

**noNg yesyesyesyesWin32 Network Help
FileTRUEWin32netyesyes10/11/98**

[About WIL Extenders](#)
[Using WIL Extenders](#)
[Compiling with WIL Extenders](#)

[Contacting Wilson WindowWare](#)
[How to get Technical Support](#)
[About this Help File](#)

[AddExtender\(filename\)](#)
[LastError\(\)](#)
[Net101](#)
[NetInfo\(requestcode\)](#)

[Win32 Network Errors](#)

[netAddDrive\(user-id, pswd, net-resource, local drive, persist\)](#)
[netAddPrinter\(user-id, pswd, net-resource, local device, persist\)](#)
[netCancelCon\(local drive, persist, forceflag\)](#)
[netDirDialog\(flag\)](#)
[netGetCon\(local drive\)](#)
[netGetUser\(netname\)](#)
[netResources\(net-resource, scope, type, usage\)](#)
[netVersion\(\)](#)

[wntAccessAdd\(server-name, resource/share-name, user-name, share-type, access-string\)](#)
[wntAccessDel\(server-name, resource/share-name, user-name, share-type\)](#)
[wntAccessGet\(server-name, resource/share-name, user-name, object-type\)](#)
[wntAccessList](#)
[wntRunAsUser](#)

[wntAddDrive\(user-id, pswd, net-resource, local drive, persist\)](#)
[wntAddPrinter\(user-id, pswd, net-resource, local device, persist\)](#)
[wntCancelCon\(local drive, persist, forceflag\)](#)
[wntChgPswd\(server/domain, user-name, old-password, new-password\)](#)
[wntDirDialog\(flag\)](#)
[wntFileClose](#)
[wntGetCon\(local drive\)](#)
[wntGetUser\(netname\)](#)
[wntGroupInfo\(server-name, group, group-type, request\)](#)
[wntListGroupes\(server-name, group-type\)](#)
[wntMemberDel\(server-name, group-name, user-name, group-type\)](#)
[wntMemberGet\(server-name, group-name, user-name, group-type\)](#)
[wntMemberGrps\(server-name, user-name, group-type, flags\)](#)
[wntMemberList\(server-name, group-name, group-type\)](#)
[wntMemberSet\(server-name, group-name, user-name, group-type\)](#)
[wntOwnerGet\(server-name, reg-key, resource-name, object-type, flag\)](#)
[wntOwnerSet\(server-name, reg-key, resource-name, object-type, user/group name\)](#)
[wntRasUserGet\(server-name, user-name, request\)](#)
[wntRasUserSet\(server-name, user-name, privilege, phone-number\)](#)
[wntResources\(net-resource, scope, type, usage\)](#)
[wntResources2\(net-resource, scope, type, usage, provider\)](#)
[wntServerList\(server-name, domain-name, server-type\)](#)
[wntServerType\(server-name\)](#)

wntServiceAt(server, domain, server-type, service-name, flags)
wntServiceInf (server-name)
wntShareAdd(server-name, resource, share-name, share-type, max-users)
wntShareDel(server-name, resource/share-name, share-type)
wntShareInfo(server-name, resource/share-name, share-type, request)
wntShareSet(server-name, resource/share-name, share-type, comment, description)
wntSvcControl(server-name, service-name, flags, control-code)
wntSvcStart(server, service-name, flags, params, delimiter)
wntSvcStatus(server, service-name, flags, request)
wntUserAdd(server-name)
wntUserAddDat(element, value)
wntUserDel(server-name, user-name)
wntUserGetDat(server-name, user-name, element)
wntUserInfo(request)
wntUserProps
wntUserRename
wntUserSetDat
wntVersion()

w9xAccessAdd
w9xAccessDel
w9xAccessGet
w9xAccessList
w9xListGroups
w9xMemberDel
w9xMemberGet
w9xMemberGrps
w9xMemberList
w9xMemberSet
w9xServiceAt
w9xServerList(server, domain, server-type)
w9xShareAdd
w9xShareDel
w9xShareInfo
w9xShareSet
w9xUserInfo
w9xUserProps
w9xVersion

w95AccessDel(server-name, resource, user-name)
w95AccessAdd(server-name, resource, user-name, access-rights, flags)
w95AddDrive(user-id, pswd, net-resource, local-drive, persist)
w95AddPrinter(user-id, pswd, net-resource, local device, persist)
w95CancelCon(local drive, persist, forceflag)
w95DirDialog(flag)
w95FileClose
w95GetCon(local name)
w95GetUser(netname)
w95Resources(net-resource, scope, type, usage)
w95ServerType
w95ServiceAt
w95ServiceInf
w95ShareAdd(server-name, resource, share-name, share-type, flags)
w95ShareInfo(server-name, share-name, request)
w95ShareSet(server-name, share-name, comment, full-password, read-password)
w95ShareDel(server-name, share-name)
w95Version()

Win32 Basic Functions

[Windows NT Functions](#)

[Windows 95 Functions](#)

[Windows 95/RADMIN Functions](#)

[Win32 Network Extenders](#)

[Suggested Approaches](#)

[Determining your platform](#)

Win32 Network Extender Help File



⁹⁵
functionio
windows

Windows 95

For use on Windows 95/98 workstations. Can control Windows 95/98 servers.

^{NT}
functions
windows

Windows NT

For use on Windows NT workstations. Can control Windows NT servers.

These Windows Interface Language Network extenders provide standard support for computers running 32 bit versions of Windows, such as Windows NT and Windows 95/98. They may be used in conjunction with other 32 bit Intel extenders.

^{9X}
functionio
windows

Windows 95/RADMIN (9X)

For use on Windows 95/98 workstations. Can control Windows NT servers. May be used in addition to the standard Windows 95/98 extender.

^{Win32}
functions
basic

Win32 Basic functions

General feature-limited extender. For backwards compatibility the functions are still supported.

The Win32 Network extenders

have been re-designed for the specific 32 bit operating systems. This might result in some minor changes, depending on how you choose to proceed with new scripts.

There are now separate extenders and additional functions for the following:

Windows 95

For use on Windows 95/98 workstations. Can control Windows 95/98 servers.

Note: All the functions have kept the same naming convention of w95[], however, they are also designed to run on Windows 98.

Windows NT

For use on Windows NT workstations. Can control Windows NT servers.

Windows 95/RADMIN (9X)

For use on Windows 95/98 workstations. Can control Windows NT servers. May be used in addition to the standard Windows 95/98 extender.

Note: In order to use the functions in this extender, you must have an NT domain server, and you must have RADMIN32.DLL and RLOCAL32.DLL present. These DLL's can be found in the Microsoft Windows 95 Service Pack 1. (location on disk admin\tools\servmgmt\)

Win32 Basic functions

General feature-limited extender. For backwards compatibility the functions are still supported.

About WIL Extenders

Using WIL Extenders

Compiling with WIL Extenders

Win32 Network Extenders

Suggested Approaches

Determining your platform

Errors

Contacting Wilson WindowWare

How to get Technical Support

About this Help File

[These functions can be used within WIL scripts or can be compiled into WIL executables](#)

Technical Support

is available for registered users. If you can't find what you're looking for, or you're having problems with your WIL scripts, be sure to look at the [suggested approaches](#) section, which will answer questions on the best direction to proceed in your development of 32 bit Networking scripts.

About WIL Extenders

WIL extender DLLs are special DLLs designed to extend the built-in function set of the WIL processor. These DLLs typically add functions not provided in the basic WIL set, such as network commands for particular networks (Novell, Windows for WorkGroups, LAN Manager and others), MAPI, TAPI, and other important Application Program Interface functions as may be defined by the various players in the computer industry from time to time. These DLLs may also include custom built function libraries either by the original authors, or by independent third party developers. (An Extender SDK is available). Custom extender DLLs may add nearly any sort of function to the WIL language, from the mundane network math or database extensions, to items that can control fancy peripherals, including laboratory or manufacturing equipment.

WIL extenders must be installed separately. Up to 10 extender DLLs may be added. The total number of added items may not exceed 200 functions and constants. The

AddExtender

function must be executed before attempting to use any functions in the extender library. The AddExtender function should be only executed once in each WIL script that requires it.

- [Using WIL Extenders](#)
- [Compiling with WIL Extenders](#)

Using WIL Extenders

Accessing the additional functionality available in WIL Extender DLLs is simple. At the top of each script in which WIL Extender commands are to be used add the appropriate extender with the [AddExtender](#) command.

```
AddExtender(extender filename)
```

The WIL interpreter will search for the the extender DLL's. If no path is specified in the **AddExtender** statement, the WIL interpreter will search the current directory, the windows directory and on directories on the path. The extender DLL's must be available or the **AddExtender** line will return an error. In general, when you run a large exe with embedded extenders, it will extract the extenders to the same directory the compiled exe is in.

The **AddExtender** function should only be executed once for each extender dll in each WIL script that requires it. Per script you can add up to 10 extender DLLs or a combined total of 200 functions.

For example to use both of the Win32 Network Extenders, two AddExtender lines would appear in the script.

```
AddExtender("WWW9532I.DLL") ;for Windows 95/98  
AddExtender("WWWNT32I.DLL") ;for Windows NT
```

- [About WIL Extenders](#)
- [Compiling with WIL Extenders](#)

Compiling with WIL Extenders

The WinBatch+Compiler has two options for compiling scripts into executables, Large EXE for Standalone PC's and Small EXE for Networked PC's. When any extender functions are used in a script, the corresponding extender must be compiled into the executable, or placed where the executable can access it.

- [About WIL Extenders](#)

- [Using WIL Extenders](#)

The Large Standalone EXE option of the Compiler has an additional button. The EXTENDERS button displays a list of extenders which can be chosen and compiled into a Standalone EXE option. More than one extender may be chosen. When a Standalone EXE is launched on a PC it looks for the necessary DLL's in the current directory, on the path and in the Windows directory. If the DLL's are not found, they are automatically written into the current directory. If for some reason, they cannot be written to that directory (perhaps the directory is set to be Read Only), the large compiled file will not run.

The DLLs can also be copied into a directory on a computers PATH and the compiled EXE will find them there and run. The Compiler has a Small EXE for Networked PC's option that takes advantage of this.

The DLLs need to be placed on the PATH only once. Subsequent EXE files installed on this same machine can be compiled under the Small EXE option.

It is generally recommended to use the filename without including the path in an AddExtender statement. However, if it is preferable to point to the current directory, (directory in which the WIL executable resides), it can be done.

```
AddExtender( StrCat( DirHome( ), "wnn3x32i.dll" ))
```

Contacting Wilson WindowWare

Wilson WindowWare, Inc.
5421 California Ave. SW
Seattle, WA 98136 USA

Orders: (800) 762-8383

Voice: (206) 938-1740

Fax: (206) 935-7129

Email: info@windowware.com

Registered users of our software get manuals, technical support, use of Wilson WindowWare on-line information services, and special offers on new versions of Wilson WindowWare products.

- [Technical Support](#)
- [About WIL Extenders](#)

How to get Technical Support

The Wilson WindowWare website is an excellent technical resource. Access to the entire Technical Support Database is at your fingertips. In the Technical Support area use the keyword search to find answers to common problems, alternate scripting methods, and sample code. Or join the Wilson WindowWare Web BBS, a new Web forum. The BBS provides an outlet for registered users to share their experiences with other users.

See the information on registering your copy (found in the WinBatch.hlp or the WinEdit.hlp files) if you haven't done so yet.

The latest versions of our software are available on-line. The places here may change at any time -- check your installation sheet for the most recent addresses.

Internet Web page: <http://www.windowware.com>

Internet Technical Support Articles & Web BBS:
<http://techsupt.windowware.com>

Internet FTP: <ftp.windowware.com> in [/wwwftp/wilson](http://wwwftp/wilson)

- [Contacting Wilson WindowWare](#)
- [About WIL Extenders](#)

About this Help File

This extender adds certain network capability to the Windows Interface Language (WIL) processing engine. Please refer to the **WIL Reference Manual** for an introduction to WIL, as well as for complete documentation of the many functions available in WIL and the programs that use it. This help file includes only topics and functions which are exclusive to this particular WIL Extender.

- [About WIL Extenders](#)
- [Using WIL Extenders](#)
- [Compiling with WIL Extenders](#)

Notational Conventions

Throughout this manual, we use the following conventions to distinguish elements of text:

ALL-CAPS

Used for filenames.

Boldface

Used for important points, programs, function names, and parts of syntax that must appear as shown.

`system`

Used for items in menus and dialogs, as they appear to the user.

`Small fixed-width`

Used for WIL sample code.

Italics

Used for emphasis, and to liven up the documentation just a bit.

Acknowledgments

This network extender developed by Morrie Wilson and Richard Merit.

Documentation written by Tina Browning, and Deana Dahley.

Win32 Network Extenders

These Windows Interface Language Network extenders provide standard support for computers running 32 bit versions of Windows, such as Windows NT and Windows 95/98.

The Windows 32 Network extender (WWNET32I.DLL) has been split into separate extenders for Windows 95/98 (WWW9532I.DLL) and Windows NT (WWWNT32I.DLL).

For Windows 95/98 use...

AddExtender("WWW9532I.DLL")

Other required DLL's: none

For Windows 95/98 to NT server use

AddExtender("WWW9X32I.DLL")

Other required DLL's: RADMIN32.DLL and RLOCAL.DLL

For Windows NT use...

AddExtender("WWWNT32I.DLL")

Other required DLL's: none

The WWNET32I.DLL, Win32 Basic Extender DLL, which supports the existing net[.] functions will still be distributed for backwards compatibility.

However, this DLL is a "wrapper" or "shell" which calls the appropriate platform-specific extender and must be used in combination with WWW9532I.DLL or WWWNT32I.DLL. One of these DLL's (depending on the operating system being used) will need to be available, either in the current directory, in the same directory as WWNET32I.DLL, or in a directory on the path.

The Win32 Basic extender remains the same.

AddExtender("WWNET32I.DLL")

This extender requires either WWW9532I.DLL or WWWNT32I.DLL, depending on the operating system.

The existing net[.] functions have been renamed to w95[.] and wnt[.], respectively. Their parameters have not changed.

- [Windows 95 Functions](#)

- [Windows 9x Functions](#)

- [Windows NT Functions](#)

- [Win32 Basic Functions](#)

- [Determining your platform](#)

- [About WIL Extenders](#)

- [Using WIL Extenders](#)

- [Compiling with WIL Extenders](#)

netAddDrive
netAddPrinter
netCancelCon
netDirDialog
netGetCon
netGetUser
netResources
netVersion

w95AccessDel
w95AccessAdd
w95AddDrive
w95AddPrinter
w95CancelCon
w95DirDialog
w95FileClose
w95GetCon
w95GetUser
w95Resources
w95ServiceAt
w95ServiceInf
w95ServerType
w95ShareAdd
w95ShareInfo
w95ShareSet
w95ShareDel
w95Version

wntAccessAdd
wntAccessDel
wntAccessGet
wntAccessList
wntAddDrive
wntAddPrinter
wntCancelCon
wntChgPswd
wntDirDialog
wntFileClose
wntGetCon
wntGetUser
wntGroupInfo
wntListGroup
wntMemberDel
wntMemberGet
wntMemberGrps
wntMemberList
wntMemberSet
wntOwnerGet
wntOwnerSet
wntRasUserGet
wntRasUserSet
wntResources
wntResources2
wntRunAsUser
wntServerList
wntServiceAt
wntServiceInf
wntShareAdd
wntShareDel
wntShareInfo
wntShareSet
wntSvcControl
wntSvcStart
wntSvcStatus
wntUserAdd
wntUserAddDat
wntUserDel
wntUserGetDat
wntUserInfo
wntUserProps
wntuserRename
wntUserSetDat
wntVersion

[How do these changes affect my scripts?](#)

Suggested Approaches

The additional extenders add a slight twist to the mix. Here are our suggestions for how to proceed.

New Scripts

If you're writing a new script, use a new platform specific extender with the new function names, **w95AddDrive** or **wntAddDrive** instead of **NetAddDrive**.

```
AddExtender("WWW9532I.DLL")  
for Windows 95/98
```

or...

```
AddExtender("WWWNT32I.DLL")  
for Windows NT.
```

or...

```
AddExtender("WWW9X32I.DLL")  
For Windows 95/98 and NT Servers.
```

If you don't know which platform you will be running on, you can add code to your script to [determine the platform](#) and conditionally execute the corresponding functions. On the other hand, you might choose to use the generic functions in the Win32 Basic Extender. In which case, all three of the Win32 extenders will need to be provided.

Existing scripts

Scripts which contain old style **net[]** functions will require special consideration.

If you're working with uncompiled WIL scripts, you have two options.

1

Replace the first three characters of the net[] functions with w95 or wnt and use the new platform specific extender. The function parameters are the same and will be unaffected.

NetAddDrive becomes either **w95AddDrive** or **wntAddDrive**.

- [Windows 95/98 Functions](#)

- [Windows 9x Functions](#)

- [Windows NT Functions](#)

- [Win32 Basic Functions](#)

- [Determining your platform](#)

- [About WIL Extenders](#)

- [Using WIL Extenders](#)

- [Compiling with WIL Extenders](#)

2

If you prefer to use the old net[] functions, **NetAddDrive** etc., they are still supported by the Win32 Basic extender, WWNET32I.DLL. As long as this extender is available, old functions like **netAddDrive** and **netCancelCon** will work exactly as they did before.

The only catch is...

You must remember to include the platform specific extender.

"WWW9532I.DLL" for Windows 95/98

"WWWNT32I.DLL" for Windows NT

Compiled Exe's

Scripts compiled with an older version of the WWNET32I.DLL will still work as long as you do not replace the dll.

For new executables, the extenders can be selected as usual from the Extender button in the Large for Standalone Exe's option or placed on the path or search drive for Small Exe's.

As usual, when a Large executable is launched it will look for copies of the DLL's. If an older version of WWNET32I.DLL is found in the current directory, on the path or in the Windows directory, it will not be replaced. This will not affect new scripts using old net[] commands.

However, if the Win32 Basic extender dll is manually replaced with the new version, old scripts will fail. Remember, the new WWNET32I.DLL points to the Windows 95/98 or Windows NT Extender DLL. It can no longer run on its own.

Determining your platform

If you need to write scripts for both Windows NT and Windows 95/98, your script can decide on the fly which platform you are using.

You'll need to have two sections of code. One with wnt[] functions and one with w95[] functions. The functions differ only in the first three characters. For example, **wntAddDrive** has the same exact parameters as **w95AddDrive**. Simply copy your code and replace the first three characters.

When you compile and distribute the code, remember to include both the Windows 95/98 Network extender, WWW9532I.DLL and the Windows NT extender, WWWNT32I.DLL.

- [Windows 95/98 Functions](#)

- [Windows 9x Functions](#)

- [Windows NT Functions](#)

- [Win32 Basic Functions](#)

- [About WIL Extenders](#)

- [Using WIL Extenders](#)

- [Compiling with WIL Extenders](#)

Example:

```
;Verify version of Windows so that we know what we are doing
;Returns from WinMetrics
;0=??? 1=Win 2=Win4Wkg 3=Win32S 4=WinNT 5=Win95
;;
;;
platform=WinMetrics(-4)
if platform == 4
  gosub WinNTdo
else
  if platform == 5
    gosub Win95do
  else
    message("oops", "WinMetrics returning %platform%")
  endif
endif
exit

:WinNTdo
AddExtender("WWWNT32I.DLL")

return
```

```
:Win95do  
AddExtender("WWW9532I.DLL")  
  
return
```

Win32 basic functions

The following WIL functions are useful when using Network extenders.

[AddExtender\(filename\)](#)
[LastError\(\)](#)
[Net101](#)
[NetInfo\(requestcode\)](#)

The following generic functions for Windows 95/98 and NT networks can be added with the WWNET32I.DLL.

Additionally, the platform specific extender DLLs (WWW9532I.DLL for Windows 95/98 and WWWNT32I.DLL for Windows NT) are required.

For Windows 95/98 workstations accessing an NT Server use the 9x extender WWW9X32I.DLL. Requires additional DLL's: RADMIN.DLL and RLOCAL.DLL. These DLL's can be found in the Microsoft Windows 95 Service Pack 1.

[netAddDrive\(user-id, pswd, net-resource, local drive, persist\)](#)
[netAddPrinter\(user-id, pswd, net-resource, local device, persist\)](#)
[netCancelCon\(local drive, persist, forceflag\)](#)
[netDirDialog\(flag\)](#)
[netGetCon\(local drive\)](#)
[netGetUser\(netname\)](#)
[netResources\(net-resource, scope, type, usage\)](#)
[netVersion\(\)](#)

• [Windows 95/98 Functions](#)

• [Windows 9x Functions](#)

• [Windows NT Functions](#)

• [Errors](#)

• [Using WIL Extenders](#)

• [Compiling with WIL Extenders](#)

windows 95 functions

The following WIL functions are useful when using Network extenders.

