



# XLNT<sup>®</sup> Command and Scripting Language



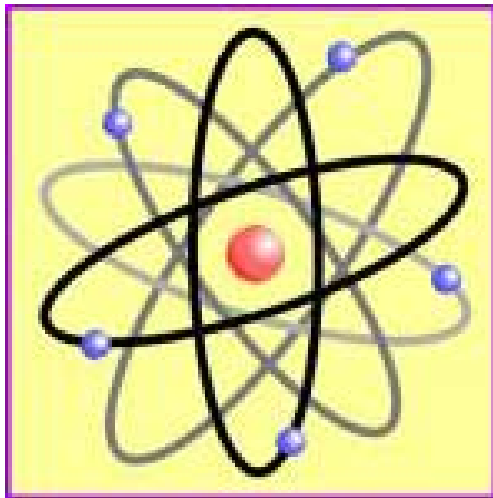
**XLNT Leverages System Administration for Microsoft Windows**



[www.advsyscon.com](http://www.advsyscon.com)



# Why a Scripting Language for Microsoft Windows?



“Although GUI tools make small tasks less intimidating for new system administrators, the tools don’t work well for enterprise and batch-oriented tasks. For example, retrieving a specific Registry value from hundreds of servers is **not a trivial** undertaking.”

Bob Wells, Windows NT Magazine



# Today's Situation

"All releases of Windows to date have needed a scripting languages to execute behind-the scenes operations like those you'd attempt with batch files."

The Cobb Group, "Exploring Windows NT"

- With the increase of Windows for Enterprise Computing, the job of administering them becomes significantly more difficult
- Whether it's doing one operation on many machines, or many operations on multiple objects. One thing is certain. GUIs just won't do!



# The Result: Many of today's Sys Admin tasks must be solved by scripting ....OR!!!!





# Differences between Command and Scripting Languages

- Command Language
  - Commands actually perform the desired tasks
  - Quicker Implementations
  - Less or no requirement to use freeware or shareware utilities to avoid having to write small programs or scripts to implement “missing” functionality
- Scripting Language
  - Usually Interpretative
  - Reusable Procedures
  - Reasonably Functional (varies by make/model)



# Languages

- Popular Scripting Languages
  - VBScript, Jscript
  - Perl, Rexx
- Popular Command Language
  - CMD (DOS)
- What if you had a solution that could augment your existing expertise in any of the above languages?



# The Advanced Language for Windows 2003 is XLNT®!

- XLNT is an Enterprise command and scripting language that accesses NT components to implement repetitive tasks of System Administration and/or Application Development, without reliance on traditional programming languages and development tools.



# XLNT - eXtended Language for Windows 2003

- Advanced Scripting and Command Language
- Designed and Built for Windows
- Extends the features of NT system components
- Allows for integration with other Scripting and Command Languages
- Completely "Network Aware"
- Easy to use



# Admin Tasks

- Provide a consistent, **automated logon** for a mixed Windows environment
  - Connect / map drives
  - Set up printers / shares that are transparent to the user
- Gather hardware/software information from **local or remote** machines
- Maintain Registry settings (local or remote)
- Perform Security operations on various objects
- Perform daily, weekly, or monthly backups of local or remote machines



# Admin Tasks

- Install Printers on Servers and Workstations
- **Schedule** administrative batch jobs to run daily, weekly, or monthly
- **Search** for a file document, driver, or program across a network Manage a corporate **website** running a webserver on Windows 2000/NT
  - Process user input --order forms, literature requests, feedback,
  - Eliminate editing HTML documents everyday



# Admin Tasks

- Purge TEMP and Internet Cache directories
- Limit users to a single domain logon
- Migrate Software Products or changes to other machines either using Microsoft's SMS more efficiently -or- avoiding "super" logon scripts



# XLNT - Common

- All Commands share a common, easy-to-understand syntax
- Parameters and Qualifiers (one or list)
- All Commands support output redirection
- Groups of commands share common qualifiers (i.e. /since, /before)
- Support of UNC specification applies to non-file related objects as well (i.e. Registry)



