# **NTLua** Scripting Windows NT Administration and Logon

Version 3.0

(scuri@tecgraf.puc-rio.br)

**NTLua** is a console application to create Windows NT administration and logon scripts.

The **NTLua** Team:

Antonio Escaño Scuri

Tecgraf - Computer Graphics Technology Group, PUC-Rio, Brazil <a href="http://www.tecgraf.puc-rio.br/~scuri/ntlua">http://www.tecgraf.puc-rio.br/~scuri/ntlua</a>

#### **Overview**

NTLua is a console application to create Windows NT administration and logon scripts. It also has a Lua module that can be loaded by "require" in other Lua applications that use the <u>LuaBinaries</u> distribution.

It can be used as a Logon scripting tool. Copy the "ntlua.exe" and DLLs to the NTLOGON share folder in your logon server, edit a file called "ntlua.lua" in the same place, and configure the user logon script to "ntlua.exe". The "ntlua.lua" script will be automatically loaded at logon time.

NTLua runs only in Windows NT systems (4.0, 2000, XP, 2003 and Vista). It does not run in Windows 9x/Me.

The NTLua script log has a blue background. Normal text is printed in white, errors are printed in red, and warnings are printed in yellow.

# **Usage**

```
ntlua [options] [script [args]]
Available options are:
   -e stat execute string 'stat'
   -l name require library 'name'
   -i enter interactive mode after executing 'script'
   -v show version information
   -- stop handling options
   - execute stdin and stop handling options
   if no options are used, it will try to load a script called "r
```

The console is the same application of the stand alone Lua interpreter with an already loaded the NTLua module and a few new features considering it runs on Windows.

# **Samples**

- Simple Script simple.lua
- Logon Script ntlua.lua

- Domain User Creation mkuser.lua
- Domain User Removal <u>rmuser.lua</u>
- Others

# **History**

02 Ago 2007 - Version 3.0 - Migration to Lua 5.1. New name space "ntlua". Changed table parameter format of **cacls** function. New **writeregbin** function. New parameter to **wait** function.

26 Apr 2002 - Version 2.1.2 - Changed the behavior of the functions: **version** and **os**. Also they now support Windows XP.

19 Apr 2002 - Version 2.1.1 - New funtions: **readinisection**, **readinikeys**, **readinisections**. (Thanks to Juan Duarte for the suggestion.)

05 Oct 2001 - Version 2.1- Now only invalid parameter errors will abort scripts. Some startup code were improved. The **wait** param in **start** function is now optional. **df** now returns more precise values and accepts a unit division.

25 Oct 2000 - Version 2.0

22 Feb 1999 - Version 1.0 (only a logon script tool, this version runs also under Windows 9x)

#### To Do's

- Windows API Services, Process (List and Kill), Shell DDE/Shell Link, Shell Special Folders
- **AppActivate / SendKeys** to control an application.
- **LogToFile** log output to a file.

# **Support**

If you interested in help, send comments, critics, suggestions, etc to **scuri@tecgraf.puc-rio.br**. Please specify system and NtLua version you are using in your message.

A new version annoucement list is available at **LuaForge**: http://lists.luaforge.net/mailman/listinfo/ntlua-announce.

The **NTLua** page at **LuaForge** is available at: <a href="http://luaforge.net/projects/ntlua/">http://luaforge.net/projects/ntlua/</a>.

This program is free for every usage. The source code is public available. The author does not offer any guaranties, nor support, etc...

Looking for Lua? <a href="http://www.lua.org/">http://www.lua.org/</a>.

#### **Author**

Antonio E. Scuri (<a href="http://www.tecgraf.puc-rio.br/~scuri">http://www.tecgraf.puc-rio.br/~scuri</a>).

# Copyright

See the <u>Copyright Notice</u>, is the same copyright used by Lua and other Tecgraf libraries.

The code for changing access control lists was used from the Platform SDK tool cacls, developed by Dave Mont, and it is copyright of Microsoft.

#### **Download**

The program source code, HTML pages, samples, and pre-compiled binaries are available at LuaForge:

http://luaforge.net/project/showfiles.php?group\_id=29

#### **Documentation**

This page is available at <a href="http://www.tecgraf.puc-rio.br/~scuri/ntlua">http://www.tecgraf.puc-rio.br/~scuri/ntlua</a>.

The documentation is also available in Windows HTML Help format.

The HTML navigation uses the WebBook tool, available at <a href="http://www.tecgraf.puc-rio.br/webbook">http://www.tecgraf.puc-rio.br/webbook</a>.

