Module 09 - Azure Automation - Troubleshooting

Exercise 1: Using Azure Automation Basic Logging

Introduction

In this lab, we will look at the various options available for writing logs from your Runbooks. This is important in a production environment, as you will need to ensure appropriate logs are written, to help troubleshoot any Runbook problems.

Summary

During this lab, we will:

- Create a Runbook
- Enable the available logging
- View log outputs

Estimated Time to Complete This Lab

30 minutes

Task 1: Create a Basic Logging Stream Test Runbook

1. [] Sign into https://portal.azure.com using your account.

2. [] Navigate to the **ContosoAutomationAccount** Automation Account.

3. [] Click **Runbooks**.

4. [] Click **+Create a runbook**.

5. [] Type in the following, then click **Create**:

    - **Name:** Start-LoggingLab
    - **Runbook type:** PowerShell Workflow

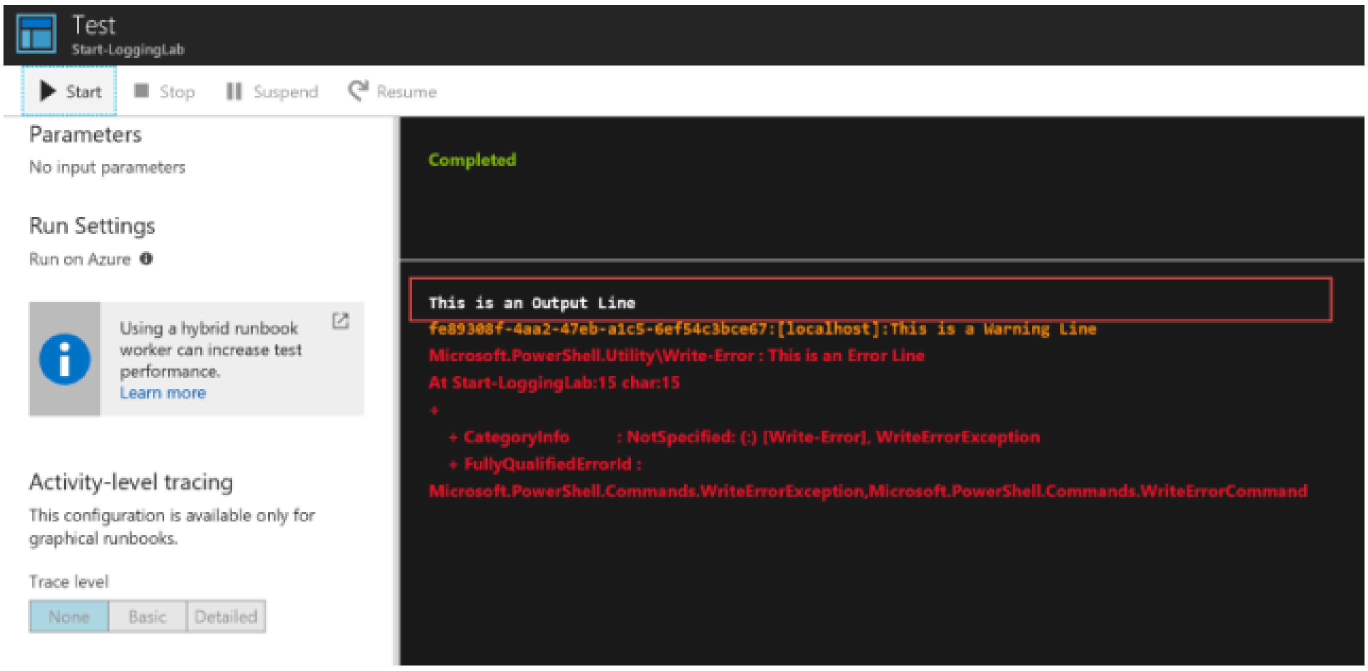
6. [] Type or copy the following code into the Canvas:

```
workflow Start-LoggingLab
{
    Write-Output "This is an Output Line"

    Write-Debug "This is a Debug Line"

    Write-Verbose "This is a Verbose Line"

    Write-Progress "This is a Progress Line"

    Write-Warning "This is a Warning Line"

    Write-Error "This is an Error Line"
}
```

> You can see that we are adding a single line for each output stream type. We will go through multiple scenarios to see what output we get.