# XLNT - Commands

- Commands
  - Active Directory (CREATE, MODIFY, DELETE, SHOW)
  - File-Oriented (APPEND, COPY, RENAME, DIRECTORY, etc)
  - Services (INSTALL, START, STOP, etc)
  - Distributed File System
  - Printer Management (INSTALL, DELETE, DEFAULT, etc)
  - Shares
  - Security (User, Group, Policies)
  - Security (Object Permissions) (SET/SHOW PERMISSIONS)
  - Intersystem communication (TELNET like)
  - System Information



# Sample XLNT Commands

- Show System
- Show Process
- Set & Show Permissions
- Security
- Services
- Foreign Commands
- Share



# XLNT Command Examples

- Access the SAM:
  - SECURITY CREATE USER Njones /DOMAIN ...
- Access Windows Share Objects:
  - SHARE ADD NTSERV /PATH=C:\MYDRIVE /TYPE=DISK
  - DISCONNECT SESSION /USER=Njones
- Access File Objects
  - SEARCH \\COMPA\C\$\SOURCE\\*.cpp "#include"
  - SET PERMISSIONS C:\TEST /ACCOUNT=GUEST:READ
- Access Performance Data
  - SHOW SYSTEM /ON=COMPA



# XLNT - Network Aware

- 99%+ of XLNT's commands support the /ON qualifier...
  - This means that commands can be targeted to another machine for execution
  - Key: **You don't need XLNT on that machine!**
  - Benefit: Saves you time and money
- If you need to run scripts on other machines, get an XLNT Run-Time License!



# XLNT - Scripting

- Script Language
  - Variables (typeless or declarative, varying datatypes)
  - Text Operations
  - Built-In Functions
  - Operators
  - Procedures
    - Sequential text file that contains one or more XLNT or CMD/DOS commands, statements, functions, etc.



# Sample XLNT Procedure

## Add LPR Port

```
$ inquire fqdm "Enter name of server machine "  
$ inquire prtq "Enter name of printer on "fqdm"  
$ addcmdtmo=f$addregistry ("HKLM",  
    "SYSTEM\ControlSet001\Control\Print\Monitors\LPRPort\Ports\"fqdm\":\"prtq\ Timeouts\  
    CommandTimeout", "REG_DWORD",%x78)  
$ adddattmo=f$addregistry ("HKLM",  
    "SYSTEM\ControlSet001\Control\Print\Monitors\LPR Port\Ports\"fqdm\":\"prtq\ Timeouts\  
    DataTimeout", "REG_DWORD",%x12c)  
$ addprtq=f$addregistry ("HKLM", -  
    "SYSTEM\ControlSet001\Control\Print\Monitors\LPR Port\Ports\"fqdm\":\"prtq\ Printer  
    Name", "REG_SZ", ""prtq")  
$ addmac=f$addregistry ("HKLM",  
    "SYSTEM\ControlSet001\Control\Print\Monitors\LPRPort\Ports\"fqdm\":\"prtq\Server Name",  
    "REG_SZ", ""fqdm")  
$ exit
```



# XLNT Built-in Functions

- Built-in “Value-Added” XLNT Function
  - F\$function-name([args,...])
  - Allows for a LEXICAL function to be invoked wherever symbols or expressions are used.
- NT Component Lexicals
  - A=F\$LOOKUPREGISTRY ("HKEY\_CURRENT\_USER", "SOFTWARE\ASCII\MYKEY\VALUE-NAME",returned\_type, returned\_value)
  - Domain = F\$ENUMDOMAIN ()
- String Parsing Lexicals
  - P1 = “MyFileName.Doc”
  - FILENAME = F\$EXTRACT(0,F\$LOCATE(".",P1),P1)



