俺の話を聞け！！LT大会 #10

2018/06/25(Mon)

マルチプロセスでコマンド実行
『VBAのちPythonときどきPowerShell』

会社名：非公開
発表者：井上　信之

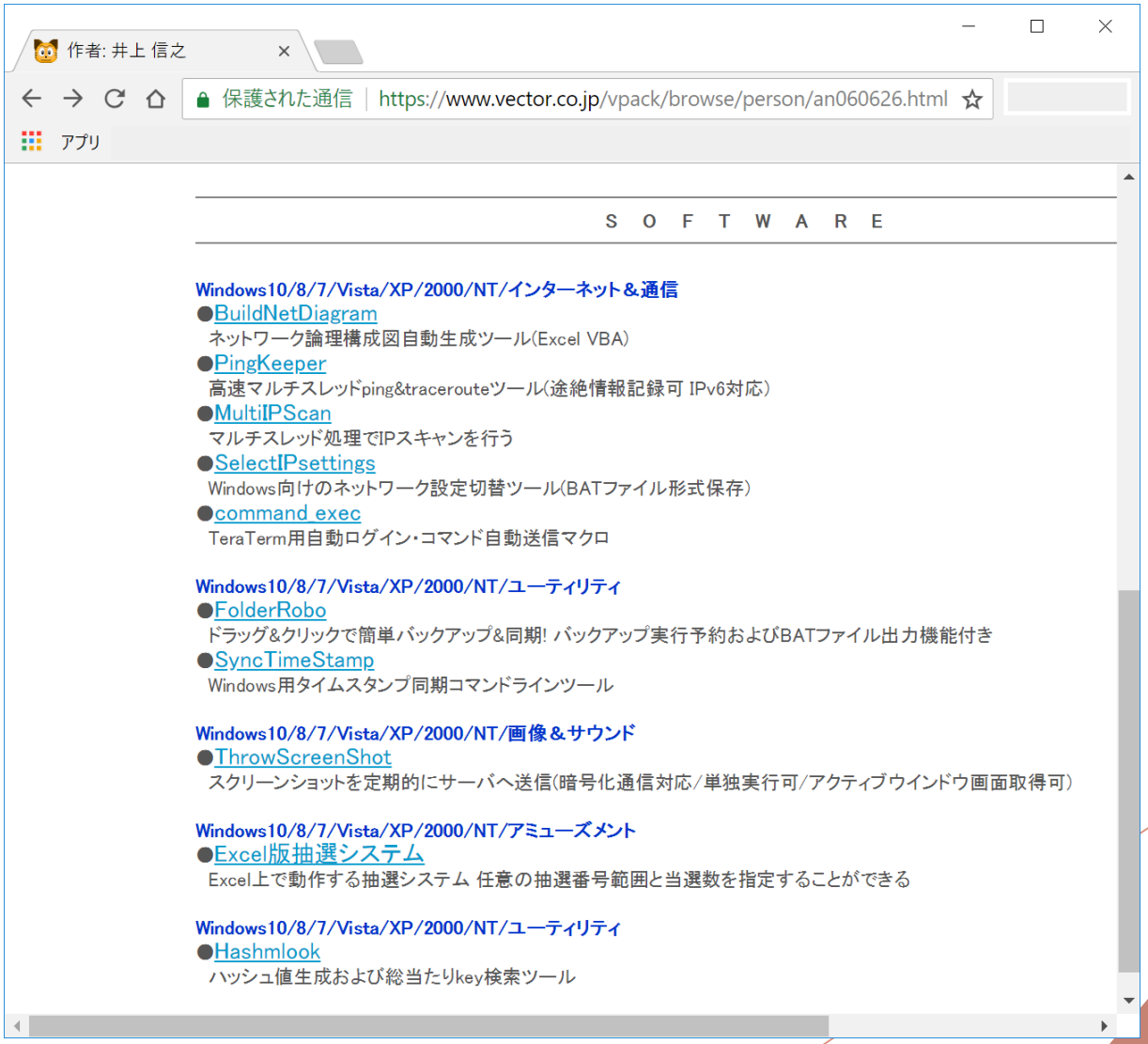
自己紹介

- ネットワーク／インフラ系の仕事

配布資料では
非公開としています。

公開ソフト一覧

https://www.vector.co.jp/vpack/browse/person/an060626.html



コマンド連続実行のニーズ

- xxxx台のネットワーク機器のconfigを収集したい
- xxxx台のネットワーク機器の設定をまとめて変更したい
- xxxx台のネットワーク機器のポートの状態を収集したい

などなど、

同じような処理を1台ずつ直列ではやっていられないとき…

コマンドを並列に実行するには…

Ping_single.bat

```
ping 192.168.0.1
ping 192.168.0.2
ping 192.168.0.3
```

これだと、前のpingコマンドが終わるまで、次のpingコマンドは実行されません。

並列に実行するには、“start“コマンドの引数として、pingコマンドを指定します。

Ping_multi.bat

```
start ping 192.168.0.1
start ping 192.168.0.2
start ping 192.168.0.3
```

コマンド プロンプト(DOS窓)が３つ起動し、pingコマンドが３個同時に実行されます。

コマンドを1,000個走らせたいときは…!?

- さすがに、BATファイルにコマンドをそのまま1000個並べるわけにはいかない。

  ブラクラ（ブラウザ・クラッシャー）状態になってしまう。

- 同時起動するコマンド数の上限を決め、上限を下回った場合に次のコマンドを起動させるようにすれば良い

コマンド並列実行サンプル
(Detune...いや、一般化バージョン)

