Snippets

Logged in users

```
$loggedInUsers = (query user) -split "\n" -replace '\s\s+', ';' | convertfrom-csv -Delimiter
';'
```

```
# example output

PS C:\> $loggedInUsers | Format-Table

USERNAME SESSIONNAME ID STATE  IDLE TIME LOGON TIME
-------- ----------- -- -----  --------- ----------
joebob   console     15 Active 6:09      1/12/1979 3:33 AM

PS C:\> $loggedInUsers.USERNAME
joebob
```

IPv6 to MAC address

```
$ipv6Address = "fe80::9657:a5ff:fe17:aeac"

# First attempt - shows all entries
Get-NetNeighbor -AddressFamily IPv6 | Where-Object {$_.IPAddress -eq "$($ipv6Address)"}

# Second attempt - only show Permananet entries
Get-NetNeighbor -AddressFamily IPv6 | Where-Object {($_.State -eq 'Permanent') -And
($_.IPAddress -eq "$($ipv6Address)")}

# Show just the Windows MAC address format with dashes
(Get-NetNeighbor -AddressFamily IPv6 | Where-Object {($_.State -eq 'Permanent') -And
($_.IPAddress -eq "$($ipv6Address)")}).LinkLayerAddress

# Normal format with colons
```

```
(Get-NetNeighbor -AddressFamily IPv6 | Where-Object {($_.State -eq 'Permanent') -And
($_.IPAddress -eq "$($ipv6Address)")}).LinkLayerAddress -replace '-', ':'
```

While a file exists or not

```
# while a file exists
While (Test-Path C:\Temp\File_I_Want_Gone.txt -ErrorAction SilentlyContinue) {
  # Do something here while the file exists
}
```

```
# while a file doesn't exists
While (!(Test-Path C:\Temp\File_I_Want_Gone.txt -ErrorAction SilentlyContinue)) {
  # Do something here while the file doesn't exists
}
```

```
# while a file exists
While (Test-Path C:\Temp\File_I_Want_Gone.txt -ErrorAction SilentlyContinue) {
  # try to delete the file, continue silently if we can't
  Remove-Item "C:\Temp\File_I_Want_Gone.txt" -ErrorAction SilentlyContinue
  # print date each time just to give some sort of feedback on the console
  Get-Date
}
```

Testing Microsoft SQL database connectivity

```
function Test-SQLConnection
{
    [OutputType([bool])]
    Param
    (
        [Parameter(Mandatory=$true,
                   ValueFromPipelineByPropertyName=$true,
                   Position=0)]
        $ConnectionString
    )
    try
    {
```

```
        $sqlConnection = New-Object System.Data.SqlClient.SqlConnection $ConnectionString;
        $sqlConnection.Open();
        $sqlConnection.Close();

        return $true;
    }
    catch
    {
        return $false;
    }
}
```

```
Test-SQLConnection "Data Source=localhost;database=someDatabase;User
ID=bogusTestUser;Password=bogusTestPassword;"
```

[Source]

Pattern matching

```
# url and ipv4 / ipv6 patterns
$hostnamePattern = "(?:https?://)?(?:www\.)?([^/:]+)"
$ipv4Pattern = "(?:http|https):\/\/((?:[0-9]{1,3}\.){3}[0-9]{1,3})"
$ipv6Pattern = "\[((?:[0-9a-fA-F]{1,4}:){7}[0-9a-fA-F]{1,4}|(?:[0-9a-fA-F]{1,4}:){1,7}:|(?:[0-
9a-fA-F]{1,4}:){1,6}:[0-9a-fA-F]{1,4}|(?:[0-9a-fA-F]{1,4}:){1,5}(:[0-9a-fA-
F]{1,4}){1,2}|(?:[0-9a-fA-F]{1,4}:){1,4}(:[0-9a-fA-F]{1,4}){1,3}|(?:[0-9a-fA-
F]{1,4}:){1,3}(:[0-9a-fA-F]{1,4}){1,4}|(?:[0-9a-fA-F]{1,4}:){1,2}(:[0-9a-fA-F]{1,4}){1,5}|[0-
9a-fA-F]{1,4}:((:[0-9a-fA-F]{1,4}){1,6})|:((:[0-9a-fA-F]{1,4}){1,7}|:)|fe80:(:[0-9a-fA-
F]{0,4}){0,4}%[0-9a-zA-Z]{1,}|::(ffff(:0{1,4}){0,1}:){0,1}((25[0-5]|(2[0-4]|1{0,1}[0-
9]){0,1}[0-9])\.){3}(25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9]))\]"

# example use - creating a WSDLocation URL Extractoratorinator
function WSDLocationURLExtractAddress {
    <#
    .DESCRIPTION
    Given a WSD Location / LocationInformation URL, attempt to extract and return the IPv4 or
IPv6 address.
    #>
    param (
        [string]$WSDLocation
    )
```

```
    $ipv4Pattern = "(?:http|https):\/\/((?:[0-9]{1,3}\.){3}[0-9]{1,3})"
    $ipv6Pattern = "\[((?:[0-9a-fA-F]{1,4}:){7}[0-9a-fA-F]{1,4}|(?:[0-9a-fA-
F]{1,4}:){1,7}:|(?:[0-9a-fA-F]{1,4}:){1,6}:[0-9a-fA-F]{1,4}|(?:[0-9a-fA-F]{1,4}:){1,5}(:[0-9a-
fA-F]{1,4}){1,2}|(?:[0-9a-fA-F]{1,4}:){1,4}(:[0-9a-fA-F]{1,4}){1,3}|(?:[0-9a-fA-
F]{1,4}:){1,3}(:[0-9a-fA-F]{1,4}){1,4}|(?:[0-9a-fA-F]{1,4}:){1,2}(:[0-9a-fA-F]{1,4}){1,5}|[0-
9a-fA-F]{1,4}:((:[0-9a-fA-F]{1,4}){1,6})|:((:[0-9a-fA-F]{1,4}){1,7}|:)|fe80:(:[0-9a-fA-
F]{0,4}){0,4}%[0-9a-zA-Z]{1,}|::(ffff(:0{1,4}){0,1}:){0,1}((25[0-5]|(2[0-4]|1{0,1}[0-
9]){0,1}[0-9])\.){3}(25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9]))\]"
    $hostnamePattern = "(?:https?://)?(?:www\.)?([^/:]+)"

    $ipv4Match = [regex]::Match($WSDLocation, $ipv4Pattern)
    $ipv6Match = [regex]::Match($WSDLocation, $ipv6Pattern)
    $hostnameMatch = [regex]::Match($WSDLocation, $hostnamePattern)

    $retval = ""
    if ($ipv4Match.Success) {
        $retval = ($ipv4Match[0].Value -split '://')[1]
    } elseif ($ipv6Match.Success) {
        $retval = ($ipv6Match[0].Value).Trim('[',']').Split('%')[0]
    } elseif ($hostnameMatch.Success) {
        $retval = ($hostnameMatch[0].Value -split '://')[1]
    } else {
        $retval = $null
    }
    return $retval
}
```

Revision #6
Created 14 May 2024 22:03:46 by bluecrow76
Updated 8 December 2025 20:39:19 by bluecrow76