# Some of XLNT's Functions

- F\$DIRECTORY - returns the current directory string
- F\$EDIT - performs string editing
- F\$ELEMENT - extracts an element from a string of elements
- F\$EXTRACT - extracts characters from a string
- F\$LENGTH - returns the length of a string
- F\$LOCATE - locates substrings
- F\$MODE - returns current command mode
- F\$PARSE - parses file specifications
- F\$PID - returns local/remote PID
- F\$READEVENT - read NT Event Log
- F\$REPORTEVENT - write NT Event Log
- F\$SEARCH - searches directories
- F\$TIME - returns current date and time as a string
- F\$LOCALTIMEZONE - returns information about the user's local time zone.
- F\$UTCTIME - returns UTC date and time, formatted in absolute time format.
- F\$FORMATDATE - returns a date string in various formats
- F\$FORMATTIME - returns a time string in various formats
- F\$GETDVIEX - extended form of the
- F\$GETDVI lexical, which obtains device information
- F\$INTEGER - returns the integer equivalent of the specified expression
- F\$STRING - returns the string equivalent of the specified expression
- F\$CVSI - converts specified bits of a character string to a signed integer
- F\$CVUI - converts the specified bits of a character string to an unsigned integer
- F\$LOADLIBRARY - loads a dynamic link library



# Additional XLNT's Functions

- F\$CHECKLIBRARY - determines if a specific DLL has been loaded
- F\$FREELIBRARY - releases a loaded DLL
- F\$ADDREGISTRY - adds a subkey and value to the registry
- F\$LOOKUPREGISTRY - looks up a key and values in the registry
- F\$DELETEREGISTRY - deletes a value from the registry
- F\$CHANGeregistry - modifies a value in the registry
- F\$FORMAT - formats a character string
- F\$MESSAGE - converts a status code to a formatted message string
- F\$MSGBOX - display Windows Dialog box
- F\$SERVICE\_STATUS - NT Service Info
- F\$GETVARIABLE - retrieves an environment variable
- F\$GETDVI - returns information about a specific device
- F\$GETSYI - returns system information
- F\$GETJPI - returns information about a specific process
- F\$ENUMDOMAIN - enumerates domain information
- F\$ENUMMACHINE - enumerates machine server information
- F\$ENUMSHAREPOINT - enumerates network share information
- F\$FILE\_ATTRIBUTES - returns information on specified file
- F\$TYPE - returns the datatype of a symbol



# XLNT -- ActiveX

- Windows Scripting Host (WSH) and ActiveX support (including IIS and WSH V2)
  - WSH is the Microsoft independent scripting host
  - XLNT supports integrated ActiveX and COM usage
  - XLNT supports WSH methods
  - XLNT can be directly invoked and manipulated (symbols, etc) from any program



# XLNT -- Developer

- Interactive Development Environment
  - Modeled after Developer Studio
  - Language Oriented Editor featuring use of color
  - Debugger
- XC “compile” command
  - Produces .EXE image
  - Image can't be modified
  - Provides source-level security of script (intellectual property)



# XLNT -- Interoperability

- CMD/DOS from XLNT
  - Transparent Foreign Command Support (user preference)
  - CMD/DOS Commands are supported within XLNT procedures
  - XLNT Error Handlers can be used with CMD/DOS commands
  - Invoke a CMD/DOS utility by file association
- XLNT from CMD/DOS
  - Add XLNT commands to existing CMD/DOS BAT scripts
  - Use XLNT's Extended Features while under CMD/DOS



# XLNT -- File Related

- All File Related Functions support
  - Local and Remote access
  - Standard Windows Security
  - Extensive wildcards, UNC and URL specifications:
    - c:\...\\*.dat
    - \\machine\c\$\test\...\\*.htm\*
    - ftp://ftp.advsyscon.com/xlnt386.exe
  - All Windows date formats and file date fields (Access, Create and Write)
  - Absolute, Delta & Combination times -- supporting various time formats



# File Deletion Script Comparison

- The Task:
  - To delete files older than the specified number of days from the specified device/directory
- Languages:
  - Visual Basic
  - XLNT



# Delete Files via Visual Basic

```
Attribute VB Name = "Module1"  
Declare Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long)  
Sub Main ()
```

```
DelOld.Show
```

```
On Error GoTo CommandError
```

```
i% = InStr(Command$, " ")  
Path$ = Left$(Command$, i% - 1)  
Days% = Val(Mid$(Command$, i% + 1, 99))
```

```
On Error GoTo 0
```

```
j% = 0  
Do  
    i% = j%  
    j% = InStr(i% + 1, Path$, "\")  
Loop While j% > 0
```

