PowerShell Get-Help on command 'Assert-MockCalled'

PS C:\Users\wahid> Get-Help Assert-MockCalled

```
NAME
    Assert-MockCalled


SYNOPSIS
    Checks if a Mocked command has been called a certain number of times
    and throws an exception if it has not.



SYNTAX
    Assert-MockCalled [-CommandName] <String> [[-Times] <Int32>]
    [[-ParameterFilter] <ScriptBlock>] [[-ModuleName] <String>] [[-Scope]
    <String>] [-Exactly] [<CommonParameters>]

    Assert-MockCalled [-CommandName] <String> [[-Times] <Int32>] -ExclusiveFilter
    <ScriptBlock> [[-ModuleName] <String>] [[-Scope] <String>] [-Exactly]
    [<CommonParameters>]



DESCRIPTION
    This command verifies that a mocked command has been called a certain number
    of times.  If the call history of the mocked command does not match the
```

parameters
passed to Assert-MockCalled, Assert-MockCalled will throw an exception.

PARAMETERS

-CommandName <String>

The mocked command whose call history should be checked.

-Times <Int32>

The number of times that the mock must be called to avoid an exception from throwing.

-ParameterFilter <ScriptBlock>

An optional filter to qualify wich calls should be counted. Only those calls to the mock whose parameters cause this filter to return true will be counted.

-ExclusiveFilter <ScriptBlock>

-ModuleName <String>

The module where the mock being checked was injected. This is optional, and must match the ModuleName that was used when setting up the Mock.

-Scope <String>

An optional parameter specifying the Pester scope in which to check for calls to the mocked command. By default, Assert-MockCalled will find all calls to the mocked command in the current Context block (if present), or the current Describe block (if there is no active Context.) Valid values are Describe, Context and It. If you use a scope of Describe or Context, the command will identify all calls to the mocked command in the current Describe / Context block, as well as all child scopes of that block.

-Exactly [<SwitchParameter>]

If this switch is present, the number specified in Times must match exactly the number of times the mock has been called. Otherwise it must match "at least" the number of times specified. If the value passed to the Times parameter is zero, the Exactly switch is implied.

<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 --------------------------

```
C:\PS>Mock Set-Content {}

{... Some Code ...}

C:\PS>Assert-MockCalled Set-Content
```

This will throw an exception and cause the test to fail if Set-Content is not called in Some Code.

-------------------------- EXAMPLE 2 --------------------------

```
C:\PS>Mock Set-Content -parameterFilter {$path.StartsWith("$env:temp\")}

{... Some Code ...}

C:\PS>Assert-MockCalled Set-Content 2 { $path -eq "$env:temp\test.txt" }
```

This will throw an exception if some code calls Set-Content on $path=$env:temp\test.txt less than 2 times

-------------------------- EXAMPLE 3 --------------------------

```
C:\PS>Mock Set-Content {}

{... Some Code ...}

C:\PS>Assert-MockCalled Set-Content 0
```

This will throw an exception if some code calls Set-Content at all

-------------------------- EXAMPLE 4 --------------------------

```
C:\PS>Mock Set-Content {}

{... Some Code ...}

C:\PS>Assert-MockCalled Set-Content -Exactly 2
```

This will throw an exception if some code does not call Set-Content Exactly two times.

-------------------------- EXAMPLE 5 --------------------------

```
PS C:\>Describe 'Assert-MockCalled Scope behavior' {

Mock Set-Content { }

    It 'Calls Set-Content at least once in the It block' {
        {... Some Code ...}

        Assert-MockCalled Set-Content -Exactly 0 -Scope It
    }
}
```

Checks for calls only within the current It block.

-------------------------- EXAMPLE 6 --------------------------

```
PS C:\>Describe 'Describe' {

Mock -ModuleName SomeModule Set-Content { }

    {... Some Code ...}

    It 'Calls Set-Content at least once in the Describe block' {
        Assert-MockCalled -ModuleName SomeModule Set-Content
    }
}
```

Checks for calls to the mock within the SomeModule module. Note that both the

Mock

and Assert-MockCalled commands use the same module name.

-------------------------- EXAMPLE 7 --------------------------

```
PS C:\>Assert-MockCalled Get-ChildItem -ExclusiveFilter { $Path -eq 'C:\' }
```

Checks to make sure that Get-ChildItem was called at least one time with the -Path parameter set to 'C:\', and that it was not called at all with the -Path parameter set to any other value.

REMARKS

To see the examples, type: "get-help Assert-MockCalled -examples".

For more information, type: "get-help Assert-MockCalled -detailed".

For technical information, type: "get-help Assert-MockCalled -full".