[AddExtender\(filename\)](#)
[LastError\(\)](#)
[Net101](#)
[NetInfo\(requestcode\)](#)

- [Win32 Basic Functions](#)
- [Windows NT Functions](#)
- [Windows 9x Functions](#)
- [Errors](#)
- [Using WIL Extenders](#)
- [Compiling with WIL Extenders](#)

The following functions for Windows 95 & Windows 98 networking can be added with the WWW9532I.DLL.

[w95AccessAdd\(server-name, resource, user-name, access-rights, flags\)](#)
[w95AccessDel\(server-name, resource, user-name\)](#)
[w95AddDrive\(user-id, pswd, net-resource, local-drive, persist\)](#)
[w95AddPrinter\(user-id, pswd, net-resource, local device, persist\)](#)
[w95CancelCon\(local drive, persist, forceflag\)](#)
[w95DirDialog\(flag\)](#)
[w95FileClose\(Server-name, path-name\)](#)
[w95GetCon\(local name\)](#)
[w95GetUser\(netname\)](#)
[w95Resources\(net-resource, scope, type, usage\)](#)
[w95ServerType\(server-name\)](#)
[w95ServiceAt\(server, domain, server-type, service-name, flags\)](#)
[w95ServiceInf\(server-name\)](#)
[w95ShareAdd\(server-name, resource, share-name, share-type, flags\)](#)
[w95ShareInfo\(server-name, share-name, request\)](#)
[w95ShareSet\(server-name, share-name, comment, full-password, read-password\)](#)
[w95ShareDel\(server-name, share-name\)](#)
[w95Version\(\)](#)

windows NT functions

The following WIL functions are useful when using Network extenders.

[AddExtender\(filename\)](#)

[LastError\(\)](#)

[Net101](#)

[NetInfo\(requestcode\)](#)

• [Win32 Basic Functions](#)

• [Windows 95/98 Functions](#)

• [Windows 9x Functions](#)

• [Errors](#)

• [Using WIL Extenders](#)

• [Compiling with WIL Extenders](#)

The following functions for Windows NT networking can be added with the WWWNT32I.DLL.

[wntAccessAdd\(server-name, resource/share-name, user-name, share-type, access-string\)](#)

[wntAccessDel\(server-name, resource/share-name, user-name, share-type\)](#)

[wntAccessGet\(server-name, resource/share-name, user-name, object-type\)](#)

[wntAccessList\(server-name, resource/share-name, object-type, flags\)](#)

[wntAddDrive\(user-id, pswd, net-resource, local drive, persist\)](#)

[wntAddPrinter\(user-id, pswd, net-resource, local device, persist\)](#)

[wntCancelCon\(local drive, persist, forceflag\)](#)

[wntChgPswd\(server/domain, user-name, old-password, new-password\)](#)

[wntDirDialog\(flag\)](#)

[wntFileClose\(server-name, file-pathname\)](#)

[wntGetCon\(local drive\)](#)

[wntGetUser\(netname\)](#)

[wntGroupInfo\(server-name, group, group-type, request\)](#)

[wntListGroup\(server-name, group-type\)](#)

[wntMemberDel\(server-name, group-name, user-name, group-type\)](#)

[wntMemberGet\(server-name, group-name, user-name, group-type\)](#)

[wntMemberGrps\(server-name, user-name, group-type, flags\)](#)

[wntMemberList\(server-name, group-name, group-type\)](#)

[wntMemberSet\(server-name, group-name, user-name, group-type\)](#)

[wntOwnerGet\(server-name, reg-key, resource-name, object-type, flag\)](#)

[wntOwnerSet\(server-name, reg-key, resource-name, object-type, user/group name\)](#)

[wntRasUserGet\(server-name, user-name, request\)](#)

[wntRasUserSet\(server-name, user-name, privilege, phone-number\)](#)

[wntResources\(net-resource, scope, type, usage\)](#)

[wntResources2](#)(net-resource, scope, type, usage, provider)
[wntRunAsUser](#)(domain/server, user-name, password, login-type, flags)
[wntServerList](#)(server-name, domain-name, server-type)
[wntServerType](#)(server-name)
[wntServiceAt](#)(server, domain, server-type, service-name, flags)
[wntServiceInf](#)(server-name)
[wntShareAdd](#)(server-name, resource, share-name, share-type, max-users)
[wntShareDel](#)(server-name, resource/share-name, share-type)
[wntShareInfo](#)(server-name, resource/share-name, share-type, request)
[wntShareSet](#)(server-name, resource/share-name, share-type, comment, description)
[wntSvcControl](#)(server, service-name, flags, control-code)
[wntSvcStart](#)(server, service-name, flags, params, delimiter)
[wntSvcStatus](#)(server, service-name, flags, request)
[wntUserAdd](#)(server-name)
[wntUserAddDat](#)(element, value)
[wntUserDel](#)(server-name, user-name)
[wntUserGetDat](#)(server-name, user-name, element)
[wntUserInfo](#)(request)
[wntUserProps](#)(server-name, user-name, request)
[wntUserSetDat](#)(server-name, user-name, element, value)
[wntUserRename](#)(server-name, old-username, new-username)
[wntVersion](#)()

windows 9X functions

The following WIL functions are useful when using Network extenders.

[AddExtender\(filename\)](#)

[LastError\(\)](#)

[Net101](#)

[NetInfo\(requestcode\)](#)

▪ [Win32 Basic Functions](#)

▪ [Windows 95/98 Functions](#)

▪ [Errors](#)

▪ [Using WIL Extenders](#)

▪ [Compiling with WIL Extenders](#)

The following functions for Windows NT networking can be added with the WWW9x321.DLL.

[w9xAccessAdd\(server-name, resource/share-name, user/group name, object-type, access-string\)](#)

[w9xAccessDel\(server-name, resource/share-name, user/group name, object-type\)](#)

[w9xAccessGet\(server-name, resource/share-name, user/group name, object-type\)](#)

[w9xAccessList\(server-name, resource/share-name, object-type, flags\)](#)

[w9xListGroup\(s\) \(server-name, group-type\)](#)

[w9xMemberDel\(server-name, group-name, user-name, group-type\)](#)

[w9xMemberGet\(server-name, group-name, user-name, group-type\)](#)

[w9xMemberGrps\(server-name, user-name, group-type, flags\)](#)

[w9xMemberList\(server-name, group-name, group-type\)](#)

[w9xMemberSet\(server-name, group-name, user-name, group-type\)](#)

[w9xServiceAt\(server, domain, server-type, service-name, flags\)](#)

[w9xServerList\(server, domain, server-type\)](#)

[w9xShareAdd\(server-name, resource, share-name, share-type, max-users\)](#)

[w9xShareDel\(server-name, resource/share-name, share-type\)](#)

[w9xShareInfo\(server-name, resource/share-name, share-type, request\)](#)

[w9xShareSet\(server-name, resource/share-name, share-type, comment, s:location\)](#)

[w9xUserInfo\(request\)](#)

[w9xUserProps\(server-name, user-name, request\)](#)

[w9xVersion\(\)](#)

Win32 Network Errors

- "185: Bad name for local drive or printer"
- "499: Unrecognized network error #"
- "500: Access is denied."
- "501: LocalName is already connected. "
- "502: Device and resource do not match. "
- "503: LocalName is invalid. "
- "504: ResourceName is not valid or cannot be located. "
- "505: The user profile is in an incorrect format. "
- "506: Unable to open the user profile to process persistent connections. "
- "507: LocalName is already in the user profile. "
- "508: Password is invalid. "
- "509: Network component is not started or invalid "
- "510: The network is not present. "
- "511: Device in use"
- "512: Device not currently connected"
- "513: Open files on device and FORCE=@FALSE"
- "514: Device not currently available"
- "515: Invalid Password"
- "516: Insufficient memory."
- "517: Not supported in current NT version (Invalid Parameter)"

AddExtender(filename)

Installs a WIL extender DLL.

Syntax:

AddExtender(filename)

Parameters:

(s) filename WIL extender DLL filename.

Returns:

(i) **@TRUE** if function succeeded.
 @FALSE if function failed.

WIL extender DLLs are special DLLs designed to extend the built-in function set of the WIL processor. These DLLs typically add functions not provided in the basic WIL set, such as network commands for particular networks (Novell, Windows for WorkGroups, LAN Manager and others), MAPI, TAPI, and other important Application Program Interface functions as may be defined by the various players in the computer industry from time to time. These DLLs may also include custom built function libraries either by the original authors, or by independent third party developers. (An Extender SDK is available). Custom extender DLLs may add nearly any sort of function to the WIL language, from the mundane network, math or database extensions, to items that can control fancy peripherals, including laboratory or manufacturing equipment.

Use this function to install extender DLLs as required. Up to 10 extender DLLs may be added. The total number of added items may not exceed 200 functions and constants. The **AddExtender** function must be executed before attempting to use any functions in the extender library. The **AddExtender** function should be only executed once in each WIL script that requires it.

The documentation for the functions added are supplied either in a separate manual or disk file that accompanies the extender DLL.

Example:

```
; Add vehicle radar processing dll controlling billboard visible to
; motorists, and link to enforcement computers.
; The WIL Extender SPEED.DLL adds functions to read a radar speed
; detector(GetRadarSpeed) , put a message on a billboard visible to
; the motorist (BillBoard), take a video of the vehicle (Camera), and
; send a message to alert enforcement personnel (Alert) that a
; motorist in violation along with a picture id number to help
; identify the offending vehicle and the speed which it was going.
;
AddExtender("SPEED.DLL")
BillBoard("Drive Safely")
While @TRUE
    ; Wait for next vehicle
    while GetRadarSpeed(<5 ; if low, then just radar noise
        Yield                ; wait a bit, then look again
    endwhile
```

```
speed=GetRadarSpeed()          ; Something is moving out there
if speed < 58
    Billboard("Drive Safely")    ; Not too fast.
else
    if speed < 63
        Billboard("Watch your Speed") ; Hmmm a hot one
    else
        if speed < 66
            Billboard("Slow Down") ; Tooooo fast
        else
            Billboard("Violation Pull Over")
            pictnum = Camera() ; Take Video Snapshot
            Alert(pictnum, speed); Pull this one over
        endif
    endif
endif
endwhile
```

See Also:

DIICall (*found in main WIL documentation*)

LastError()

Returns the most-recent error encountered during the current WIL program.

Syntax:

LastError()

Parameters:

None

Returns:

(i) most-recent WIL error code encountered.

In addition to the normal behavior of the LastError function documented in the WIL Reference Guide, if the most recent error occurred in a WIL Extender, then a number assigned by the Extender will be returned. The numbers are documented in the appendix of this Extender document.

It may be possible to obtain error numbers not documented. The "Notes" section of the WIL manual has been provided to allow you to keep records of undocumented error codes.

Example:

```
ErrorMode(@OFF)
FileCopy("data.dat", "c:\backups\*.*", @FALSE)
ErrorMode(@CANCEL)
If LastError() == 1006
    Message("Error", "Please call Tech Support at 555-9999.")
endif
Message("LastError", " Done.")
```

See Also:

Debug, ErrorMode *(both found in main WIL documentation)*

Net101

All network functionality for WIL is performed via "WIL Extenders", add-on DLLs for WIL, which contain Network commands for assorted networks.

NetInfo is the only WIL network function. It returns the types of the networks currently active on the local machine, and can be used to help determine which network extenders should be loaded in multi-network environments.

Documentation for the various network extenders are found either in a manual for a particular extender or in an associated disk file.

See Also:

[NetInfo](#), [AddExtender](#), [DllCall](#) (*found in main WIL documentation*)

NetInfo(1) return values:

NONE	No networks active
MSNET	Microsoft Network
LANMAN	LAN Manager
WINNET	Windows Network (Windows for WorkGroups, etc.)
NETWARE	Novell NetWare
VINES	Banyan Vines
OTHER2	Other (code 0x20)
OTHER4	Other (code 0x40)
OTHER8	Other (code 0x80)

32 Bit Windows

NetInfo(0) will always return the string "WINNT" for 32 bit Windows platforms, regardless of whether the platform is Windows 95/98 or Windows NT.

Under Windows 95/98, **NetInfo(1)** will return a list of installed network client ID's, delimited with the standard file delimiter (by default, a tab).

Possible client ID's, with their corresponding descriptions, are:

Client ID	Description
3OPEN	3Com 3+Open (all versions)
3SHARE	3Com 3+Share (all versions)
DLR	IBM OS/2 LAN Server (versions below 1.2)
DLR12	IBM OS/2 LAN Server (version 1.2)
DLR13	IBM OS/2 LAN Server (versions 1.2, 1.3, and 1.2 without /API)
DLR13CSD	IBM OS/2 LAN Server (version 1.3 CSD 5015/5050)
DLR20	IBM OS/2 LAN Server (version 2.0)
FTPNFS	FTP Software NFS Client (InterDrive 95)
LANMAN	Microsoft Real Mode LAN Manager
LANT5	Artisoft LANtastic (version 5.X and above)
MSNET	Real mode MS-Net Compatible
NETWARE3	Novell NetWare (Workstation Shell 3.X [NETX])
NETWARE4	Novell NetWare (Workstation Shell 4.0 and above [VLM])
NOVELL32	Novell NetWare Client 32
NWREDIR	Client for NetWare Networks
PATHWKS	DEC PATHWORKS (versions below 4.0)
PATHWKS40	DEC PATHWORKS (version 4.x)
PCLP	IBM PC LAN Program (all versions)
PCNFS50	SunSoft PC-NFS (version 5.0)
VINES552	Banyan DOS/Windows 3.1 client
VREDIR	Client for Microsoft Networks

Under Windows NT, **NetInfo(1)** will return a list of installed network provider ID's, delimited with the standard file delimiter (by default, a tab).

Possible providers, with their corresponding descriptions, are:

Provider ID	Description
LanmanWorkstation	Microsoft Windows Network (NT client)
NetWareWorkstation	NetWare Services (Novell Netware client)
NWCWorkstation	NetWare or Compatible Network (Microsoft Netware client)

Example:

```
a=NetInfo(0)
if a=="MULTINET"
    b=NetInfo(1)
    count=ItemCount(b," ")
    Message("Multinet supporting %count% networks", b)
else
    Message("Installed Network", a)
endif
```

See Also:

[Net101](#), [AddExtender](#), [DllCall](#) *(found in main WIL documentation)*

netAddDrive(user-id, pswd, net-resource, local drive, persist)

Maps a drive.

Syntax:

```
netAddDrive(user-id, pswd, net-resource, local-drive, persist)
```

Parameters:

(s) user-id	user-id or @DEFAULT for current user
(s) pswd	password or @DEFAULT for current password or @NONE for no password
(s) net-resource	UNC netname of net resource
(s) local drive	local drive id e.g. ("K:") or @NONE for connect only
(s) persist	@TRUE - Specifies persistent connection, one that will automatically reconnect when you reboot windows. @FALSE - Specifies a temporary connection.

Returns:

(i) always 1

This function allows a connection to be made to a net resource, and, optionally, a drive to be mapped to the net resource.

Example:

```
AddExtender("WWNET32I.DLL")
netAddDrive(@default,@default,"\\SERVER\PUB","E:",@false)
RunWait("E:\EXCEL\EXCEL.EXE","E")
netCancelCon("E:",@false,@false)
```

See Also:

[netDirDialog](#), [netCancelCon](#)

netAddPrinter(user-id, pswd, net-resource, local device, persist)

Maps a printer resource to a local port.

Syntax:

```
netAddPrinter(user-id, pswd, net-resource, local device, persist)
```

Parameters:

(s) user-id	user-id or @DEFAULT for current user
(s) pswd	password or @DEFAULT for current password or @NONE for no password
(s) net-resource	UNC netname of net resource
(s) local device	local printer port e.g. ("lpt1") or @NONE for connect only
(s) persist	@TRUE - Specifies persistent connection, one that will automatically reconnect when you reboot windows. @FALSE - Specifies a temporary connection.

Returns:

(i)	@TRUE if the port was mapped; @FALSE the port was not mapped.
-----	--

This function allows a connection to be made to a net resource, and, optionally, a local device to be mapped to the net resource.

Example:

```
AddExtender("WVNET32I.DLL")  
netAddPrinter(@default,@default,"\\SERVER\LJ4","lpt2",@false)
```

See Also:

[netDirDialog](#), [netCancelCon](#)

netCancelCon(local drive, persist, forceflag)

Breaks a network connection.

Syntax:

netCancelCon(local drive, persist, forceflag)

Parameters:

- (s) local drive the mapped device.
- (s) persist **@TRUE** - update persistent connection table ;
@FALSE - do not update persistent connection table to remove this device
- (i) forceflag **@TRUE** to break the connection regardless of open files.
@FALSE - if files are open, connection will not be broken.

Returns:

- (i) **@TRUE** if successful; **@FALSE** if unsuccessful.

When a mapped local drive is specified, only that connection will be closed.

If persist is set to **@TRUE**, then the persistent connection will be updated to remove this drive mapping from the list of persistent connections.

If forceflag is set to **0**, **netCancelCon** will not break the connection if any files on that connection are still open. If forceflag is set to **1**, the connection will be broken regardless.

Example:

```
AddExtender("WNET32I.DLL")
netAddDrive(@default,@default,"\\SERVER\Pub\EXCEL","E:",@false)
RunWait("E:\EXCEL.EXE","\E")
netCancelCon("E:",@false,@false)
```

See Also:

[netAddDrive](#), [netDirDialog](#)

netDirDialog(flag)

Brings up a network drive connect/disconnect dialog box

Syntax:

netDirDialog(flag)

Parameters:

(i) flag **@FALSE**=disconnect dialog
 @TRUE=connect dialog

Returns:

(i) always 1

This function prompts the user with either a standard Connect or Disconnect dialog box. The user may make or break network drive mappings via the dialog box.

Example:

```
AddExtender("WWNET32I.DLL")
err=netDirDialog(@TRUE)
runwait("excel.exe", "/e")
netDirDialog(@FALSE)
```

See Also:

[netAddDrive](#)

netGetCon(local drive)

Returns the name of a connected network resource.

Syntax:

netGetCon(local name)

Parameters:

(s) local name local drive name.

Returns:

(i) name of a network resource.

netGetCon returns the name of the network resource currently connected to a 'local name'. If the resource is not mapped a null string will be returned.

Example:

```
AddExtender("WWNET32I.DLL")
netsrc=netGetCon("K:")
if netsrc="" then Message("Drive K: is","not mapped")
else Message("Drive K: is mapped to",netsrc)
```

See Also:

[netAddDrive](#), [netDirDialog](#)

netGetUser(netname)

Returns the name of the user currently logged into the network.

Syntax:

netGetUser(netname)

Parameters:

(s) netname name of network or **@DEFAULT** for default network.

Returns:

(s) the user name.

This function will interrogate the network and return the current user name. **@default** will return the user id of the local user.

Example:

```
AddExtender ("WWNET32I.DLL")
username=netGetUser (@default)
Message("Current User is",username)
```

See Also:

[netGetCon](#)

netResources(net-resource, scope, type, usage)

Itemizes network resources.

Syntax:

netResources(net-resource, scope, type, usage)

Parameters:

- | | |
|------------------|--|
| (s) net-resource | a UNC (e.g., "\\FredPC"), a domain (e.g., "SALES"), or ("") for the root of the network. |
| (i) scope | see below. |
| (i) type | see below. |
| (i) usage | see below. |

Returns:

- | | |
|-----|--|
| (s) | a tab-delimited list of network resources. |
|-----|--|

This function returns a tab-delimited list of network resources which are located immediately below the specified 'net-resource'.

'Scope' can be one of the following:

Req #	Meaning
1	All currently connected resources
2	All resources on the network
3	All remembered (persistent) connections

'Type' can be one of the following:

Req #	Meaning
0	All resources
1	All disk resources
2	All print resources
3	All disk and print resources

'Usage' can be one of the following:

Req #	Meaning
0	All resources
1	All connectable resources
2	All container resources
3	All connectable and container resources

Note: 'usage' is ignored unless 'scope' == 2.

Example:

```
AddExtender("WWNET32I.DLL")
servers = netResources("OFFICE", 2, 1, 1)
server = TextSelect("Available servers", servers, @TAB)
shares = netResources(server, 2, 1, 1)
share = TextSelect("Shared directories on %server%", shares, @TAB)
```


netVersion()

Returns the version of this Extender DLL.

Syntax:

netVersion()

Parameters:

none

Returns:

(i) the version number of this extender Dll.

This function is used to check the version number of this Dll in cases where older DLL's exist and alternate processing is desirable. Version numbers of newer versions will be larger than that of older versions.

Example:

```
AddExtender("WWNET32I.DLL")
a=netVersion( )
Message("Dll Version",a)
```

wntAccessAdd(server-name, resource/share-name, user/group name, object-type, access-string)

Adds or updates access (permission) records for a resource.