(continued on next slide)



# Visual Basic (cont'd.)

```
If i% <= 0 Then
    MsgBox "Could not find any backslash (" + Chr$(34) + "\" + Chr$(34) -
        + ") in the path. Aborting", vbCritical, "Error"
    GoTo Finish
End If

Directory$ = Left$(Path$, i% - 1)
FileWildcard$ = Mid$(Path$, i% + 1, 255)

DelOld.Print "Searching directory:"
DelOld.Print " " & Directory$
DelOld.Print "Searching files: " & FileWildcard$
DelOld.Print "Deleting files older than " & Days% & " days"

If Days% < 0 Then
    MsgBox "The number of days given in the second command line parameter was
        found to be " & Days%
        & ". However, it must be a positive number of days. Aborting", vbCritical,
        "Error"
    GoTo Finish
End If
```

(continued on next slide)



# Visual Basic (cont'd.)

```
If Mid$(Directory$, 2, 2) <> ":\" Then
    MsgBox "The path given in the first command line parameter must begin with a
        drive letter, a colon and a backslash. " & "Aborting", vbCritical, "Error"
    GoTo Finish
End If

On Error GoTo DriveError
ChDrive Directory$
On Error GoTo 0

ChDir Directory$
If CurDir$ <> Directory$ Then
    MsgBox "Could not change to directory/path: " & Directory$ & Chr$(13) & Chr$(10) &
        "Aborting", vbCritical, "Error"
    GoTo Finish
End If
```

(continued on Next Slide)



# Visual Basic (cont'd.)

```
FileName$ = Dir$(FileWildcard$)
```

```
DeletedFiles% = 0
```

```
If FileName$ <> "" Then
```

```
  Do
```

```
    If Not DelOld.Visible Then End
```

```
    DoEvents
```

```
    If Now - FileDateTime(FileName$) > Days% Then
```

```
      DelOld.Print "Deleting: " & FileName$
```

```
      DelOld.Refresh
```

```
      Kill FileName$
```

```
      DeletedFiles% = DeletedFiles% + 1
```

```
      DeletedFilesOnThisScreen% = DeletedFilesOnThisScreen% + 1
```

```
      If DelOld.CurrentY >= DelOld.Height - 1000 Then
```

(continued on next slide)



# Visual Basic (cont'd.)

```
Seconds% = 6 * DeletedFilesOnThisScreen%  
If Seconds% > 30 Then Seconds% = 30  
DelOld.Print "Waiting " & Seconds% & " Seconds..."  
DelOld.Refresh  
Sleep 1000# * Seconds%  
DelOld.Cls  
DeletedFilesOnThisScreen% = 0  
End If  
End If  
FileName$ = Dir$  
Loop While FileName$ <> ""  
End If  
  
DelOld.Print DeletedFiles% & " Files deleted. Normal program end."  
Seconds% = 6 * DeletedFilesOnThisScreen% (continued on Next Slide)
```



# Visual Basic (cont'd.)

```
If Seconds% > 30 Then Seconds% = 30
DelOld.Print "Waiting " & Seconds% & " Seconds..."
DelOld.Refresh
Sleep 1000# * Seconds%
GoTo Finish
```

```
CommandError:
Resume CommandError2
CommandError2:
If Command$ = "" Then
    MsgBox "Required command parameters not entered"
Else
    MsgBox "Found command line parameters"
End If
GoTo Finish
```

(continued on Next Slide)



# Visual Basic (cont'd.)

```
DriveError:
Resume DriveError2
DriveError2:
MsgBox "Could not change to drive " * Left$(Path$, 2) & Chr$(13) &
Chr$(10) &
    "Aborting", vbCritical, "Error"
GoTo Finish

Finish:
End
End Sub
```



# Delete Files via XLNT

```
$ delete/log/before=-'p1'- 'p2'\*.*  
$ exit
```

“p1” is the first parameter (days) and  
“p2” would be the directory specification



# XLNT - Functions

- User Defined Functions
  - Provides extensibility to the language
  - Allows inclusion of all Win32 and user-written API functions
  - Accessed through the DECLARE statement
  - Function Support
    - Ability to invoke any routine in any DLL
    - FUNCTION PROTOTYPE
    - DECLARE FUNCTION datatype name library [argument,...[options]]
  - Loading the DLL the function resides in
  - Invoking the Function



# Remote System Reboot

- The following is a script that reboots a Windows computer.
- This example, Reboot.xcp, shows how XLNT can directly call Win32 API functions.



