PowerShell Get-Help on command 'Uninstall-Package'

PS C:\Users\wahid> Get-Help Uninstall-Package

```
NAME
    Uninstall-Package

SYNOPSIS
    Uninstalls one or more software packages.



SYNTAX
    Uninstall-Package [-AdditionalArguments <System.String[]>] [-AllVersions]
    [-Force] [-ForceBootstrap] [-Confirm] [-WhatIf] [<CommonParameters>]

    Uninstall-Package [-AdditionalArguments <System.String[]>] [-AllVersions]
    [-Force] [-ForceBootstrap] [-Confirm] [-WhatIf] [<CommonParameters>]

    Uninstall-Package [-AllowClobber] [-AllowPrereleaseVersions] [-AllVersions]
    [-Force] [-ForceBootstrap] [-InstallUpdate] [-NoPathUpdate]
    [-PackageManagementProvider <System.String>] [-Scope {CurrentUser | AllUsers}]
    [-SkipPublisherCheck] [-Type {Module | Script | All}] [-Confirm] [-WhatIf]
    [<CommonParameters>]

    Uninstall-Package [-AllowClobber] [-AllowPrereleaseVersions] [-AllVersions]
    [-Force] [-ForceBootstrap] [-InstallUpdate] [-NoPathUpdate]
    [-PackageManagementProvider <System.String>] [-Scope {CurrentUser | AllUsers}]
    [-SkipPublisherCheck] [-Type {Module | Script | All}] [-Confirm] [-WhatIf]
    [<CommonParameters>]

    Uninstall-Package [-AllVersions] [-Destination <System.String>]
    [-ExcludeVersion] [-Force] [-ForceBootstrap] [-Scope {CurrentUser | AllUsers}]
    [-SkipDependencies] [-Confirm] [-WhatIf] [<CommonParameters>]

    Uninstall-Package [-AllVersions] [-Destination <System.String>]
    [-ExcludeVersion] [-Force] [-ForceBootstrap] [-Scope {CurrentUser | AllUsers}]
    [-SkipDependencies] [-Confirm] [-WhatIf] [<CommonParameters>]

    Uninstall-Package [-AllVersions] [-Force] [-ForceBootstrap]
    [-IncludeSystemComponent] [-IncludeWindowsInstaller] [-Confirm] [-WhatIf]
    [<CommonParameters>]

    Uninstall-Package [-AllVersions] [-Force] [-ForceBootstrap]
    [-IncludeSystemComponent] [-IncludeWindowsInstaller] [-Confirm] [-WhatIf]
    [<CommonParameters>]

    Uninstall-Package [-InputObject]
    <Microsoft.PackageManagement.Packaging.SoftwareIdentity[]> [-AllVersions]
    [-Force] [-ForceBootstrap] [-Confirm] [-WhatIf] [<CommonParameters>]

    Uninstall-Package [-Name] <System.String[]> [-AllVersions] [-Force]
    [-ForceBootstrap] [-MaximumVersion <System.String>] [-MinimumVersion
    <System.String>] [-ProviderName {Bootstrap | NuGet | PowerShellGet}]
    [-RequiredVersion <System.String>] [-Confirm] [-WhatIf] [<CommonParameters>]
```

DESCRIPTION

The `Uninstall-Package` cmdlet uninstalls one or more software packages from the local computer. To find installed packages, use the `Get-Package` cmdlet.

!INCLUDE [nuget-module (../../includes/nuget-module.md)]

PARAMETERS

-AdditionalArguments <System.String[]>

Specifies additional arguments.

-AllowClobber <System.Management.Automation.SwitchParameter>

Overrides warning messages about conflicts with existing commands.

Overwrites existing commands that have the same name as commands being installed.

-AllowPrereleaseVersions <System.Management.Automation.SwitchParameter>

Allows packages marked as prerelease to be uninstalled.

-AllVersions <System.Management.Automation.SwitchParameter>

Indicates that this cmdlet uninstalls all versions of the package.

-Destination <System.String>

Specifies a string of the path to the input object.

-ExcludeVersion <System.Management.Automation.SwitchParameter>

Switch to exclude the version number in the folder path.