Syntax:

```
wntAccessAdd(server-name, resource/share-name, user/group-name,
              object-type, access-string)
```

Parameters:

- | | |
|-------------------------|---|
| (s) server-name | name of a network file server or empty string ("") to indicate the current machine. |
| (s) resource/share-name | identifies the object to be modified. |
| (s) user/group-name | name of a user or of a group to whom access is being granted. If necessary, it can be fully qualified as 'server\user'. |
| (i) object-type | identifies the type of the 'resource/share-name' object. See below. |
| (i) access-string | the type of access that is being granted. Either a predefined access type, or a delimited list. See below. |

Returns:

- | | |
|-----|----|
| (s) | 1. |
|-----|----|

WntAccessAdd can be used for looking up user/group names and share names (if 'resource/share-name' specifies a share).

Object-Type

'Object-type' indicates the type of object identified by 'resource/share-name', and can be one of the following:

Req#	object type
100	share (e.g., a directory accessed through a share)
200	printer (accessed through a share)
300	file or directory in an NTFS partition
301	directory in an NTFS partition, and all its subdirectories
302	directory in an NTFS partition, and all files in the directory
303	directory in an NTFS partition, and all its subdirectories, and all files in the directory and all its subdirectories
400	registry key
401	registry key, and all its subkeys

If **'object-type'** = 100 (share), then 'resource/share-name' should specify the name of the share.

If **'object-type'** = 300 (file or directory in an NTFS partition), then 'resource/share-name' should be UNC (ie, "\\Server\Share\Dir").

If **'object-type'** = 400 (registry key), then 'resource/share-name' should be the handle of an open registry key (opened with the **RegOpenKey** function), or a predefined registry handle. (Registration Functions are listed in the WIL Reference Manual under "Registration Database Operations".)

Otherwise, 'resource/share-name' should specify the name of the resource (e.g., "HP LaserJet III" or "C:\UTIL\MYFILE.TXT").

Access-String

'Access-string' specifies the type of access that is being granted. It can be either (A) a predefined access type, or (B) a delimited list of one or more specific 'access-records'. Both are described below:

(A) Predefined access types:

These get translated into specific access records, as shown. It is possible that the appropriate values may vary depending on your system configuration, or among different versions of Windows NT:

Access-string	Meaning	Specific equivalent
"DirNT:List"	List	"0:2:1179817"
"DirNT:Read"	Read	"0:9:-1610612736
"DirNT:Add"	Add	0:2:1179817"
"DirNT:AddRead"	Add & Read	"0:2:1180086"
"DirNT:Change"	Change	"0:9:-1610612736
"DirNT:Full"	Full Control	0:2:1180095"
"DirNT:None"	No Access	"0:9:-536805376
		0:2:1245631"
"DirShare:Read"	Read	"0:9:268435456
"DirShare:Change"	Change	0:2:2032127"
"DirShare:Full"	Full Control	"1:9:268435456
"DirShare:None"	No Access	1:2:2032127"
		"0:0:1179817"
"File:Read"	Read	"0:0:1245631"
"File:Change"	Change	"0:0:2032127"
"File:Full"	Full Control	"1:0:2032127"
"File:None"	No Access	
		"0:0:1179817"
"Print:Print"	Print	"0:0:1245631"
"Print:Manage"	Manage	"0:0:2032127"
"Print:Full"	Documents	"1:0:2032127"
"Print:None"	Full Control	
	No Access	"0:0:131080"

"Reg:Read"	Read	"0:0:131072
"Reg:Full"	Full Control	0:0:268435456"
		"0:0:983052
		0:0:268435456"
		"1:0:983052
		1:0:268435456"
		"0:2:131097"
		"0:2:983103"

Note: "DirShare" is a directory accessed through a share ("object-type" = 100). "DirNT" is a directory in an NTFS partition, accessed through NT security ("object-type" = 300).

(B) Specific 'access-records':

This can be a single record, or a list of records (maximum of 10) delimited with vertical bars (|). Each record is in the format:

```
record-type:access-flags:access-rights
```

where 'record-type', 'access-flags', and 'access-rights' are each a decimal number, separated with colons (:).

It is not expected that most users will want to manually create or edit 'access-records' strings. Instead, you can use the **wntAccessGet** function to return an 'access-records' string in the proper format for use with this function. This is useful for transferring access rights from one user to another.

A brief description of the fields in the 'access-records' string follows. Please note that any detailed explanation is beyond the scope of this document, but might be obtained from the WIN32 SDK programmers' documentation available from Microsoft and other publishers.

'record-type' (one of the following):

- 0 ACCESS_ALLOWED_ACE_TYPE
- 1 ACCESS_DENIED_ACE_TYPE

'access-flags' (0, or, one or more of the following):

- 1 OBJECT_INHERIT_ACE
- 2 CONTAINER_INHERIT_ACE
- 4 NO_PROPAGATE_INHERIT_ACE
- 8 INHERIT_ONLY_ACE

'access-rights' (one or more of the following, usually several, depending on the object type.)

Note: For all practical purposes, this is a complete list:

- 1 FILE_LIST_DIRECTORY
- 1 (Dir)

2	FILE_READ_DATA (File)
2	FILE_ADD_FILE (Dir)
4	FILE_WRITE_DATA (File)
4	FILE_ADD_SUBDIRECTOR
8	Y (Dir)
16	FILE_APPEND_DATA (File)
32	FILE_READ_EA
32	FILE_WRITE_EA
64	FILE_TRAVERSE (Dir)
128	FILE_EXECUTE (File)
256	FILE_DELETE_CHILD
65536	FILE_READ_ATTRIBUTES
131072	FILE_WRITE_ATTRIBUTES
262144	DELETE
524288	READ_CONTROL
1048576	WRITE_DAC
268435456	WRITE_OWNER
536870912	SYNCHRONIZE
1073741824	GENERIC_ALL
-2147483648	GENERIC_EXECUTE
	GENERIC_WRITE
	GENERIC_READ

If any access records already exist for 'resource/share-name' for the specified 'user/group name', they will be removed before adding the records specified by 'access-string'.

Note: If you have not previously added any permissions to a share, it may implicitly have some default permissions. For example, when you create a share for a directory, it defaults to giving "Full Control" access to "Everyone". When **wntAccessAdd** is used to create an access record for such a share, it will supersede those default permissions (i.e., the default permissions will be removed). If you wish to keep the default permissions, use **wntAccessAdd** to set them explicitly.

To use **wntAccessAdd** to change the permissions on a file on a remote server (ie, not the local machine), you need to specify the file name as a UNC, eg:

```
wntAccessAdd("server2", "\\server2\C$\test", "Everyone", 300, "DirNT:Read")
```

("C\$" and "D\$" are standard admin shares to "C:" and "D:", etc.)

Example:

```
;This example sets the share called "Public" on the current machine so
;that any member of the group "Everybody" has full access to the contents
;of the directory associated with the "Public" share. This function does
;not affect any permissions that may have been set with a NTFS file system
;with respect to the directory associated with the share.
;
AddExtender("WWWNT32I.DLL")
wntAccessAdd("", "Public", "Everybody", 100, "DirShare:Full")
```

See Also:

[wntAccessDel](#)

wntAccessDel(server-name, resource/share-name, user-name, share-type)

Removes an access (permission) records from a resource.

Syntax:

wntAccessDel(server-name, resource/share-name, user-name, share-type)

Parameters:

(s) server-name	name of a network file server or empty string ("") to indicate the current machine.
(s) resource/share-name	identifies the object to be modified.
(s) user-name	name of a user or of a group to whom access is being granted. If necessary, it can be fully qualified as 'server\user'.
(i) share-type	identifies the object type of the 'resource/share-name'.
100	share (e.g., a directory accessed through a share)
200	printer (accessed through a share)
300	file or directory in an NTFS partition
301	directory in an NTFS partition, and all its subdirectories
302	directory in an NTFS partition, and all files in the directory
303	directory in an NTFS partition, and all its subdirectories, and all files in the directory and all its subdirectories
400	registry key
401	registry key, and all its subkeys

Returns:

(i) **@TRUE** if records were deleted, **@FALSE** if no records were found.

Example:

```
; This example shows the removal of all specific references to the  
; group "Supervisors" in relation to a file "D:\NTFS\blackbook.txt"  
; on a NTFS file system for the current system. Note that this could  
; either remove specific permission to access the file, or it could  
; remove specific access denial to the file.
```

```
AddExtender("WWWNT32I.DLL")  
rslt=wntAccessDel("", "D:\NTFS\blackbook.txt", "Supervisors", 300)  
if rslt  
    Message("wntAccessDel", "Access records deleted")  
else  
    Message("wntAccessDel", "No access records found")  
endif
```

See Also:

[wntAccessAdd](#)

wntAccessGet(server-name, resource/share-name, user-name, share-type)

Returns access (permission) records for a resource.

Syntax:

```
wntAccessGet(server-name, resource/share-name, user-name, share-type)
```

Parameters:

- | | |
|-------------------------|---|
| (s) server-name | name of a network file server or empty string ("") to indicate the current machine. |
| (s) resource/share-name | identifies the object to be modified. |
| (s) user-name | name of a user or of a group to whom access is being granted. If necessary, it can be fully qualified as 'server\user'. |
| (i) share-type | identifies the type of the 'resource/share-name' object. |

Returns:

- | | |
|-----|--|
| (i) | a delimited list of access records or ("") if no records were found. |
|-----|--|

WntAccessGet returns a list of access records for 'resource/share-name' for the specified 'user/group name', delimited with vertical bars (|). If there are no appropriate records, it returns a blank string ("").

See **wntAccessAdd** for information on the format of the access records.

Note: If no permissions were previously added to a share, it may implicitly have some default permissions. For example, when a share is created for a directory, it defaults to giving "Full Control" access to "Everyone", although there may not actually be any access records for the share. Therefore, **wntAccessGet** may return a blank string (""). However, implicit permissions will become actual permissions when from the File Manager (or Explorer) the "Permissions" dialog for the share is brought up and "OK" is selected. **wntAccessGet** can then retrieve the actual permissions.

Example:

```
records=wntAccessGet("", "Public", "jdoe", 100)
if records=="
    Message("wntAccessGet", "No records found for share 'Public'")
else
    Message("wntAccessGet", strcat("'Public' Share Records are", @crlf, records))
endif
```

See Also:

[wntAccessAdd](#), [wntAccessDel](#)

wntAccessList(server-name, resource/share-name, object-type, flags)

Returns list of users who have access (permission) records for a resource.

Syntax:

wntAccessList(server-name, resource/share-name, object-type, access string)

Parameters:

- (s) server-name name of a network file server or empty string ("") to indicate the current machine.
- (s) resource/share-name identifies the object to be modified.
- (i) object-type indicates the type of object identified by 'resource/share-name'. See below.
- (i) flags specifies the format of the returned names.

Returns:

- (s) a tab-delimited list of users and groups who have access records for "resource/share-name"; ie, users and groups for whom permissions have explicitly been set. Returns a blank string ("") if there are no appropriate records.

Object-Type

'Object-type' indicates the type of object identified by 'resource/share-name', and can be one of the following:

Req#	object type
100	share (e.g., a directory accessed through a share)
200	printer (accessed through a share)
300	file or directory in an NTFS partition
400	registry key

If '**object-type**' = 100 (share), then 'resource/share-name' should specify the name of the share.

If '**object-type**' = 400 (registry key), then 'resource/share-name' should be the handle of an open registry key (opened with the **RegOpenKey** function), or a predefined registry handle. (Registration Functions are listed in the WIL Reference Manual under "Registration Database Operations".)

Otherwise, 'resource/share-name' should specify the name of the resource (e.g., "HP LaserJet III" or "C:\UTIL\MYFILE.TXT").

Flags

'Flag' specifies the format of the returned names, and can be one of the following:

- 0 account (eg, "johndoe")
- 1 domain\account (eg, "OFFICE\johndoe")

Note: For built-in accounts which are predefined by the system (eg, "Administrators"), only the account name is returned, regardless of the "flag" setting.

Example:

```
;This example sets the share called "Public" on the current machine so
;that any member of the group "Everybody" has full access to the contents
;of the directory associated with the "Public" share. This function does
;not affect any permissions that may have been set with a NTFS file system
;with respect to the directory associated with the share.
;
AddExtender("WWWNT32I.DLL")
wntAccessAdd("", "Public", "Everybody", 100, "DirShare:Full")
users = wntAccessList("", "Public", 100, 0)
```

See Also:

[wntAccessDel, wntAccessAdd](#)

wntAddDrive(user-id, pswd, net-resource, local drive, persist)

Maps a drive.

Syntax:

wntAddDrive(user-id, pswd, net-resource, local-drive, persist)

Parameters:

(s) user-id	user-id or @DEFAULT for current user
(s) pswd	password or @DEFAULT for current password or @NONE for no password
(s) net-resource	UNC netname of net resource
(s) local drive	local drive id e.g. ("K:") or @NONE for connect only
(s) persist	@TRUE - Specifies persistent connection, one that will automatically reconnect when you reboot windows. @FALSE - Specifies a temporary connection.

Returns:

(i) always 1

This function allows a connection to be made to a net resource, and, optionally, a drive to be mapped to the net resource.

Example:

```
AddExtender("WWWNT32I.DLL")
wntAddDrive(@default,@default,"\\SERVER\Pub","E:",@false)
RunWait("E:\EXCEL\EXCEL.EXE","E")
wntCancelCon("E:",@false,@false)
```

See Also:

[wntDirDialog](#), [wntCancelCon](#)

wntAddPrinter(user-id, pswd, net-resource, local device, persist)

Maps a printer resource to a local port.

Syntax:

```
wntAddPrinter(user-id, pswd, net-resource, local device, persist)
```

Parameters:

(s) user-id	user-id or @DEFAULT for current user
(s) pswd	password or @DEFAULT for current password or @NONE for no password
(s) net-resource	UNC netname of net resource
(s) local device	local printer port e.g. ("lpt1") or @NONE for connect only
(s) persist	@TRUE - Specifies persistent connection, one that will automatically reconnect when you reboot windows. @FALSE - Specifies a temporary connection.

Returns:

(i)	@TRUE if the port was mapped; @FALSE the port was not mapped.
-----	--

This function allows a connection to be made to a net resource, and, optionally, a local device to be mapped to the net resource.

Example:

```
AddExtender("WWWNT32I.DLL")  
wntAddPrinter(@default,@default,"\\SERVER\LJ4","lpt2",@false)
```

See Also:

[wntDirDialog](#), [wntCancelCon](#)

wntCancelCon(local drive, persist, forceflag)

Breaks a network connection.

Syntax:

wntCancelCon(local drive, persist, forceflag)

Parameters:

- (s) local drive the mapped device.
- (s) persist **@TRUE** - update persistent connection table ;
@FALSE - do not update persistent connection table to remove this device.
- (i) forceflag **@TRUE** to break the connection regardless of open files.
@FALSE - if files are open, connection will not be broken.

Returns:

- (i) **@TRUE** if successful; **@FALSE** if unsuccessful.

When a mapped local drive is specified, only that connection will be closed.

If persist is set to **@TRUE**, then the persistent connection will be updated to remove this drive mapping from the list of persistent connections.

If forceflag is set to **0**, **wntCancelCon** will not break the connection if any files on that connection are still open. If forceflag is set to **1**, the connection will be broken regardless.

Example:

```
AddExtender("WWWNT32I.DLL")
wntAddDrive(@default,@default,"\\SERVER\PUB\EXCEL","E:",@false)
RunWait("E:\EXCEL.EXE","\E")
wntCancelCon("E:",@false,@false)
```

See Also:

[wntAddDrive](#), [wntDirDialog](#)

wntChgPswd(server/domain, user-name, old-password, new-password)

Changes a user's password.

Syntax:

```
wntChgPswd(server/domain, user-name, old-password, new-password)
```

Parameters:

- | | |
|-------------------|---|
| (s) server/domain | either a server name (e.g., "\\MYSERVER") or domain name (e.g., "SALES") on which the password will be changed, or a blank string ("") for the current user's login domain. |
| (s) user-name | the name of the user whose password will be changed, or a blank string ("") for the current user's login name. |
| (s) old-password | the user's password on "server/domain" or "*UNKNOWN*" (see below). |
| (s) new-password | the new password to be set or a blank string ("") to indicate no password is desired. |

Returns:

- | | |
|-----|---|
| (i) | 1 |
|-----|---|

old-password

wntChgPswd can be used to specify a new password without knowing the old password, if you are a member of the Administrators or Account Operators local group. To do this, specify "*UNKNOWN*" as the old password. In this case, the "user" parameter must specify an actual user name (ie, it cannot be a blank string).

Example:

```
old=AskPassword("Change Password","Enter old password")
while @TRUE
    new1=AskPassword("Change Password","Enter new password")
    new2=AskPassword("Change Password","Enter new password again")
    if new1==new2 then break
    Message("Try Again","The new passwords you entered do not match")
endwhile

rslt=wntChgPswd("", "", old, new1)

if rslt
    Message("Password","Successfully changed")
else
    Message("Password","Not changed")
endif
```

wntDirDialog(flag)

Brings up a network drive connect/disconnect dialog box

Syntax:

wntDirDialog(flag)

Parameters:

(i) flag **@FALSE**=disconnect dialog
 @TRUE=connect dialog

Returns:

(i) **1**

This function prompts the user with either a standard Connect or Disconnect dialog box. The user may make or break network drive mappings via the dialog box.

Example:

```
AddExtender("WWWNT32I.DLL")
err=wntDirDialog(@TRUE)
runwait("excel.exe", "/e")
wntDirDialog(@FALSE)
```

See Also:

[wntAddDrive](#)

wntFileClose(server-name, file-pathname)

Close all network connections to a file.

Syntax:

```
wntFileClose(server-name, file-pathname)
```

Parameters:

- | | |
|-------------------|---|
| (s) server-name | "server-name" is the name of a remote server on which the function will execute, or a blank string ("") to indicate the local computer. |
| (s) file-pathname | "file-pathname" is a fully-qualified file name (eg, "C:\DOC\MYFILE.TXT").
NOTE: The file name MUST be fully-qualified. |

Returns:

- | | |
|-----|---|
| (i) | Returns the number of connections which existed (and were closed) for the specified file. |
|-----|---|

Note: this function will not work to do a wntFileClose on a local file that was opened by yourself on the local machine, even if it's done over a mapped drive with a UNC.

Example:

;the "C:\DOC\MYFILE.TXT" on the local machine needs to be opened by someone else over the network.

```
AddExtender("WWWNT32I.DLL")  
wntFileClose("", "C:\DOC\MYFILE.TXT")
```

See Also:

[wntAddDrive](#)

wntGetCon(local drive)

Returns the name of a connected network resource.

Syntax:

wntGetCon(local name)

Parameters:

(s) local name local drive name.

Returns:

(i) name of a network resource.

wntGetCon returns the name of the network resource currently connected to a 'local name'. If the resource is not mapped a null string will be returned.

Example:

```
AddExtender ("WWWNT32I.DLL")
netsrc=wntGetCon ("K:")
if netsrc="" then Message("Drive K: is","not mapped")
else Message("Drive K: is mapped to",netsrc)
```

See Also:

[wntAddDrive](#), [wntDirDialog](#)

wntGetUser(netname)

Returns the name of the user currently logged into the network.

Syntax:

wntGetUser(netname)

Parameters:

(s) netname - name of network or **@DEFAULT** for default network.

Returns:

(s) the user name.

This function will interrogate the network and return the current user name. **@default** will return the user id of the local user.

Example:

```
AddExtender("WWWNT32I.DLL")
username=wntGetUser(@default)
Message("Current User is",username)
```

See Also:

[wntGetCon](#)

wntGroupInfo(server-name, group, group-type, request)

Returns information about a group.

Syntax:

wntGroupInfo(server-name, group, group-type, request)

Parameters:

- | | |
|-----------------|---|
| (s) server-name | the UNC name of the server on which the function will execute (eg, "\\MYSERVER"), or "" for the local computer. |
| (s) group | name of a group. |
| (s) group-type | can be @LOCALGROUP or @GLOBALGROUP . |
| (s) request | specifies the information to be returned, and can be one of the following:
0 group name
1 group comment |

Returns:

- | | |
|-----|-------------------|
| (s) | Returns a string. |
|-----|-------------------|

"request" specifies the information to be returned, and can be one of the following:

- | | |
|---|---------------|
| 0 | group name |
| 1 | group comment |

Example:

```
AddExtender("WWWNT32I.DLL")
comment = wntGroupInfo("", "Administrators", @LOCALGROUP, 1)
Message("Comment for 'Administrators'", comment)
```

See Also:

[wntListGroup](#)s

wntListGroup(server-name, group-type)

Returns tab-delimited list of all user groups on a specified server.

Syntax:

wntListGroup(server-name, group-type)

Parameters:

(s) server-name name of a network file server ("\\MYSERVER")
or ("") for a local PC.

(i) group-type @LOCALGROUP or @GLOBALGROUP.

Returns:

(i) a tab-delimited list of user groups.

Example:

```
AddExtender("WWWNT32I.DLL")
localgroups=wntListGroup("",@LOCALGROUP)
AskItemList("Local Groups",localgroups,@tab,@sorted,@single)
globalgroups=wntListGroup("",@GLOBALGROUP)
AskItemList("Global Groups",globalgroups,@tab,@sorted,@single)
```

See Also:

[wntMemberList](#)

wntMemberDel(server-name, group-name, user-name, ,group-type)

Deletes the specified user from the specified group on the specified server.

