PowerShell Get-Help on command 'Invoke-RestMethod'

PS C:\Users\wahid> Get-Help Invoke-RestMethod

```
NAME
    Invoke-RestMethod


SYNOPSIS
    Sends an HTTP or HTTPS request to a RESTful web service.



SYNTAX
    Invoke-RestMethod [-Uri] <System.Uri> [-Body <System.Object>] [-Certificate
    <System.Security.Cryptography.X509Certificates.X509Certificate>]
    [-CertificateThumbprint <System.String>] [-ContentType <System.String>]
    [-Credential <System.Management.Automation.PSCredential>] [-DisableKeepAlive]
    [-Headers <System.Collections.IDictionary>] [-InFile <System.String>]
    [-MaximumRedirection <System.Int32>] [-Method {Default | Get | Head | Post |
    Put | Delete | Trace | Options | Merge | Patch}] [-OutFile <System.String>]
    [-PassThru] [-Proxy <System.Uri>] [-ProxyCredential
    <System.Management.Automation.PSCredential>] [-ProxyUseDefaultCredentials]
    [-SessionVariable <System.String>] [-TimeoutSec <System.Int32>]
    [-TransferEncoding {chunked | compress | deflate | gzip | identity}]
    [-UseBasicParsing] [-UseDefaultCredentials] [-UserAgent <System.String>]
    [-WebSession <Microsoft.PowerShell.Commands.WebRequestSession>]
```

[<CommonParameters>]

DESCRIPTION

The `Invoke-RestMethod` cmdlet sends HTTP and HTTPS requests to Representational State Transfer (REST) web services that return richly structured data.

PowerShell formats the response based to the data type. For an RSS or ATOM feed, PowerShell returns the Item or Entry XML nodes. For JavaScript Object Notation (JSON) or XML, PowerShell converts, or deserializes, the content into `[PSCustomObject]` objects.

> [!NOTE] > When the REST endpoint returns multiple objects, the objects are received as an array. If you pipe > the output from `Invoke-RestMethod` to another command, it is sent as a single `[Object[]]` > object. The contents of that array are not enumerated for the next command on the pipeline.

This cmdlet is introduced in Windows PowerShell 3.0.

> [!NOTE] > By default, script code in the web page may be run when the page is being parsed to populate the > `ParsedHtml` property. Use the UseBasicParsing switch to suppress this.

PARAMETERS

-Body <System.Object>

Specifies the body of the request. The body is the content of the request that follows the headers. You can also pipe a body value to `Invoke-RestMethod`.

The Body parameter can be used to specify a list of query parameters or specify the content of the response.

When the input is a GET request, and the body is an IDictionary (typically, a hash table), the body is added to the URI as query parameters. For other request types (such as POST), the body is set as the value of the request body in the standard name=value format.

> [!WARNING] > The verbose output of a POST body will end with `with -1-byte payload`, even though > the size of the body is both known and sent in the `Content-Length` HTTP header.

-Certificate <System.Security.Cryptography.X509Certificates.X509Certificate>

Specifies the client certificate that is used for a secure web request. Enter a variable that contains a certificate or a command or expression that gets the certificate.

To find a certificate, use `Get-PfxCertificate` or use the `Get-ChildItem` cmdlet in the Certificate (`Cert:`) drive. If the certificate is not valid or does not have sufficient authority, the command fails.

-CertificateThumbprint <System.String>

Specifies the digital public key certificate (X509) of a user account that has permission to send the request. Enter the certificate thumbprint of the certificate.

Certificates are used in client certificate-based authentication. Certificates can only be mapped only to local user accounts, not domain accounts.

To see the certificate thumbprint, use the `Get-Item` or `Get-ChildItem` command to find the certificate in `Cert:\CurrentUser\My`.

-ContentType <System.String>

Specifies the content type of the web request.

If this parameter is omitted and the request method is POST, `Invoke-RestMethod` sets the content type to "application/x-www-form-urlencoded". Otherwise, the content type is not specified in the call.

-Credential <System.Management.Automation.PSCredential>

Specifies a user account that has permission to send the request. The default is the current user.

Type a user name, such as User01 or Domain01\User01 , or enter a PSCredential object generated by the `Get-Credential` cmdlet.

Credentials are stored in a PSCredential (/dotnet/api/system.management.automation.pscredential)object and the password is stored as a SecureString (/dotnet/api/system.security.securestring).

> [!NOTE] > For more information about SecureString data protection, see > How secure is SecureString? (/dotnet/api/system.security.securestring#how-secure-is-securestring).

-DisableKeepAlive <System.Management.Automation.SwitchParameter>

Sets the KeepAlive value in the HTTP header to False. By default, KeepAlive is True. KeepAlive establishes a persistent connection to the server to facilitate subsequent requests.

-Headers <System.Collections.IDictionary>

Specifies the headers of the web request. Enter a hash table or dictionary.

To set UserAgent headers, use the UserAgent parameter. You cannot use this parameter to specify UserAgent or cookie headers.