.. "Make it Reusable, Make it Simple, Make it Small" ...

# Reference of the "ntlua" Name Space

#### **Variables**

```
_version - NTLua version number
```

\_script\_filename - the full path of the current script being executed from the command line

**\_script\_name** - just the file name of the script without path

**\_script\_path** - just the path of the script without file name

**Console** (available only when using the ntlua console)

**cls()** - Clears the console string.

pause() - Displays the message "Press any key to continue..." and stops the script execution.

getkey() - Waits until the user press a key and returns that key as a string.

**getline()** - Get keys until the user press Enter, then returns the keys in a string.

**ok()** - Returns a non nil value if no function failed. Scrips will abort only if there are Lua errors or invalid parameters.

#### Interaction

echo/print(text) - Prints the text string on the console. Always include a line break.

beep() - Sounds a beep.

wait(time, idle\_func) - Delay processing for time milliseconds. If
idle\_func is not nil, then it is called while the wait time is not reached.
idle\_func must be a function like:

```
function idle_func()
{
    print(".")
}

msgbox(text, title, type, icon) - Displays the standard message box dialog. type can be MB_OK, MB_OKCANCEL, MB_RETRYCANCEL, MB_YESNO, MB_YESNOCANCEL, MB_ABORTRETRYIGNORE. icon can be MB_ICONEXCLAMATION, MB_ICONINFORMATION, MB_ICONQUESTION, MB_ICONSTOP.

System Information

computername() - Returns the current computer name.

username/whoami() - Returns the current user name.

version() - Returns a detailed Operating System string description. Ex:

Microsoft Windows 2000 Workstation [Version 5.0.2195] Service Paos() - Returns the same information above but in a Lua table.

{
    system = "2000"
```

```
system = "2000",
type = "Workstation",
major = 5,
minor = 0,
build = 2195,
service = "Service Pack 1"
}
```

**datetime()** - Returns a string describing the current date and time (ex. "Thu Oct 26 11:01:51 2000").

**ipaddress()** - Returns the computer IP address if any configured. Return nil if not.

#### **Environment**

**setenv(name, value)** - Sets an environment variable in the user environment.

getsysenv(name) - Returns an environment variable if its set in the system
environment.

**getusrenv(name)** - Returns an environment variable if its set in the user environment.

**expandenv(name)** - Expands an environment variable that contains references to other environment variables. Returns the result.

# Registry

writeregbin(bkey, key\_name, value\_name, value, size) - Writes a string as a binary key in the registry. key\_name is the path. bkey can be: KEY\_CLASSES\_ROOT, KEY\_CURRENT\_USER, KEY\_LOCAL\_MACHINE, KEY\_USERS. The string can contain zeros.

writeregkey(bkey, key\_name, value\_name, value) - Writes a string in the registry. key\_name is the path. bkey can be: KEY\_CLASSES\_ROOT, KEY\_CURRENT\_USER, KEY\_LOCAL\_MACHINE, KEY\_USERS.

readregkey(bkey, key\_name, value\_name) - Reads a string from the registry and returns it. key\_name is the path. bkey can be: KEY\_CLASSES\_ROOT, KEY\_CURRENT\_USER, KEY\_LOCAL\_MACHINE, KEY\_USERS.

#### **Utilities**

writeinikey(filename, section, value\_name, value) - Writes a string in a ".INI" file.

readinikey(filename, section, value\_name) - Reads a string from a
".INI" file and returns it.

**readinikeys(filename, section)** - Reads all the keys from a ".INI" file section and returns them as a table.

**readinisection(filename, section)** - Reads all the keys and values from a ".INI" file section and returns them as a table indexed by the key names.

**readinisections(filename)** - Reads all the sections names from a ".INI" file and returns them as a table.

**logevent(servername, type, msg, datastr)** - Sends an event to the application log of the specified computer. If **servername** is nil it uses the local computer. **type** can be: EVENTLOG\_ERROR\_TYPE, EVENTLOG\_WARNING\_TYPE, EVENTLOG\_INFORMATION\_TYPE, EVENTLOG\_AUDIT\_SUCCESS, EVENTLOG\_AUDIT\_FAILURE.

**netsend(servername, dstname, msg)** - Sends a message to s specified user or computer. If **servername** is nil it uses the local computer.