Syntax:

```
wntMemberDel(server-name, group-name, user-name, group-type)
```

Parameters:

- | | |
|-----------------|--|
| (s) server-name | name of a network file server ("\\MYSERVER") or ("") for a local PC. |
| (s) group-name | name of the group. |
| (s) user-name | name of the current user. |
| (i) group-type | @LOCALGROUP or @GLOBALGROUP . |

Returns:

- | | |
|-----|--|
| (i) | @TRUE if successful;
@FALSE if the user is not a member of the group. |
|-----|--|

Assuming that the person running this script has sufficient authority to delete users from the specified group, this function will delete the specified user from the group.

Example:

```
AddExtender("WWWNT32I.DLL")
rslt=wntMemberDel("", "testgroup", "jdoe", @LOCALGROUP)
if rslt
    Message("jdoe", "removed from testgroup")
else
    Message("jdoe", "is not a member of testgroup")
endif
```

See Also:

[wntMemberGet](#), [wntMemberSet](#)

wntMemberGet(server-name, group-name, user-name, group-type)

Determines if the specified user is a member of the specified group on the specified server.

Syntax:

```
wntMemberGet(server-name, group-name, user-name, group-type)
```

Parameters:

(s) server-name	name of a network file server ("\\MYSERVER") or ("") for a local PC.
(s) group-name	name of the group.
(s) user-name	name of the current user.
(i) group-type	@LOCALGROUP or @GLOBALGROUP

Returns:

(i)	@TRUE if successful; @FALSE if unsuccessful.
-----	---

Assuming that the person running this script has sufficient authority to query members of the specified group, this function will allow the person to determine if the user is a member of the specified group or not.

Example:

```
AddExtender("WWWNT32I.DLL")
rslt=wntMemberGet("", "testgroup", "jdoe", @LOCALGROUP)
if rslt
    Message("jdoe", "is a member of testgroup")
else
    Message("jdoe", "is not a member of testgroup")
endif
```

See Also:

[wntMemberSet](#), [wntMemberDel](#)

wntMemberGrps(server-name, user-name, group-type, flags)

Returns a list of groups to which the specified user belongs.

Syntax:

wntMemberGrps(server-name, user-name, group-type, flags)

Parameters:

- | | |
|-----------------|---|
| (s) server-name | name of a network file server ("\\MYSERVER") or ("") for a local PC. |
| (s) user-name | name of the current user. |
| (i) group-type | @LOCALGROUP or @GLOBALGROUP |
| (i) flags | 1 - with group-type of @LOCALGROUP to include groups in which user is an indirect member. Otherwise, set to 0. |

Returns:

- | | |
|-----|---------------------------------|
| (i) | a tab delimited list of groups. |
|-----|---------------------------------|

Example:

```
AddExtender("WWWNT32I.DLL")
groups=wntMemberGrps("", "jdoe", @LOCALGROUP, 1)
AskItemList("jdoe is associated with", groups, @tab, @sorted, @single)
```

See Also:

[wntMemberSet](#), [wntMemberDel](#)

wntMemberList(server-name, group-name, group-type)

Returns a list of the members of a user group.

Syntax:

```
wntMemberList(server-name, group-name, group-type)
```

Parameters:

- | | |
|-----------------|--|
| (s) server-name | name of a network file server ("\\MYSERVER") or ("") for a local PC. |
| (s) group-name | name of the group. |
| (i) group-type | @LOCALGROUP or @GLOBALGROUP |

Returns:

- (i) a tab delimited list of users.

Example:

```
AddExtender("WWWNT32I.DLL")
people=wntMemberList("", "Everybody", @LOCALGROUP)
AskItemList("Member of group Everybody", people, @tab, @sorted, @single)
```

See Also:

[wntMemberSet](#), [wntMemberDel](#)

wntMemberSet(server-name, group-name, user-name, group-type)

Sets the specified user as a member of the specified group on the specified server.

Syntax:

```
wntMemberSet(server-name, group-name, user-name, group-type)
```

Parameters:

- (s) server-name name of a network file server ("\\MYSERVER") or ("") for a local PC.
- (s) group-name name of the group.
- (s) user-name name of the current user.
- (i) group-type **@LOCALGROUP** or **@GLOBALGROUP**.

Returns:

- (i) **@TRUE** if successful;
 @FALSE if the user is already a member of the group.

Assuming that the person running this script has sufficient authority to add users to the specified group, this function will add the specified user to the group.

Example:

```
AddExtender("WWWNT32I.DLL")
rslt=wntMemberSet("", "testgroup", "jdoe", @LOCALGROUP)
if rslt
    Message("jdoe", "is a NEW member of testgroup")
else
    Message("jdoe", "is ALREADY a member of testgroup")
endif
```

See Also:

[wntMemberDel](#), [wntMemberGet](#)

wntOwnerGet(server-name, reg-key, resource-name, object-type, flag)

Returns the owner of an object.

Syntax:

wntOwnerGet(server-name, reg-key, resource-name, object-type, flag)

Parameters:

- | | |
|-------------------|--|
| (s) server-name | name of a network file server ("\\MYSERVER") or ("") for a local PC. |
| (i) reg-key | handle of an open registry key or 0. See below. |
| (s) resource-name | identifies the object to be accessed. |
| (i) object-type | indicates the type of object identified by 'Resource-name'. |
| (i) flag | specifies the format of the returned string. |

Returns:

- | | |
|-----|--|
| (s) | the name of the account that owns the object, or a blank string ("") if the object has no owner. |
|-----|--|

'**Server-name**' can be the name of the server on which the function will execute, or a blank string ("") to indicate the current machine. This is used for looking up user/group names.

'**Reg-key**' is used only if the object is a registry key. If 'Object-type' = 400 (registry key), then 'Reg-key' should be the handle of an open registry key (opened with the "RegOpenKey" function), or a predefined registry handle (listed in the WIL Reference under "Registration Database Operations"). Otherwise, 'Reg-key' is ignored, and should be specified as 0.

'**Resource-name**' identifies the object to be accessed. If 'Object-type' = 400 (registry key), then 'Resource-name' can be a subkey string relative to 'Reg-key', or a blank string ("") if 'Reg-key' represents the actual object to be accessed. Otherwise, 'Resource-name' should specify the name of the object (eg, "C:\UTIL\MYFILE.TXT").

'**Object-type**' indicates the type of object identified by 'Resource-name', and can be one of the following:

Req #	Meaning
300	file or directory in an NTFS partition
400	registry key

'**Flag**' specifies the format of the returned string, and can be one of the following:

Req #	Meaning
0	account (eg, "johndoe")
1	domain\account (eg, "OFFICE\johndoe")

Note: For built-in accounts which are predefined by the system (eg, "Administrators"), only the

account name is returned, regardless of the 'Flag' setting.

Example:

```
;For a file:
AddExtender("WWWNT32I.DLL")
ErrorMode(@OFF)
owner = wntOwnerGet("", 0, "f:\test\myfile.txt", 300, 0)
ErrorMode(@ON)
Message("Owner is", owner)

;For a registry key:
AddExtender("WWWNT32I.DLL")
ErrorMode(@OFF)
owner = wntOwnerGet("", @REGMACHINE, "Software\Test", 400, 0)
ErrorMode(@ON)
Message("Owner is", owner)
```

See Also:

[wntOwnerSet](#), [wntMemberGet](#)

wntOwnerSet(server-name, reg-key, resource-name, object-type, user/group name)

Takes ownership of an object.

Syntax:

wntOwnerSet(server-name, reg-key, resource-name, object-type, user/group name)

Parameters:

(s) server-name	name of a network file server ("\\MYSERVER") or ("") for a local PC.
(i) reg-key	handle of an open registry key or 0.
(s) resource-name	identifies the object to be accessed.
(i) object-type	indicates the type of object identified by 'Resource-name'.
(s) user/group name	name of user/group who is taking ownership of the object.

Returns:

(i)	1
-----	---

'User/group name' is the name of the user or group who is taking ownership of the object. If necessary, it can be fully qualified as "server\user". It can be either the current user, or "Administrators". You can specify a blank string ("") to indicate "Administrators".

'Object-type' indicates the type of object identified by 'Resource-name', and can be one of the following:

Req #	Meaning
300	file or directory in an NTFS partition
400	registry key

In addition to taking ownership of an object, you may also wish to reset the object's access (permission) records to the system default (which grants full control to everyone). To do this, add 50 to the values for 'Object-type' listed above, ie:

Req #	Meaning
350	file or directory in an NTFS partition -- reset access records
450	registry key -- reset access records

After resetting the object's access records, you can use the wntAccessAdd function to change the object's permissions.

See wntOwnerGet for additional parameter information.

Example:

```
;For a file:  
AddExtender ("WWWNT32I.DLL")
```

```
wntOwnerSet("", 0, "f:\test\myfile.txt", 350, "Administrators")
```

;For a registry key:

```
AddExtender("WWWNT32I.DLL")
```

```
wntOwnerSet("", @REGMACHINE, "Software\Test", 450, "Administrators")
```

See Also:

[wntOwnerGet, wntMemberGet](#)

wntRasUserGet (server-name, user-name, request)

Gets RAS information for a user.

Syntax:

wntRasUserGet(server-name, user-name, request)

Parameters:

- (s) server-name is the UNC name of the primary or backup domain controller that has the user account database (eg, "\\MYSERVER"). If the machine on which the user account resides is a standalone NT workstation or server, you can specify the UNC name of that machine, or "" for the local computer.
- (s) user-name is the name of a user.
- (i) request specifies the information to be returned, and can be one of the following:
1 - Dial-in privilege or
2 - Callback phone number

Returns:

- (i/s) Value Returns one of the following, depending on the request.
- | | |
|---|---------------------------|
| 1 | (i) Dial-in privilege |
| 2 | (s) Callback phone number |

"Privilege"

Dial-in privilege specifies both the dial-in permission and the callback mode, and will be one of the following values:

Value	Meaning
1	No dial-in permission / No callback privilege
2	No dial-in permission / Callback number specified by caller
4	No dial-in permission / Callback number preset by administrator
9	Dial-in permission / No callback privilege
10	Dial-in permission / Callback number specified by caller
12	Dial-in permission / Callback number preset by administrator

Example:

```
AddExtender ("WWWNT32I.DLL")
flags=wntRasUserGet (\\SERV, "user1", 1)
Message ("RASFlags on User1", flags)
wntRasUserSet (server, "user1", "9", "")
flags=wntRasUserGet (\\SERV, "user1", 1)
Message ("RASFlags on User1", flags)
```

See Also:

[wntRasUserSet](#)

wntRasUserSet(server-name, user-name, privilege, phone-number)

Sets RAS information for a user.

Syntax:

wntRasUserSet (server-name, user-name, privilege, phone-number)

Parameters:

- (s) server-name is the UNC name of the primary or backup domain controller that has the user account database (eg, "\\MYSERVER"). If the machine on which the user account resides is a standalone NT workstation or server, you can specify the UNC name of that machine, or "" for the local computer.
- (s) user-name is the name of a user.
- (i) request specifies the information to be returned, and can be one of the following:
- (i) privilege see below.
- (s) phone-number specifies an administrator-preset callback phone number, if appropriate. You can specify a blank string ("") to leave the currently-set number (if any) unchanged.

Returns:

- (i) Returns 1.

"Privilege"

Value	Meaning
1	No dial-in permission / No callback privilege
2	No dial-in permission / Callback number specified by caller
4	No dial-in permission / Callback number preset by administrator
9	Dial-in permission / No callback privilege
10	Dial-in permission / Callback number specified by caller
12	Dial-in permission / Callback number preset by administrator

Example:

```
AddExtender("WWWNT32I.DLL")
flags=wntRasUserGet(\\SERV,"user1",1)
Message("RASFlags on User1", flags)
wntRasUserSet(server,"user1","9","")
flags=wntRasUserGet(\\SERV,"user1",1)
Message("RASFlags on User1", flags)
```

See Also:

[wntRasUserGet](#)

wntResources(net-resource, scope, type, usage)

Itemizes network resources.

Syntax:

wntResources(net-resource, scope, type, usage)

Parameters:

- | | |
|------------------|---|
| (s) net-resource | a UNC (e.g., "\\FredPC"), a domain (e.g., "SALES") or ("") for the root of the network. |
| (i) scope | see below. |
| (i) type | see below. |
| (i) usage | see below. |

Returns:

- | | |
|-----|--|
| (s) | a tab-delimited list of network resources. |
|-----|--|

This function returns a tab-delimited list of network resources which are located immediately below the specified 'net-resource'.

'Scope' can be one of the following:

Req #	Meaning
1	All currently connected resources
2	All resources on the network
3	All remembered (persistent) connections

'Type' can be one of the following:

Req #	Meaning
0	All resources
1	All disk resources
2	All print resources
3	All disk and print resources

'Usage' can be one of the following:

Req #	Meaning
0	All resources
1	All connectable resources
2	All container resources
3	All connectable and container resources

Note: 'usage' is ignored unless 'scope' == 2.

Example:

```
AddExtender("WWWNT32I.DLL")
servers = wntResources("OFFICE", 2, 1, 1)
server = TextSelect("Available servers", servers, @TAB)
shares = wntResources(server, 2, 1, 1)
share = TextSelect("Shared directories on %server%", shares, @TAB)
```

See Also:

[WntResources2](#)

wntResources2(net-resource, scope, type, usage, provider)

Itemizes network resources.

(This functions supersedes wntResources.)

Syntax:

wntResources2(net-resource, scope, type, usage, provider)

Parameters:

- | | |
|------------------|---|
| (s) net-resource | a server UNC (e.g., "\\FredPC"), a domain (e.g., "SALES"), or the name of a container object with a domain (e.g. "Directory Services"). Cannot be the a network provider. Specify an empty string ("") for the root of the network or the root of the network provider. |
| (i) scope | see below. |
| (i) type | see below. |
| (i) usage | see below. |
| (s) provider | name of the network provider under which 'net resource' is located. |

Returns:

- | | |
|-----|---|
| (s) | a tab-delimited list of network resources located immediately below the specified 'net-resource'. |
|-----|---|

Optional Returns:

- | | |
|-----|--|
| (s) | If you specify blank strings ("") for both 'net-resource' and 'provider', this function will return a tab delimited list of network providers. |
| (s) | If you specify a blank string ("") for net-resource' and a non-blank 'provider', this function will return a tab delimited list of network resources located beneath 'provider'. |

'Scope' can be one of the following:

Req #	Meaning
1	All currently connected resources
2	All resources on the network
3	All remembered (persistent) connections

'Type' can be one of the following:

Req #	Meaning
0	All resources
1	All disk resources
2	All print resources
3	All disk and print resources

Note: 'net-resource', 'usage', and 'provider' are ignored unless 'scope' is 2. That is, you can itemize all current or persistent connections network-wide, eg:

```
wntResources2("", 1, 0, 1, "")
```

Or you can itemize all available resources under a specific 'net-resource', eg:

```
wntResources2("\\JELLO", 2, 0, 1, "Microsoft Windows Network")
```

But you can not do both at the same time, ie, you can not itemize current connections under a specific 'net-resource':

Req #	Meaning
0	All resources
1	All connectable resources
2	All container resources
3	All connectable and container resources

Note: A "connectable" object is an object that can be connected to, such as a directory or a printer. A "container" object is an object that contains other objects, such as a domain or a server.

'Provider' is the name of the network provider under which 'net-resource' is located. Most commonly 'provider' is "Microsoft Windows Network" (for Windows NT) or "NetWare Services" (for NetWare).

Example:

```
AddExtender("WWWNT32I.DLL")
;;examples for the different types of resources.
providers = wntResources2("", 2, 0, 2, "")
domains = wntResources2("", 2, 0, 2, "Microsoft Windows Network")
servers = wntResources2("OFFICE", 2, 0, 2, "Microsoft Windows Network")
shares = wntResources2("\\SERVER", 2, 0, 1, "Microsoft Windows Network")
```

See Also:

[WntResources](#)

wntRunAsUser(domain/server, user-name, password, login-type, flags)

Run as a different user.

Syntax:

wntRunAsUser(domain/server, user-name, password, login-type, flags)

Parameters:

- | | |
|-------------------|--|
| (s) domain/server | the name of the domain or server on which the specified user account resides, or "." to indicate that the local account database should be searched for the user, or "" to indicate that the local account database and (then) any trusted domain account databases should be searched for the user. |
| (s) user-name | the name of the user to run as. |
| (s) password | the specified user's login password. |
| (i) login-type | see below. |
| (i) flags | reserved, and should be set to 0. |

Returns:

- | | |
|-----|-----------|
| (i) | always 1. |
|-----|-----------|

login-type can be one of the following:

Type	Meaning
2	Interactive login
3	Network login
4	Batch login
5	Service login

Note that the specified user must have the appropriate user rights assigned to be able to log in as a batch job or service.

This function causes the specified user to be impersonated only for the duration of the currently-running instance of the WIL Interpreter.

In order to use this function, the currently-logged-in user must have the "Act as part of the operating system" user right assigned. This can be set from the "Policy" menu in the NT User Manager (note that the "Show Advanced User Rights" option in the dialog box must be checked).

Example:

```
AddExtender("WWWNT32I.DLL")
;run as new user
user="Joe"
ret = wntRunAsUser( "", user, "", 3, 0)
Message("Running as new user:",user)
exit
```

See Also:

[wntUserInfo](#), [wntUserAdd](#)

wntServerList(server-name, domain-name, server-type)

Lists servers in a domain.

Syntax:

wntServerList(server-name, domain-name, server-type)

Parameters:

- (s) server-name the UNC name of the server on which the function will execute (e.g., "\\MYSERVER"), or ("") for the local computer.
- (s) domain-name the name of the domain which will be used (e.g., "SALES"), or ("") for the primary domain.
- (i) server-type identifies the type of servers which will be examined. See below.

Returns:

- (i) a tab-delimited list of server UNC names (e.g., "\\MYSERVER").

Note: An NT workstation can be considered to be a "server".

Server-type

Specify -1 for all servers. Or, specify one or more of the following flags, combined using the binary OR ("|") operator.

Req#	Server Type
1	All LAN Manager workstation
2	All LAN Manager server
4	Any server running with Microsoft SQL Server
8	Primary domain controller
16	Backup domain controller
32	Server running the timesource service
64	Apple File Protocol servers
128	Novell servers
256	LAN Manager 2.x Domain Member
512	Server sharing print queue
1024	Server running dialin service
2048	Xenix server
4096	Windows NT (either workstation or server)
8192	Server running Windows for Workgroups
16384	Microsoft File and Print for Netware
32768	Windows NT Non-DC server
65536	Server that can run the browser service
131072	Server running a browser service as backup
262144	Server running the master browser service
524288	Server running the domain master browser
4194304	Windows 95 or newer

Note: This function can take a while to run, depending on how many servers are in the domain. Also, it will only return the names of servers which it is able to access, which requires that the user have browse access to their service control managers.

Example:

```
AddExtender("WWWNT32I.DLL")
;return a list of all Windows NT (either workstation or server)
servers = wntServerList("", "", 4096)
AskItemList("list of Windows NT workstations or
servers", servers, @tab, @sorted, @single)
exit
```

See Also:

wntServerType(server-name)

Returns a server's platform.

Syntax:

wntServerType(server-name)

Parameters:

(s) server-name the UNC name of a server (eg, "\\SERVER1"),
or a blank string ("") to indicate the local machine.

Returns:

(i) See below.

Returns one of the following values:

Value	Meaning
0	Invalid server name
1	Other
2	Windows for Workgroups
3	Windows 95 or later
4	Windows NT

Example:

; This example returns a servers platform

```
AddExtender("WWWNT32I.DLL")
type=wntServerType("\\Server1")
switch type
case 0
  message("Error","invalid server name")
  break
case 1
  message("Server Type is:","Other")
  break
case 2
  message("Server Type is:","Windows for Workgroups")
  break
case 3
  message("Server Type is:","Windows 95 or later")
  break
case 4
  message("Server Type is:","Windows NT")
  break
EndSwitch
```

See Also:

[w95ServerType](#)

wntServiceAt(server, domain, server-type, service-name, flags)

Lists all servers in a domain which contain a specified service.

Syntax:

wntServiceAt(server, domain, server-type, service-name, flags)

Parameters:

- (s) server-name the UNC name of the server on which the function will execute (e.g., "\\MYSERVER"), or ("") for the local computer.
- (s) domain the name of the domain which will be used (e.g., "SALES"), or ("") for the primary domain.
- (i) server-type identifies the type of servers which will be examined. See below.
- (s) service-name the name of the service to be looked for.
- (i) flags specifies information on the service being looked for. See below.

Returns:

- (i) a tab-delimited list of server UNC names (e.g., "\\MYSERVER").

Note: An NT workstation can be considered to be a "server".

Server-type

Specify **-1** for all servers. Or, specify one or more of the following flags, combined using the binary OR ("|") operator.