8. [] Click **Save** -> **Test pane**.
9. [] Click **Start**. (Be sure to run this on **Azure**, not on a Hybrid worker)
10. [] We can see that it has only three stream outputs:

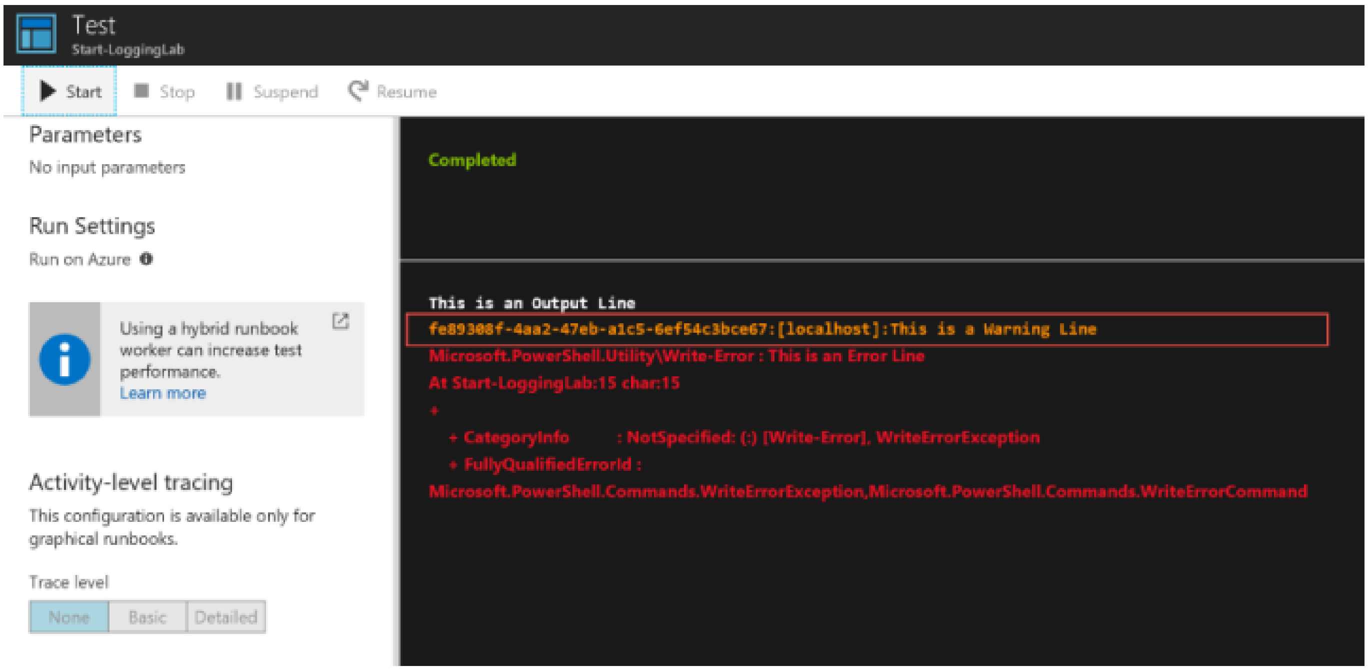
Write-Output



```
This is an Output Line
fe89308f-4aa2-47eb-a1c5-6ef54c3bce67:[localhost]:This is a Warning Line
Microsoft.PowerShell.Utility\Write-Error : This is an Error Line
At Start-LoggingLab:15 char:15
+
  + CategoryInfo          : NotSpecified: (:) [Write-Error], WriteErrorException
  + FullyQualifiedErrorId :
Microsoft.PowerShell.Commands.WriteErrorException,Microsoft.PowerShell.Commands.WriteErrorCommand
```