-InFile <System.String>

Gets the content of the web request from a file.

Enter a path and file name. If you omit the path, the default is the current location.

-MaximumRedirection <System.Int32>

Determines how many times Windows PowerShell redirects a connection to an alternate Uniform Resource Identifier (URI) before the connection fails. The default value is 5. A value of 0 (zero) prevents all redirection.

-Method <Microsoft.PowerShell.Commands.WebRequestMethod>

Specifies the method used for the web request. The acceptable values for this parameter are:

- `Default`

- `Delete`

- `Get`

- `Head`

- `Merge`

- `Options`

- `Patch`

- `Post`

- `Put`

- `Trace`

-OutFile <System.String>

Saves the response body in the specified output file. Enter a path and file name. If you omit the path, the default is the current location.

By default, `Invoke-RestMethod` returns the results to the pipeline.

-PassThru <System.Management.Automation.SwitchParameter>

This parameter is valid only when the OutFile parameter is also used in the command. The intent is to have the results written to the file and to the pipeline.

> [!NOTE] > When you use the PassThru parameter, the output is written to the pipeline but the file is > empty. For more information, see > PowerShell Issue #15409 (https://github.com/PowerShell/PowerShell/issues/15409).

-Proxy <System.Uri>

Uses a proxy server for the request, rather than connecting directly to the Internet resource. Enter the URI of a network proxy server.

-ProxyCredential <System.Management.Automation.PSCredential>

Specifies a user account that has permission to use the proxy server that is specified by the Proxy parameter. The default is the current user.

Type a user name, such as "User01" or "Domain01\User01", or enter a PSCredential object, such as one generated by the `Get-Credential` cmdlet.

This parameter is valid only when the Proxy parameter is also used in the command. You cannot use the ProxyCredential and ProxyUseDefaultCredentials parameters in the same command.

-ProxyUseDefaultCredentials <System.Management.Automation.SwitchParameter>

Uses the credentials of the current user to access the proxy server that is specified by the Proxy parameter.

This parameter is valid only when the Proxy parameter is also used in the command. You cannot use the ProxyCredential and ProxyUseDefaultCredentials parameters in the same command.

-SessionVariable <System.String>

Creates a variable containing the web request session. Enter a variable name without the dollar sign (`$`) symbol.

When you specify a session variable, `Invoke-RestMethod` creates a web request session object and assigns it to a variable with the specified name in your PowerShell session. You can use the variable in your session as soon as the command completes.

Unlike a remote session, the web request session isn't a persistent connection. It's an object that contains information about the connection and the request, including cookies, credentials, the maximum redirection value, and the user agent string. You can use it to share state and data among web requests.

To use the web request session in subsequent web requests, specify the session variable in the value of the WebSession parameter. PowerShell uses the data in the web request session object when establishing the new connection. To override a value in the web request session, use a cmdlet parameter, such as UserAgent or Credential . Parameter values take precedence over values in the web request session.

You can't use the SessionVariable and WebSession parameters in the same command.

-TimeoutSec <System.Int32>

Specifies how long the request can be pending before it times out. Enter a value in seconds. The default value, 0, specifies an indefinite time-out.

A Domain Name System (DNS) query can take up to 15 seconds to return or time out. If your request contains a host name that requires resolution, and you set TimeoutSec to a value greater than zero, but less than 15 seconds, it can take 15 seconds or more before a WebException is thrown, and your request times out.

-TransferEncoding <System.String>

Specifies a value for the transfer-encoding HTTP response header. The acceptable values for this parameter are:

- `Chunked`

- `Compress`

- `Deflate`

- `GZip`

- `Identity`

-Uri <System.Uri>

Specifies the Uniform Resource Identifier (URI) of the Internet resource to which the web request is sent. This parameter supports HTTP, HTTPS, FTP, and FILE values.

This parameter is required. The parameter name (Uri) is optional.

-UseBasicParsing <System.Management.Automation.SwitchParameter>

Indicates that the cmdlet uses basic parsing. The cmdlet returns the raw

HTML in a String object.

-UseDefaultCredentials <System.Management.Automation.SwitchParameter>

Uses the credentials of the current user to send the web request.

-UserAgent <System.String>

Specifies a user agent string for the web request.

The default user agent is similar to "Mozilla/5.0 (Windows NT; Windows NT 6.1; en-US) WindowsPowerShell/3.0" with slight variations for each operating system and platform.

To test a website with the standard user agent string that is used by most Internet browsers, use the properties of the PSUserAgent (/dotnet/api/microsoft.powershell.commands)class, such as Chrome, FireFox, Internet Explorer, Opera, and Safari.

-WebSession <Microsoft.PowerShell.Commands.WebRequestSession>

Specifies a web request session. Enter the variable name, including the dollar sign (`$`).

To override a value in the web request session, use a cmdlet parameter, such as UserAgent or Credential . Parameter values take precedence over values in the web request session.