Req#	Server Type
1	All LAN Manager workstation
2	All LAN Manager server
4	Any server running with Microsoft SQL Server
8	Primary domain controller
16	Backup domain controller
32	Server running the timesource service
64	Apple File Protocol servers
128	Novell servers
256	LAN Manager 2.x Domain Member
512	Server sharing print queue
1024	Server running dialin service
2048	Xenix server
4096	Windows NT (either workstation or server)
8192	

16384	Server running Windows for Workgroups
32768	Microsoft File and Print for Netware
65536	Windows NT Non-DC server
131072	Server that can run the browser service
262144	Server running a browser service as backup
524288	Server running the master browser service
4194304	Server running the domain master browser
-2147483648	Windows 95 or newer Domain announcement

Service-Name

'Service name' is the name of a service (e.g., "Spooler") or driver (e.g., "Atdisk"). The name can be specified either as the "display name" which is listed in Control Panel (name-type = 0) or the "service name" which is the actual registry key for the service (name-type = 1000). The SDK documentation describes them as:

DisplayName = a string that is to be used by user interface programs to identify the service.
 ServiceName = a string that names a service in a service control manager database.

So, the following two commands will yield identical results:

```
servers = wntServiceAt("", "", -1, "Browser", 101) ; display name
servers = wntServiceAt("", "", -1, "Computer Browser", 1001) ;service name
```

Flags

'Flags' specifies information on the service being looked for. It consists of one entry from each of the following three groups, added together:

Req# service type

- 1 services
- 2 drivers
- 3 both

Req# service state

- 100 active services
- 200 inactive services
- 300 both

Name type indicates what the 'service-name' parameter represents.

Req# name type

- 0 display name (the name shown in Control Panel)
- 1000 service name (the actual registry key name)

Note: This function can take a while to run, depending on how many servers are in the domain. Also, it will only return the names of servers which it is able to access, which requires that the user have browse access to their service control managers.

Example:

```
;return a list of all servers running the "Spooler" service  
servers = wntServiceAt("", "", -1, "Spooler", 101)  
AskItemList("Lan Manager Servers",servers,@tab,@sorted,@single)
```

```
;return a list of all NT machines with an "Atdisk" driver installed  
servers = wntServiceAt("", "", 4096, "Atdisk", 302)  
AskItemList("Lan Manager Servers",servers,@tab,@sorted,@single)
```

wntServiceInf

Returns information about a server's type.

Syntax:

wntServiceInf(server-name)

Parameters:

(s) server-name the UNC name of a server (eg, "\\SERVER1"),
or a blank string ("") to indicate the local machine.

Returns:

(i) a bitmask indicating the type of server, or 0 on error.
The individual flag bits in the bitmask can be extracted
using the binary AND("&") operator. See below.

Return Values	Server Type
1	All LAN Manager workstation
2	All LAN Manager server
4	Any server running with Microsoft SQL Server
8	Primary domain controller
16	Backup domain controller
32	Server running the timesource service
64	Apple File Protocol servers
128	Novell servers
256	LAN Manager 2.x Domain Member
512	Server sharing print queue
1024	Server running dialin service
2048	Xenix server
4096	Windows NT (either workstation or server)
8192	Server running Windows for Workgroups
16384	Microsoft File and Print for Netware
32768	Windows NT Non-DC server
65536	Server that can run the browser service
131072	Server running a browser service as backup
262144	Server running the master browser service
524288	Server running the domain master browser
1048576	Unknown service
2097152	Unknown service
4194304	Windows 95 or newer
-2147483648	Domain announcement

Example:

```
AddExtender("WWWNT32I.DLL")
servertype=wntServiceInf("\\SERVER1")
title=strcat("Server Type : ",servertype)
if servertype == 0
    Message("Error","There was an Error with the function wntServiceInf")
Else
    if (servertype & 4194304)
        message(title, "Windows 95 or newer")
    Else
        if (servertype & 128)
            message(title, "Novell server")
        Else
            If ( servertype & (8|16))
                message(title,"Primary or Backup Domain Controller")
            Else
                Message(title, "Other")
            EndIf
        Endif
    Endif
Endif
```

See Also:

[wntServerType](#)

wntShareAdd(server-name, resource, share-name, share-type, max-users)

Shares a resource.

Syntax:

wntShareAdd(server-name, resource, share-name, share-type, max-users)

Parameters:

- | | |
|-----------------|---|
| (s) server-name | name of a network file server or empty string ("") to indicate the current machine. |
| (s) resource | identifies the object to be shared. Resource can be a directory name (e.g., "c:\util"), a printer object (e.g., "HP LaserJet III"), or the name of a sharable device. |
| (s) share-name | name by which other users will access the resource. |
| (i) share-type | identifies the type of the 'resource' object.
0 = directory, 1 = printer, and 2 = device. |
| (i) max-users | the maximum number of users allowed, or -1 for the highest possible number. |

Returns:

- | | |
|-----|----|
| (s) | 1. |
|-----|----|

If the 'share-type' is 1 for printer both the 'server-name' and 'max-users' are ignored. Set 'server-name' to a blank string ("") and 'max-users' to -1.

Example:

```
; This example adds a share called "Public" to the C:\TEMP
; directory. It sets max-users to -1 to provide unlimited
; concurrent access to the directory.
AddExtender("WWWNT32I.DLL")
wntShareAdd("", "C:\TEMP", "Public", 0, -1)
```

See Also:

[wntShareDel](#), [wntShareSet](#)

wntShareDel(server-name, resource/share-name, share-type)

UN-shares a resource.

Syntax:

wntShareDel(server-name, resource/share-name, share-type)

Parameters:

- (s) server-name name of a network file server or empty string ("") to indicate the current machine.
- (s) resource/share-name identifies the object to be modified.
- (i) share-type identifies the type of the 'resource' object.
0 = directory, 1 = printer, and 2 = device.

Returns:

- (s) **@TRUE** if the share was deleted;
@FALSE if there was no share with the specified name.

If 'share-type' = 1 (printer), then 'resource/share-name' should specify the object's resource name (e.g., "HP LaserJet III"). Otherwise, it should specify the object's share name.

Example:

```
; This example removes the "Public" share from the system.
AddExtender("WWWNT32I.DLL")
rslt=wntShareDel("", "Public", 0)
if rslt
    Message("wntShareDel", "Share PUBLIC deleted")
else
    Message("wntShareDel", "Share PUBLIC not found")
endif
```

See Also:

[wntShareAdd](#), [wntShareSet](#)

wntShareInfo(server-name, resource/share-name, share-type, request)

Returns information about a shared resource.

Syntax:

wntShareInfo(server-name, resource/share-name, share-type, request)

Parameters:

(s) server-name	name of a network file server or empty string ("") to indicate the current machine.
(s) resource/share-name	identifies the object to be modified.
(i) share-type	identifies the type of the 'resource' object. 0 = directory, 1 = printer, and 2 = device.
(i) request	specifies the information to be returned. See below.

Returns:

(s)/(i) a string or integer, depending on "request".

"request" specifies the information to be returned, and can be one of the following:

- 0 (s) share name
- 1 (s) resource
- 2 (s) comment
- 3 (s) location
- 6 (i) share type
- 8 (i) max users
- 9 (i) current users

Note: This function can only be performed by members of the Administrators or Account Operators local group, or those with Communication, Print, or Server operator group membership.

Example:

```
AddExtender("WWWNT32I.DLL")
comment = wntGroupInfo("", "Administrators", 0, 2)
Message("Comment for 'Administrators'", comment)
```

See Also:

[wntShareAdd](#), [wntShareSet](#)

wntShareSet(server-name, resource/share-name, share-type, comment, description)

Sets additional share information for a resource.

Syntax:

```
wntShareSet(server-name, resource/share-name, share-type, comment, location)
```

Parameters:

- | | |
|-------------------------|---|
| (s) server-name | name of a network file server or empty string ("") to indicate the current machine. |
| (s) resource/share-name | identifies the object to be modified. |
| (i) share-type | identifies the type of the 'resource' object.
0 = directory, 1 = printer, and 2 = device. |
| (s) comment | text string used to describe the share or a blank string ("") if no comment is desired. |
| (s) location | a text string that can be used to describe the location of the printer, 'share-type' 1, or a blank string ("") if no location description is desired. A blank string ("") must be specified for any other 'share-type'. |

Returns:

- (s) 1.

If 'share-type' = 1 (printer), then 'resource/share-name' should specify the object's resource name (e.g., "HP LaserJet III"). Otherwise, it should specify the object's share name.

Example:

```
; This example adds a comment to the Public share. Other network users can  
; see the comment when they browser the network.  
AddExtender("WWWNT32I.DLL")  
wntShareSet("", "Public", 0, "Public Access Directory on my machine", "")
```

See Also:

[wntShareAdd](#), [wntShareDel](#)

wntSvcControl(server-name, service-name, flags, control-code)

Stops or controls a service.

Syntax:

wntSvcControl(server, service-name, flags, control-code)

Parameters:

- (s) server-name the UNC name of the server on which the function will execute (e.g., "\\MYSERVER"), or ("") for the local computer.
- (s) service-name the name of the service to be looked for.
- (i) flags specifies the type of name that "service-name" represents:
 - 0 display name (the name shown in Control Panel)
 - 1000 service name (the actual registry key name)
- (i) control code specifies the operation to be performed. See below.

Returns:

- (i) returns 1.

Control codes:

Value	Name	Meaning
1	SERVICE_CONTROL_STOP	Requests the service to stop.
2	SERVICE_CONTROL_PAUSE	Requests the service to pause.
3	SERVICE_CONTROL_CONTINUE	Requests the paused service to resume.
4	SERVICE_CONTROL_INTERROGATE	Requests the service to update immediately its current status information to the service control manager.

Or it can be a user-defined control code in the range of 128 to 255, inclusive.

Example:

```
wntSvcControl("", "Spooler", 0, 1) ; stop the Spooler service
```

See Also:

[wntSvcStart](#)

wntSvcStart(server, service-name, flags, params, delimiter)

Starts a service.

Syntax:

wntSvcStart(server, service-name, flags, params, delimiter)

Parameters:

- | | |
|------------------|--|
| (s) server-name | the UNC name of the server on which the function will execute (e.g., "\\MYSERVER"), or ("") for the local computer. |
| (s) service-name | the name of the service to be looked for. |
| (i) flags | specifies the type of name that "service-name" represents:
0 display name (the name shown in Control Panel)
1000 service name (the actual registry key name) |
| (s) params | specifies a delimited list of parameters to be passed to the service, or a blank string ("") for no parameters. You can select any character you wish to be the delimiter, and must set the "delimiter" parameter accordingly.

Note: driver services do not receive parameters. |
| (s) delimiter | specifies a single character, indicating the delimiter used for "params". For example, if "params" specifies a space-delimited list, then "delimiter" should be set to " ". If "params" is a blank string, then "delimiters" is ignored and can be set to a blank string. |

Returns:

- | | |
|-----|------------|
| (i) | returns 1. |
|-----|------------|

Example:

```
AddExtender("WWWNT32I.DLL")  
wntSvcStart("", "Spooler", 0, "", "") ; start the Spooler service
```

See Also:

[wntSvcControl](#), [wntSvcStatus](#)

wntSvcStatus(server, service-name, flags, request)

Returns status information on a service.

Syntax:

wntSvcStatus(server, service-name, flags, request)

Parameters:

- | | |
|------------------|--|
| (s) server-name | the UNC name of the server on which the function will execute (e.g., "\\MYSERVER"), or ("") for the local computer. |
| (s) service-name | the name of the service to be looked for. |
| (i) flags | specifies the type of name that "service-name" represents:
0 display name (the name shown in Control Panel)
1000 service name (the actual registry key name) |
| (i) request | specifies the information to be returned. See below. |

Returns:

- | | |
|-----|-------------|
| (i) | an integer. |
|-----|-------------|

"request" specifies the information to be returned, and can be one of the following:

- | | |
|---|-------------------------|
| 1 | ServiceType |
| 2 | CurrentState |
| 3 | ControlsAccepted |
| 4 | Win32ExitCode |
| 5 | ServiceSpecificExitCode |
| 6 | CheckPoint |
| 7 | WaitHint |

Further information on these values follows:

ServiceType:

The value returned includes one of the following service type flags to indicate the type of service. In addition, for a SERVICE_WIN32 service, the SERVICE_INTERACTIVE_PROCESS flag might be set, indicating that the service process can interact with the desktop.

Value	Name	Meaning
1	SERVICE_KERNEL_DRIVER	A service type flag that indicates a Windows

		NT device driver.
2	SERVICE_FILE_SYSTEM_DRIVER	A service type flag that indicates a Windows NT file system driver.
16	SERVICE_WIN32_OWN_PROCESS	A service type flag that indicates a Win32 service that runs in its own process.
32	SERVICE_WIN32_SHARE_PROCESS	A service type flag that indicates a Win32 service that shares a process with other services.
256	SERVICE_INTERACTIVE_PROCESS	A flag that indicates a Win32 service process that can interact with the desktop.

CurrentState:

Indicates the current state of the service. One of the following values is specified:

Value	Name	Meaning
1	SERVICE_STOPPED	The service is not running.
2	SERVICE_START_PENDING	The service is starting.
3	SERVICE_STOP_PENDING	The service is stopping.
4	SERVICE_RUNNING	The service is running.
5	SERVICE_CONTINUE_PENDING	The service continue is pending.
6	SERVICE_PAUSE_PENDING	The service pause is pending.
7	SERVICE_PAUSED	The service is paused.

ControlsAccepted:

Specifies the control codes that the service will accept and process. A user interface process can control a service by specifying a control command in the ControlService function. By default, all services accept the SERVICE_CONTROL_INTERROGATE value. Any or all of the following flags can be specified to enable the other control codes.

Value	Name	Meaning
1	SERVICE_ACCEPT_STOP	The service can be stopped. This enables the SERVICE_CONTROL_STOP value.
2	SERVICE_ACCEPT_PAUSE_CONTINUE	The service can be paused and continued. This enables the SERVICE_CONTROL_PAUSE and SERVICE_CONTROL_CONTINUE values.
4	SERVICE_ACCEPT_SHUTDOWN	The service is notified when system shutdown occurs. This enables the system to send a SERVICE_CONTROL_SHUTDOWN value to the service. The ControlService

function cannot send this control code.

Win32ExitCode:

Specifies a Win32 error code that the service uses to report an error that occurs when it is starting or stopping. To return an error code specific to the service, the service must set this value to `ERROR_SERVICE_SPECIFIC_ERROR` to indicate that the `dwServiceSpecificExitCode` member contains the error code. The service should set this value to `NO_ERROR` when it is running and on normal termination.

ServiceSpecificExitCode:

Specifies a service specific error code that the service returns when an error occurs while the service is starting or stopping. This value is ignored unless the `dwWin32ExitCode` member is set to `ERROR_SERVICE_SPECIFIC_ERROR`.

CheckPoint:

Specifies a value that the service increments periodically to report its progress during a lengthy start, stop, or continue operation. For example, the service should increment this value as it completes each step of its initialization when it is starting up. The user interface program that invoked the operation on the service uses this value to track the progress of the service during a lengthy operation. This value is not valid and should be zero when the service does not have a start, stop, or continue operation pending.

WaitHint:

Specifies an estimate of the amount of time, in milliseconds, that the service expects a pending start, stop, or continue operation to take before the service makes its next call to the `SetServiceStatus` function with either an incremented `dwCheckPoint` value or a change in `dwCurrentState`. If the amount of time specified by `dwWaitHint` passes, and `dwCheckPoint` has not been incremented, or `dwCurrentState` has not changed, the service control manager or service control program can assume that an error has occurred.

Example:

```
AddExtender("WWWNT32I.DLL")
state = wntSvcStatus("", "Spooler", 0, 2) ; get the current state of the Spooler
service

Switch state
  case 1 ; if it's stopped, start it
    wntSvcStart("", "Spooler", 0, "", "")
    break
  case 7 ; if it's paused, resume it
    wntSvcControl("", "Spooler", 0, 3)
    break
EndSwitch
```

See Also:

[wntSvcStart](#), [wntSvcControl](#)

wntUserAdd(server-name)

Adds a user account.

Syntax:

```
wntUserAdd(server-name)
```

Parameters:

(s) server-name the UNC name of the server on which the function will execute (eg, "\\MYSERVER"), or "" for the local computer.

Returns:

(i) always 1.

Before calling this function, you must use **wntUserAddDat** to set parameters for the new user account. At a minimum, you must set the "name" element. The other elements will receive default values.

Calling this function does not reset elements in the user parameter structure. So, you can set various elements, add a user, then change just the "name" element and add another user. All other elements will retain their previous values. To clear all elements, call **wntUserAddDat** specifying blank strings for "variable" and "value".

Example:

```
AddExtender("WWWNT32I.DLL")
wntUserAddDat("name", "jdoe")
wntUserAddDat("full_name", "John Doe")
wntUserAdd("")
exit
```

See Also:

[wntUserAddDat](#)

wntUserAddDat(element, value)

Sets parameter information for **wntUserAdd**. (This function sets values for elements in a user parameter structure, which is used by the wntUserAdd function.)

Syntax:

Parameters:

(s) element see below.

(s / i) value see below.

Returns:

(i) **@TRUE** on success,
 @FALSE if there was a problem.

"element" can be one of the following elements in the structure. Its type (string or integer) is shown in parentheses, followed by a description of its corresponding "value":

"name" (s):

Specifies the name of the user account. The number of characters in the name cannot exceed 256.

"password" (s):

The password for the user specified in the "name" element. The length cannot exceed 256 bytes. By convention, Windows NT limits the length of passwords to 14 characters. This convention allows LAN Manager, Windows 3.x, Windows for Workgroups 3.x, and Windows 95 clients to access a Windows NT server using the account.

"home_dir" (s):

Points to a string containing the path of the home directory of the user specified in "user_name". The string can be null.

"comment" (s):

Points to a string that contains a comment. The string can be a null string, or it can have any number of characters before the terminating null character.

"flags" (i):

Contains values that determine several features. This element can be any of the following values:

Value	Name	Meaning
2	UF_ACCOUNTDISABLE	The user's account is disabled.
8	UF_HOMEDIR_REQUIRED	The home directory is required. This value is ignored in Windows NT.
16	UF_LOCKOUT	The account is currently locked out.
32	UF_PASSWRD_NOTREQD	No password is required.
64	UF_PASSWRD_CANT_CHANGE	The user cannot change the password.
65536	UF_DONT_EXPIRE_PASSWD	Don't expire password.

The following values describe the account type. Only one value can be set.

Value	Name	Meaning
256	UF_TEMP_DUPLICATE_ACCOUNT	This is an account for users whose primary account is in another domain. This domain, but not to any domain that trusts this domain. The User Manager refers to this account type as a local user account.
512	UF_NORMAL_ACCOUNT	This is a default account type that represents a typical user.
2048	UF_INTERDOMAIN_TRUST_ACCOUNT	This is a permit to trust account for a Windows NT domain that trusts other domains.
4096	UF_WORKSTATION_TRUST_ACCOUNT	This is a computer account for a Windows NT Workstation or Windows NT Server that is a member of this domain.
8192	UF_SERVER_TRUST_ACCOUNT	This is a computer account for a Windows NT Backup Domain Controller that is a member of this domain.

"script_path" (s):

Points to a string specifying the path of the user's logon script, .CMD, .EXE, or .BAT file. The string can be null.

"full_name" (s):

Points to a string that contains the full name of the user. This string can be a null string, or it can have any number of characters before the terminating null character.

"usr_comment" (s):

Points to a string that contains a user comment. This string can be a null string, or it can have any number of characters before the terminating null character.

"workstations" (s):

Points to a string that contains the names of workstations from which the user can log on. As many as eight workstations can be specified; the names must be separated by commas (.). If you do not want to restrict the number of workstations, use a null string. To disable logons from all workstations to this account, set the UF_ACCOUNTDISABLE (2) value in the "flags" element.

"acct_expires" (i):

Specifies when the account will expire. This value is stored as the number of seconds elapsed since 00:00:00, January 1, 1970. A value of -1 indicates that the account never expires.

"max_storage" (i):

Specifies the maximum amount of disk space the user can use. Use -1 to use all available disk space.

"logon_hours" (s):

Points to a 21-byte (168 bits) bit string that specifies the times during which the user can log on. Each bit represents a unique hour in the week. The first bit (bit 0, word 0) is Sunday, 0:00 to 0:59; the second bit (bit 1, word 0) is Sunday, 1:00 to 1:59; and so on. A null pointer in this element means there is no time restriction.

Note: Bit 0 in word 0 represents Sunday from 0:00 to 0:59 only if you are in the GMT time zone. In

all other cases you must adjust the bits according to your time zone offset (for example, GMT minus 8 hours for PST).

"country_code" (i):

Specifies the country code for the user's language of choice.

"code_page" (i):

Specifies the code page for the user's language of choice.

"profile" (s):

Specifies a path to the user's profile. This value can be a null string, a local absolute path, or a UNC path.

"home_dir_drive" (s):

Specifies the drive letter assigned to the user's home directory for logon purposes.

