HOW TO GET REMOTE SYSTEM INFORMATION – PART 2

26/02/2018 by STEPHANOS — 5 COMMENTS

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

How To Get Remote System Information – Part 2

Scenario:

How to get remote system information – Part 2

For those who have not read Part of the series, we have discussed about creating UI with PowerShell and getting information from local and remote computer systems. In Part 2 we are going to discuss about some changes that have been applied on the script and the reason behind of those changes. The changes on the script have been done to increase the functionality, stability and be able to retrieve more information from the systems.

As you will be able to see in the synopsis of the script the below changes have been done.

- Added Fan information
- Added Battery Information
- Added Portable Battery Information
- Added Network Settings Information
- Added ping connection test of remote system
- Added Remote Desktop connection to the remote system
- Warning for the use of Win32_Product class
- Added option for Win32Reg_AddRemovePrograms class.
- Added visibility to Taskbar
- Added Help information

Let's check every point in more detail

The first four points from the above list, allow us to retrieve extra information from the local or remote system. As we have mentioned also in Part 1, this depends on the system if it can provide the information. If the system is not

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

information. For battery and portable battery, the script will get all the information available if the system is able to provide it.

For network settings, the script will get all the network devices with the related information for each device. The information for each device is the description and MAC address, if it is DHCP enabled, DHCP leased and DHCP server. It will get DNS related information, such as, DNS Domain, DNS hostname, DNS registration status and other. Of course we will get also, the IP settings of the network device like IP Address, Default Gateway, Subnet mask.

Ping Test and Remote Desktop Connection

As an extra functionality of the script I have added ping test to the remote system and remote desktop connection. For the ping test, we have used the command Test-Connection directly from PowerShell using the default setting. This will ping the remote system and provide us the result same way as a normal ping test with Command Prompt. If the Computer Name field is empty, we will receive an error on the screen that we have to provide a computer name to perform the test.

For the remote desktop connection, we actually launch the default application by Microsoft to connect to the remote systems. When we click this button, the application will be launched directly trying to connect to the specific remote system, asking for credentials to connect to it. If the computer name field is empty, the application will still be launched but it will be empty or filled with the latest connection that you have made using Remote Desktop Connection.

By the above additions in the functionality of the script we can directly get information of the hardware of the system. We are able to see the installed software of the system along with the processes running and services on the system. We can check the network settings of the system and perform a ping test to it. And at the same time will be able to directly connect to the remote system using the default remote desktop application without the need of going

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

Software Information Issue

In Part 1, we were able to get the information for the software installed on a system using Win32_Product. As you may or may not know the specific class is not stable and most of the times can cause an issue on the system. The problem appears when you are trying to apply any filter. When you are querying the system to get the information for the software, the problem appears. Below is the warning from Microsoft

> ***Warning Win32_Product*** *is not query optimized. Queries such as â€œselect * from Win32_Product where (name like â€˜Sniffer%â€™)â€□ require WMI to use the MSI provider to enumerate all of the installed products and then parse the full list sequentially to handle the "where" clause. This process also initiates a consistency check of packages installed, verifying and repairing the install. With an account with only user privileges, as the user account may not have access to quite a few locations, may cause delay in application launch and an event 11708 stating an installation failure. For more information, see KB Article 794524.*

There are two alternative ways to get the software information. The first one is to use another class which is not installed in Windows by default, called Win32Reg_AddRemovePrograms. This class can be found only if you have SCCM client installed on your computer. The second alternative is to check through the registry to get the information.

Solution to the software information problem

A temporary solution to the above situation was to enable both classes in the script. When someone will click on the â€œSoftwareâ€□ button, then the script will pop up another form and will ask you to select which class you want to use. If you select Win32Reg_AddRemovePrograms, then the script will use that class and get the information from the local or remote system that you

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

yes then the script will get the information using Win32_Product class. If the user will select No the warning message will disappear and he will need to select another option or close the form with the options.

With this way, the script warns the administrator for the use of Win32_Product class usage and has the option **NOT** to perform the action. To accomplish the above, a new form has been designed within the function of â€œSoftware_infoâ€□. At the point of software information we have nested functions that allow us to perform the above operations. During the process I have described above, We are using a form and a message box. The message boxes have specific values in terms of buttons. You cannot define your own buttonsâ€™ text, but you can define what will done when the specific button will be clicked. As we want to have custom buttons for the class options and we are not able to have it with message box, we create our own custom form.