# Remote System Reboot

```
$ NULL[0,8]=%x00
$ base = F$GETVAR("WINDIR")
$ INQUIRE COMP_NAME "Machine to reboot "
$ if comp_name .eqs. "" then abort "Machine-name must be specified."
$ INQUIRE SHUT_TIME "Seconds to shutdown [5] "
$ if shut_time .eqs. "" then shut_time="5"
$ stop_time=f$integer(shut_time)
$ inquire reboot_restart "Restart after Reboot [Y] "
$ reboot=1
$ if reboot_restart .eqs. "" then reboot_restart="T"
$ if .not. reboot_restart then reboot=0
$ DECLARE FUNCTION dword "InitiateSystemShutdownA" advapi32
  string,string,dword,dword,dword STDCALL
```

(continued on next slide)



# Remote System Reboot

```
$ VER32 = 0
$ IF .NOT. F$CHECKLIBRARY("base'\SYSTEM32\advapi32.dll")
$ THEN
$   VER32=F$LOADLIBRARY("""base'\SYSTEM32\advapi32.dll")
$   IF .NOT. VER32 THEN ABORT "$LOADLIBRARY failed for
advapi32.dll"
$ ENDIF
$ x = COMP_NAME+NULL
$ xm = "SHUT DOWN by Remote Site "$local_machine'
User:"$USERNAME"+NULL
$ REPLY = InitiateSystemShutdownA(x,xm,stop_time,1,reboot)
$ WRITE $STDOUT "The Action resulted in: "REPLY"
$ if VER32 then RESULT = F$FREELIBRARY(VER32)
$ EXIT
```



# Sample Windows-specific Command Groups



- User Management
- Object Security
- Service Management



# User Management

Control and Maintain Windows User, Group and Policy level security:

- SECURITY CREATE
- SECURITY MODIFY
- SECURITY DELETE
- SECURITY SHOW



# SECURITY SHOW

## \$ security show user test /domain /full

User Account from MY-DOMAIN  
User Name = TEST  
Full Name = John J. User  
Comment =  
Country Code = 0  
Account Active = Yes  
Account Locked = No  
Account Expires = Never  
Password Last Set = 7/22/2002 12:10:50 PM  
Password Expires = N/A  
Password Changed = 7/22/2002 12:10:50 PM  
Password Required = Yes  
User may change password = yes  
Workstations Allowed = All  
Home Directory = c:\test  
Profile Path =  
Logon Script =  
Last Logon = 2/17/2006 1:42:40 PM  
Logon Error Count = 0  
Logon Success Count = 0

```
$ security show user testuser2/domain/full
Windows NT 5.1 Security information for Picasso on 2/17/2006 1:43:19 PM

User Account from
User Name = TEST
Full Name = Test
Comment = ASCII - TEST - Test (No Mailbox)
Country Code = 0
Account Enabled = Yes
Account Locked = No
Account Expires = Never
Password Last Set = 7/22/2002 12:10:50 PM
Password Changed = 7/22/2002 12:10:50 PM
Password Expires = Never
Password Required = Yes
User may change password = Yes
Workstations Allowed = All
Home Directory =
Home Directory Drive =
Profile Path =
Logon Script =
Logon Server = Any Logon Server
Last Logon = 2/17/2006 1:42:40 PM
Last Logoff = Never
Logon Error Count = 0
Logon Success Count = 65535
Logon Hours Allowed = ALL
RAS Dialing Privilege = NO
Primary Global Group = Domain Users
Local Groups = None
Global Groups = Domain Users, Domain Admins
TEST has the following rights
-None

The following Global groups have rights:

Domain Users has the following rights
-None

Domain Admins has the following rights
-None
```