本番環境用から、最低限の機能を取り出し、
デチューン、いや、一般化したサンプルコードを、
GitHubにて公開しています。

https://github.com/gx3n-inue/multiExec_from_VBA

https://github.com/gx3n-inue/multiExec_from_Python3

https://github.com/gx3n-inue/multiExec_from_PowerShell

ただし、

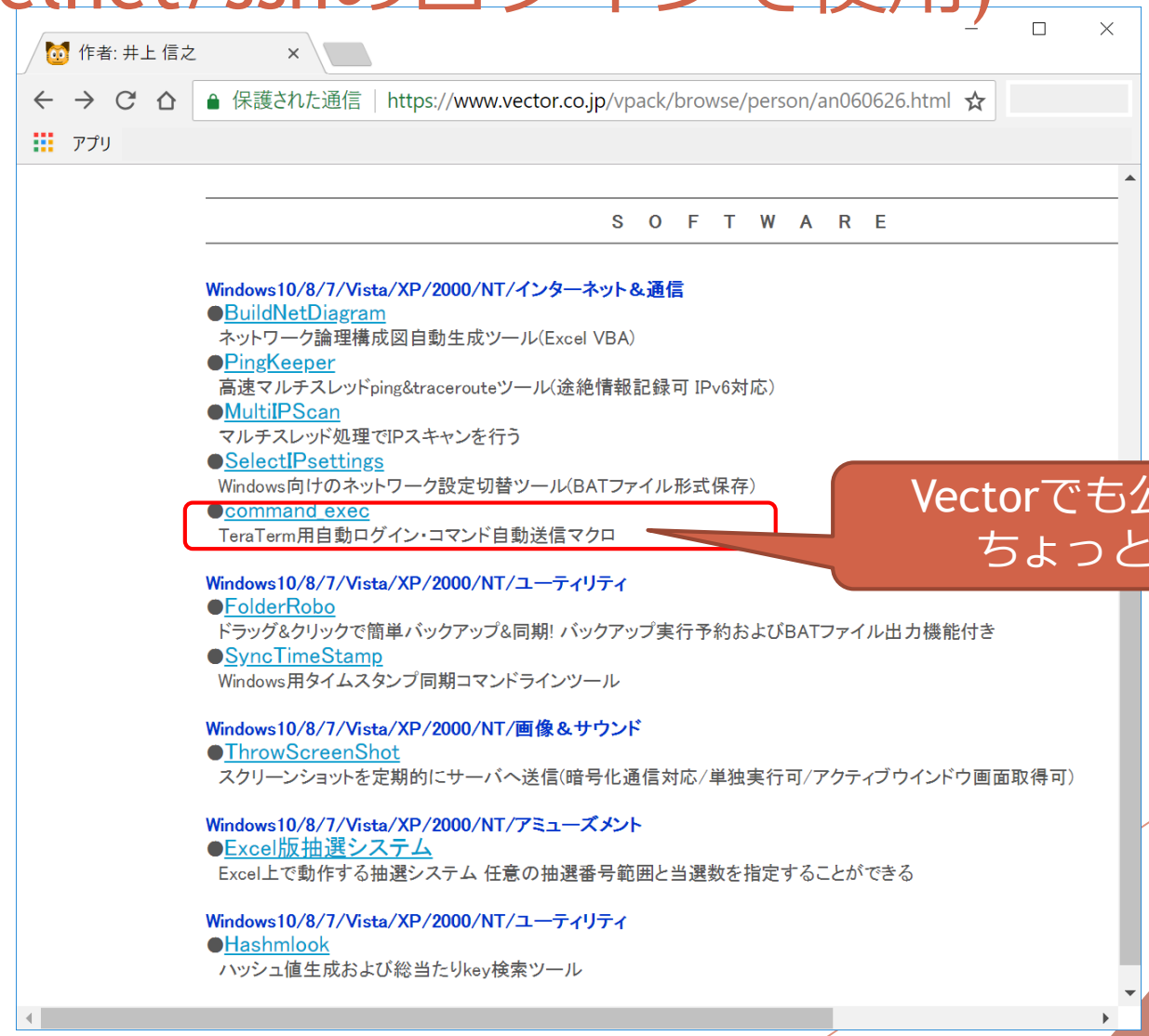
・コマンドの実行に失敗しても無視

・TeraTermのパス等も環境に合わせて変更が必要

なので、注意してください。

処理に成功したかどうかは、logファイルを調べ、コマンドの実行結果を調べる処理も必要。（本番環境ではそうしてる）

呼び出しているTeraTerm用マクロ(telnet/sshのログインで使用)



Excel VBAからの外部コマンドの実行方法

- Shell関数を利用する
  - MS-DOSの頃から存在するAPI
  - 戻り値はプロセスID
- WSH(Windows Script Host)オブジェクトを利用する
  - テキストファイルに記述したスクリプトを実行するスクリプト実行環境。Windows 98から搭載された。
  - COMを通じてレジストリの操作やWMIへのアクセスが可能であるなど、強力な機能を持っている。
  - WSHスクリプトを利用したウイルスや脆弱性攻撃などが増えたため、新しいスクリプト実行環境であるWindows PowerShellが登場した。

Excel VBA/Python3/PowerShell でのコマンド起動方法

	Excel VBA	Python3	PowerShell
Shell関数 (Win32API) (Kernel32.dllのWinExec()?)	利用可 (事前の宣言が必要)	利用可 (事前の宣言が必要)	利用可 (inlineでC#/VB.NETで処理を記述)
WSH (COMオブジェクト)	利用可	利用可	利用可
System.Diagnostics.Process (.NET Framework)	× (DLLの自作が必要)	利用可（少し面倒） （ほかに、Iron Python もある）	Process コマンドレット • Get-Process • Start-Process • Stop-Process • Wait-Process • Debug-Process