Requirements

First of all as we are using CIM commands you will need to have at least PowerShell 3.0 installed on the computer/ server that you will run the script. Secondly the user that will run the script need to have remote management permission on the remote system to be able to connect and get the information. The last thing that you need to ensure is that the remote system will need to run WINRM to be able to connect to it. WMI commands are using DCOM connection to remote system. CIM commands are using WSMAN to connect to the remote system. If you are not running WINRM on them then you will not be able to get any information. Note that script will work if you run it locally as CIM uses DCOM to connect to the local system and if no computer name will be specified the script will provide information of the local system.

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

Hope you like it. If you have any questions or anything else please let me know in the comments below.

Stay tuned for the next part of this series.

Related Links:

- How to get remote system information – Part 1
- How to get remote system information – Part 3
- How to get remote system information – Part 4
- Win32_Product class (Windows) – MSDN – Microsoft
- PowerShell script to get computer information
- Get-CimInstance (CimCmdlets) – Microsoft Docs
- Out-String – Microsoft Docs
- Add-Type – Microsoft Docs
- New-Object – Microsoft Docs

Solution / Script:

```
<#
.SYNOPSIS
  Name: Get-SysInfo.ps1
  The purpose of this script is to retrieve information of remote system

.DESCRIPTION
  This is a simple script with UI to retrieve information of remote syst
  software and peripherals.

  It will gather hardware specifications, peripherals, installed softwar
  and Operating System through a very simple and functioning GUI.

.RELATED LINKS
  Home

.NOTES
  Version:      1.1
  Updated:      25-02-2018      - Added Fan Information
```

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

```
                                - Warning for the use of Win32_Product cl
                                - Added option for Win32Reg_AddRemoveProg
                                - Added visibility to Taskbar

  Release Date: 22-02-2018

  Author: Stephanos Constantinou

.EXAMPLE
  Run the Get-SysInfo script to retrieve the information.
  Get-SysInfo.ps1
#>

$System_info = {
   $ComputerName = $txt_ComputerName.Text
   $System = Get-CimInstance -Class Win32_ComputerSystem -ComputerName $
   $lbl_sysinfo.Text = $System | FL -Property Name,
                                              Manufacturer,

                                              Model,
                                              PartOfDomain,
                                              Domain,
                                              Workgroup,
                                              DNSHostName,
                                              NumberOfProcessors,
                                              NumberOfLogicalProcessors,
                                              TotalPhysicalMemory,
                                              CurrentTimeZone,
                                              DaylightInEffect,
                                              HypervisorPresent,
                                              PrimaryOwnerName,
                                              UserName | Out-String}

$bios_info = {
   $ComputerName = $txt_ComputerName.Text
   $Bios = Get-CimInstance -Class Win32_BIOS -ComputerName $ComputerName
   $lbl_sysinfo.Text = $Bios | FL -Property Name,
                                            SerialNumber,
                                            Version,
                                            BIOSVersion,
                                            ReleaseData | Out-String}

$CPU_info = {
```

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

```
                                        Name,
                                        Caption,
                                        L2CacheSize,
                                        L3CacheSize,
                                        LoadPercentage,
                                        CurrentClockSpeed | Out-Strin

$RAM_info = {
   $ComputerName = $txt_ComputerName.Text
   $RAM = Get-CimInstance -Class Win32_PhysicalMemory -ComputerName $Com
   $lbl_sysinfo.Text = $RAM | FL -Property Tag,
                                        DeviceLocator,
                                        Manufacturer,
                                        PartNumber,
                                        SerialNumber,
                                        Capacity,
                                        Speed | Out-String}

$MB_info = {
   $ComputerName = $txt_ComputerName.Text
   $MB = Get-CimInstance -Class Win32_BaseBoard -ComputerName $ComputerN
   $lbl_sysinfo.Text = $MB | FL -Property Manufacturer,
                                        Model,
                                        Version | Out-String}

$PhysicalDrives_info = {
   $ComputerName = $txt_ComputerName.Text
   $PhysicalDrives = Get-CimInstance -Class Win32_DiskDrive -ComputerNam
   $lbl_sysinfo.Text = $PhysicalDrives | FL -Property DeviceID,
                                                FirmwareRevision,
                                                Manufacturer,
                                                Model,
                                                MediaType,
                                                SerialNumber,
                                                InterfaceType,
                                                Partitions,
                                                Size,
                                                TotalCylinders,
                                                TotalHeads,
                                                TotalSectors,
                                                TotalTracks,
                                                TracksPerCylinderB
                                                SectorsPerTrack
```

