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

PowerShell Get-Help on command 'New-WebServiceProxy'

PS C:\Users\wahid> Get-Help New-WebServiceProxy

NAME

New-WebServiceProxy

SYNOPSIS

Creates a Web service proxy object that lets you use and manage the Web service in PowerShell.

SYNTAX

New-WebServiceProxy [-Uri] <System.Uri> [[-Class] <System.String>] [[-Namespace] <System.String>] [-Credential <System.Management.Automation.PSCredential>] [<CommonParameters>]

New-WebServiceProxy [-Uri] <System.Uri> [[-Class] <System.String>] [[-Namespace] <System.String>] [-UseDefaultCredential] [<CommonParameters>]

DESCRIPTION

The `New-WebServiceProxy` cmdlet lets you use a Web service in PowerShell. The cmdlet connects to a Web service and creates a Web service proxy object in PowerShell. You can use the proxy object to manage the Web service.

A Web service is an XML-based program that exchanges data over a network, especially over the Internet. The Microsoft .NET Framework provides Web service proxy objects that represent the Web service as a .NET Framework object.

PARAMETERS

-Class <System.String>

Specifies a name for the proxy class that the cmdlet creates for the Web service. The value of this parameter is used together with the Namespace parameter to provide a fully qualified name for the class. The default value is generated from the Uniform Resource Identifier (URI).

-Credential <System.Management.Automation.PSCredential>

Specifies a user account that has permission to perform this action. The default is the current user. This is an alternative to using the UseDefaultCredential parameter.

Type a user name, such as User01 or Domain01\User01, or enter a PSCredential object, such as one generated by the `Get-Credential` cmdlet. If you type a user name, this cmdlet prompts you for a password.

-Namespace <System.String>

Specifies a namespace for the new class.

The value of this parameter is used together with the value of the Class parameter to generate a fully qualified name for the class. The default value is Microsoft.PowerShell.Commands.NewWebserviceProxy.AutogeneratedTypes plus a type that is generated from the URI.

You can set the value of the Namespace parameter so that you can access

multiple Web services that have the same name.

-Uri <System.Uri>
Specifies the URI of the Web service. Enter a URI or the path and filename of a file that contains a service description.

The URI must return an `.asmx` page or to a page that returns a service description. To return a service description of a Web service that was created using ASP.NET, append "?WSDL" to the URL of the Web service (for example, `http://www.contoso.com/MyWebService.asmx?WSDL`).

-UseDefaultCredential <System.Management.Automation.SwitchParameter>
Indicates that this cmdlet uses the default credential. This cmdlet sets the UseDefaultCredential property in the resulting proxy object to True. This is an alternative to using the Credential parameter.

<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: Create a proxy for a Web service ---------

```
$calc = New-WebServiceProxy -Uri
"http://www.dneonline.com/calculator.asmx?wsdl"
```

Example 2: Create a proxy for a Web service and specify namespace and class

```
$URI = "http://www.dneonline.com/calculator.asmx?wsdl"
$calc = New-WebServiceProxy -Uri $URI -Namespace "WSProxy" -Class "Calculator"
```

The command uses the Uri parameter to specify the URI and the Namespace and Class parameters to specify the namespace and class of the object.

------ Example 3: Display methods of a Web service proxy ------

```
$calc | Get-Member -MemberType method

TypeName: WSProxy.Calculator

Name                 MemberType Definition
----                 ---------- ----------
Abort                Method     void Abort()
Add                  Method     int Add(int intA, int intB)
AddAsync             Method     void AddAsync(int intA, int intB), void
AddAsync(int intA,
BeginAdd             Method     System.IAsyncResult BeginAdd(int intA,
int intB, System.Asy
BeginDivide          Method     System.IAsyncResult BeginDivide(int intA,
int intB, System.
BeginMultiply        Method     System.IAsyncResult BeginMultiply(int
intA, int intB, Syste
BeginSubtract        Method     System.IAsyncResult BeginSubtract(int
intA, int intB, Syste
CancelAsync          Method     void CancelAsync(System.Object userState)
CreateObjRef         Method     System.Runtime.Remoting.ObjRef
CreateObjRef(type requestedT
Discover             Method     void Discover()
Dispose              Method     void Dispose(), void IDisposable.Dispose()
Divide               Method     int Divide(int intA, int intB)
DivideAsync          Method     void DivideAsync(int intA, int intB),
void DivideAsync(int
EndAdd               Method     int EndAdd(System.IAsyncResult
asyncResult)
EndDivide            Method     int EndDivide(System.IAsyncResult
```

asyncResult)
EndMultiply Method int EndMultiply(System.IAsyncResult
asyncResult)
EndSubtract Method int EndSubtract(System.IAsyncResult
asyncResult)
Equals Method bool Equals(System.Object obj)
GetHashCode Method int GetHashCode()
GetLifetimeService Method System.Object GetLifetimeService()
GetType Method type GetType()
InitializeLifetimeService Method System.Object InitializeLifetimeService()
Multiply Method int Multiply(int intA, int intB)
MultiplyAsync Method void MultiplyAsync(int intA, int intB),
void MultiplyAsync(
Subtract Method int Subtract(int intA, int intB)
SubtractAsync Method void SubtractAsync(int intA, int intB),
void SubtractAsync(
ToString Method string ToString()

This example uses the `Get-Member` cmdlet to display the methods of the Web service proxy object in the `$calc` variable. We uses these methods in the following example.

Notice that the TypeName of the proxy object, WebServiceProxy, reflects the namespace and class names that were specified in the previous example.

---------- Example 4: Use a Web service proxy ----------

PS> $calc.Multiply(6,7)
42

This example uses the Web service proxy stored in the `$calc` variable. The command uses the Multiply method of the proxy.

REMARKS

To see the examples, type: "get-help New-WebServiceProxy -examples".

For more information, type: "get-help New-WebServiceProxy -detailed".

For technical information, type: "get-help New-WebServiceProxy -full".

For online help, type: "get-help New-WebServiceProxy -online"