"password_expired" (i):

Determines whether the password of the user has expired. Specify nonzero to indicate that the user must change password at next logon.

If "variable" and "value" are both set to blank strings (""), all values will be cleared from the user parameter structure.

You can specify a value of "*NULL*" to set a string element to a NULL pointer, which is not the same as a NULL string ("").

Example:

```
AddExtender("WWWNT32I.DLL")
wntUserAddDat("name", "jdoe")
wntUserAddDat("full_name", "John Doe")
wntUserAdd("")
exit
```

See Also:

[wntUserAdd](#)

wntUserDel(server-name, user-name)

Deletes a user account.

Syntax:

```
wntUserDel(server-name, user-name)
```

Parameters:

- | | |
|-----------------|--|
| (s) server-name | is the UNC name of the server on which the function will execute (eg, "\\MYSERVER"), or "" for the local computer. |
| (s) user-name | is the name of the user to be deleted. |

Returns:

- | | |
|-----|-----------|
| (i) | always 1. |
|-----|-----------|

Example:

```
AddExtender("WWWNT32I.DLL")  
wntUserDel("//Server1, "jdoe")
```

See Also:

[wntUserAdd](#), [wntUserAddDat](#)

wntUserGetDat(server-name, user-name, element)

Returns parameter information for a user account.

Syntax:

```
wntUserGetDat(server-name, user-name, element)
```

Parameters:

- | | |
|-----------------|--|
| (s) server-name | is the UNC name of the server on which the function will execute (eg, "\\MYSERVER"), or "" for the local computer. |
| (s) user-name | name of a user who has an account on "server-name". |
| (s) element | specifies the element to be returned. See wntUserAddDat for a list of valid elements. |

Returns:

- | | |
|-----|---|
| (i) | Returns a string or integer value, depending on "element".. |
|-----|---|

Example:

```
user="joe"  
theflags=wntUserGetDat("\\SERVER",user,"flags")  
if theflags & 2  
    Message(user,"Account Disabled")  
endif  
if theflags & 32  
    Message(user,"Password not required")  
endif
```

See Also:

[wntUserAddDat](#)

wntUserInfo(request)

Returns information about the currently logged-on user.

Syntax:

wntUserInfo(request)

Parameters:

(i) request specifies the information to be returned. See below.

Returns:

(s) returns a string.

"request" specifies the information to be returned, and can be one of the following:

V	Name	Meaning
0	user name	name of the user currently logged on to the workstation
1	logon domain	domain name of the user account of the user currently logged on to the workstation
2	other domains	space-delimited list of other LAN Manager domains browsed by the workstation
3	logon server	name of the computer that authenticated the server

Example:

```
AddExtender("WWWNT32I.DLL")
server = wntUserInfo(3)
Message("Logon server", server)
```

See Also:

[wntUserProps](#)

wntUserProps

Returns information about a network user.

Syntax:

```
wntUserProps(server-name, user-name, request)
```

Parameters:

- | | |
|-----------------|---|
| (s) server-name | is the name of the server on which the function will execute, or a blank string ("") to indicate the current machine. |
| (s) user-name | is the name of a user who has an account on "servername". |
| (i) request | specifies the information to be returned, and can be one of the following: |

Returns:

- | | |
|-----|------------------------------|
| (s) | returns a string. See below. |
|-----|------------------------------|

Note: the user-name parameter can be obtained, using the function **wntGetUser**.

<u>Value</u>	<u>Returns</u>
0	Username
1	Full name
2	Description
3	User profile path
4	Login script name
5	Home directory
6	Home directory logon drive
7	Privilege level ("GUEST", "USER", or "ADMIN")

Example:

```
AddExtender("WWWNT32I.DLL")
server="\\Server"
user=wntGetUser(@default)

response=wntUserProps(server, user, 0)
message("The user name is:",response)

response=wntUserProps(server, user, 1)
message("The full name is:",response)

response=wntUserProps(server, user, 2)
message("The description is:",response)

response=wntUserProps(server, user, 3)
message("The user profile path is:",response)
```

```
response=wntUserProps(server, user, 4)
message("The login script name is:",response)

response=wntUserProps(server, user, 5)
message("The home directory is:",response)

response=wntUserProps(server, user, 6)
message("The home directory logon drive is:",response)

response=wntUserProps(server, user, 7)
message("The Privilege level is:",response)

message("Finished","Retrieving information about a network user")
exit
```

See Also:

[wntUserInfo](#)

wntUserRename

Renames a user account.

Syntax:

```
wntUserRename(server-name, old-username, new-username)
```

Parameters:

- | | |
|------------------|--|
| (s) server-name | UNC name of the server on which the function will execute (eg, "\\MYSERVER"), or " " for the local computer. |
| (s) old-username | An existing account name. |
| (s) new-username | New name to be given to the account. |

Returns:

- | | |
|-----|---|
| (i) | @TRUE on success,
@FALSE if there was a problem. |
|-----|---|

See Also:

[wntUserInfo](#)

wntUserSetDat

Modifies parameter information for a user account.

Syntax:

wntUserSetDat(server-name, user-name, element, value)

Parameters:

- (s) server-name is the name of the server on which the function will execute, or a blank string ("") to indicate the current machine.
- (s) user-name is the name of a user who has an account on "servername".
- (s) element See below.
- (s) value See below.

Returns:

- (s) @TRUE on success,
@FALSE if there was a problem.

"element" can be one of the following elements in the structure. Its type (string or integer) is shown in parentheses, followed by a description of its corresponding "value":

"name" (s):

Specifies the name of the user account. The number of characters in the name cannot exceed 256.

"password" (s):

The password for the user specified in the "name" element. The length cannot exceed 256 bytes. By convention, Windows NT limits the length of passwords to 14 characters. This convention allows LAN Manager, Windows 3.x, Windows for Workgroups 3.x, and Windows 95 clients to access a Windows NT server using the account.

"home_dir" (s):

Points to a string containing the path of the home directory of the user specified in "user_name". The string can be null.

"comment" (s):

Points to a string that contains a comment. The string can be a null string, or it can have any number of characters before the terminating null character.

"flags" (i):

Contains values that determine several features. This element can be any of the following values:

Value	Name	Meaning
2	UF_ACCOUNTDISABLE	The user's account is disabled.
8	UF_HOMEDIR_REQUIRED	The home directory is required. This value is ignored in Windows NT.
16	UF_LOCKOUT	The account is currently locked out.
32	UF_PASSWRD_NOTREQD	No password is required.
64	UF_PASSWRD_CANT_CHANG	The user cannot change the password.

E
65536 UF_DONT_EXPIRE_PASSWD Don't expire password.

The following values describe the account type. Only one value can be set.

Value	Name	Meaning
256	UF_TEMP_DUPLICATE_ACCOUNT	This is an account for users whose primary account is in another domain. This domain, but not to any domain that trusts this domain. The User Manager refers to this account type as a local user account.
512	UF_NORMAL_ACCOUNT	This is a default account type that represents a typical user.
2048	UF_INTERDOMAIN_TRUST_ACCOUNT	This is a permit to trust account for a Windows NT domain that trusts other domains.
4096	UF_WORKSTATION_TRUST_ACCOUNT	This is a computer account for a Windows NT Workstation or Windows NT Server that is a member of this domain.
8192	UF_SERVER_TRUST_ACCOUNT	This is a computer account for a Windows NT Backup Domain Controller that is a member of this domain.

"script_path" (s):

Points to a string specifying the path of the user's logon script, .CMD, .EXE, or .BAT file. The string can be null.

"full_name" (s):

Points to a string that contains the full name of the user. This string can be a null string, or it can have any number of characters before the terminating null character.

"usr_comment" (s):

Points to a string that contains a user comment. This string can be a null string, or it can have any number of characters before the terminating null character.

"workstations" (s):

Points to a string that contains the names of workstations from which the user can log on. As many as eight workstations can be specified; the names must be separated by commas (.). If you do not want to restrict the number of workstations, use a null string. To disable logons from all workstations to this account, set the UF_ACCOUNTDISABLE (2) value in the "flags" element.

"acct_expires" (i):

Specifies when the account will expire. This value is stored as the number of seconds elapsed since 00:00:00, January 1, 1970. A value of -1 indicates that the account never expires.

"max_storage" (i):

Specifies the maximum amount of disk space the user can use. Use -1 to use all available disk space.

"logon_hours" (s):

Points to a 21-byte (168 bits) bit string that specifies the times during which the user can log on. Each bit represents a unique hour in the week. The first bit (bit 0, word 0) is Sunday, 0:00 to 0:59;

the second bit (bit 1, word 0) is Sunday, 1:00 to 1:59; and so on. A null pointer in this element means there is no time restriction.

Note: Bit 0 in word 0 represents Sunday from 0:00 to 0:59 only if you are in the GMT time zone. In all other cases you must adjust the bits according to your time zone offset (for example, GMT minus 8 hours for PST).

"country_code" (i):

Specifies the country code for the user's language of choice.

"code_page" (i):

Specifies the code page for the user's language of choice.

"profile" (s):

Specifies a path to the user's profile. This value can be a null string, a local absolute path, or a UNC path.

"home_dir_drive" (s):

Specifies the drive letter assigned to the user's home directory for logon purposes.

"password_expired" (i):

Determines whether the password of the user has expired. Specify nonzero to indicate that the user must change password at next logon.

If "variable" and "value" are both set to blank strings (""), all values will be cleared from the user parameter structure.

You can specify a value of "*NULL*" to set a string element to a NULL pointer, which is not the same as a NULL string ("").

Note: This function cannot rename an account. It can only modify parameter information for a user account. To rename a account use the function [wntUserRename](#).

Example:

```
AddExtender("WWWNT32I.DLL")
wntUserSetDat("", "joed", "full_name", "John Doe")
exit
```

See Also:

[wntUserAddDat](#), [wntUserRename](#), [wntUserInfo](#)

wntVersion()

Returns the version of this Extender DLL.

Syntax:

wntVersion()

Parameters:

none

Returns:

(i) the version of number of this extender Dll.

This function is used to check the version number of this Dll in cases where older DLL's exist and alternate processing is desirable. Version numbers of newer versions will be larger than that of older versions.

Example:

```
AddExtender("WWWNT32I.DLL")
a=wntVersion()
Message("Dll Version",a)
```


w9xAccessAdd(server-name, resource/share-name, user/group name, object-type, access-string)

Adds or updates access (permission) records for a resource.

Syntax:

```
w9xAccessAdd(server-name, resource/share-name, user/group-name,
              object-type, access-string)
```

Parameters:

- | | |
|-------------------------|---|
| (s) server-name | the UNC name of the NT server on which the function will execute (e.g., "\\MYSERVER"). |
| (s) resource/share-name | identifies the object to be modified. |
| (s) user/group-name | name of a user or of a group to whom access is being granted. If necessary, it can be fully qualified as 'server/user'. |
| (i) object-type | identifies the type of the 'resource/share-name' object. See below. |
| (i) access-string | the type of access that is being granted. Either a predefined access type, or a delimited list. See below. |

Returns:

- | | |
|-----|----|
| (s) | 1. |
|-----|----|

W9xAccessAdd can be used for looking up user/group names and share names (if 'resource/share-name' specifies a share).

Object-Type

'Object-type' indicates the type of object identified by 'resource/share-name', and can be one of the following:

Req#	object type
100	share (e.g., a directory accessed through a share)
200	printer (accessed through a share)
300	file or directory in an NTFS partition
400	registry key

If 'object-type' = 100 (share), then 'resource/share-name' should specify the name of the share.

If 'object-type' = 300 (file or directory in an NTFS partition), then 'resource/share-name' should be UNC (ie, "\\Server\Share\Dir").

If 'object-type' = 400 (registry key), then 'resource/share-name' should be the handle of an open

registry key (opened with the **RegOpenKey** function), or a predefined registry handle. (Registration Functions are listed in the WIL Reference Manual under "Registration Database Operations".)

Otherwise, 'resource/share-name' should specify the name of the resource (e.g., "HP LaserJet III" or "C:\UTIL\MYFILE.TXT").

Access-String

'Access-string' specifies the type of access that is being granted. It can be either (A) a predefined access type, or (B) a delimited list of one or more specific 'access-records'. Both are described below:

(A) Predefined access types:

These get translated into specific access records, as shown. It is possible that the appropriate values may vary depending on your system configuration, or among different versions of Windows NT:

Access-string	Meaning	Specific equivalent
"DirNT:List"	List	"0:2:1179817"
"DirNT:Read"	Read	"0:9:-1610612736 0:2:1179817"
"DirNT:Add"	Add	"0:2:1180086"
"DirNT:AddRead"	Add & Read	"0:9:-1610612736 0:2:1180095"
"DirNT:Change"	Change	"0:9:-536805376 0:2:1245631"
"DirNT:Full"	Full Control	"0:9:268435456 0:2:2032127"
"DirNT:None"	No Access	"1:9:268435456 1:2:2032127"
"DirShare:Read"	Read	"0:0:1179817"
"DirShare:Change"	Change	"0:0:1245631"
"DirShare:Full"	Full Control	"0:0:2032127"
"DirShare:None"	No Access	"1:0:2032127"
"File:Read"	Read	"0:0:1179817"
"File:Change"	Change	"0:0:1245631"
"File:Full"	Full Control	"0:0:2032127"
"File:None"	No Access	"1:0:2032127"
"Print:Print"	Print	"0:0:1179817"
"Print:Manage"	Manage	"0:0:1245631"
"Print:Full"	Documents	"0:0:2032127"
"Print:None"	Full Control	"1:0:2032127"
	No Access	"0:0:131080"
"Reg:Read"	Read	"0:0:131072 0:0:268435456"
"Reg:Full"	Full Control	"0:0:983052 0:0:268435456"
		"1:0:983052 1:0:268435456"

"0:2:131097"

"0:2:983103"

Note: "DirShare" is a directory accessed through a share ("object-type" = 100). "DirNT" is a directory in an NTFS partition, accessed through NT security ("object-type" = 300).

(B) Specific 'access-records':

This can be a single record, or a list of records (maximum of 10) delimited with vertical bars (|). Each record is in the format:

```
record-type:access-flags:access-rights
```

where 'record-type', 'access-flags', and 'access-rights' are each a decimal number, separated with colons (:).

It is not expected that most users will want to manually create or edit 'access-records' strings. Instead, you can use the **w9xAccessGet** function to return an 'access-records' string in the proper format for use with this function. This is useful for transferring access rights from one user to another.

A brief description of the fields in the 'access-records' string follows. Please note that any detailed explanation is beyond the scope of this document, but might be obtained from the WIN32 SDK programmers' documentation available from Microsoft and other publishers.

'record-type' (one of the following):

- 0 ACCESS_ALLOWED_ACE_TYPE
- 1 ACCESS_DENIED_ACE_TYPE

'access-flags' (0, or, one or more of the following):

- 1 OBJECT_INHERIT_ACE
- 2 CONTAINER_INHERIT_ACE
- 4 NO_PROPAGATE_INHERIT_ACE
- 8 INHERIT_ONLY_ACE

'access-rights' (one or more of the following, usually several, depending on the object type.)

Note: This is not a complete list):

- 1 FILE_LIST_DIRECTORY
- 1 (Dir)
- 2 FILE_READ_DATA (File)
- 2 FILE_ADD_FILE (Dir)
- 4 FILE_WRITE_DATA (File)
- 4 FILE_ADD_SUBDIRECTOR
- 8 Y (Dir)
- 16 FILE_APPEND_DATA (File)
- 32 FILE_READ_EA

32	FILE_WRITE_EA
64	FILE_TRAVERSE (Dir)
128	FILE_EXECUTE (File)
256	FILE_DELETE_CHILD
65536	FILE_READ_ATTRIBUTES
131072	FILE_WRITE_ATTRIBUTES
262144	DELETE
524288	READ_CONTROL
1048576	WRITE_DAC
268435456	WRITE_OWNER
536870912	SYNCHRONIZE
1073741824	GENERIC_ALL
-2147483648	GENERIC_EXECUTE
	GENERIC_WRITE
	GENERIC_READ

If any access records already exist for 'resource/share-name' for the specified 'user/group name', they will be removed before adding the records specified by 'access-string'.

Note: If you have not previously added any permissions to a share, it may implicitly have some default permissions. For example, when you create a share for a directory, it defaults to giving "Full Control" access to "Everyone". When **w9xAccessAdd** is used to create an access record for such a share, it will supersede those default permissions (i.e., the default permissions will be removed). If you wish to keep the default permissions, use **w9xAccessAdd** to set them explicitly.

Example:

```
;This example sets the share called "Public" on the current machine so
;that any member of the group "Everybody" has full access to the contents
;of the directory associated with the "Public" share. This function does
;not affect any permissions that may have been set with a NTFS file system
;with respect to the directory associated with the share.
;
AddExtender("WWW9X32I.DLL")
w9xAccessAdd("\\MYSERVER", "Public", "Everybody", 100, "DirShare:Full")
```

See Also:

[w9xAccessDel](#)

w9xAccessDel(server-name, resource/share-name, user-name, share-type)

Removes an access (permission) records from a resource.

Syntax:

w9xAccessDel(server-name, resource/share-name, user-name, share-type)

Parameters:

(s) server-name	the UNC name of the NT server on which the function will execute (e.g., "\\MYSERVER").
(s) resource/share-name	identifies the object to be modified.
(s) user-name	name of a user or of a group to whom access is being granted. If necessary, it can be fully qualified as 'server\user'.
share-type	identifies the object type of the 'resource/share-name'. 100 share (e.g., a directory accessed through a share) 200 printer (accessed through a share) 300 file or directory in an NTFS partition 400 registry key

Returns:

- (i) **@TRUE** if records were deleted,
@FALSE if no records were found.

Example:

```
; This example shows the removal of all specific references to the  
; group "Supervisors" in relation to a file "D:\NTFS\blackbook.txt"  
; on a NTFS file system for the current system. Note that this could  
; either remove specific permission to access the file, or it could  
; remove specific access denial to the file.
```

```
AddExtender("WWW9X32I.DLL")  
rslt=w9xAccessDel("", "D:\NTFS\blackbook.txt", "Supervisors", 300)  
if rslt  
    Message("w9xAccessDel", "Access records deleted")  
else  
    Message("w9xAccessDel", "No access records found")  
endif
```

See Also:

[w9xAccessAdd](#)

w9xAccessGet(server-name, resource/share-name, user-name, share-type)

Returns access (permission) records for a resource.

Syntax:

```
w9xAccessGet(server-name, resource/share-name, user-name, share-type)
```

Parameters:

- (s) server-name the UNC name of the NT server on which the function will execute (e.g., "\\MYSERVER").
- (s) resource/share-name identifies the object to be modified.
- (s) user-name name of a user or of a group to whom access is being granted. If necessary, it can be fully qualified as 'server\user'.
- (i) share-type identifies the type of the 'resource/share-name' object.

Returns:

- (i) a delimited list of access records or ("") if no records were found.

W9xAccessGet returns a list of access records for 'resource/share-name' for the specified 'user/group name', delimited with vertical bars (|). If there are no appropriate records, it returns a blank string ("").

See **w9xAccessAdd** for information on the format of the access records.

Note: If no permissions were previously added to a share, it may implicitly have some default permissions. For example, when a share is created for a directory, it defaults to giving "Full Control" access to "Everyone", although there may not actually be any access records for the share. Therefore, **w9xAccessGet** may return a blank string (""). However, implicit permissions will become actual permissions when from the File Manager (or Explorer) the "Permissions" dialog for the share is brought up and "OK" is selected. **w9xAccessGet** can then retrieve the actual permissions.

Example:

```
records=w9xAccessGet("", "Public", "jdoe", 100)
if records=="
    Message("w9xAccessGet", "No records found for share 'Public'")
else
Message("w9xAccessGet", strcat("'Public' Share Records are", @crlf, records))
endif
```

See Also:

[w9xAccessDel](#), [w9xAccessAdd](#)

w9xAccessList(server-name, resource/share-name, object-type, flags)

Returns list of users who have access (permission) records for a resource.

Syntax:

w9xAccessList(server-name, resource/share-name, object-type, access string)

Parameters:

- (s) server-name the UNC name of the NT server on which the function will execute (e.g., "\\MYSERVER").
- (s) resource/share-name identifies the object to be modified.
- (i) object-type indicates the type of object identified by 'resource/share-name'. See below.
- (i) flags specifies the format of the returned names.

Returns:

- (s) a tab-delimited list of users and groups who have access records for "resource/share-name"; ie, users and groups for whom permissions have explicitly been set. Returns a blank string ("") if there are no appropriate records.

Object-Type

'Object-type' indicates the type of object identified by 'resource/share-name', and can be one of the following:

Req#	object type
100	share (e.g., a directory accessed through a share)
200	printer (accessed through a share)
300	file or directory in an NTFS partition
400	registry key

If '**object-type**' = 100 (share), then 'resource/share-name' should specify the name of the share.

If '**object-type**' = 400 (registry key), then 'resource/share-name' should be the handle of an open registry key (opened with the **RegOpenKey** function), or a predefined registry handle. (Registration Functions are listed in the WIL Reference Manual under "Registration Database Operations".)