**shutdown(servername, message, timeout, force, reboot)** - Initiates a shutdown and optional restart of the specified computer. **message** is optional and **timeout** is in seconds. If **servername** is nil it uses the local computer.

winexit(flag, force) - Either logs off the current user, shuts down the system, or shuts down and restarts the system. flag can be EWX\_LOGOFF, EWX\_POWEROFF, EWX\_REBOOT, EWX\_SHUTDOWN. force will force processes to terminate.

#### **Process**

**execute(command\_line)** - Starts the command line as the Explorer "open" command. So you can execute folders, documents and executables.

**start(command\_line, wait)** - Starts the command line creating a new process. **wait** forces the execution to wait for the process to terminate, can be omitted, default is no wait.

# **Net Drive Map**

netuse(driveletter, path) - Connects a drive letter to a network path.
netdel(driveletter) - Disconnects a mapped drive letter.

#### **Share**

**netshareadd(servername, sharename, path, comment)** - Adds a share to the specified computer. **comment** is optional. If **servername** is nil it uses the local computer.

**netsharedel(servername, sharename)** - Removes a share from the specified computer. If **servername** is nil it uses the local computer.

**netshareenum(servername, doshare\_func)** - Calls the specified function for each share in the specified computer. If **servername** is nil it uses the local computer. **doshare\_func** must be a function like:

```
function do_share(sharename, path, type)
{
  if (abort) then
    return 0
  else
    return 1;
  end
}
```

**type** can be: "FOLDER", "PRINTER", "DEVICE", "IPC", "SPECIAL". If **shareenum\_func** is nil it will print all the shares.

#### User

For all the functions: **servername** is the computer where the function actually executes. If nil executes at the local computer. **local** (1 or 0) specifies that the group is a local group.

netuseradd(servername, username, full\_name, comment, password,
profile\_path, script\_path, home\_dir\_drive, home\_dir\_path) Creates a new user. full\_name, comment, password, profile\_path,
script\_path, home\_dir\_drive, home\_dir\_path are optional and can be
nil.

**netuserrename**(**servername**, **oldusername**, **newusername**) - Renames the user.

```
netusersetinfo(servername, username, param_name, param) -
Changes user parameters. param_name can be: "fullname", "comment",
"profilepath", "scriptpath", "homedrive", "homepath".
```

**netusergetinfo(servername, username)** - Returns the user full name, comment, profile path, script path, home dir drive and home dir path.

netuserdel(servername, username) - Removes the user.

netuserenum(servername, douser\_func) - Calls the specified function for each user. douser\_func must be a function like:

```
function do_user(username)
{
  if (abort) then
    return 0
  else
    return 1;
  end
}
```

netusergroupsenum(servername, username, local,
dousergroup\_func) - Calls the specified function for each group the user
belongs to. dousergroup\_func must be a function like:

```
function do_usergroup(groupname)
{
  if (abort) then
    return 0
  else
    return 1;
  end
}
```

**ifmember(servername, username, local, groupname)** - Checks if the user belongs to a group. Returns nil otherwise.

netuserchangepassword(domainname, username, oldpassword,
newpassword) - Change the user password. domainname can be nil to
specify the current domain. username can be nil to specify the current user.

#### **Groups**

For all the functions: **servername** is the computer where the function actually executes. If nil executes at the local computer. **local** (1 or 0) specifies that the group is a local group.

**netgroupadduser(servername, local, groupname, username)** - Adds a user to a group.

netgroupdeluser(servername, local, groupname, username) - Removes a user from a group.

**netgroupadd(servername, local, groupname, comment)** - Creates a new group. **comment** is optional, can be nil.

netgroupdel(servername, local, groupname) - Removes a group.

netgrouprename(servername, local, oldgroupname, newgroupname) - Renames a group (this is not available in the UserManager, dont' know why).

**netgroupsetinfo(servername, local, groupname, comment)** - Changes group comment.

**netgroupgetinfo(servername, local, groupname)** - Returns the group comment.

netgroupenum(servername, local, dogroup\_func) - Calls the specified
function for each group. dogroup\_func must be a function like:

```
function do_group(groupname)
{
   if (abort) then
     return 0
   else
     return 1;
   end
}
```

#### Domain

netgetdomain() - Returns the current user domain.

**netgetpdc(servername, domainname)** - Returns the name of the Primary Domain Controller of the specified domain. If **servername** is nil it uses the local computer.

