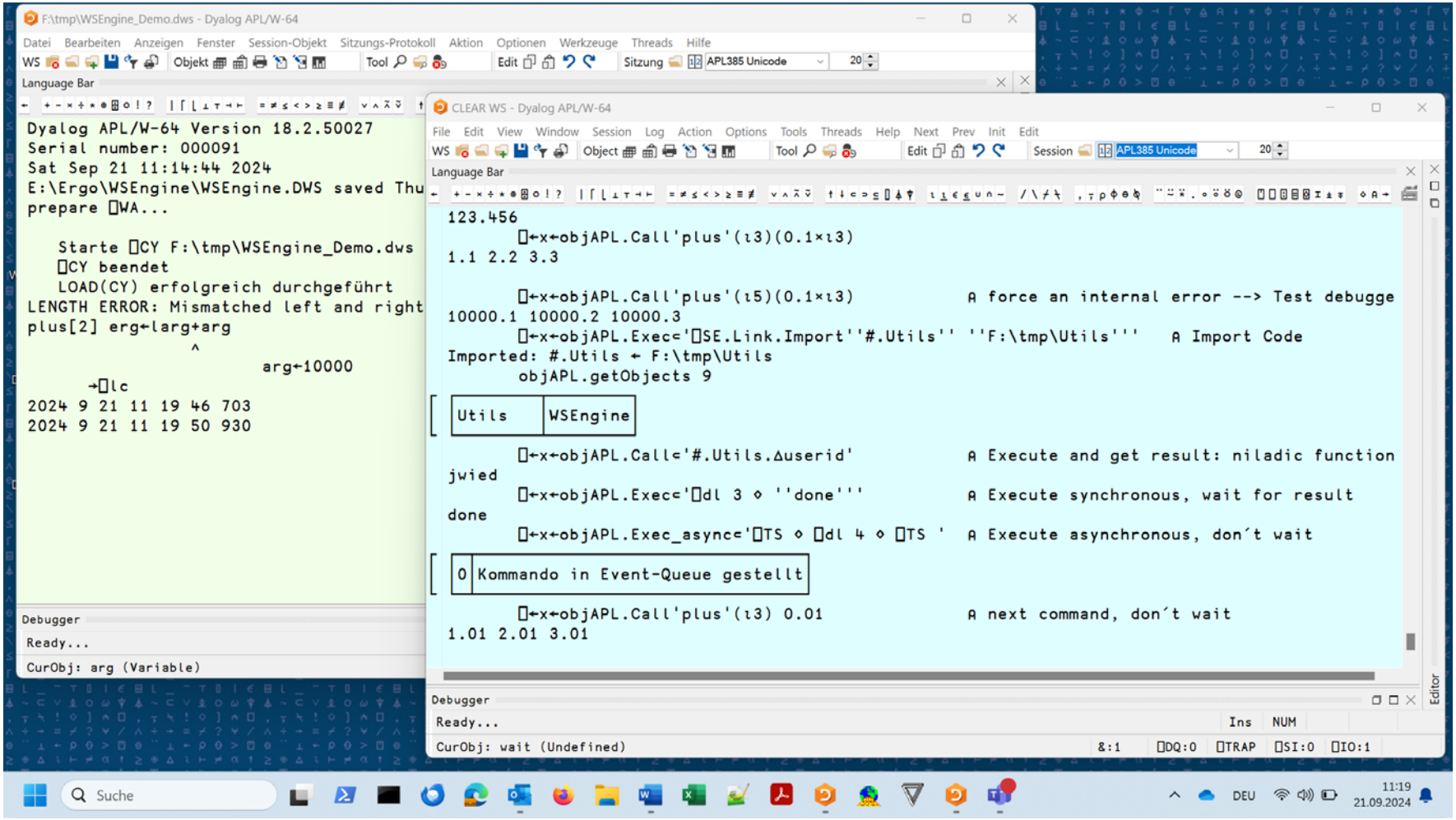
Demo: APL ↔ dyalog.WSEngine



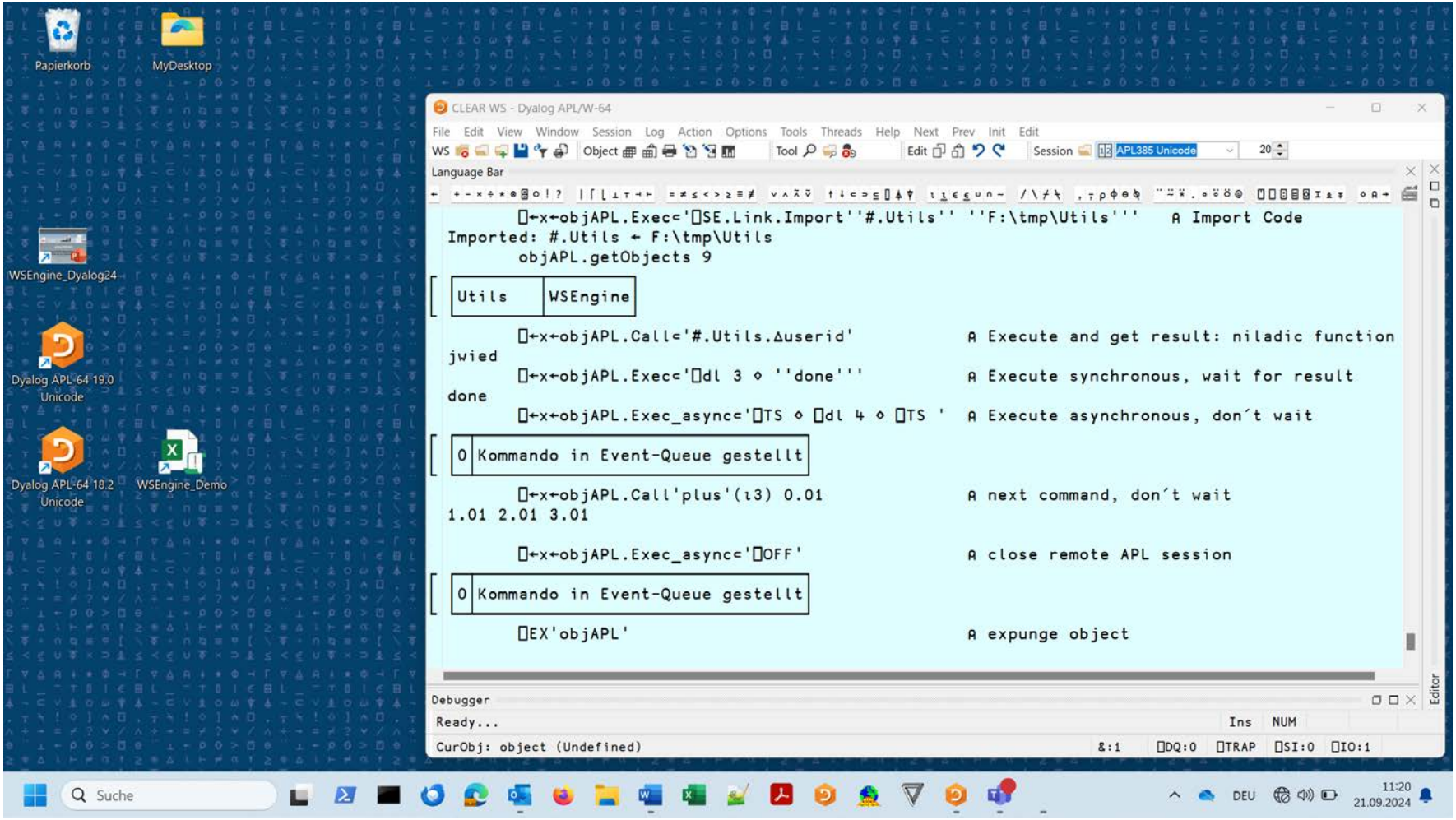
Demo: APL ↔ dyalog.WSEngine

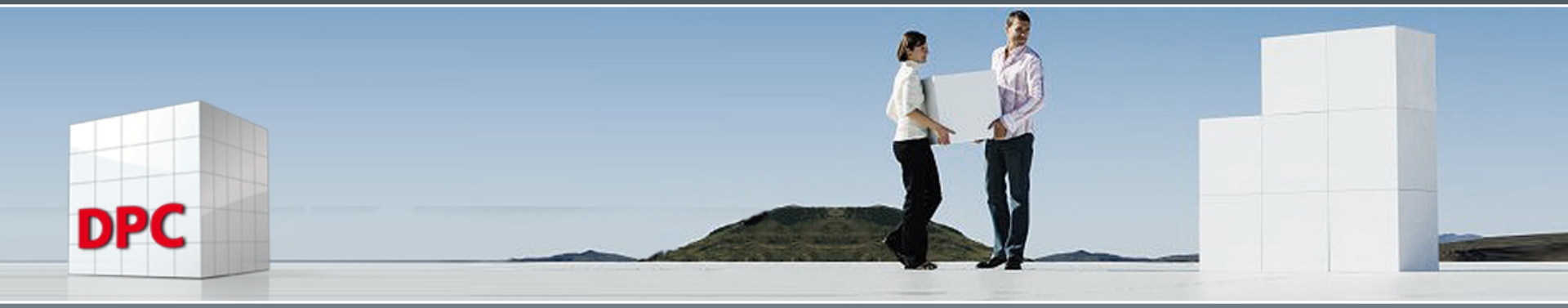


Demo: APL ↔ dyalog.WSEngine



Demo: APL ↔ dyalog.WSEngine





dyalog.WSEngine

Thank you for listening

Jürgen Wiedemann
DPC

Dyalog ´24
17. September 2024