Otherwise, 'resource/share-name' should specify the name of the resource (e.g., "HP LaserJet III" or "C:\UTIL\MYFILE.TXT").

Flags

'Flag' specifies the format of the returned names, and can be one of the following:

- 0 account (eg, "johndoe")
- 1 domain\account (eg, "OFFICE\johndoe")

Note: For built-in accounts which are predefined by the system (eg, "Administrators"), only the account name is returned, regardless of the "flag" setting.

Example:

```
;This example sets the share called "Public" on the current machine so
;that any member of the group "Everybody" has full access to the contents
;of the directory associated with the "Public" share. This function does
;not affect any permissions that may have been set with a NTFS file system
;with respect to the directory associated with the share.
;
AddExtender("WWW9X32I.DLL")
w9xAccessAdd("", "Public", "Everybody", 100, "DirShare:Full")
users = w9xAccessList("", "Public", 100, 0)
```

See Also:

[w9xAccessDel, w9xAccessAdd](#)

w9xListGroup(server-name, group-type)

Returns tab-delimited list of all user groups on a specified server.

Syntax:

w9xListGroup(server-name, group-type)

Parameters:

- | | |
|-----------------|--|
| (s) server-name | the UNC name of the NT server on which the function will execute (e.g., "\\MYSERVER"). |
| (i) group-type | @GLOBALGROUP. |

Returns:

- | | |
|-----|--------------------------------------|
| (i) | a tab-delimited list of user groups. |
|-----|--------------------------------------|

Example:

```
AddExtender("WWW9X32I.DLL")
globalgroups=w9xListGroup("",@GLOBALGROUP)
AskItemList("Global Groups",globalgroups,@tab,@sorted,@single)
```

See Also:

[w9xMemberList](#)

w9xMemberDel(server-name, group-name, user-name, group-type)

Deletes the specified user from the specified group on the specified server.

Syntax:

```
w9xMemberDel(server-name, group-name, user-name, group-type)
```

Parameters:

- | | |
|-----------------|--|
| (s) server-name | the UNC name of the NT server on which the function will execute (e.g., "\\MYSERVER"). |
| (s) group-name | name of the group. |
| (s) user-name | name of the current user. |
| (i) group-type | @GLOBALGROUP . |

Returns:

- | | |
|-----|--|
| (i) | @TRUE if successful;
@FALSE if the user is not a member of the group. |
|-----|--|

Assuming that the person running this script has sufficient authority to delete users from the specified group, this function will delete the specified user from the group.

Example:

```
AddExtender("WWW9X32I.DLL")
rslt=w9xMemberDel("", "testgroup", "jdoe", @GLOBALGROUP)
if rslt
    Message("jdoe", "removed from testgroup")
else
    Message("jdoe", "is not a member of testgroup")
endif
```

See Also:

[w9xMemberGet](#), [w9xMemberSet](#)

w9xMemberGet(server-name, group-name, user-name, group-type)

Determines if the specified user is a member of the specified group on the specified server.

Syntax:

```
w9xMemberGet(server-name, group-name, user-name, group-type)
```

Parameters:

(s) server-name	the UNC name of the NT server on which the function will execute (e.g., "\\MYSERVER").
(s) group-name	name of the group.
(s) user-name	name of the current user.
(i) group-type	@GLOBALGROUP

Returns:

(i)	@TRUE if successful; @FALSE if unsuccessful.
-----	---

Assuming that the person running this script has sufficient authority to query members of the specified group, this function will allow the person to determine if the user is a member of the specified group or not.

Example:

```
AddExtender("WWW9X32I.DLL")
rslt=w9xMemberGet("", "testgroup", "jdoe", @GLOBALGROUP)
if rslt
    Message("jdoe", "is a member of testgroup")
else
    Message("jdoe", "is not a member of testgroup")
endif
```

See Also:

[w9xMemberSet](#), [wntMemberDel](#)

w9xMemberGrps(server-name, user-name, group-type, flags)

Returns a list of groups to which the specified user belongs.

Syntax:

w9xMemberGrps(server-name, user-name, group-type, flags)

Parameters:

- | | |
|-----------------|--|
| (s) server-name | the UNC name of the NT server on which the function will execute (e.g., "\\MYSERVER"). |
| (s) user-name | name of the current user. |
| (i) group-type | @GLOBALGROUP |
| (i) flags | 0. |

Returns:

- | | |
|-----|---------------------------------|
| (i) | a tab delimited list of groups. |
|-----|---------------------------------|

Example:

```
AddExtender("WWW9X32I.DLL")
groups=w9xMemberGrps("", "jdoe", @GLOBALGROUP, 0)
AskItemList("jdoe is associated with", groups, @tab, @sorted, @single)
```

See Also:

[w9xMemberSet](#), [wntMemberDel](#)

w9xMemberList(server-name, group-name, group-type)

Returns a list of the members of a user group.

Syntax:

w9xMemberList(server-name, group-name, group-type)

Parameters:

- | | |
|-----------------|--|
| (s) server-name | the UNC name of the NT server on which the function will execute (e.g., "\\MYSERVER"). |
| (s) group-name | name of the group. |
| (i) group-type | @GLOBALGROUP |

Returns:

- | | |
|-----|--------------------------------|
| (i) | a tab delimited list of users. |
|-----|--------------------------------|

Example:

```
AddExtender("WWW9X32I.DLL")
people=w9xMemberList("", "Everybody", @GLOBALGROUP)
AskItemList("Member of group Everybody", people, @tab, @sorted, @single)
```

See Also:

[w9xMemberSet](#), [wntMemberDel](#)

w9xMemberSet(server-name, group-name, user-name, group-type)

Sets the specified user as a member of the specified group on the specified server.

Syntax:

w9xMemberSet(server-name, group-name, user-name, group-type)

Parameters:

- | | |
|-----------------|--|
| (s) server-name | the UNC name of the NT server on which the function will execute (e.g., "\\MYSERVER"). |
| (s) group-name | name of the group. |
| (s) user-name | name of the current user. |
| (i) group-type | @GLOBALGROUP. |

Returns:

- | | |
|-----|--|
| (i) | @TRUE if successful;
@FALSE if the user is already a member of the group. |
|-----|--|

Assuming that the person running this script has sufficient authority to add users to the specified group, this function will add the specified user to the group.

Example:

```
AddExtender("WWW9X32I.DLL")
rslt=w9xMemberSet("", "testgroup", "jdoe", @GLOBALGROUP)
if rslt
    Message("jdoe", "is a NEW member of testgroup")
else
    Message("jdoe", "is ALREADY a member of testgroup")
endif
```

See Also:

[w9xMemberDel](#), [w9xMemberGet](#)

w9xServiceAt(server, domain, server-type, service-name, flags)

Lists all servers in a domain which contain a specified service.

Syntax:

w9xServiceAt(server, domain, server-type, service-name, flags)

Parameters:

- (s) server-name the UNC name of the NT server on which the function will execute (e.g., "\\MYSERVER").
- (s) domain the name of the domain which will be used (e.g., "SALES"), or ("") for the primary domain.
- (i) server-type identifies the type of servers which will be examined. See below.
- (s) service-name the name of the service to be looked for.
- (i) flags specifies information on the service being looked for. See below.

Returns:

- (i) a tab-delimited list of server UNC names (e.g., "\\MYSERVER").

Note: An NT workstation can be considered to be a "server".

Server-type

Specify -1 for all servers. Or, specify one or more of the following flags, combined using the binary OR ("|") operator.

Req#	Server Type
1	All LAN Manager workstation
2	All LAN Manager server
4	Any server running with Microsoft SQL Server
8	Primary domain controller
16	Backup domain controller
32	Server running the timesource service
64	Apple File Protocol servers
128	Novell servers
256	LAN Manager 2.x Domain Member
512	Server sharing print queue
1024	Server running dialin service
2048	Xenix server
4096	Windows NT (either workstation or server)
8192	Server running Windows for Workgroups
16384	Microsoft File and Print for Netware
32768	

65536	Windows NT Non-DC server
131072	Server that can run the browser service
262144	Server running a browser service as backup
524288	Server running the master browser service
4194304	Server running the domain master browser
-2147483648	Windows 95 or newer Domain announcement

Service-Name

'Service name' is the name of a service (e.g., "Spooler") or driver (e.g., "Atdisk"). The name can be specified either as the "display name" which is listed in Control Panel (name-type = 0) or the "service name" which is the actual registry key for the service (name-type = 1000). The SDK documentation describes them as:

DisplayName = a string that is to be used by user interface programs to identify the service.

ServiceName = a string that names a service in a service control manager database.

So, the following two commands will yield identical results:

```
servers = w9xServiceAt("", "", -1, "Browser", 101) ; display name
servers = w9xServiceAt("", "", -1, "Computer Browser", 1001) ;service name
```

Flags

'Flags' specifies information on the service being looked for. It consists of one entry from each of the following three groups, added together:

Req# service type

- 1 services
- 2 drivers
- 3 both

Req# service state

- 100 active services
- 200 inactive services
- 300 both

Name type indicates what the 'service-name' parameter represents.

Req# name type

- 0 display name (the name shown in Control Panel)
- 1000 service name (the actual registry key name)

Note: This function can take a while to run, depending on how many servers are in the domain.

Also, it will only return the names of servers which it is able to access, which requires that the user have browse access to their service control managers.

Example:

```
;return a list of all servers running the "Spooler" service  
servers = w9xServiceAt("", "", -1, "Spooler", 101)  
AskItemList("Lan Manager Servers", servers, @tab, @sorted, @single)
```

```
;return a list of all NT machines with an "Atdisk" driver installed  
servers = w9xServiceAt("", "", 4096, "Atdisk", 302)  
AskItemList("Lan Manager Servers", servers, @tab, @sorted, @single)
```

w9xServerList(server, domain, server-type)

Lists all servers in a domain.

Syntax:

w9xServerList(server, domain, server-type)

Parameters:

- (s) server-name the UNC name of the NT server on which the function will execute (e.g., "\\MYSERVER").
- (s) domain the name of the domain which will be used (e.g., "SALES"), or ("") for the primary domain.
- (i) server-type identifies the type of servers which will be examined. See below.

Returns:

- (i) Returns a tab-delimited list of server UNC names (eg, "\\MYSERVER").

Note: An NT workstation can be considered to be a "server".

Server-type

Specify -1 for all servers. Or, specify one or more of the following flags, combined using the binary OR ("|") operator.

Req#	Server Type
1	All LAN Manager workstation
2	All LAN Manager server
4	Any server running with Microsoft SQL Server
8	Primary domain controller
16	Backup domain controller
32	Server running the timesource service
64	Apple File Protocol servers
128	Novell servers
256	LAN Manager 2.x Domain Member
512	Server sharing print queue
1024	Server running dialin service
2048	Xenix server
4096	Windows NT (either workstation or server)
8192	Server running Windows for Workgroups
16384	Microsoft File and Print for Netware
32768	Windows NT Non-DC server
65536	Server that can run the browser service
131072	Server running a browser service as backup
262144	Server running the master browser service
524288	Server running the domain master browser
4194304	Windows 95 or newer
-2147483648	Domain announcement

Note: This function can take a while to run, depending on how many servers are in the domain.

Also, it will only return the names of servers which it is able to access, which requires that the user have browse access to their service control managers.

Example:

```
AddExtender("www9x32i.dll")
;return a list of all servers
servers = w9xServerlist("", "", -1) ;Specify -1 for all servers
AskItemList("Lan Manager Servers", servers, @tab, @sorted, @single)

;return a list of all NT machines
servers = w9xServerlist("", "", 4096)
AskItemList("NT machines", servers, @tab, @sorted, @single)
```

w9xShareAdd(server-name, resource, share-name,share-type, max-users)

Shares a resource.

Syntax:

```
w9xShareAdd(server-name, resource, share-name,share-type, max-users)
```

Parameters:

- | | |
|-----------------|---|
| (s) server-name | the UNC name of the NT server on which the function will execute (e.g., "\\MYSERVER"). |
| (s) resource | identifies the object to be shared. Resource can be a directory name (e.g., "c:\util"), a printer object (e.g., "HP LaserJet III"), or the name of a sharable device. |
| (s) share-name | name by which other users will access the resource. |
| (i) share-type | identifies the type of the 'resource' object. 0 = directory, 1 = printer, and 2 = device. |
| (i) max-users | the maximum number of users allowed, or -1 for the highest possible number. |

Returns:

- | | |
|-----|----|
| (s) | 1. |
|-----|----|

If the 'share-type' is 1 for printer both the 'server-name' and 'max-users' are ignored. Set 'server-name' to a blank string ("") and 'max-users' to -1.

Example:

```
; This example adds a share called "Public" to the C:\TEMP
; directory. It sets max-users to -1 to provide unlimited
; concurrent access to the directory.
AddExtender("WWW9X32I.DLL")
w9xShareAdd("", "C:\TEMP", "Public", 0, -1)
```

See Also:

[w9xShareDel](#), [w9xShareSet](#)

w9xShareDel(server-name, resource/share-name, share-type)

UN-shares a resource.

Syntax:

w9xShareDel(server-name, resource/share-name, share-type)

Parameters:

- | | |
|-------------------------|--|
| (s) server-name | the UNC name of the NT server on which the function will execute (e.g., "\\MYSERVER"). |
| (s) resource/share-name | identifies the object to be modified. |
| (i) share-type | identifies the type of the 'resource' object.
0 = directory, 1 = printer, and 2 = device. |

Returns:

- | | |
|-----|--|
| (s) | @TRUE if the share was deleted;
@FALSE if there was no share with the specified name. |
|-----|--|

If 'share-type' = 1 (printer), then 'resource/share-name' should specify the object's resource name (e.g., "HP LaserJet III"). Otherwise, it should specify the object's share name.

Example:

```
; This example removes the "Public" share from the system.
AddExtender("WWW9X32I.DLL")
rslt=w9xShareDel("", "Public", 0)
if rslt
    Message("w9xShareDel", "Share PUBLIC deleted")
else
    Message("w9xShareDel", "Share PUBLIC not found")
endif
```

See Also:

[w9xShareAdd](#), [w9xShareSet](#)

w9xShareInfo(server-name, resource/share-name, share-type, request)

Returns information about a shared resource.

Syntax:

w9xShareInfo(server-name, resource/share-name, share-type, request)

Parameters:

(s) server-name	the UNC name of the NT server on which the function will execute (e.g., "\\MYSERVER").
(s) resource/share-name	identifies the object to be modified.
(i) share-type	identifies the type of the 'resource' object. 0 = directory, 1 = printer, and 2 = device.
(i) request	specifies the information to be returned. See below.

Returns:

(s)/(i)	a string or integer, depending on "request".
---------	--

"request" specifies the information to be returned, and can be one of the following:

- 0 (s) share name
- 1 (s) resource
- 2 (s) comment
- 3 (s) location
- 6 (i) share type
- 8 (i) max users
- 9 (i) current users

Note: This function can only be performed by members of the Administrators or Account Operators local group, or those with Communication, Print, or Server operator group membership.

Example:

```
AddExtender("WWW9X32I.DLL")
comment = w9xGroupInfo("", "Administrators", @GLOBALGROUP, 1)
Message("Comment for 'Administrators'", comment)
```

See Also:

[w9xShareAdd](#), [w9xShareSet](#)

w9xShareSet(server-name, resource/share-name, share-type, comment, description)

Sets additional share information for a resource.

Syntax:

```
w9xShareSet(server-name, resource/share-name, share-type, comment, location)
```

Parameters:

(s) server-name	the UNC name of the NT server on which the function will execute (e.g., "\\MYSERVER").
(s) resource/share-name	identifies the object to be modified.
(i) share-type	identifies the type of the 'resource' object. 0 = directory, 1 = printer, and 2 = device.
(s) comment	text string used to describe the share or a blank string ("") if no comment is desired.
(s) location	a text string that can be used to describe the location of the printer, 'share-type' 1, or a blank string ("") if no location description is desired. A blank string ("") must be specified for any other 'share-type'.

Returns:

(s) 1.

If 'share-type' = 1 (printer), then 'resource/share-name' should specify the object's resource name (e.g., "HP LaserJet III"). Otherwise, it should specify the object's share name.

Example:

```
; This example adds a comment to the Public share. Other network users can  
; see the comment when they browser the network.  
AddExtender("WWW9X32I.DLL")  
w9xShareSet("", "Public", 0, "Public Access Directory on my machine", "")
```

See Also:

[w9xShareAdd](#), [w9xShareDel](#)

w9xUserInfo(request)

Returns information about the currently logged-on user.

Syntax:

w9xUserInfo(request)

Parameters:

(i) request specifies the information to be returned.
See below.

Returns:

(s) returns a string.

"request" specifies the information to be returned, and can be one of the following:

Value	Name	Meaning
0	user name	name of the user currently logged on to the workstation
1	logon domain	domain name of the user account of the user currently logged on to the workstation

Example:

```
AddExtender("WWW9X32I.DLL")  
domain = w9xUserInfo(1)  
Message("Logon domain", domain)
```


w9xUserProps

Returns information about a network user.

Syntax:

```
w9xUserProps(server-name, user-name, request)
```

Parameters:

- (s) server-name is the name of the server on which the function will execute, or a blank string ("") to indicate the current machine.
- (s) user-name is the name of a user who has an account on "servername".
- (i) request specifies the information to be returned, and can be one of the following:

Returns:

- (s) returns a string. See below.

Request codes 2 - 6 will likely return a blank string.

Value	Returns
-------	---------

- 0 Username
- 1 Full name
- 2 Description
- 3 User profile path
- 4 Login script name
- 5 Home directory
- 6 Home directory logon drive
- 7 Privilege level ("GUEST", "USER", or "ADMIN")

Example:

```
AddExtender("WWW9X32I.DLL")
server="\\Server"
user="John"

response=w9xUserProps(server, user, 0)
message("The user name is:",response)

response=w9xUserProps(server, user, 1)
message("The full name is:",response)

response=w9xUserProps(server, user, 2)
message("The description is:",response)

response=w9xUserProps(server, user, 3)
message("The user profile path is:",response)
```

```
response=w9xUserProps(server, user, 4)
message("The login script name is:",response)

response=w9xUserProps(server, user, 5)
message("The home directory is:",response)

response=w9xUserProps(server, user, 6)
message("The home directory logon drive is:",response)

response=w9xUserProps(server, user, 7)
message("The Privilege level is:",response)

message("Finished","Retrieving information about a network user")
exit
```

See Also:

[w9xuserinfo](#)

w9xVersion()

Returns the version of this Extender DLL.

Syntax:

w9xVersion()

Parameters:

none

Returns:

(i) the version of number of this extender Dll.

This function is used to check the version number of this Dll in cases where older DLL's exist and alternate processing is desirable. Version numbers of newer versions will be larger than that of older versions.

Example:

```
AddExtender("WWW9X32I.DLL")
a=w9xVersion()
Message("Dll Version",a)
```

w95AccessAdd(server-name, resource, user-name, access-rights, flags)

Adds or updates an access (permission) record for a resource.

Syntax:

w95AccessAdd(server-name, resource, user-name, access-rights, flags)

Parameters:

- | | |
|-------------------|---|
| (s) server-name | name of a network file server or empty string ("") to indicate the current machine. |
| (s) resource | identifies the object to be shared. Resource can be a directory name (e.g., "c:\util"), a printer object (e.g., "\PRINT\HP LaserJet III"), or the name of a sharable device. Printers must be specified in the form, "\PRINT\ <share-name>", where "<share-name>" is substituted with the actual share name.</share-name> |
| (s) user-name | name of a user or of a group to whom access is being granted. |
| (i) access-rights | the type of access that is being granted to "user-name". (determined by the operating system in use) |
| (i) flags | 0 - reserved for future use. |

Returns:

- | | |
|-----|----|
| (s) | 1. |
|-----|----|

Access-rights

Under Windows 95/98, '**access-rights**' can be one of the following standard permission types:

Access request	Meaning
@ACC_READ_95	Read-only access
@ACC_FULL_95	Full access

If 'resource' specifies a printer, '**access-rights**' should be @ACC_FULL_95.

'**Access rights**' can also be a combination of the following custom permission flags, combined using the binary "OR" ("|") operator:

@ACC_READ	Read
@ACC_WRITE	Write
@ACC_CREATE	Create
@ACC_DELETE	Delete
@ACC_ATTRIB	Change attributes

@ACC_LIST List files
@ACC_CONTROL Change access control

If an access record already exists for 'resource' for the specified user, it will be updated to the specified 'access rights'.

Note: This function requires that user-level access control be used. (Check the configuration under "Network" settings.)

Example:

```
; This example shows how to add access records for jdoe to a particular  
; directory. Because of an idiosyncrasy of the 95 OS, the directory name  
; is specified instead of a more logical share name. C'est la vie.  
; This example allows full access rights to the directory for jdoe  
AddExtender("WWW9532I.DLL")  
w95AccessAdd("", "C:\TEMP", "jdoe", @ACC_FULL95, 0)
```

See Also:

[w95AccessDel](#)

w95AccessDel(server-name, resource, user-name)

Removes an access (permission) record from a resource.

