=+++++++++++===========URL-==================================

//copy file : <https://www.codeproject.com/Articles/16768/Deployment-made-simple-using-Powershell>

====================…….====================………==================

$scenes = Invoke-RestMethod '<http://central/rest/viewers/ENTRANCETV/scenes>'

$scenes.children | ForEach-Object { $elementIndex = [array]::IndexOf($scenes.children, $\_); Write-Host $elementIndex ": " $\_.name }

$index = Read-Host 'Which Scene (Selected number)?'

$selectedScene = $scenes.children[$index]

$sceneUrl = '<http://central/rest/viewers/ENTRANCETV/scenes/>'

$sceneUrl += $selectedScene.name

$headers = New-Object "System.Collections.Generic.Dictionary[[String],[String]]"

$headers.Add("X-HTTP-Method-Override", 'GETPAYLOAD')

$scene = Invoke-RestMethod $sceneUrl -Headers $headers

$sceneToPut = New-Object PSObject -Property @{

    renderers = $scene.renderers

    name = $scene.name

}

$json = $sceneToPut | ConvertTo-Json -Depth 10

Write-Host $json

$headers = New-Object "System.Collections.Generic.Dictionary[[String],[String]]"

$headers.Add("X-HTTP-Method-Override", 'PUTPAYLOAD')

Invoke-RestMethod '<http://central/rest/viewers/ENTRANCETV/canvases/default>' -Method Post -Headers $headers -Body $json -ContentType 'application/json'

=============………..=====================……….========================

$headers = New-Object "System.Collections.Generic.Dictionary[[String],[String]]"

$headers.Add("X-HTTP-Method-Override", 'GETPAYLOAD')

$scene = Invoke-RestMethod '<http://central/rest/viewers/ENTRANCETV/scenes/PW-I>' -Headers $headers

$json = $scene | ConvertTo-Json -Depth 10

$headers["X-HTTP-Method-Override"] = 'PUTPAYLOAD'

Invoke-RestMethod '<http://central/rest/viewers/ENTRANCETV/canvases/default>' -Method Post -Headers $headers -Body $json -ContentType 'application/json'

============…………………………….=========POPUP

$wshell = New-Object -ComObject Wscript.Shell

$wshell.Popup("Operation Completed",0,"Done",0x1)

Comobject = Wscript.shell,scripting.filesystemobject

================……….

New-PSDrive -Name "S" -Root "[\\Server01\Scripts](file:///Server01/Scripts)" -Persist -PSProvider "FileSystem"

(Get-WmiObject -Class Win32\_Service -Filter "name='WinRM'" -ComputerName Server01).StopService()

***How to setup Internt proxy***

$regKey="HKCU:\Software\Microsoft\Windows\CurrentVersion\Internet Settings"

Set-ItemProperty -path $regKey ProxyEnable -value 1

Set-ItemProperty -path $regKey ProxyServer -value $proxy

Set-ItemProperty -Path "HKCU:\Software\Microsoft\Windows\CurrentVersion\Internet Settings" -Name proxyserver squid.ind.iae.net:3128

Set-ItemProperty -Path "HKCU:\Software\Microsoft\Windows\CurrentVersion\Internet

Settings" -Name proxyOverride squid.ind.iae.net:3128

***Passing Credential……………….***

Enter encrypted password once and store to file; define variables

$path = "C:\temp\"

$passwd = Read-Host "enter desired password" -AsSecureString

$encpwd = ConvertFrom-SecureString $passwd

$encpwd > $path\passw.bin

============---------=================

# define path to store password and input password

$path = "C:\temp"

# get the encrypted password from the path

$encpwd = Get-Content c:\temp\filename.bin

# convert file to secure string

$passwd = ConvertTo-SecureString $encpwd

# define needed credential

$cred = new-object System.Management.Automation.PSCredential ind\administrator',$passwd

$encpwd = Get-Content .\passw.bin

$passwd = ConvertTo-SecureString $encpwd

$cred = new-object System.Management.Automation.PSCredential -ArgumentList

ind\administrator,$passwd

$credential = new-object -typename System.Management.Automation.PSCredential -argumentlist "myusername",$pass

do {

sleep -seconds 1

$mailboxExists = get-mailboxpermission -Identity "CN=$displayName,$DN" -User "NT AUTHORITY\SELF" -ErrorAction SilentlyContinue |fw IsValid

write-host "." -nonewline

} while (!$mailboxExists)

>> command |out-null /// wait till program ends.

>>> Function with mandatory parameter…..