{400}

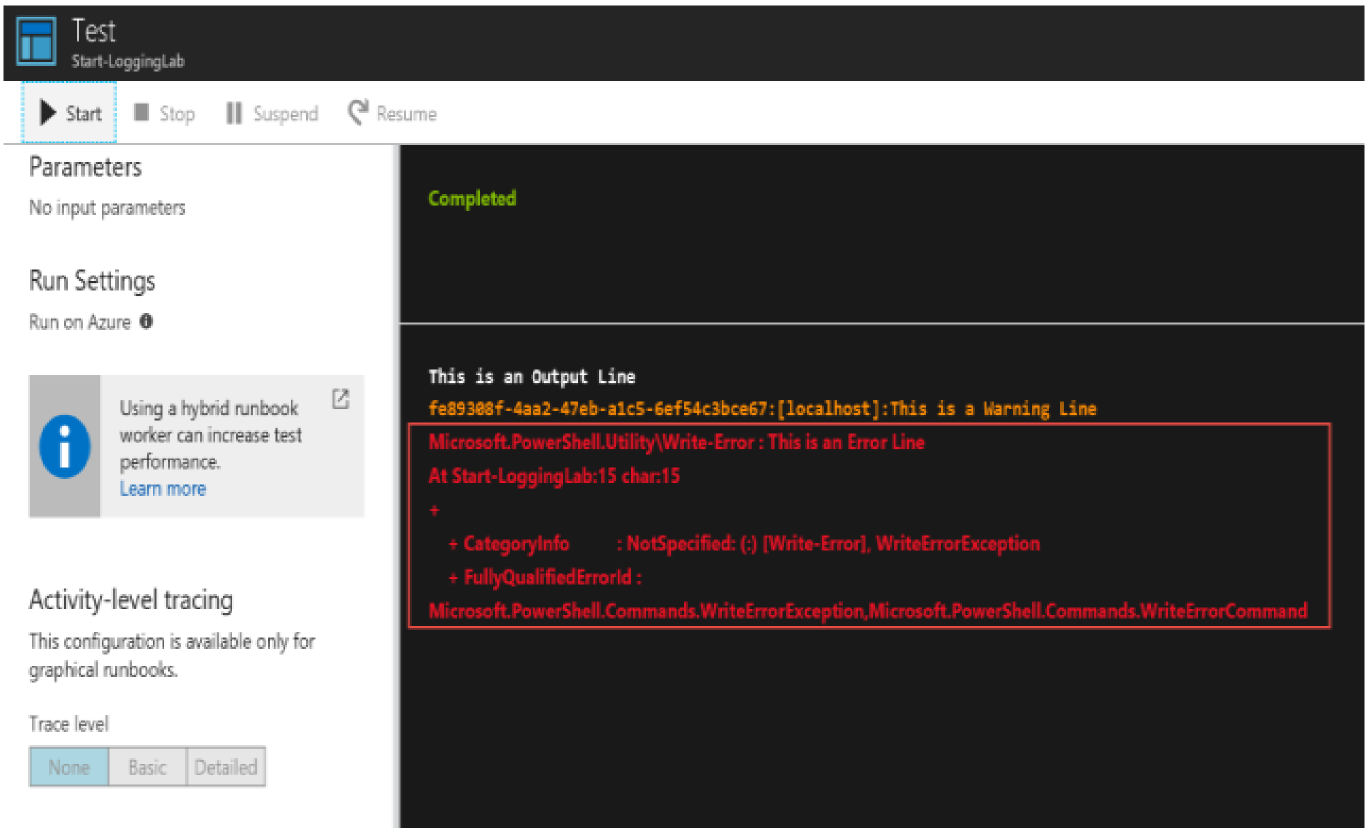
Write-Warning



```
This is an Output Line
fe89308f-4aa2-47eb-a1c5-6ef54c3bce67:[localhost]:This is a Warning Line
Microsoft.PowerShell.Utility\Write-Error : This is an Error Line
At Start-LoggingLab:15 char:15
+
  + CategoryInfo          : NotSpecified: (:) [Write-Error], WriteErrorException
  + FullyQualifiedErrorId :
Microsoft.PowerShell.Commands.WriteErrorException,Microsoft.PowerShell.Commands.WriteErrorCommand
```

{400}

Write-Error



```
This is an Output Line
fe89308f-4aa2-47eb-a1c5-6ef54c3bce67:[localhost]:This is a Warning Line
Microsoft.PowerShell.Utility\Write-Error : This is an Error Line
At Start-LoggingLab:15 char:15
+
  + CategoryInfo          : NotSpecified: (:) [Write-Error], WriteErrorException
  + FullyQualifiedErrorId :
Microsoft.PowerShell.Commands.WriteErrorException,Microsoft.PowerShell.Commands.WriteErrorCommand
```

{400}

> [!note] Debug, Verbose, and Progress are missing

11. [] Close the **Test** blade.