-Force <System.Management.Automation.SwitchParameter>

Forces the command to run without asking for user confirmation.

-ForceBootstrap <System.Management.Automation.SwitchParameter>

Forces PackageManagement to automatically install the package provider for the specified package.

-IncludeSystemComponent <System.Management.Automation.SwitchParameter>

Specifies that this cmdlet uninstalls system components.

-IncludeWindowsInstaller <System.Management.Automation.SwitchParameter>

Indicates that this cmdlet uninstalls the package through Windows Installer.

-InputObject <Microsoft.PackageManagement.Packaging.SoftwareIdentity[]>

Accepts pipeline input that specifies the package's SoftwareIdentity object from the `Get-Package` cmdlet. InputObject accepts the SoftwareIdentity object as a `Get-Package` value or a variable that contains the object.

-InstallUpdate <System.Management.Automation.SwitchParameter>

Indicates that `Uninstall-Package` uninstalls updates.

-MaximumVersion <System.String>

Specifies the maximum allowed package version that you want to uninstall. If you don't specify this parameter, `Uninstall-Package` uninstalls the package's newest version.

-MinimumVersion <System.String>

Specifies the minimum allowed package version that you want to uninstall. If you don't add this parameter, `Uninstall-Package` uninstalls the package's newest version that satisfies any version specified by the MaximumVersion parameter.

-Name <System.String[]>

Specifies one or more package names. Multiple package names must be separated by commas.

-NoPathUpdate <System.Management.Automation.SwitchParameter>

NoPathUpdate only applies to the `Install-Script` cmdlet. NoPathUpdate is a dynamic parameter added by the provider and isn't supported by

`Uninstall-Package`.

-PackageManagementProvider <System.String>

Specifies the PackageManagement provider.

-ProviderName <System.String[]>

Specifies one or more package provider names to search for packages. You can get package provider names by running the `Get-PackageProvider` cmdlet.

-RequiredVersion <System.String>

Specifies the exact allowed version of the package that you want to uninstall. If you don't add this parameter, `Uninstall-Package` uninstalls the package's newest version that satisfies any version specified by the MaximumVersion parameter.

-Scope <System.String>

Specifies the scope for which to uninstall the package. The acceptable values for this parameter are as follows:

- CurrentUser

- AllUsers

-SkipDependencies <System.Management.Automation.SwitchParameter>

Skips the uninstallation of software dependencies.

-SkipPublisherCheck <System.Management.Automation.SwitchParameter>

Allows you to get a package version that is newer than your installed version. For example, an installed package that is digitally signed by a trusted publisher but a new version isn't digitally signed.

-Type <System.String>

Specifies whether to search for packages with a module, a script, or both.

    The acceptable values for this parameter are as follows:

    - Module

    - Script

    - All

-Confirm <System.Management.Automation.SwitchParameter>
    Prompts you for confirmation before running the cmdlet.

-WhatIf <System.Management.Automation.SwitchParameter>
    Shows what would happen if `Uninstall-Package` cmdlet is run. The cmdlet
    isn't 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).

---------------- Example 1: Uninstall a package ----------------

```
PS> Uninstall-Package -Name NuGet.Core
```

------ Example 2: Use the pipeline to uninstall a package ------

```
PS> Get-Package -Name NuGet.Core -RequiredVersion 2.14.0 | Uninstall-Package
```

The `Get-Package` cmdlet uses the Name and RequiredVersion parameters to
specify a package. A SoftwareIdentity object is sent down the pipeline. The
`Uninstall-Package` cmdlet receives the object as an InputObject and removes

the package.

As an alternative, the `Uninstall-Package` cmdlet can specify a value for the InputObject parameter:

`Uninstall-Package -InputObject ( Get-Package -Name NuGet.Core -RequiredVersion 2.14.0 )`

REMARKS

To see the examples, type: "get-help Uninstall-Package -examples".

For more information, type: "get-help Uninstall-Package -detailed".

For technical information, type: "get-help Uninstall-Package -full".

For online help, type: "get-help Uninstall-Package -online"