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

```
$LogicalDrives_info = {
   $ComputerName = $txt_ComputerName.Text
   $LogicalDrives = Get-CimInstance -Class Win32_LogicalDisk -ComputerNa
   $lbl_sysinfo.Text = $LogicalDrives | FL -Property DeviceID,
                                                     Description,
                                                     VolumeName,
                                                     ProviderName,
                                                     Size,
                                                     FreeSpace,
                                                     VolumeSerialNumber,
                                                     FileSystem,
                                                     Compressed | Out-St

$GPU_info = {
   $ComputerName = $txt_ComputerName.Text
   $GPU = Get-CimInstance -Class Win32_VideoController -ComputerName $Co
   $lbl_sysinfo.Text = $GPU | FL -Property DeviceID,
                                          Name,
                                          VideoProcessor,

                                          AdapterDACType,
                                          AdapterRAM,
                                          DriverDate,
                                          DriverVersion,
                                          VideoModeDescription,
                                          CurrentBitsPerPixel,
                                          CurrentHorizontalResolution,
                                          CurrentVerticalResolution,
                                          CurrentNumberOfColors,
                                          CurrentRefreshRate,
                                          MaxRefreshRate,
                                          MinRefreshRate,
                                          Status | Out-String}

$Network_info = {
   $ComputerName = $txt_ComputerName.Text
   $Network = Get-CimInstance -Class Win32_NetworkAdapter -ComputerName
   $lbl_sysinfo.Text = $Network | FL -Property DeviceID,
                                               Name,
                                               Manufacturer,
                                               ProductName,
                                               ServiceName,
                                               MACAddress,
                                               AdapterType
```

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

```
$NetSettings_info = {
   $ComputerName = $txt_ComputerName.Text
   $NetSettings = Get-CimInstance -Class Win32_NetworkAdapterConfigurati
   $lbl_sysinfo.Text = $NetSettings | FL -Property Description,
                                                   DHCPEnabled,
                                                   DHCPLeaseObtained,
                                                   DNSDomain,
                                                   DNSDomainSuffixSearch
                                                   DHCPServer,
                                                   DNSHostName,
                                                   DNSServerSearchOrder,
                                                   DomainDNSRegistration
                                                   FullDNSRegistrationEn
                                                   IPEnabled,
                                                   IPAddress,
                                                   DefaultIPGateway,
                                                   IPSubnet,
                                                   MACAddress,
                                                   ServiceName | Out-Str

$Monitor_info = {
   $ComputerName = $txt_ComputerName.Text
   $Monitor = Get-CimInstance -Class Win32_DesktopMonitor -ComputerName
   $lbl_sysinfo.Text = $Monitor | FL -Property DeviceID,
                                               Name,
                                               MonitorManufacturer,
                                               MonitorType,
                                               PixelsPerXLogicalInch,
                                               PixelPerYLogicalInch,
                                               ScreenHeight,
                                               ScreenWidth,
                                               Status | Out-String}

$OS_info = {
   $ComputerName = $txt_ComputerName.Text
   $OS = Get-CimInstance -Class Win32_OperatingSystem -ComputerName $Com
   $lbl_sysinfo.Text = $OS | FL -Property Name,
                                          Manufacturer,
                                          Caption,
                                          Version,
                                          MUILanguages
```

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

```
                                        PortableOperatingSystem,
                                        Primary,
                                        BootDevice,
                                        LastBootUpTime,
                                        LocalDateTime,
                                        CurrentTimeZone,
                                        RegisteredUser,
                                        SerialNumber,
                                        SystemDevice,
                                        SystemDirectory,
                                        SystemDrive,
                                        WindowsDirectory,
                                        EncryptionLevel,
                                        FreePhysicalMemory,
                                        FreeSpaceInPagingFiles,
                                        FreeVirtualMemory,
                                        SizeStoredInPagingFiles,
                                        TotalVirtualMemorySize,
                                        TotalVisibleMemorySize,
                                        Status | Out-String}

$Keyboard_info = {
   $ComputerName = $txt_ComputerName.Text
   $Keyboard = Get-CimInstance -Class Win32_Keyboard -ComputerName $Compu
   $lbl_sysinfo.Text = $Keyboard | FL -Property Description,
                                                Caption,
                                                NumberOfFunctionKeys | Ou

$Mouse_info = {
   $ComputerName = $txt_ComputerName.Text
   $Mouse = Get-CimInstance -Class Win32_PointingDevice -ComputerName $Co
   $lbl_sysinfo.Text = $Mouse | FL -Property Description,
                                             Name,
                                             HardwareType,
                                             Manufacturer | Out-String}

$CDROM_info = {
   $ComputerName = $txt_ComputerName.Text
   $CDROM = Get-CimInstance -Class Win32_CDROMDrive -ComputerName $Comput
   $lbl_sysinfo.Text = $CDROM | FL -Property Drive,
                                             Name,
                                             Caption,
                                             Description
```