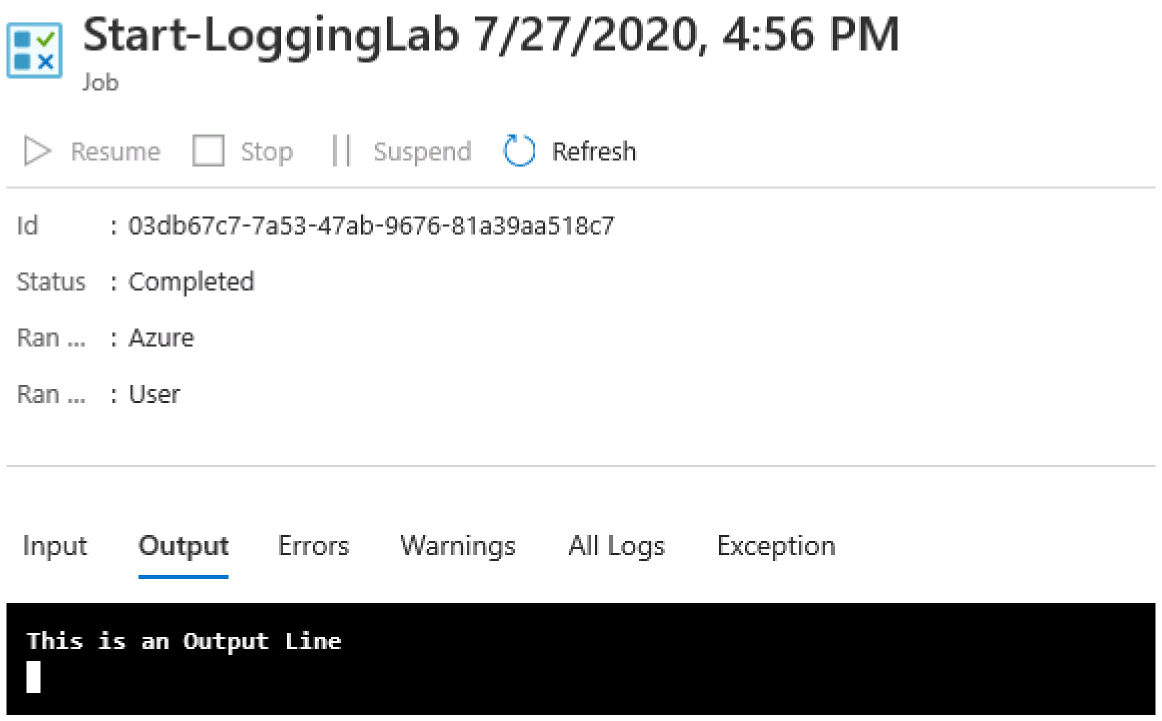
12. [] We will now publish the runbook and look at how we see the output from a Job.

13. [] Click **Publish** -> **Yes**.

14. [] Click **Start** -> **Yes**.

15. [] The Job blade will expand out. After the **Status** shows as **Completed**, click **Output**.

Notice that we only see the **Output** line.



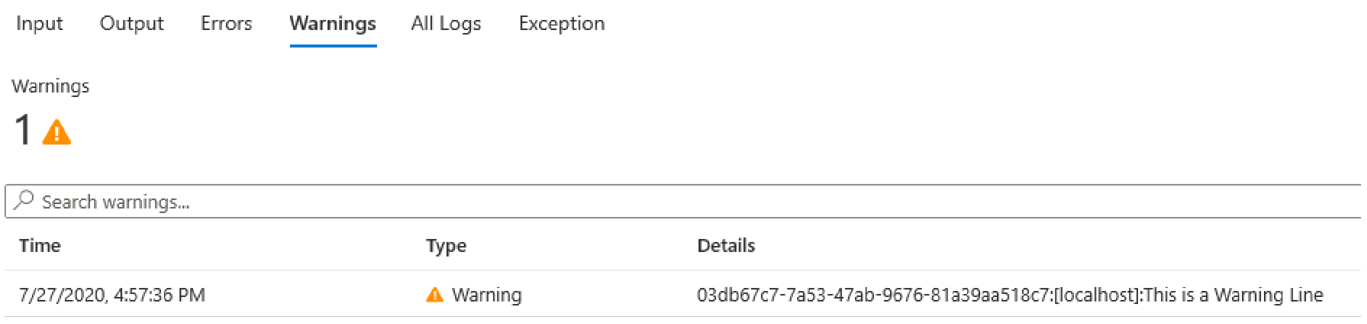
Start-LoggingLab 7/27/2020, 4:56 PM
Job

Id : 03db67c7-7a53-47ab-9676-81a39aa518c7
Status : Completed
Ran ... : Azure
Ran ... : User

```
This is an Output Line
```

{400}

16. [] Click **Warnings** to view output.