**getdir(dir, file\_mask, subdir)** - Returns a table with all the folder structure of the specified folder. Subfolder are treated as new sub tables, and files are values. **file\_mask** can restrict the file selection, if nil the default is "\*.\*". **subdir** allows recursion of subfolders.

forfiles(dir, file\_mask, subdir, dofile\_func) - Calls the specified
function for each file selected by the file\_mask inside the dir folder
structure. subdir allows recursion of subfolders. dofile\_func must be a
function like:

```
function do_file(filename, status)
{
  if (abort) then
    return 0
  else
    return 1;
  end
}
```

**status** is 1 when entering the specified folder, -1 leaving the specified folder, 0 is a file.

cacls(filefilter, subdir, continue, clear, grant{}, replace{},
revoke{}, deny{}) - Changes access control lists of the specified files.
subdir allows recurtion of subfolders. the function stops if an error occurs,
continue allows to ignore errors. clear will clear all the actual acls before
doing any operations, so replace and deny can not be used, and must be nil.
grant and replace are tables with the list of users and permissions to grant
or replace respectively in the acls (ex: {user1="F", test2= "R"}). revoke
and deny are tables with the list of users to revoke from the list or deny
access respectively. Permissions can be: "F", "C", "R", "N" (Full, Change,
Read, None). The table can also be nil, but not all of them at the same time.
For example:

```
cacls("*.pdb", 0, 0, 0, {scuri="F", support="C"}, nil, {"SYSTEM"
```

**dumpacls(filefilter, subdir)** - Dumps (print) access control lists of the specified files. **subdir** allows recurtion of subfolders.

exist(filename) - Returns a non nil value if the file exists.

windir() - Returns the Windows folder.

sysdir() - Returns the Windows System folder.

getcwd/pwd() - Returns the current folder.

mkdir/md(dirname) - Creates a folder.

rmdir/rd(dirname) - Removes an empty folder.

chdir/cd(dirname) - Changes the current folder. If dir is not specified
returns the current folder.

**copy/cp(srcfilename, dstfilename)** - Copies a file. Overwrite destination if exists.

move/mv/rename(srcfilename, dstfilename) - Moves or renames a file. Overwrite destination if exists.

**copydir(srcdirname, dstdirname)** - Copies the contents of a folder to another.

**deltree/prune(dirname)** - Removes a folder and all its contents including subfolders.

del/remove/rm/erase(filename) - Removes the file specified.

**access(filename, mode)** - Determine file-access permission. **mode** can be: "x", "w", "r", "f" (existence, can write, can read, can read or write). Returns a non nil value if mode is allowed.

attrib(filename, mask) - Changes the attributes of a file. mask can be a combination of the following: "a", "h", "o", "r", "s", "t" (archive, hidden, offline, read-only, system, temporary). If mask is nil returns the current attributes mask.

filesize(filename) - Returns the size of the file in bytes.

**filetitle(filename)** - Returns the name of the file without any path.

(filename = path / title)

**filepath(filename)** - Returns the path of the file.

**filefullpath()** - Returns the complete filename of a file. You can specify just the title and you get the current folder and add to the file tile.

**filetime(filename)** - Returns 3 strings regarding to file time: *creationtime*, *lastaccesstime*, *lastwritetime*. Uses the same format as the **datetime** function.

**fileshortpath(filename)** - Returns the short name of the file (8.3 characters).

**bintype(filename)** - Returns the binary type of the executable. It can be: "Win32", "DOS", "OS216", "PIF", "POSIX", "Win16" or nil if unknown.

where(filename) - Searches for the file in the PATH and returns the complete file name if found. Returns nil otherwise.

**df/diskfree(filename, div)** - Returns 3 values regarding to disk space: *available*, *total* and *free*. If per-user quotas are in use, the first value may be less than the total number of free bytes on the disk. If **filename** is nil uses the current folder. **div** can be "b", "Mb", "Kb" and "Gb". **div** is optional, and the default is "Mb".

### **NTLua License**

This product is free software: it can be used for both academic and commercial purposes at absolutely no cost. There are no royalties or GNU-like "copyleft" restrictions. It is licensed under the terms of the MIT license reproduced below, and so is compatible with GPL and also qualifies as Open Source software. It is not in the public domain, Tecgraf and Petrobras keep its copyright. The legal details are below.

The spirit of this license is that you are free to use the library for any purpose at no cost without having to ask us. The only requirement is that if you do use it, then you should give us credit by including the copyright notice below somewhere in your product or its documentation. A nice, but optional, way to give us further credit is to include a Tecgraf logo in a web page for your product.

The code is designed and implemented by Antonio Escaño Scuri at Tecgraf/PUC-Rio in Brazil. The implementation is not derived from licensed software.

Copyright © 1999-2007 Antonio Escaño Scuri.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,

DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.