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

```
$Sound_info = {
   $ComputerName = $txt_ComputerName.Text
   $Sound = Get-CimInstance -Class Win32_SoundDevice -ComputerName $Comp
   $lbl_sysinfo.Text = $Sound | FL -Property DeviceID,
                                             Name,
                                             Manufacturer,
                                             ProductName | Out-String}

$Printers_info = {
   $ComputerName = $txt_ComputerName.Text
   $Printers = Get-CimInstance -Class Win32_Printer -ComputerName $Compu
   $lbl_sysinfo.Text = $Printers | FL -Property DeviceID,
                                                Name,
                                                HorizontalResolution,
                                                VerticalResolution,
                                                Default,
                                                DriverName,

                                                Direct,
                                                Network,
                                                Local,
                                                Hidden,
                                                KeepPrintedJobs,
                                                PrintJobDataType,
                                                PrintProcessor,
                                                PortName,
                                                Shared,
                                                ServerName,
                                                SpoolEnabled,
                                                WorkOffline,
                                                CapabilityDescriptions,
                                                Status | Out-String}

$Fan_info = {
   $ComputerName = $txt_ComputerName.Text
   $Fan = Get-CimInstance -Class Win32_Fan -ComputerName $ComputerName
   $lbl_sysinfo.Text = $Fan | FL -Property Name,
                                           Caption,
                                           Description,
                                           InstallDate,
                                           ActiveCooling,
                                           DesiredSpeed
```

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

```
   $lbl_sysinfo.Text = $Battery | FL -Property * | Out-String}

$PortBattery_info = {
   $ComputerName = $txt_ComputerName.Text
   $PortBattery = Get-CimInstance -Class Win32_Battery -ComputerName $Co
   $lbl_sysinfo.Text = $PortBattery | FL -Property * | Out-String}

$Software_info = {
   $ComputerName = $txt_ComputerName.Text

   $Product = {
      $Warning = [System.Windows.MessageBox]::Show('Are you sure that yo

      switch ($Warning){
         Yes {$SoftwareOption.Close()
              $Software = Get-CimInstance -Class Win32Reg_Product -Compu
              $lbl_sysinfo.Text = $Software | FL -Property Name,
                                                           Version,
                                                           Description,
                                                           Vendor,
                                                           InstallDate,
                                                           InstallLocati
                                                           HelpLink,
                                                           URLInfoAbout,
                                                           URLUpdateInfo
         No {Break}
      }
   }

   $AddRemove = {
      $SoftwareOption.Close()
      $Software = Get-CimInstance -Class Win32Reg_AddRemovePrograms -Com
      $lbl_sysinfo.Text = $Software | FL -Property DisplayName,
                                                   InstallDate,
                                                   Publisher,
                                                   Version | Out-String}

   $SoftwareOption = New-Object system.Windows.Forms.Form
   $SoftwareOption.Text = "Class Option"
   $SoftwareOption.Size = New-Object System.Drawing.Size(500,130)
   $SoftwareOption.AutoSize = $False
   $SoftwareOption.AutoScroll = $False
```

