View and Delete Local Profile List

delete_profile_with_gui_v2.ps1

```
########################################
# AUTHOR   : Ryan Mutschler
# DATE     : 8-15-2014
# EDIT     : 8-15-2014
# COMMENT  : GUI interface to delete user profiles from remote desktop session host
server
#
# VERSION  : 1    (Initial release)
# VERSION  : 2 - added support to select multiple users at once
########################################









#Setup script variables
#add computers to computers variable to search for profiles on those computers
[array] $Computers = "dcwipvmhsj001"
$log = "C:\temp\log.txt"
$date = Get-Date

#Reset variables
$selecteduser = ""
$profilelist = @()



Function SetupForm {
#Setup the form
[void] [System.Reflection.Assembly]::LoadWithPartialName("System.Windows.Forms")
```

```
[void] [System.Reflection.Assembly]::LoadWithPartialName("System.Drawing")

$objForm = New-Object System.Windows.Forms.Form
$objForm.Text = "Select user(s)"
$objForm.Size = New-Object System.Drawing.Size(300,320)
$objForm.StartPosition = "CenterScreen"
$btnDelete = New-Object System.Windows.Forms.Button
$btnDelete.Location = New-Object System.Drawing.Size(120,240)
$btnDelete.Size = New-Object System.Drawing.Size(75,23)
$btnDelete.Text = "Delete Profile"
$objForm.Controls.Add($btnDelete)

#When a user clicks the delete button get the details of the logged in users and calls the
DeleteProfile function
$btnDelete.Add_Click({
#calls the delete profile function
    DeleteProfile
    }
    )

$CancelButton = New-Object System.Windows.Forms.Button
$CancelButton.Location = New-Object System.Drawing.Size(200,240)
$CancelButton.Size = New-Object System.Drawing.Size(75,23)
$CancelButton.Text = "Cancel"
$CancelButton.Add_Click({$objForm.Close()})
$objForm.Controls.Add($CancelButton)

$objLabel = New-Object System.Windows.Forms.Label
$objLabel.Location = New-Object System.Drawing.Size(10,20)
$objLabel.Size = New-Object System.Drawing.Size(280,20)
$objLabel.Text = "Please select user to delete profile:"
$objForm.Controls.Add($objLabel)

$objListBox = New-Object System.Windows.Forms.ListBox
$objListBox.Location = New-Object System.Drawing.Size(10,40)
$objListBox.Size = New-Object System.Drawing.Size(260,300)
$objListBox.Height = 180
$objListBox.SelectionMode = "MultiExtended"

#Run through each computer in the computers variable to compile a list of unique user accounts
```

```
across all servers
ForEach ($computer in $Computers) {
#use WMI to find all users with a profile on the servers
    Try{
        [array]$users = Get-WmiObject -ComputerName $computer Win32_UserProfile -filter
"LocalPath Like 'C:\\Users\\%'" -ea stop
        }
    Catch {
         Write-Warning "$($error[0]) "
        Break
        }

#compile the profile list and remove the path prefix leaving just the usernames
$profilelist = $profilelist + $users.localpath -replace "C:\\users\\"

#filter the user names to show only unique values left to prevent duplicates from profile
existing on multiple computers
$uniqueusers = $profilelist | Select-Object -Unique | Sort-Object
}

#adds the unique users to the combo box
ForEach($user in $uniqueusers) {
    [void] $objListBox.Items.Add($user)
}

$objForm.Controls.Add($objListBox)
$objForm.Topmost = $True
$objForm.Add_Shown({$objForm.Activate()})
[void] $objForm.ShowDialog()

}

Function DeleteProfile {
ForEach ($x in $objListBox.SelectedItems) {
    #Add the path prefix back to the selected user
    $selecteduser = $x
    $selectedUser = "C:\Users\$selecteduser"

    #This section reads through all the computers and deletes the profile from all the
computers - it catches any errors.
```

```
    ForEach ($computer in $Computers) {
        Try {
            (Get-WmiObject -ComputerName $computer Win32_UserProfile | Where-Object
{$_.LocalPath -eq $selecteduser}).Delete()
            Write-Host -ForegroundColor Green "$selecteduser has been deleted from $computer"
            Add-Content $log "$date $selecteduser profile has been deleted from $computer"
        }



        Catch [System.Management.Automation.MethodInvocationException]{
            Write-Host -ForegroundColor Red "ERROR: Profile is currently locked on $computer -
please use log off user script first"
            Add-Content $log "$date $selecteduser Profile is currently locked on $computer -
please use log off user script first"
        }

        Catch [System.Management.Automation.RuntimeException] {
            Write-Host -ForegroundColor Yellow -BackgroundColor Blue "INFO: $selecteduser
Profile does not exist on $computer"
            Add-Content $log "$date INFO: $selecteduser Profile does not exist on $computer"
        }

        Catch {
            Write-Host -ForegroundColor Red "ERROR: an unknown error occoured. The error
response was $error[0]"
            Add-Content $log "$date ERROR: an unknown error occoured. The error response was
$error[0]"
        }
     }
     }

#Add a label to say process is complete
$objLabel1 = New-Object System.Windows.Forms.Label
$objLabel1.Location = New-Object System.Drawing.Size(10,100)
$objLabel1.Size = New-Object System.Drawing.Size(280,20)
$objLabel1.Text = "Deletion complete, check log for more details."
$objForm.Controls.Add($objLabel1)
```

```
#Add a view log button to view the log file
$LogButton = New-Object System.Windows.Forms.Button
$LogButton.Location = New-Object System.Drawing.Size(50,150)
$LogButton.Size = New-Object System.Drawing.Size(75,23)
$LogButton.Text = "View Log"
$LogButton.Add_Click({Invoke-Item $log})
$objForm.Controls.Add($LogButton)

}

#Check script was run as admin
If (-NOT ([Security.Principal.WindowsPrincipal]
[Security.Principal.WindowsIdentity]::GetCurrent()).IsInRole([Security.Principal.WindowsBuiltI
nRole] "Administrator"))
{
[System.Windows.Forms.MessageBox]::Show("It doesn't appear you have run this PowerShell
session with administrative rights, the script may not function correctly. If no users are
displayed please ensure you run the script again using administrive rights.")
}



#Start the form
SetupForm
```

```
$pc = qwinsta /server:SERVERNAME | select-string "Disc" | select-string -notmatch "services"

if ($pc)
{
  $pc| % {

  logoff ($_.tostring() -split ' +')[2] /server:SERVERNAME

  }
}
```

Not sure what this is anymore

```
gwmi win32_userprofile |
```

```
select @{LABEL=”last used”;EXPRESSION={$_.ConvertToDateTime($_.lastusetime)}},


LocalPath, SID
| ft -a | out-file "C:\Temp\log.txt"
```

Revision #3
Created 2023-11-10 04:51:19 UTC by Ryan
Updated 2025-03-12 15:27:48 UTC by Ryan