# Object Security

Control Access to Windows securable objects:

- Files
- Shares
- Registry Keys
- Printers
- Services

```
XLNT Session for User: XLNTUSER
$ show permissions "Test Printer" /object=printer
Permissions for Printer Objects on ORION 07-May-1998 15:37:05

Object : Test Printer
Owner  : NT-ASCII\xlntt1
        NT-BEIA\Domain Admins          Print <Print>
        NT-BEIA\Domain Guests         Print <Print>
        NT-BEIA\IUSR_THOR              Print <Print>
        NT-BEIA\bbordonaro             Print <Print>
        CREATOR OWNER                  Manage Documents <Manage>
        Everyone                       Print <Print>
        Administrators                  Full Control <Full>
        Power Users                     Full Control <Full>
        NT-ASCII\xlntt1                 Full Control <Full>
$
```



# SET PERMISSIONS Command

Allows object ownership and permissions to be granted or revoked for any securable object.

```
$ set permissions c:\test /account=jjones:full:full
$ set permissions/revoke c:\test /account=everyone
$ set permissions/object=printer \\server\printer17 -
    /account=(administrators:full, everyone:print)
$ set permissions/object=regkey
\\server\HKLM\Testkey\Test
    /account=(customers:read, everyone:noaccess)
```



# SET PERMISSIONS on Cluster

Allows object permissions to be changed based on a textual SID.

```
$ set permissions c:\test /account=(S-1-1-1-0:full)
```

This is especially important for Cluster operations since today the resources must be failed over to set proper user/group trustees. XLNT avoids this requirement!



# SHOW PERMISSIONS

Displays permissions currently set on the designated object

**\$ show permissions/object=printer Printer17**

Permissions for Printer Objects on ABC 3-Jun-1999 12:43:02

Object: Printer17

Owner: Administrators

CREATOR OWNER

Everyone

Administrators

Power Users

Manage Documents (Manage)

Print (Print)

Full Control (Full)

Full Control (Full)



# Service Management

## Control and Maintain Windows Services:

- INSTALL SERVICE
- START SERVICE
- SHOW SERVICE
- CONFIGURE SERVICE
- PAUSE SERVICE
- RESET SERVICE
- RESUME SERVICE
- STOP SERVICE
- REMOVE SERVICE



# Sample XLNT Application

- Windows Desktop Logon
  - Use XLNT to create a global logon script for all workstation users
  - Map an XLNT procedure to an executable image
  - Allows someone to track all workstation logons for a domain
- The following is a Logon Script using XLNT



# Logon Script

```
$ if f$userinfo  
    ($username,,,"IFMEMBER","PAYROLL")  
$ then  
$     net use o: \\server\payrollshare  
$ endif  
$ exit
```



# XLNT and ActiveX Scripting

- Unique ability to combine commands and ActiveX object manipulation
- Two procedures on Web Site:
  - Populate Excel spreadsheet from SAM
  - Populate SAM from Excel spreadsheet
- Exchange Scripting Agent Support



# XLNT - Product Editions

- XLNT Professional Edition
  - IDE Script Editor and Debugger
  - Script EXE Generator
- XLNT Standard Edition
- XLNT Enterprise Run-Time Edition
  - Requires Professional Edition
  - Supports .EXE Scripts Only



# XLNT<sup>®</sup> from Advanced Systems Concepts

- Operating Environments
  - Windows
  - Supported on:
    - Intel & Intel based
- Microsoft Logo Certified
  - BackOffice
  - Windows NT/9x



Designed for



Microsoft<sup>®</sup>  
Windows NT<sup>®</sup>  
Windows<sup>®</sup>95





# Questions



E-mail: [support@advsyscon.com](mailto:support@advsyscon.com)

[sales@advsyscon.com](mailto:sales@advsyscon.com)

US & Canada 800-229-2724 ext. 180

International 973-539-2660 ext. 180