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

```
    $SoftwareOption.Opacity = 1
    $SoftwareOption.FormBorderStyle = "Fixed3D"
    $SoftwareOption.StartPosition = "CenterScreen"

    $lbl_SoftwareOption = New-Object System.Windows.Forms.Label
    $lbl_SoftwareOption.Location = New-Object System.Drawing.Point(20,10)
    $lbl_SoftwareOption.Size = New-Object System.Drawing.Size(500,25)
    $lbl_SoftwareOption.Text = "Please select the class that you want to
    $lbl_SoftwareOption.Font = $Font
    $SoftwareOption.Controls.Add($lbl_SoftwareOption)

    $btn_Product = New-Object System.Windows.Forms.Button
    $btn_Product.Location = New-Object System.Drawing.Point(10,50)
    $btn_Product.Size = New-Object System.Drawing.Size(230,25)
    $btn_Product.Text = "Win32_Product"
    $btn_Product.Font = $Font
    $btn_Product.Add_Click($Product)
    $SoftwareOption.Controls.Add($btn_Product)


    $btn_AddRemove = New-Object System.Windows.Forms.Button
    $btn_AddRemove.Location = New-Object System.Drawing.Point(250,50)
    $btn_AddRemove.Size = New-Object System.Drawing.Size(230,25)
    $btn_AddRemove.Text = "Win32_AddRemovePrograms"
    $btn_AddRemove.Font = $Font
    $btn_AddRemove.Add_Click($AddRemove)
    $SoftwareOption.Controls.Add($btn_AddRemove)

    $SoftwareOption.ShowDialog()
}

$Process_info = {
    $ComputerName = $txt_ComputerName.Text
    $Process = Get-CimInstance -Class Win32_Process -ComputerName $Comput
    $lbl_sysinfo.Text = $Process | FL -Property ProcessName,
                                                Path,
                                                CreationDate | Out-String

$Services_info = {
    $ComputerName = $txt_ComputerName.Text
    $Services = Get-CimInstance -Class Win32_Service -ComputerName $Compu
    $lbl_sysinfo.Text = $Services | FL -Property Name,
                                                 DisplayName
```

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

```
$Ping_Test_info = {
   $ComputerName = $txt_ComputerName.Text

   If ($ComputerName -eq ""){
      $lbl_sysinfo.ForeColor = "Red"
      $lbl_sysinfo.Text = "Please provide a computer name to test the co
   else {
      $Ping_Test = Test-Connection $ComputerName
      $lbl_sysinfo.Text = $Ping_Test | Out-String}}

$RDP_Connection = {
   $ComputerName = $txt_ComputerName.Text
   mstsc /v:$ComputerName}

Add-Type -AssemblyName System.Windows.Forms

$Font = New-Object System.Drawing.Font("Consolas",12,[System.Drawing.Fon


$MainForm = New-Object system.Windows.Forms.Form
$MainForm.Text = "Computer Information"
$MainForm.Size = New-Object System.Drawing.Size(1200,800)
$MainForm.AutoScroll = $True
$MainForm.MinimizeBox = $True
$MainForm.MaximizeBox = $True
$MainForm.WindowState = "Normal"
$MainForm.SizeGripStyle = "Hide"
$MainForm.ShowInTaskbar = $True
$MainForm.Opacity = 1
$MainForm.StartPosition = "CenterScreen"
$MainForm.ShowInTaskbar = $True
$MainForm.Font = $Font

$lbl_ComputerName = New-Object System.Windows.Forms.Label
$lbl_ComputerName.Location = New-Object System.Drawing.Point(0,5)
$lbl_ComputerName.Size = New-Object System.Drawing.Size(150,20)
$lbl_ComputerName.Font = $Font
$lbl_ComputerName.Text = "Computer Name"
$MainForm.Controls.Add($lbl_ComputerName)

$lbl_sysinfo = New-Object System.Windows.Forms.Label
$lbl_sysinfo.Location = New-Object System.Drawing.Point(155,50)
```