Unlike a remote session, the web request session is not a persistent connection. It is an object that contains information about the connection and the request, including cookies, credentials, the maximum redirection value, and the user agent string. You can use it to share state and data among web requests.

To create a web request session, enter a variable name (without a dollar sign) in the value of the SessionVariable parameter of an

    `Invoke-RestMethod` command. `Invoke-RestMethod` creates the session and
    saves it in the variable. In subsequent commands, use the variable as the
    value of the WebSession parameter.

    You cannot use the SessionVariable and WebSession parameters in the same
    command.

<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: Get the PowerShell RSS feed ------------

```
Invoke-RestMethod -Uri https://devblogs.microsoft.com/powershell/feed/ |
  Format-Table -Property Title, pubDate

Title                                                          pubDate
-----                                                          -------
Join the PowerShell 10th Anniversary Celebration!                    Tue, 08
Nov 2016 23:00:04 +0000
DSC Resource Kit November 2016 Release                               Thu, 03
Nov 2016 00:19:07 +0000
PSScriptAnalyzer Community Call - Oct 18, 2016                       Thu, 13
Oct 2016 17:52:35 +0000
New Home for In-Box DSC Resources                                    Sat, 08
Oct 2016 07:13:10 +0000
New Social Features on Gallery                                       Fri, 30
Sep 2016 23:04:34 +0000
PowerShellGet and PackageManagement in PowerShell Gallery and GitHub Thu, 29
Sep 2016 22:21:42 +0000
PowerShell Security at DerbyCon                                      Wed, 28
```

```
Sep 2016 01:13:19 +0000
DSC Resource Kit September Release                    Thu, 22
Sep 2016 00:25:37 +0000
PowerShell DSC and implicit remoting broken in KB3176934        Tue, 23
Aug 2016 15:07:50 +0000
PowerShell on Linux and Open Source!                  Thu, 18
Aug 2016 15:32:02 +0000
```

This command uses the `Invoke-RestMethod` cmdlet to get information from the PowerShell Blog RSS feed. The command uses the `Format-Table` cmdlet to display the values of the Title and pubDate properties of each blog in a table.

------------------------- Example 2 -------------------------

```
$Cred = Get-Credential

# Next, allow the use of self-signed SSL certificates.

[System.Net.ServicePointManager]::ServerCertificateValidationCallback = {
$True }

# Create variables to store the values consumed by the Invoke-RestMethod
command.
# The search variable contents are later embedded in the body variable.

$Server = 'server.contoso.com'
$Url = "https://${server}:8089/services/search/jobs/export"
$Search = "search index=_internal | reverse | table
index,host,source,sourcetype,_raw"

# The cmdlet handles URL encoding. The body variable describes the search
criteria, specifies CSV as
# the output mode, and specifies a time period for returned data that starts
two days ago and ends
```

```
# one day ago. The body variable specifies values for parameters that apply to
the particular REST
# API with which Invoke-RestMethod is communicating.

$Body = @{
    search = $Search
    output_mode = "csv"
    earliest_time = "-2d@d"
    latest_time = "-1d@d"
}

# Now, run the Invoke-RestMethod command with all variables in place,
specifying a path and file
# name for the resulting CSV output file.

Invoke-RestMethod -Method Post -Uri $url -Credential $Cred -Body $body
-OutFile output.csv

{"preview":true,"offset":0,"result":{"sourcetype":"contoso1","count":"9624"}}

{"preview":true,"offset":1,"result":{"sourcetype":"contoso2","count":"152"}}

{"preview":true,"offset":2,"result":{"sourcetype":"contoso3","count":"88494"}}

{"preview":true,"offset":3,"result":{"sourcetype":"contoso4","count":"15277"}}
```

--------------- Example 3: Pass multiple headers ---------------

```
$headers = @{
    'userId' = 'UserIDValue'
    'token' = 'TokenValue'
}
```

```
Invoke-RestMethod -Uri $uri -Method Post -Headers $headers -Body $body
```

APIs often require passed headers for authentication, validation etc.

--------------- Example 3: Submitting form data ---------------

```
$R = Invoke-WebRequest https://website.com/login.aspx
$R.Forms[0].Name = "MyName"
$R.Forms[0].Password = "MyPassword"
Invoke-RestMethod https://website.com/service.aspx -Body $R.Forms[0]
```

----- Example 4: Enumerate returned items on the pipeline -----

```
$uri = 'https://api.github.com/repos/microsoftdocs/powershell-docs/issues'
$x = 0
Invoke-RestMethod -Uri $uri | ForEach-Object { $x++ }
$x
1

$x = 0
(Invoke-RestMethod -Uri $uri) | ForEach-Object { $x++ }
$x
30

$x = 0
Invoke-RestMethod -Uri $uri | Write-Output | ForEach-Object { $x++ }
$x
30
```

REMARKS

To see the examples, type: "get-help Invoke-RestMethod -examples".

For more information, type: "get-help Invoke-RestMethod -detailed".

For technical information, type: "get-help Invoke-RestMethod -full".

For online help, type: "get-help Invoke-RestMethod -online"