[CmdletBinding()]

param (

[Parameter(Mandatory=$True)]

[string]$computername,

[Parameter(Mandatory=$True)]

[string]$logfile,

[int]$attemptcount = 5

)

=======================================….=================================

function RestartServiceIfStopped([string]$service)

{

$serviceStatus = (Get-Service $service).Status

if ($serviceStatus -eq 'Stopped'){

Start-Service -displayname $service

Write-Host "Started $service"

}

elseif ($serviceStatus -eq 'Running'){

Write-Host "Service $service is running"

}

#Write-EventLog -logname application -source MyApplication -eventid 1 -message "$service is running"

}

RestartServiceIfStopped "Spooler"

=================

Send-MailMessage -To edwilson@contoso.net -From "ed@iammred.net"

-SmtpServer ex1.contoso.net -Subject "Service Status SQL1"

-body ((gsv -cn sql1 -Name mssql\*) | out-string)

=========================Wait for installation========

$appSetup = Get-Process setup

$appSetup.WaitForExit()

==============Disable firewall==============

Set-NetFirewallProfile -Profile Domain,Public,Private -Enabled False

================Remote application Install+++++++++

$dirGM17 = "$dirOSApps\GlobalMapper\v17"

$strArgs = "/S /v/qn"

Write-Progress -Activity "Installing software..." -Status "Running installer..." -PercentComplete 50

Start-Process -FilePath $dirGM17\GM17.1.exe -ArgumentList $strArgs -Verb runAs -WorkingDirectory $dirGM17

$appSetup = Get-Process GM17.1

$appSetup.WaitForExit()

=====================….====================

Get-psdrive & get-psprovider

=======================……===============================

$teststring = "test=keep this, but not this."

$teststring -replace ".\*=" -replace ",.\*"

.\*= means any number of characters up to and including an equals sign.

,.\* means a comma followed by any number of characters.

==========================……………..========================

[DateTime] $LAST\_CHANGE\_DATETIME =

[DateTime]::Now.AddDays(-10)

=================To copy only Modified files……………..

$srcPath="C:\test"

[DateTime]$LAST\_CHANGE\_DATETIME=[DateTime]::Now

$changedFiles = get-childitem $srcPath -exclude "\*.bat" -Recurse |

where { $\_ -is [System.IO.FileInfo]

-and $\_.LastWriteTime -ge $LAST\_CHANGE\_DATETIME

-or $\_.Name -eq $DEFAULT\_ASPX

-or $\_.Name -eq "ERRORLOG" }

=======================================……..=============

///create Directory ….

if (!([System.IO.Directory]::Exists("c:\test\new"))){[System.IO.Directory]::CreateDirectory("c:\test\create")}

===============New possession for multiple computer

$a,$b = New-PSSession Boeing1,Boeing5

////Filter option

gwmi -Class win32\_operatingsystem |select PSComputername,caption,version,@{Name="servicepack";Expression={$\_.csdversion}},installdate|out-gridview

////to check string status

If ([string]::IsNullOrEmpty($caelumVersion) -eq $true)

//Invoke-command –scriptblock{ powershell.exe vlc.exe/sAll /msi /norestart ALLUSERS=1 EULA\_ACCEPT=YES

//Filter command

Get-Item -Path "C:\Windows\System32\\*" -Filter \*.exe

//$env:Path += ";C:\Program Files\GnuWin32\bin"

==================…….=====================……..===============

Compare-Object -ReferenceObject (Get-Content C:\Users\Lori\Documents\PCapps.txt) -DifferenceObject (Get-Content C:\Users\Lori\Documents\LAPTOPapps.txt)

=======================………=======================

//Uninstall Programm …….

HKLM:\Software\Microsoft\Windows\CurrentVersion\Uninstall\\*

HKLM:\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Uninstall\\*

=======================….========================

**Uninstall Package :**

PS> $app = Get-WmiObject -Class Win32\_Product -Filter "Name = '<Package Name>'"

PS> $app.Uninstall()

Silient App installation

PS> $setup=Start-Process "setup.exe" -ArgumentList "/s" -Wait

if ($setup.exitcode -eq 0)

write-host "Software Installed Succesfully"

Remote app onstalation

PS> Invoke-Command -Computername $server -ScriptBlock {Start-process "[\\servershare\Software\setup.exe](file:///servershare/Software/setup.exe)"}

Find process in wmi object

Get-WmiObject -Class Win32\_process -Filter "Name='winlogon.exe'"

If statement

if($win.ProcessId -eq 0){Write-Host hello} else {write-host sucess -ForegroundColor Green}

===========Export-csv=========

$Name = @()

$Phone = @()

Import-Csv H:\Programs\scripts\SomeText.csv |`

ForEach-Object {

$Name += $\_.Name

$Phone += $\_."Phone Number"

}

$inputNumber = Read-Host -Prompt "Phone Number"

if ($Phone -contains $inputNumber)

{

Write-Host "Customer Exists!"