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

```
$MainForm.Controls.Add($lbl_SysInfo)

$txt_ComputerName = New-Object System.Windows.Forms.TextBox
$txt_ComputerName.Location = New-Object System.Drawing.Point(150,5)
$txt_ComputerName.Size = New-Object System.Drawing.Size(200,20)
$txt_ComputerName.Font = $Font
$MainForm.Controls.Add($txt_ComputerName)

$btn_System = New-Object System.Windows.Forms.Button
$btn_System.Location = New-Object System.Drawing.Point(5,50)
$btn_System.Size = New-Object System.Drawing.Size(145,25)
$btn_System.Font = $Font
$btn_System.Text = "System"
$btn_System.Add_Click($System_info)
$MainForm.Controls.Add($btn_System)

$btn_BIOS = New-Object System.Windows.Forms.Button
$btn_BIOS.Location = New-Object System.Drawing.Point(5,75)
$btn_BIOS.Size = New-Object System.Drawing.Size(145,25)

$btn_BIOS.Font = $Font
$btn_BIOS.Text = "BIOS"
$btn_BIOS.Add_Click($bios_info)
$MainForm.Controls.Add($btn_BIOS)

$btn_CPU = New-Object System.Windows.Forms.Button
$btn_CPU.Location = New-Object System.Drawing.Point(5,100)
$btn_CPU.Size = New-Object System.Drawing.Size(145,25)
$btn_CPU.Font = $Font
$btn_CPU.Text = "CPU"
$btn_CPU.Add_Click($cpu_info)
$MainForm.Controls.Add($btn_CPU)

$btn_RAM = New-Object System.Windows.Forms.Button
$btn_RAM.Location = New-Object System.Drawing.Point(5,125)
$btn_RAM.Size = New-Object System.Drawing.Size(145,25)
$btn_RAM.Font = $Font
$btn_RAM.Text = "RAM"
$btn_RAM.Add_Click($ram_info)
$MainForm.Controls.Add($btn_RAM)

$btn_MB = New-Object System.Windows.Forms.Button
$btn_MB.Location = New-Object System.Drawing.Point(5,150)
$btn_MB.Size = New-Object System.Drawing.Size(145,25)
```

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

```
$btn_PhysicalDrives = New-Object System.Windows.Forms.Button
$btn_PhysicalDrives.Location = New-Object System.Drawing.Point(5,175)
$btn_PhysicalDrives.Size = New-Object System.Drawing.Size(145,25)
$btn_PhysicalDrives.Font = $Font
$btn_PhysicalDrives.Text = "Physical Drives"
$btn_PhysicalDrives.Add_Click($PhysicalDrives_info)
$MainForm.Controls.Add($btn_PhysicalDrives)

$btn_LogicalDrives = New-Object System.Windows.Forms.Button
$btn_LogicalDrives.Location = New-Object System.Drawing.Point(5,200)
$btn_LogicalDrives.Size = New-Object System.Drawing.Size(145,25)
$btn_LogicalDrives.Font = $Font
$btn_LogicalDrives.Text = "Logical Drives"
$btn_LogicalDrives.Add_Click($LogicalDrives_info)
$MainForm.Controls.Add($btn_LogicalDrives)

$btn_Graphics = New-Object System.Windows.Forms.Button
$btn_Graphics.Location = New-Object System.Drawing.Point(5,225)
$btn_Graphics.Size = New-Object System.Drawing.Size(145,25)
$btn_Graphics.Font = $Font
$btn_Graphics.Text = "Graphics"
$btn_Graphics.Add_Click($GPU_info)
$MainForm.Controls.Add($btn_Graphics)

$btn_Network = New-Object System.Windows.Forms.Button
$btn_Network.Location = New-Object System.Drawing.Point(5,250)
$btn_Network.Size = New-Object System.Drawing.Size(145,25)
$btn_Network.Font = $Font
$btn_Network.Text = "Network"
$btn_Network.Add_Click($Network_info)
$MainForm.Controls.Add($btn_Network)

$btn_NetSettings = New-Object System.Windows.Forms.Button
$btn_NetSettings.Location = New-Object System.Drawing.Point(5,275)
$btn_NetSettings.Size = New-Object System.Drawing.Size(145,25)
$btn_NetSettings.Font = $Font
$btn_NetSettings.Text = "Net Settings"
$btn_NetSettings.Add_Click($NetSettings_info)
$MainForm.Controls.Add($btn_NetSettings)

$btn_Monitors = New-Object System.Windows.Forms.Button
$btn_Monitors.Location = New-Object System.Drawing.Point(5,300)
```