Syntax:

w95AccessDel(server-name, resource, user-name)

Parameters:

- | | |
|-----------------|---|
| (s) server-name | name of a network file server or empty string ("") to indicate the current machine. |
| (s) resource | identifies the object to be shared. Resource can be a directory name (e.g., "c:\util"), a printer object (e.g., "\PRINT\HP LaserJet III"), or the name of a sharable device. Printers must be specified in the form, "\PRINT\ <share-name>", where "<share-name>" is substituted with the actual share name.</share-name> |
| (s) user-name | name of a user or the name of a group. |

Returns:

- | | |
|-----|--|
| (i) | @TRUE if records were deleted,
@FALSE if no records were found. |
|-----|--|

Note: This function requires that user-level access control be used. (Check the configuration under "Network" settings.)

Example:

```
; This example shows how to remove access records for jdoe to a particular
; directory. Because of an idiosyncrasy of the 95 OS, the directory name
; is specified instead of a more logical share name. C'est la vie.
; Note that removing access records could remove access records
; that either specifically permit or deny access to the resource.
AddExtender("WWW9532I.DLL")
```

```
rslt=w95AccessDel("", "C:\TEMP", "jdoe")
if rslt
    Message("w95AccessDel", "jdoe records to D:\TEMP via net shares removed.")
else
    Message("w95AccessDel", "No access records found")
endif
```

See Also:

[w95AccessAdd](#)

w95AddDrive(user-id, pswd, net-resource, local-drive, persist)

Maps a drive.

Syntax:

w95AddDrive(user-id, pswd, net-resource, local-drive, persist)

Parameters:

- | | |
|------------------|---|
| (s) user-id | user-id or @DEFAULT for current user |
| (s) pswd | password or @DEFAULT for current password or @NONE for nopassword |
| (s) net-resource | UNC netname of net resource |
| (s) local drive | local drive id e.g. ("K:") or @NONE for connect only |
| (s) persist | @TRUE Specifies persistent connection, one that will automatically reconnect when you reboot windows.
@FALSE Specifies a temporary connection. |

Returns:

- | | |
|-----|----------|
| (i) | always 1 |
|-----|----------|

This function allows a connection to be made to a net resource, and, optionally, a drive to be mapped to the net resource.

Example:

```
AddExtender("WWW9532I.DLL")
w95AddDrive(@default,@default,"\\SERVER\PUB","E:",@false)
RunWait("E:\EXCEL\EXCEL.EXE","\E")
w95CancelCon("E:",@false,@false)
```

See Also:

[w95DirDialog](#), [w95CancelCon](#)

w95AddPrinter(user-id, pswd, net-resource, local device, persist)

Maps a printer resource to a local port.

Syntax:

```
w95AddPrinter(user-id, pswd, net-resource, local device, persist)
```

Parameters:

(s) user-id	user-id or @DEFAULT for current user
(s) pswd	password or @DEFAULT for current password or @NONE for no password
(s) net-resource	UNC netname of net resource
(s) local device	local printer port e.g. ("lpt1") or @NONE for connect only
(s) persist	@TRUE - Specifies persistent connection, one that will automatically reconnect when you reboot windows. @FALSE - Specifies a temporary connection.

Returns:

(i)	@TRUE if the port was mapped; @FALSE the port was not mapped.
-----	--

This function allows a connection to be made to a net resource, and, optionally, a local device to be mapped to the net resource.

Example:

```
AddExtender("WWW9532I.DLL")  
w95AddPrinter(@default,@default,"\\SERVER\LJ4","lpt2",@false)
```

See Also:

[w95DirDialog](#), [w95CancelCon](#)

w95CancelCon(local drive, persist, forceflag)

Breaks a network connection.

Syntax:

w95CancelCon(local drive, persist, forceflag)

Parameters:

- | | |
|-----------------|--|
| (s) local drive | the mapped device. |
| (s) persist | @TRUE - update persistent connection table
@FALSE - do not update persistent connection table to remove this device |
| (i) forceflag | @TRUE - breaks the connection regardless of open files.
@FALSE - if files are open, connection will not be broken. |

Returns:

- | | |
|-----|--|
| (i) | @TRUE if successful; @FALSE if unsuccessful. |
|-----|--|

When a mapped local drive is specified, only that connection will be closed.

If persist is set to **@TRUE**, then the persistent connection will be updated to remove this drive mapping from the list of persistent connections.

If forceflag is set to **0**, **w95CancelCon** will not break the connection if any files on that connection are still open. If forceflag is set to **1**, the connection will be broken regardless.

Example:

```
AddExtender("WWW9532I.DLL")
w95AddDrive(@default,@default,"\\SERVER\Pub","E:",@false)
RunWait("E:\EXCEL\EXCEL.EXE","E")
w95CancelCon("E:",@false,@false)
```

See Also:

[W95AddDrive](#), [W95DirDialog](#)

w95DirDialog(flag)

Brings up a network drive connect/disconnect dialog box

Syntax:

w95DirDialog(flag)

Parameters:

(i) flag **@FALSE**=disconnect dialog
 @TRUE=connect dialog

Returns:

(i) **1**

This function prompts the user with either a standard Connect or Disconnect dialog box. The user may make or break network drive mappings via the dialog box.

Example:

```
AddExtender("WWW9532I.DLL")
err=w95DirDialog(@TRUE)
runwait("excel.exe", "/e")
w95DirDialog(@FALSE)
```

See Also:

[W95AddDrive](#)

w95FileClose(server-name, file-pathname)

Close all network connections to a file.

Syntax:

w95FileClose(server-name, file-pathname)

Parameters:

- (s) server-name "server-name" is the name of a remote server on which the function will execute, or a blank string ("") to indicate the local computer.
- (s) file-pathname "file-pathname" is a fully-qualified file name (eg,"C:\DOC\MYFILE.TXT").
NOTE: The file name
MUST be fully-qualified.

Returns:

- (i) @TRUE on success, or
@FALSE if the specified file was not open.

Note: this function will not work to do a w95FileClose on a local file that was opened by yourself on the local machine, even if it's done over a mapped drive with a UNC.

Note: you will get the 597 error if you are trying to do a w95FileClose on a file to which you do not have administrator access rights .

Example:

;;the "C:\DOC\MYFILE.TXT" on the local machine needs to be opened by
;;someone else over the network.

```
AddExtender("WWW9532I.DLL")
Dirchange(origdir)
```

```
run("excel.exe", strcat(origdir,"TestBook1.xls")) ;or any Excel file you've created
WinWaitExist("~Excel",3)
```

```
flocate=strcat(origdir,"TestBook1.xls")
```

```
Display(2,"The File is Located in:", flocate)
```

```
b=fileexist(strcat(origdir,"TestBook1.xls"))
Display(2,"FileExist for TestBook1.xls", b);this should display a 2 if open
```

```
a=w95FileClose("", flocate)
Display(2,"w95FileClose =", a)
```

See Also:

[W95AddDrive](#)

w95GetCon(local name)

Returns the name of a connected network resource.

Syntax:

w95GetCon(local name)

Parameters:

(s) local name local drive name.

Returns:

(i) name of a network resource.

w95GetCon returns the name of the network resource currently connected to a 'local name'. If the resource is not mapped a null string will be returned.

Example:

```
AddExtender("WWW9532I.DLL")
netsrc=w95GetCon("K:")
if netsrc="" then Message("Drive K: is","not mapped")
else Message("Drive K: is mapped to",netsrc)
```

See Also:

[W95AddDrive](#), [W95DirDialog](#)

w95GetUser(netname)

Returns the name of the user currently logged into the network.

Syntax:

w95GetUser(netname)

Parameters:

(s) netname - name of network or **@DEFAULT** for default network.

Returns:

(s) the user name.

This function will interrogate the network and return the current user name. **@default** will return the user id of the local user.

Example:

```
AddExtender("WWW9532I.DLL")
username=w95GetUser(@default)
Message("Current User is",username)
```

See Also:

[w95GetCon](#)

w95Resources(net-resource, scope, type, usage)

Itemizes network resources.

Syntax:

w95Resources(net-resource, scope, type, usage)

Parameters:

- | | |
|------------------|--|
| (s) net-resource | a UNC (e.g., "\\FredPC"), a domain (e.g., "SALES"), or ("") for the root of the network. |
| (i) scope | see below. |
| (i) type | see below. |
| (i) usage | see below. |

Returns:

- | | |
|-----|--|
| (s) | a tab-delimited list of network resources. |
|-----|--|

This function returns a tab-delimited list of network resources which are located immediately below the specified 'net-resource'.

'Scope' can be one of the following:

Req #	Meaning
1	All currently connected resources
2	All resources on the network
3	All remembered (persistent) connections

'Type' can be one of the following:

Req #	Meaning
0	All resources
1	All disk resources
2	All print resources
3	All disk and print resources

'Usage' can be one of the following:

Req #	Meaning
0	All resources
1	All connectable resources
2	All container resources
3	All connectable and container resources

Note: 'usage' is ignored unless 'scope' == 2.

Example:

```
ResourceRoot=""
ScopeConnectted=1
ScopeAll=2
ScopePersistant=3
TypeAll=0
TypeDisk=1
TypePrint=2
TypeDiskAndPrint=3
UsageAll=0
UsageConnectable=1
UsageContainer=2
UsageConnAndCont=4
oncancel="exit"
AddExtender (strcat (DirHome (), "WWW9532I.DLL"))
aaa=ResourceRoot
aaalist=w95Resources (aaa, ScopeAll, TypeAll, UsageAll)
level=0

while 1
  aaa%level%=aaa
  if level==0 then ButtonNames ("Down","Exit")
    else ButtonNames ("Down","Up")
  level=level+1
  goto downalevel

  :upalevel
  if level==0 then exit
  level=level-1
  if level==0 then ButtonNames ("Down","Exit")
    else ButtonNames ("Down","Up")
  aaa=aaa%level%

  :downalevel
  oncancel="goto upalevel"
  aaalist=w95Resources (aaa, ScopeAll, TypeAll, UsageAll)
  aaa=AskItemList ("Resource Viewer",aaalist,@tab,@sorted,@single)

endwhile
exit
:CANCEL
%oncancel%
exit
```

w95ServiceAt(server, domain, server-type, service-name, flags)

Lists all servers in a domain which contain a specified service.

Syntax:

w95ServiceAt(server, domain, server-type, service-name, flags)

Parameters:

- (s) server-name the UNC name of the server on which the function will execute (e.g., "\\MYSERVER"), or ("") for the local computer.
- (s) domain the name of the domain which will be used (e.g., "SALES"), or ("") for the primary domain.
- (i) server-type identifies the type of servers which will be examined. See below.
- (s) service-name the name of the service to be looked for.
- (i) flags specifies information on the service being looked for. See below.

Returns:

- (i) a tab-delimited list of server UNC names (e.g., "\\MYSERVER").

Server-type

Specify -1 for all servers. Or, specify one or more of the following flags, combined using the binary OR ("|") operator.

Req#	Server Type
1	All LAN Manager workstation
2	All LAN Manager server
4	Any server running with Microsoft SQL Server
8	Primary domain controller
16	Backup domain controller
32	Server running the timesource service
64	Apple File Protocol servers
128	Novell servers
256	LAN Manager 2.x Domain Member
512	Server sharing print queue
1024	Server running dialin service
2048	Xenix server
4096	Windows NT (either workstation or server)
8192	Server running Windows for Workgroups
16384	Microsoft File and Print for Netware
32768	Windows NT Non-DC server
65536	

131072	Server that can run the browser service
262144	Server running a browser service as backup
524288	Server running the master browser service
4194304	Server running the domain master browser
-2147483648	Windows 95 or newer Domain announcement

Service-Name

'Service name' is the name of a service (e.g., "Spooler") or driver (e.g., "Atdisk"). The name can be specified either as the "display name" which is listed in Control Panel (name-type = 0) or the "service name" which is the actual registry key for the service (name-type = 1000). The SDK documentation describes them as:

DisplayName = a string that is to be used by user interface programs to identify the service.
 ServiceName = a string that names a service in a service control manager database.

So, the following two commands will yield identical results:

```
servers = wntServiceAt("", "", -1, "Browser", 101) ; display name
servers = wntServiceAt("", "", -1, "Computer Browser", 1001) ;service name
```

Flags

'Flags' specifies information on the service being looked for. It consists of one entry from each of the following three groups, added together:

Req# service type

1	services
2	drivers
3	both

Req# service state

100	active services
200	inactive services
300	both

Name type indicates what the 'service-name' parameter represents.

Req# name type

0	display name (the name shown in Control Panel)
1000	service name (the actual registry key name)

Note: This function can take a while to run, depending on how many servers are in the domain. Also, it will only return the names of servers which it is able to access, which requires that the user have browse access to their service control managers.

Example:

```
;return a list of all servers running the "Spooler" service  
servers = w95ServiceAt("", "", -1, "Spooler", 101)  
AskItemList("Lan Manager Servers", servers, @tab, @sorted, @single)
```

```
;return a list of all 95 machines with an "Atdisk" driver installed  
servers = w95ServiceAt("", "", 4096, "Atdisk", 302)  
AskItemList("Lan Manager Servers", servers, @tab, @sorted, @single)
```

w95ServiceInf

Returns information about a server's type.

Syntax:

w95ServiceInf(server-name)

Parameters:

(s) server-name the UNC name of a server (eg, "\\SERVER1"),
or a blank string ("") to indicate the local machine.

Returns:

(i) a bitmask indicating the type of server, or 0 on error.
The individual flag bits in the bitmask can be extracted
using the binary AND("&") operator. See below.

Return Values	Server Type
1	All LAN Manager workstation
2	All LAN Manager server
4	Any server running with Microsoft SQL Server
8	Primary domain controller
16	Backup domain controller
32	Server running the timesource service
64	Apple File Protocol servers
128	Novell servers
256	LAN Manager 2.x Domain Member
512	Server sharing print queue
1024	Server running dialin service
2048	Xenix server
4096	Windows NT (either workstation or server)
8192	Server running Windows for Workgroups
16384	Microsoft File and Print for Netware
32768	Windows NT Non-DC server
65536	Server that can run the browser service
131072	Server running a browser service as backup
262144	Server running the master browser service
524288	Server running the domain master browser
1048576	Unknown service
2097152	Unknown service
4194304	Windows 95 or newer
-2147483648	Domain announcement

Example:

```
AddExtender("WWW9532I.DLL")
servertype=w95ServiceInf("\\SERVER1")
title=strcat("Server Type : ",servertype)
if servertype == 0
  Message("Error","There was an Error with the function w95ServiceInf")
Else
  if (servertype & 4194304)
    message(title, "Windows 95 or newer")
  Else
    if (servertype & 128)
      message(title, "Novell server")
    Else
      If ( servertype & (8|16))
        message(title,"Primary or Backup Domain Controller")
      Else
        Message(title, "Other")
      EndIf
    Endif
  Endif
Endif
```

See Also:

[w95ServerType](#) , [wntServerType](#)

w95ServerType

Returns a server's platform.

Syntax:

w95ServerType(server-name)

Parameters:

(s) server-name the UNC name of a server (eg, "\\SERVER1"),
or a blank string ("") to indicate the local machine.

Returns:

(i) See below.

Returns one of the following values:

Value	Meaning
-------	---------

- | | |
|---|------------------------|
| 0 | Invalid server name |
| 1 | Other |
| 2 | Windows for Workgroups |
| 3 | Windows 95 or later |
| 4 | Windows NT |

Example:

; This example returns a servers platform

```
AddExtender("WWW9532I.DLL")
type=w95ServerType("\\Server1")
switch type
case 0
  message("Error","invalid server name")
  break
case 1
  message("Server Type is:","Other")
  break
case 2
  message("Server Type is:","Windows for Workgroups")
  break
case 3
  message("Server Type is:","Windows 95 or later")
  break
case 4
  message("Server Type is:","Windows NT")
  break
EndSwitch
```

See Also:

[wntServerType](#)

w95ShareAdd(server-name, resource, share-name, share-type, flags)

Shares a resource.

Syntax:

w95ShareAdd(server-name, resource, share-name, share-type, flags)

Parameters:

- (s) server-name name of a network file server or empty string ("") to indicate the current machine.
- (s) resource identifies the object to be shared. Resource can be a directory name (e.g., "c:\util"), a printer object (e.g., "HP LaserJet III"), or the name of a sharable device.
- (s) share-name name by which other users will access the resource.
- (i) share-type see below.
- (i) flags see below.

Returns:

- (s) 1.

Share-type

The type of 'resource' is identified by the parameter 'share-type' and can be one of the following.

Req#	share type
0	directory
1	printer
2	device

Flags

'flags' specifies the access type being granted to other users. If share-level access control is being used, then 'flags' can be one of the following:

100001	read-only
100002	full
100003	depends on password

If user-level access control is being used or if 'resource' specifies a printer then 'flags' should be set to 100002.

Example:

```
; This example adds a share "Public" referencing the C:\TEMP  
; directory. Share access will depend on the specified password
```

```
AddExtender("WWW9532I.DLL")  
w95ShareAdd("", "C:\TEMP", "Public", 0, 100003)  
w95ShareSet("", "Public", "My Public Directory", "myfullpswd", "myreadonlypswd")
```

See Also:

[w95ShareDel](#), [w95ShareSet](#)

w95ShareInfo(server-name, share-name, request)

Returns information about a shared resource.

Syntax:

w95ShareInfo(server-name, share-name, request)

Parameters:

- | | |
|-----------------|---|
| (s) server-name | name of a network file server or empty string ("") to indicate the current machine. |
| (s) share-name | name by which other users will access the resource. |
| (i) request | see below. |

Returns:

- | | |
|---------|--|
| (s)/(i) | a string or integer, depending on "request". |
|---------|--|

Request

Specifies the information to be returned, and can be one of the following:

Req#	Info returned
0	(s) share name
1	(s) resource
2	(s) comment
4	(s) full password
5	(s) read password
6	(i) share type
7	(i) flags

See w95ShareAdd and w95ShareSet for information on these values.

Example:

```
AddExtender("WWW9532I.DLL")
w95ShareAdd("", "C:\TEMP", "Public", 0, 100003)
type=w95ShareInfo("", "Public", 6)
message("share type", type)
```

See Also:

[w95ShareDel](#), [w95ShareAdd](#)

w95ShareSet(server-name, share-name, comment, full-password, read-password)

Sets additional share information for a resource.

Syntax:

w95ShareSet(server-name, share-name, comment, full-password, read-password)

Parameters:

(s) server-name	name of a network file server or empty string ("" to indicate the current machine.
(s) share-name	name by which other users will access the resource.
(s) comment	a text string used to describe the share or an empty string ("").
(i) full-password	password or an empty string (""), see below.
(i) read-password	password or an empty string (""), see below.

Returns:

(s) 1.

The parameters '**full-password**' and '**read-password**' are determined the type of access control and other specified parameters.

If share-level access control is being used, and 'share-name' specifies a directory, then '**full-password**' specifies the password that is required for full access, and '**read-password**' specifies the password that is required for read-only access.

If share-level access control is being used, and 'share-name' specifies a printer, then '**full-password**' specifies the password that is required for access. Set '**read-password**' to ("" an empty string.

If user-level access control is being used, then both '**full-password**' and '**read-password**' should be set with empty strings, ("").

Example:

```
; This example adds a share "Public" referencing the C:\TEMP  
; directory. Share access will depend on the specified password
```

```
AddExtender("WWW9532I.DLL")  
w95ShareAdd("", "C:\TEMP", "Public", 0, 100003)  
w95ShareSet("", "Public", "My Public Directory", "myfullpswd", "myreadonlypswd")
```

See Also:

[w95ShareAdd](#), [w95ShareDel](#)

w95ShareDel(server-name, share-name)

UN-shares a resource.

Syntax:

```
w95ShareDel(server-name, share-name)
```

Parameters:

- | | |
|-----------------|--|
| (s) server-name | name of a network file server or an empty string ("") to indicate the current machine. |
| (s) share-name | name by which other users will access the resource. |

Returns:

- | | |
|-----|--|
| (s) | @TRUE if the share was deleted;
@FALSE if there was no share with the specified name. |
|-----|--|

Example:

;This example removes the "Public" share

```
AddExtender("WWW9532I.DLL")
rslt=w95ShareDel("", "Public")
if rslt
    Message("w95ShareDel", "Share Public removed")
else
    Message("w95ShareDel", "Share Public not found")
endif
```

See Also:

[w95ShareAdd](#), [w95ShareSet](#)

[w95GetUser](#)

w95Version()

Returns the version of this Extender DLL.

Syntax:

w95Version()

Parameters:

none

Returns:

(i) the version of number of this extender Dll.

This function is used to check the version number of this Dll in cases where older DLL's exist and alternate processing is desirable. Version numbers of newer versions will be larger than that of older versions.

Example:

```
AddExtender("WWW9532I.DLL")
a=w95Version()
Message("Dll Version",a)
```

