Command Syntax Part 2

How do you tell which Parameters are optional and which arguments are required?

To answer that question we need PowerShell's Help system

Type **help** get-service -showwindow

Scroll down to the Syntax section and what I have done is I've copied the Syntax section into my scripting Pane. So you can use show window and I'll view the syntax from the scripting pane.

Now watch this I'm going **to type get-service**, notice that this **cmdlet runs** without **adding any parameters.**

Now check this out type **get-eventlog** press return, notice what it says:

```
Get-EventLog
cmdlet Get-EventLog at command pipeline position
1
Supply values for the following parameters:
LogName:
```

So, the **question** is why did **get-service run** when **get-eventlog** required a **value** for the -**logname parameter**?

1 The answer is in the syntax of the help files for get-service and get-eventlog

Lets checkout **get-service first**. Let's start with **parameter set #2 and set #3**

Optional [Parameter Argument] **Notice** that all the **parameters and the arguments** are surrounded by **square brackets**.

This means that adding the **parameters are optional** and not needed. So,

2 Get-service will run **without adding any parameters**.

Now let's take a look at the **syntax for get-eventlog**

If you want to follow along type help get-eventlog -showwindow

Again I've copied the syntax for get-eventlog to my scripting pane

Let's go through the **syntax**, notice that almost **every parameter and every argument are surrounded by square brackets**,

that means that they are **all optional** or **not needed**.

Notice **-logname** is **surrounded** by **square brackets** but the **argument is not**.

Required Argument

That means, because there are **square brackets** around the parameter -**logname**, the name **logname is optional** but the **argument <string> is required.**

That is **why** when you ran **get-eventlog without** any parameters, **PowerShell asks** for a **value** for **-logname**

Now go back to get-eventlog, logname:

type Application and **press return**, and the **command runs.**

3 Let's go ahead and clear the screen cls

4 Positional parameters [Param] (Use get-eventlog scroll down to #Position)

Type help get-eventlog -showwindow

Scroll down until you see the **parameter attributes list**. We are going to be working with **three parameters** from this list,

-InstanceID, -logname and **-newest**

From the **list** notice that the **parameter -logname** has a position of **0**,

-InstanceID has a position of **1** and **-newest** has a position of **named**.

Now what does this mean.

0, 1 and name, refer to the **actual position** that the **cmdlet** must be **placed** in the **order** of the **cmdlets**.

So **-logname** is **positional**, it's **position is 0** which is the **first position**.

The parameter **-instanceID** position is **1** which is **after -logname** in the **order of cmdlets**

Ok lets check that out

So type **get-eventlog application 0,1**

(I **don't have** to type the parameter **-logname** or the parameter **-InstanceID**, because they are **both optional because the're surrounded by square brackets**

Now take a look at the **argument** for **-InstanceID** because there are **two square** brackets

within the **angle brackets** the parameter -InstanceID can take **multiple arguments**. Now press return and that runs.

Now lets see if we can **move these values out of order**. **Move the 0,1** in **front** of **application** and **press return**.

An we get **an error** because **Powershell expects** the **positional parameter -logname** or the **value type application** to be 5 the **first in the list**.

6 Named Parameters

Take a look at **-Newest**,

notice that the **position** is **named**.

Named means that you can put **-newest anywhere** in the **order of parameters** and it **will work**. Let's check it out

Type **get-eventlog -newest 5 application 0,1** and that worked.

We see that we moved the parameter with a position called named and moved that in front of the positional parameter -logname and we see that the command ran