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

```
$MainForm.Controls.Add($btn_Monitors)

$btn_OS = New-Object System.Windows.Forms.Button
$btn_OS.Location = New-Object System.Drawing.Point(5,325)
$btn_OS.Size = New-Object System.Drawing.Size(145,25)
$btn_OS.Font = $Font
$btn_OS.Text = "OS"
$btn_OS.Add_Click($OS_info)
$MainForm.Controls.Add($btn_OS)

$btn_Keyboard = New-Object System.Windows.Forms.Button
$btn_Keyboard.Location = New-Object System.Drawing.Point(5,350)
$btn_Keyboard.Size = New-Object System.Drawing.Size(145,25)
$btn_Keyboard.Font = $Font
$btn_Keyboard.Text = "Keyboard"
$btn_Keyboard.Add_Click($Keyboard_info)
$MainForm.Controls.Add($btn_Keyboard)

$btn_Mouse = New-Object System.Windows.Forms.Button

$btn_Mouse.Location = New-Object System.Drawing.Point(5,375)
$btn_Mouse.Size = New-Object System.Drawing.Size(145,25)
$btn_Mouse.Font = $Font
$btn_Mouse.Text = "Mouse"
$btn_Mouse.Add_Click($Mouse_info)
$MainForm.Controls.Add($btn_Mouse)

$btn_CDROM = New-Object System.Windows.Forms.Button
$btn_CDROM.Location = New-Object System.Drawing.Point(5,400)
$btn_CDROM.Size = New-Object System.Drawing.Size(145,25)
$btn_CDROM.Font = $Font
$btn_CDROM.Text = "CDROM"
$btn_CDROM.Add_Click($CDROM_info)
$MainForm.Controls.Add($btn_CDROM)

$btn_Sound = New-Object System.Windows.Forms.Button
$btn_Sound.Location = New-Object System.Drawing.Point(5,425)
$btn_Sound.Size = New-Object System.Drawing.Size(145,25)
$btn_Sound.Font = $Font
$btn_Sound.Text = "Sound"
$btn_Sound.Add_Click($Sound_info)
$MainForm.Controls.Add($btn_Sound)

$btn_Printers = New-Object System.Windows.Forms.Button
```

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

```
$btn_Printers.Add_Click($Printers_info)
$MainForm.Controls.Add($btn_Printers)

$btn_Fan = New-Object System.Windows.Forms.Button
$btn_Fan.Location = New-Object System.Drawing.Point(5,475)
$btn_Fan.Size = New-Object System.Drawing.Size(145,25)
$btn_Fan.Font = $Font
$btn_Fan.Text = "Fan"
$btn_Fan.Add_Click($Fan_info)
$MainForm.Controls.Add($btn_Fan)

$btn_Battery = New-Object System.Windows.Forms.Button
$btn_Battery.Location = New-Object System.Drawing.Point(5,500)
$btn_Battery.Size = New-Object System.Drawing.Size(145,25)
$btn_Battery.Font = $Font
$btn_Battery.Text = "Battery"
$btn_Battery.Add_Click($Battery_info)
$MainForm.Controls.Add($btn_Battery)


$btn_PortBattery = New-Object System.Windows.Forms.Button
$btn_PortBattery.Location = New-Object System.Drawing.Point(5,525)
$btn_PortBattery.Size = New-Object System.Drawing.Size(145,25)
$btn_PortBattery.Font = $Font
$btn_PortBattery.Text = "Port Battery"
$btn_PortBattery.Add_Click($PortBattery_info)
$MainForm.Controls.Add($btn_PortBattery)

$btn_Software = New-Object System.Windows.Forms.Button
$btn_Software.Location = New-Object System.Drawing.Point(5,550)
$btn_Software.Size = New-Object System.Drawing.Size(145,25)
$btn_Software.Font = $Font
$btn_Software.Text = "Software"
$btn_Software.Add_Click($Software_info)
$MainForm.Controls.Add($btn_Software)

$btn_Process = New-Object System.Windows.Forms.Button
$btn_Process.Location = New-Object System.Drawing.Point(5,575)
$btn_Process.Size = New-Object System.Drawing.Size(145,25)
$btn_Process.Font = $Font
$btn_Process.Text = "Process"
$btn_Process.Add_Click($Process_info)
$MainForm.Controls.Add($btn_Process)
```

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

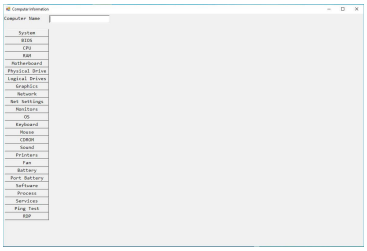
```
$btn_Services.Add_Click($Services_info)
$MainForm.Controls.Add($btn_Services)

$btn_Ping = New-Object System.Windows.Forms.Button
$btn_Ping.Location = New-Object System.Drawing.Point(5,625)
$btn_Ping.Size = New-Object System.Drawing.Size(145,25)
$btn_Ping.Font = $Font
$btn_Ping.Text = "Ping Test"
$btn_Ping.Add_Click($Ping_Test_info)
$MainForm.Controls.Add($btn_Ping)

$btn_RDP = New-Object System.Windows.Forms.Button
$btn_RDP.Location = New-Object System.Drawing.Point(5,650)
$btn_RDP.Size = New-Object System.Drawing.Size(145,25)
$btn_RDP.Font = $Font
$btn_RDP.Text = "RDP"
$btn_RDP.Add_Click($RDP_Connection)
$MainForm.Controls.Add($btn_RDP)


$MainForm.ShowDialog
```