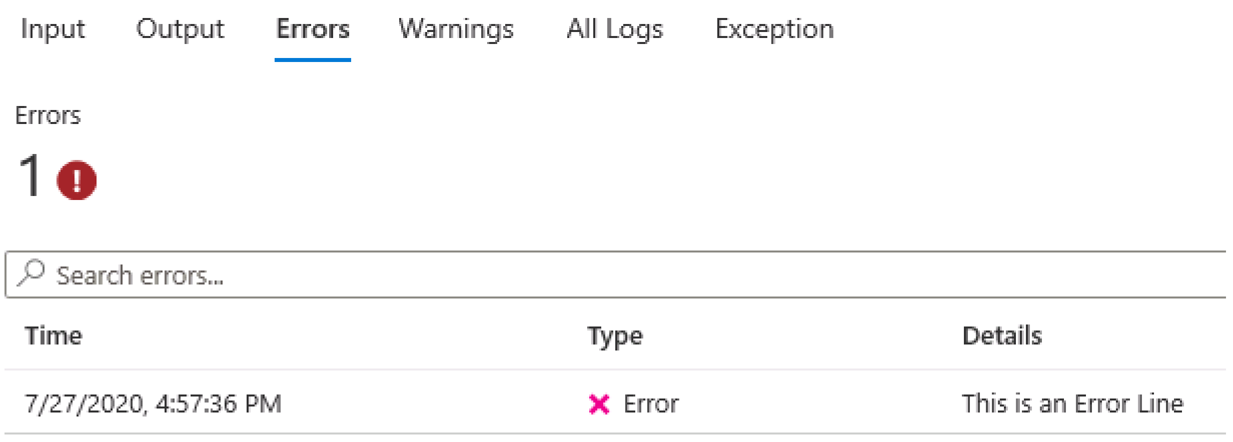


Warnings
1

Time	Type	Details
7/27/2020, 4:57:36 PM	Warning	03db67c7-7a53-47ab-9676-81a39aa518c7:[localhost]:This is a Warning Line

{400}

17. [] Click **Errors**.



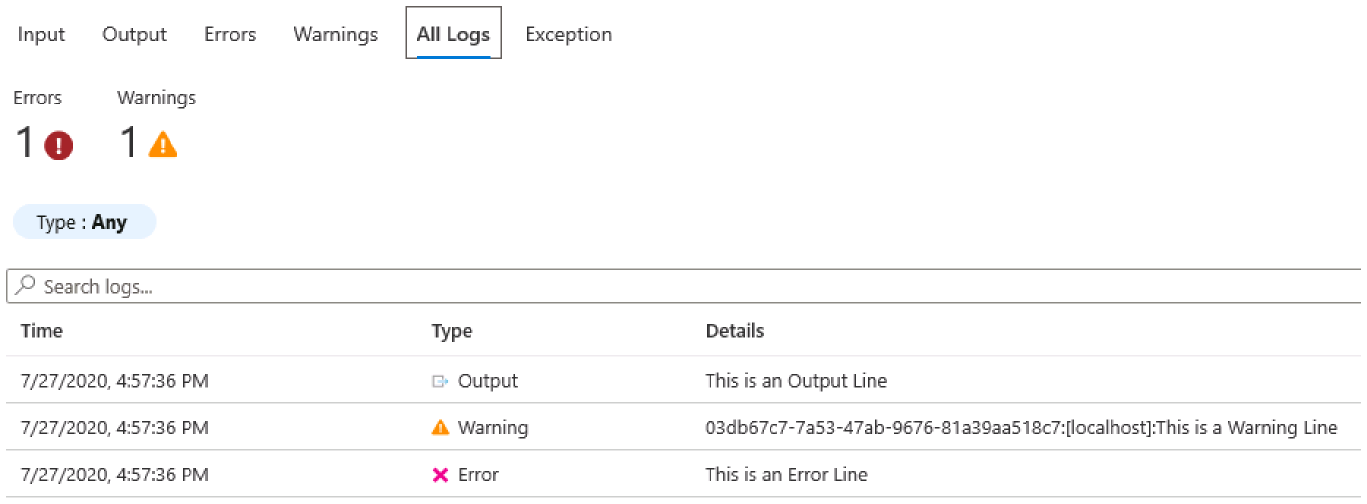
Errors
1

Time	Type	Details
7/27/2020, 4:57:36 PM	Error	This is an Error Line

{400}

18. [] Finally, we can look at all the logs by clicking **All Logs**.

19. []**All Logs** shows us each stream all together.