$Where = [array]::IndexOf($Phone, $inputNumber)

Write-Host "Customer Name: " $Name[$Where]

}

=========================================

=======HTML File=======

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | $Header = @"  <style>  TABLE {border-width: 1px; border-style: solid; border-color: black; border-collapse: collapse;}  TH {border-width: 1px; padding: 3px; border-style: solid; border-color: black; background-color: #6495ED;}  TD {border-width: 1px; padding: 3px; border-style: solid; border-color: black;}  </style>  "@ |

Get-PSDrive | ConvertTo-Html -Property Name,Used,Provider,Root,CurrentLocation -Head $Header | Out-File -FilePath PSDrives.html

Graphical user interface, text, application, table

Description automatically generated

Example :

$Header = @"

<style>

TABLE {border-width: 1px; border-style: solid; border-color: black; border-collapse: collapse;}

TH {border-width: 1px; padding: 3px; border-style: solid; border-color: black; background-color: #6495ED;}

TD {border-width: 1px; padding: 3px; border-style: solid; border-color: black;}

</style>

"@

Get-PSDrive | ConvertTo-Html -Property Name,Used,Provider,Root,CurrentLocation -Head $Header | Out-File -FilePath C:\temp\ps.html

$ss = [System.IO.File]::ReadAllText('c:\temp\ps.html')

$body= "<p>Below is the most recent report or expired users and users expiring in the next 7 days.</p>

<p>If you created or manage any of the them system or test accounts, please update them. Additionally, if you recognize any terminated employees, please disable them for the good of the empire.</p>

<p>This message was generated automatically.</p>

"+" $ss "

$splat = @{

To = "kiran.chandran@walmartlabs.com"

#CC = "itgroup@me.com"

From = "idc@walmart.com"

Subject = "Test"

Body = $body

SmtpServer = "smtp-gw1.wal-mart.com"

BodyAsHtml = $true

}

Send-MailMessage @splat



Convert CSV to DICT :

$ge= Import-Csv C:\Temp\GE.csv

$data=@{}

$ge|%{$data[$\_.key]=$\_.value}

Graphical user interface, text, application

Description automatically generated

####### Adding Password to Powershell script :

$username = "homeoffice\ad-kmc00bg"

$password = "Boeing@123"

$secureStringPwd = $password | ConvertTo-SecureString -AsPlainText -Force

$secureStringText = $secureStringPwd | ConvertFrom-SecureString

Set-Content "C:\temp\ExportedPassword.txt" $secureStringText

$username = "homeoffice\ad-kmc00bg"

$pwdTxt = Get-Content "C:\temp\ExportedPassword.txt"

$securePwd = $pwdTxt | ConvertTo-SecureString

$creds = New-Object System.Management.Automation.PSCredential -ArgumentList $username, $securePwd

$Session = New-PSSession -ConfigurationName Microsoft.Exchange -ConnectionUri <https://webmail.wal-mart.com/Powershell?serializationLevel=Full> -Authentication Basic -Credential $creds -AllowRedirection

if ($Session -ne $null){

Import-PSSession $Session -AllowClobber}

########### Sharepoint onine update List item+++

#Config Variables

$SiteURL = "<https://teams.wal-mart.com/sites/Swamy>"

$ListName ="test1"

$ListItemID ="2"

#Connect to PNP Online

Connect-PnPOnline -Url $SiteURL –UseWebLogin

#Get List Item to Update

$ListItem = Get-PnPListItem -List $ListName -Id $ListItemID -ErrorAction Stop

#Update List Item - Internal Names of the columns : Value

Set-PnPListItem -List $ListName -Identity $ListItem -Values @{"Title" = "testSha"; "qty"="000"}

#Read more: <https://www.sharepointdiary.com/2015/07/sharepoint-online-update-list-items-using-powershell.html#ixzz6VNdwlcrE>

+++++++++++++++++++++++++++++++++++++=

+++++ String trimming

$user = $onlinemailboxupn.IndexOf(“@”)

$left = $onlinemailboxupn.Substring(0,$user)

$onlinemailboxupn1 = “$left” + “@walmart.onmicrosoft.com”

Get-Item 'C:\ProgramData\VMware\VMware Blast\Blast-Worker-SessionId2.log'|Select-String -Pattern '(loss|rtt)'|select-string -Pattern "2021-07-29" > C:\Temp\test2.txt

++++2 foreach loop=====

$data = @()

$user|foreach{

$userid = $\_

foreach($i in $((gho $\_).name))

{

$v=get-vm $i|select Name,NumCpu,MemoryGB,@{Name="user";Expression={$userid}}

$data+=$v

}}

=========