



python



PowerShell

FPDF Library
PDF generator

Full credit is given to the above companies including the OS that this PDF file was generated!

PowerShell Get-Help on command 'Debug-Process'

PS C:\Users\wahid> Get-Help Debug-Process

NAME

Debug-Process

SYNOPSIS

Debugs one or more processes running on the local computer.

SYNTAX

Debug-Process [-Id] <System.Int32[]> [-Confirm] [-WhatIf] [<CommonParameters>]

Debug-Process -InputObject <System.Diagnostics.Process[]> [-Confirm] [-WhatIf]

[<CommonParameters>]

Debug-Process [-Name] <System.String[]> [-Confirm] [-WhatIf]

[<CommonParameters>]

DESCRIPTION

The `Debug-Process` cmdlet attaches a debugger to one or more running processes on a local computer. You can specify the processes by their process name or process ID (PID), or you can pipe process objects to this cmdlet.

This cmdlet attaches the debugger that is currently registered for the process. Before using this cmdlet, verify that a debugger is downloaded and correctly configured.

PARAMETERS

`-Id <System.Int32[]>`

Specifies the process IDs of the processes to be debugged. The `Id` parameter name is optional.

To find the process ID of a process, type ``Get-Process``.

`-InputObject <System.Diagnostics.Process[]>`

Specifies the process objects that represent processes to be debugged.

Enter a variable that contains the process objects or a command that gets the process objects, such as the ``Get-Process`` cmdlet. You can also pipe process objects to this cmdlet.

`-Name <System.String[]>`

Specifies the names of the processes to be debugged. If there is more than one process with the same name, this cmdlet attaches a debugger to all processes with that name. The `Name` parameter is optional.

`-Confirm <System.Management.Automation.SwitchParameter>`

Prompts you for confirmation before running the cmdlet.

`-WhatIf <System.Management.Automation.SwitchParameter>`

Shows what would happen if the cmdlet runs. The cmdlet is not run.

`<CommonParameters>`

This cmdlet supports the common parameters: `Verbose`, `Debug`, `ErrorAction`, `ErrorVariable`, `WarningAction`, `WarningVariable`,

OutBuffer, PipelineVariable, and OutVariable. For more information, see [about_CommonParameters](https://go.microsoft.com/fwlink/?LinkID=113216) (https://go.microsoft.com/fwlink/?LinkID=113216).

-- Example 1: Attach a debugger to a process on the computer --

```
PS C:\> Debug-Process -Name "Windows Powershell"
```

This command attaches a debugger to the PowerShell process on the computer.

Example 2: Attach a debugger to all processes that begin with the specified string

```
PS C:\> Debug-Process -Name "SQL*"
```

This command attaches a debugger to all processes that have names that begin with SQL.

----- Example 3: Attach a debugger to multiple processes -----

```
PS C:\> Debug-Process "Winlogon", "Explorer", "Outlook"
```

This command attaches a debugger to the Winlogon, Explorer, and Outlook processes.

----- Example 4: Attach a debugger to multiple process IDs -----

```
PS C:\> Debug-Process -Id 1132, 2028
```

This command attaches a debugger to the processes that have process IDs 1132 and 2028.

Example 5: Use Get-Process to get a process then attach a debugger to it

```
PS C:\> Get-Process "Windows PowerShell" | Debug-Process
```

This command attaches a debugger to the PowerShell processes on the computer.

It uses the `Get-Process` cmdlet to get the PowerShell processes on the`

computer, and it uses a pipeline operator (|) to send the processes to the `Debug-Process` cmdlet.

To specify a particular PowerShell process, use the ID parameter of `Get-Process`.

Example 6: Attach a debugger to a current process on the local computer

```
PS C:\> $PID | Debug-Process
```

This command attaches a debugger to the current PowerShell processes on the computer.

The command uses the `\$PID` automatic variable, which contains the process ID of the current PowerShell process. Then, it uses a pipeline operator (|) to send the process ID to the `Debug-Process` cmdlet.

For more information about the `\$PID` automatic variable, see [about_Automatic_Variables](#)

(../Microsoft.PowerShell.Core/About/about_Automatic_Variables.md).

Example 7: Attach a debugger to the specified process on multiple computers

```
PS C:\> Get-Process -ComputerName "Server01", "Server02" -Name "MyApp" |  
Debug-Process
```

This command attaches a debugger to the MyApp processes on the Server01 and Server02 computers.

The command uses the `Get-Process` cmdlet to get the MyApp processes on the Server01 and Server02 computers. It uses a pipeline operator to send the processes to the `Debug-Process` cmdlet, which attaches the debuggers.

Example 8: Attach a debugger to a process that uses the InputObject parameter

```
PS C:\> $P = Get-Process "Windows PowerShell"
```

```
PS C:\> Debug-Process -InputObject $P
```

This command attaches a debugger to the PowerShell processes on the local computer.

The first command uses the `Get-Process` cmdlet to get the PowerShell processes on the computer. It saves the resulting process object in the variable named `$P`.

The second command uses the `InputObject` parameter of the `Debug-Process` cmdlet to submit the process object in the `$P` variable.

REMARKS

To see the examples, type: `"get-help Debug-Process -examples"`.

For more information, type: `"get-help Debug-Process -detailed"`.

For technical information, type: `"get-help Debug-Process -full"`.

For online help, type: `"get-help Debug-Process -online"`