Time	Type	Details
7/27/2020, 4:57:36 PM	Output	This is an Output Line
7/27/2020, 4:57:36 PM	Warning	03db67c7-7a53-47ab-9676-81a39aa518c7:[localhost]:This is a Warning Line
7/27/2020, 4:57:36 PM	Error	This is an Error Line

{400}

> Ok, so what has happened to the other streams? First let's look at enabling those in the portal.

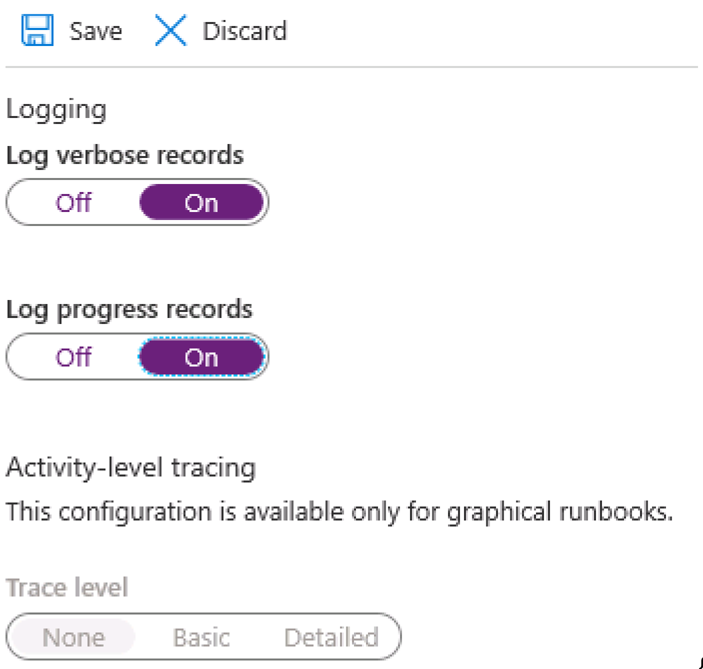
Task 2: Explore other Streams: Verbose, Debug, and Progress

1. [] If open, close the **Streams** and **All Logs** blade. Next, close the **Job** blade.

2. [] From the **Start-LoggingLab** runbook, look at the menu on the left and click **Logging and tracing**

3. [] Change the following:

   - **Log verbose records:** On
   - **Log progress records:** On



Logging

Log verbose records: Off [✓ On]

Log progress records: Off [✓ On]

Activity-level tracing

This configuration is available only for graphical runbooks.