PowerShellからのコマンド実行方法

方法１）直接exeを入力する

```
>notepad.exe test.txt
```

方法２）& に続けてコマンドを入力する

```
>& notepad.exe
```

方法３）Start-Processコマンドレットを使用する

```
>Start-Process -FilePath notepad.exe -ArgumentList test.txt -Wait
```

VBAでの外部コマンドの起動2-1
(WSH(Execメソッド)版　コマンドの終了を待つ)

コマンドの終了を待つ場合

```
Dim sh As New IWshRuntimeLibrary.WshShell   ' WshShellクラスオブジェクト
Dim ex As WshExec                           ' Execメソッド戻り値

Set ex = sh.Exec(”実行コマンド”)

If (ex is not None):
  pid = ex.ProcessID

  ' コマンドが終了まで待機する
  Do While (ex.Status = WshRunning)
    Sleep 100
  Loop
End If
```

WSHShellクラスのExecメソッド実行後、ループ内でプロセスの終了シグナルを待つ。

Python3での外部コマンドの起動2-1
(WSH(Execメソッド)版　コマンドの終了を待つ)

コマンドの終了を待つ場合

```
shell = win32com.client.Dispatch("WScript.Shell")
ex = shell.Exec(" 実行コマンド" )

if (ex is not None):
  pid = ex.ProcessID

  while (ex.status == 0):
    sleep(0.1)
```

WSHShellクラスのExecメソッド実行後、ループ内でプロセスの終了シグナルを待つ。

PowerShellでの外部コマンドの起動2-1
(WSH(Execメソッド)版　コマンドの終了を待つ)

コマンドの終了を待つ場合

```
$WshShell = New-Object -ComObject WScript.Shell
$ex = $WshShell.Exec(”実行コマンド”)

if ($ex) {
     $processId = $ex.ProcessID

     while ( $ex.status -eq 0) {
          Start-Sleep 0.1
     }
}
```

WSHShellクラスのExecメソッド実行後、ループ内でプロセスの終了シグナルを待つ。

※PowerShell では、変数$PIDは予約されています。

外部コマンドの起動
WSH(Execメソッド)コマンドの終了を待つ

Excel VBA	Python3	PowerShell

Dim sh As New IWshRuntimeLibrary.WshShell
 	
Dim ex As WshExec
Set ex = sh.Exec(”実行コマンド”)
 	
If (ex is not None):
  pid = ex.ProcessID
 	
  Do While (ex.Status = WshRunning)
  Sleep 100
  Loop
End If
	shell = win32com.client.Dispatch("WScript.Shell")
	
	ex = shell.Exec(”実行コマンド”)
	
	if (ex is not None):
	  pid = ex.ProcessID
	
	  while (ex.status == 0):
	  sleep(0.1)
		$WshShell = New-Object -ComObject WScript.Shell
		
		$ex = $WshShell.Exec(”実行コマンド”)
		
		if ($ex) {
		  $processId = $ex.ProcessID
		
		  while ($ex.status -eq 0) {
		  Start-Sleep 0.1
		  }
		}

外部コマンドの起動

.NET Framework(System.Diagnostics.Processオブジェクト)

Python3	PowerShell
import clr import System　# .NET Framework clr.AddReference('System.Diagnostics.Process') ps = System.Diagnostics.Process() ps.Start("notepad.exe") ps.WaitForExit()	$proc = Start-Process -PassThru "notepad.exe" $proc.WaitForExit()

実行中のPIDの取得方法

- WMI (Windows Management Infomation)を利用

  プロセス一覧を取得し、あらかじめ保存しておいた値とProcessIDが一致するプロセスが存在するかをチェックする

指定したPIDのプロセスが存在しているか調べる(VBA)