Share Tweet Share Share Pin

Summary

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok



Article Name	How to get remote system information - Part 2
Description	How to get remote system information with PowerShell. Getting remote computer or server information has never been easier with this PowerShell script and it GUI. Stephanos Constantinou Blog - PowerShell Scripting
Author	Stephanos
Publisher Name	Stephanos Constantinou Blog
Publisher Logo	

FILED UNDER: POWERSHELL SCRIPTS

TAGGED WITH: ADD-TYPE, GET-CIMINSTANCE, NEW-OBJECT, OUT-STRING, WIN32_BASEBOARD, WIN32_BATTERY, WIN32_BIOS, WIN32_CDROMDRIVE, WIN32_COMPUTERSYSTEM, WIN32_DESKTOPMONITOR, WIN32_DISKDRIVE, WIN32_FAN, WIN32_KEYBOARD, WIN32_LOGICALDISK, WIN32_NETWORKADAPTER, WIN32_NETWORKADAPTERCONFIGURATION, WIN32_OPERATINGSYSTEM, WIN32_PHYSICALMEMORY, WIN32_POINTINGDEVICE, WIN32_PORTABLEBATTERY, WIN32_PRINTER, WIN32_PROCESS, WIN32_PROCESSOR, WIN32_PRODUCT, WIN32_SERVICE, WIN32_SOUNDDEVICE, WIN32_VIDEOCONTROLLER, WIN32REG_ADDREMOVEPROGRAMS

Comments

rcjay272 says

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

Just found your site and this post. I've been wanting to create a GUI for several of my scripts and this fits perfectly.
One question I have is on the creation of the GUI how do you go about designing it?
Meaning…Do you have an idea of what you want to gather then mock up the button layout first? I looking for help in the design.
While playing around I found for some of my examples I started from scratch and then moved to building the button structure and finally adding the code for each specific button.

Is there a better way to design the GUI? What is your process?

Thanks again for the script example…it was easy to follow and understand.
Rob.

Reply

Stephanos says
13/03/2018 at 14:32

Hello Rob,

Thank you for your comments.

When I have a GUI in my scripts, I usually draw a layout based on what I would like to achieve and then I proceed with the coding for the UI and button actions at the same time. If any changes are needed during the process, I act accordingly. I don't know if this is the correct procedure as I am not a UI expert, but it works for me.

Thank you

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

Trackbacks

How to get remote system information - Part 2 - How to Code .NET says:
26/02/2018 at 19:00
[...] on February 25, 2018by admin submitted by /u/SConstantinou [link] [comments] No comments [...]

How to get remote system information - Part 1 - Stephanos Constantinou says:
06/03/2018 at 10:24
[...] How to get remote system information – Part 2 – Stephanos Constantinou says: 26/02/2018 at 16:22 [...]

How to get remote system information - Part 3 - Stephanos Constantinou says:
14/03/2018 at 17:34
[...] How to get remote system information – Part 2 [...]

Leave a Reply

Your email address will not be published.

Comment

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

Name

Email

Website

POST COMMENT

This site uses Akismet to reduce spam. Learn how your comment data is processed.

Recent Posts

ICS Cube Product Review 26/04/2019
PowerShell Module SysInfo v1.2.0 15/03/2019
PowerShell Module SysInfo v1.1.2 13/11/2018
PowerShell Module

Categories

Modules Cmdlets (57)
PowerShell Modules (5)
PowerShell Scripts (38)
PowerShell Tutorials (35)
Software Reviews (2)

Archives

Blogroll

Planet PowerShell
Reddit – PowerShell
PowerShell Magazine
PowerShell.org
PowerShell Team Blog
Hey, Scripting Guy! Blog
Mike F Robbins
PowerShell Explained with Kevin Marquette

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok

© 2020 · Stephanos Constantinou Blog

We use cookies to ensure that we give you the best experience on our website. If you continue to use this site we will assume that you are happy with it.

Ok