Trace level: None Basic Detailed

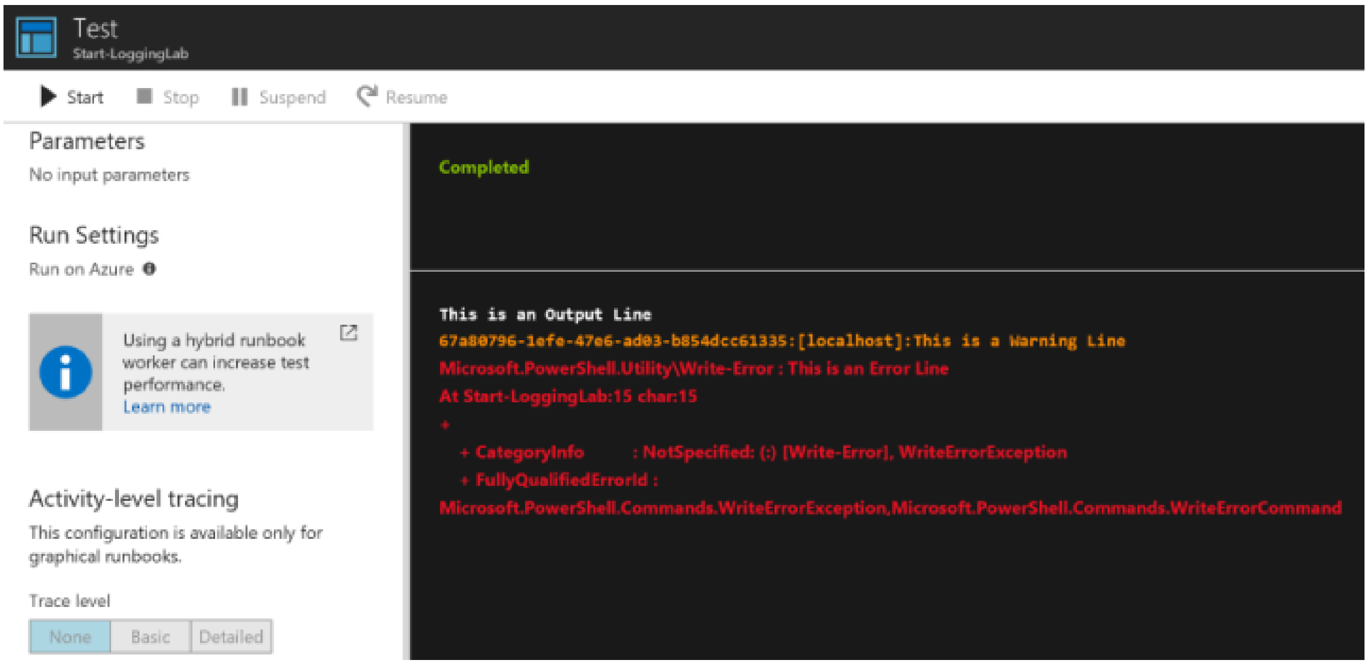
{400}

4. [] Click **Save**.

5. [] Click **Overview** > **Edit**.

6. [] Click **Test pane**.

7. [] Click **Start**.



```
Completed

This is an Output Line
67a80796-1efe-47e6-ad03-b854dcc61335:[localhost]:This is a Warning Line
Microsoft.PowerShell.Utility\Write-Error : This is an Error Line
At Start-LoggingLab:15 char:15
+
    + CategoryInfo          : NotSpecified: (:) [Write-Error], WriteErrorException
    + FullyQualifiedErrorId :
Microsoft.PowerShell.Commands.WriteErrorException,Microsoft.PowerShell.Commands.WriteErrorCommand
```

{400}

> Notice that nothing actually changed; even though we enabled Progress and Verbose logs.

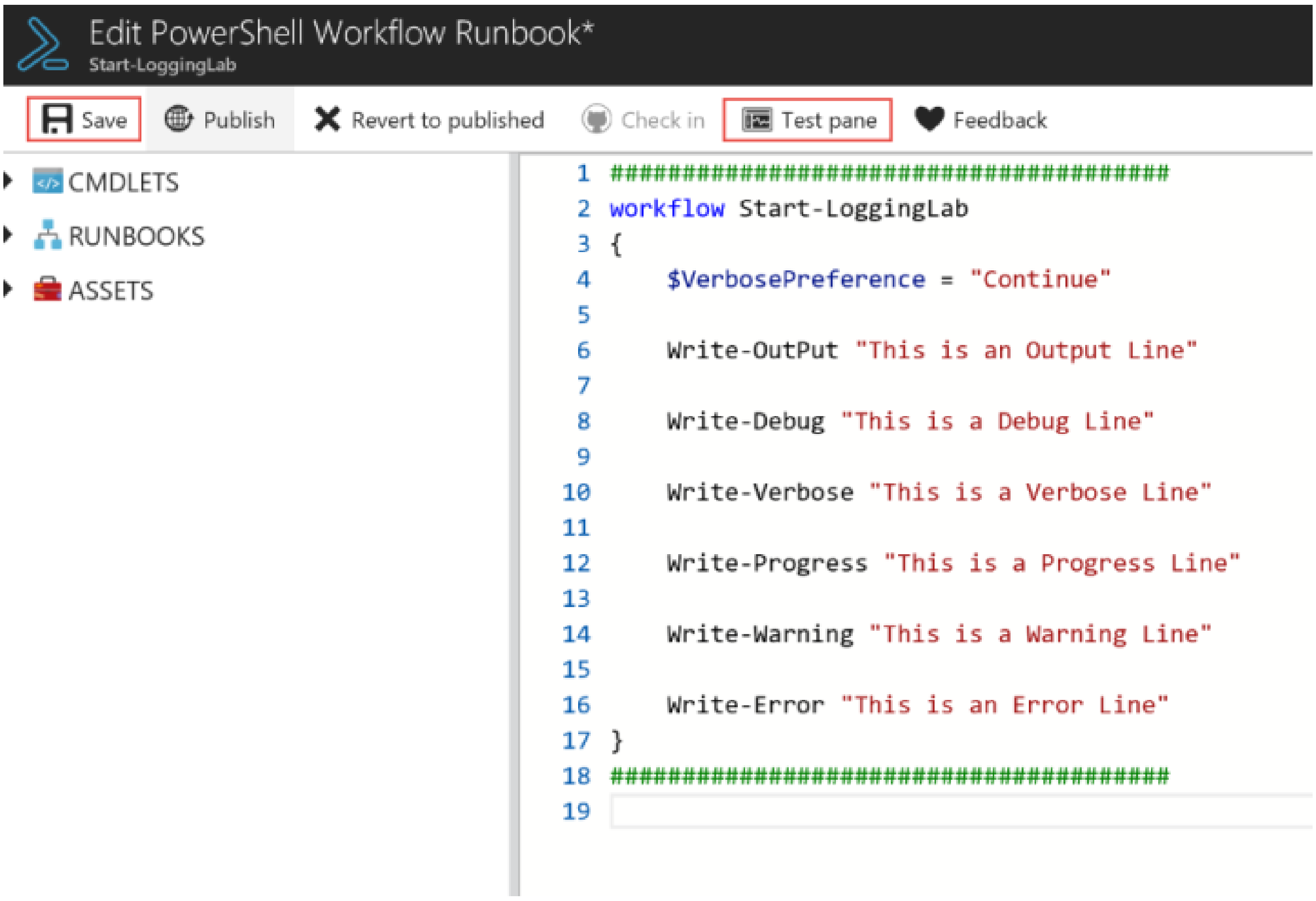
8. [] We can enable Verbose logging for the Test Output by setting `$VerbosePreference` in the script.