指定したPIDのプロセスが存在しているか調べる

```
Public Function IsExist_targetProcesses(target_pid As Integer) As Boolean
    (中略)
    ' WMI Win32_Process classのオブジェクトから PIDを検索する
    Dim Locator: Set Locator = CreateObject("WbemScripting.SWbemLocator")
    Dim Server: Set Server = Locator.ConnectServer
    Dim objSet: Set objSet = Server.ExecQuery("Select * From Win32_Process")
    Dim obj

    For Each obj In objSet
        If obj.ProcessID = target_pid Then
            ' 見つかった場合
            IsExist_targetProcesses_from_WMI = True
            Exit Function
        End If
    Next

    IsExist_targetProcesses_from_WMI = False

End Function
```

指定したPIDのプロセスが存在しているか調べる(Python3)

指定したPIDのプロセスが存在しているか調べる

```
def IsExist_targetProcesses(target_pid):
  if (target_pid == 0):
    return 0

  # WMI Win32_Process classのオブジェクトからPIDを検索する
  locator = win32com.client.Dispatch("WbemScripting.SWbemLocator")
  server = locator.ConnectServer()
  objSet = server.ExecQuery("Select * From Win32_Process")

  for obj in objSet:
    if (obj.ProcessID == target_pid):
      return 1

  return 0
```

指定したPIDのプロセスが存在しているか調べる(PowerShell)

指定したPIDのプロセスが存在しているか調べる

```
function IsExist_targetProcesses([int]$target_pid)
{
    return @(Get-Process -Id $target_pid -ErrorAction 0).Count
}
```

指定したPIDのプロセスが存在しているか調べる(比較)

Excel VBA

```
Public Function IsExist_targetProcesses(target_pid As Integer) As Boolean
    (中略)
    ' WMI Win32_Process classのオブジェクトから PIDを検索する
    Dim Locator: Set Locator = CreateObject("WbemScripting.SWbemLocator")
    Dim Server: Set Server = Locator.ConnectServer
    Dim objSet: Set objSet = Server.ExecQuery("Select * From Win32_Process")
    Dim obj

    For Each obj In objSet
        If obj.ProcessID = target_pid Then
            ' 見つかった場合
            IsExist_targetProcesses_from_WMI = True
            Exit Function
        End If
    Next

    IsExist_targetProcesses_from_WMI = False

End Function
```

Python3

```
def IsExist_targetProcesses(target_pid):
  if (target_pid == 0):
    return 0

  # WMI Win32_Process classのオブジェクトからPIDを検索する
  locator = win32com.client.Dispatch("WbemScripting.SWbemLocator")
  server = locator.ConnectServer()
  objSet = server.ExecQuery("Select * From Win32_Process")

  for obj in objSet:
    if (obj.ProcessID == target_pid):
      return 1

  return 0
```

PowerShell

```
function IsExist_targetProcesses([int]$target_pid)
{
    return @(Get-Process -Id $target_pid -ErrorAction 0).Count
}
```

それでは、実演…

- ~~接続先のNW機器のデモ用として、AWS EC2 Ubuntu Server 16.04 に連続ログインし、適当なコマンドを実行する。~~
- ~~実演用に、telnet, snmpを許可しています。（私のIPアドレスだけ）~~

と思いましたが、実行結果は前回と同じなので、別のネタ…

おまけ）
PowerShellで演奏する電子オルゴール

- PowerShellで演奏する電子オルゴール

  https://qiita.com/gx3n-inue/items/8ca1d987d58235e7f434
  https://github.com/gx3n-inue/PS_PlayBox

  プログラムの入力および実行には、テキストエディタだけあれば十分です。

  - プログラムの構成
    1. メインプログラム(PS_PlayBox.ps1)
    2. MIDIのWin32APIを呼び出すC#モジュール(myMIDI.cs)
    3. 音階とMIDIノート番号の対応定義ファイル(note-number.dat)
    4. 演奏したい楽譜ファイル(*.txt)

PS_Play_Box.ps1の実行時にC#のモジュールが読み込まれますが、
事前コンパイルは必要ありません。

実行方法）

>.¥PS_PlayBox.ps1 <演奏したい楽譜ファイル> <音色番号>