9. [] Close the Test blade.

10. [] Let's add the following line at line 4 of our script:

```
$VerbosePreference = "Continue"
```

11. [] Click **Save** > **Test pane**.

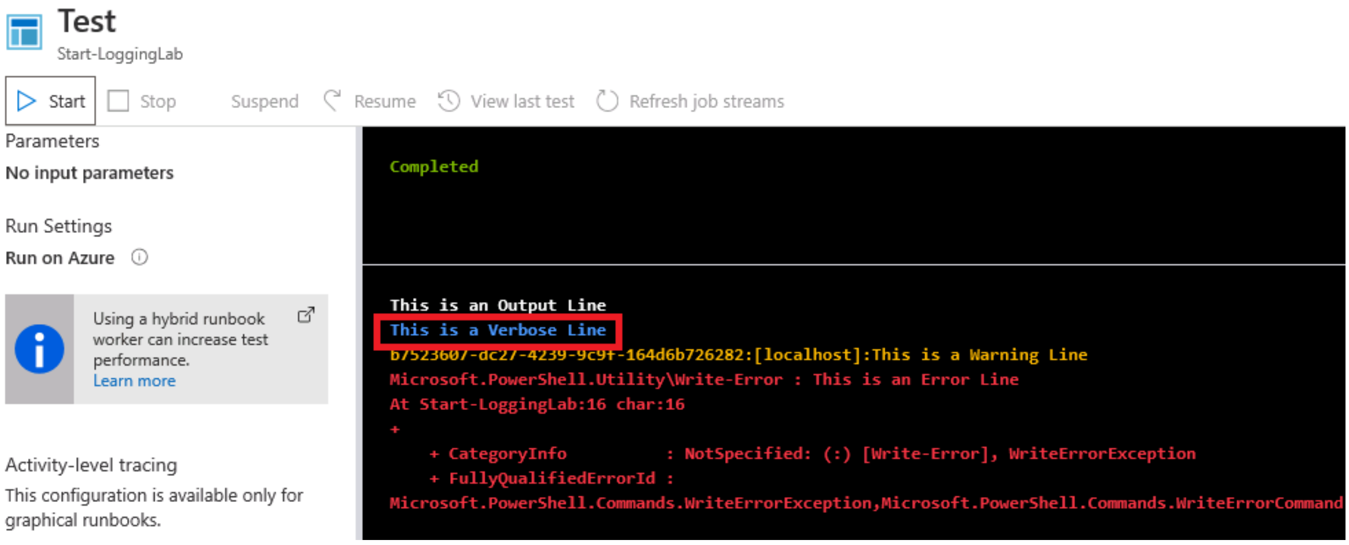


```
##########################################
workflow Start-LoggingLab
{
    $VerbosePreference = "Continue"

    Write-OutPut "This is an Output Line"

    Write-Debug "This is a Debug Line"

    Write-Verbose "This is a Verbose Line"

    Write-Progress "This is a Progress Line"

    Write-Warning "This is a Warning Line"

    Write-Error "This is an Error Line"
}
##########################################
```

{400}

12. [] Click **Start**.

13. [] Notice that we now have an extra line with **This is a Verbose line**.



```
Completed

This is an Output Line
This is a Verbose Line
b7523607-dc27-4239-9c9f-164d6b726282:[localhost]:This is a Warning Line
Microsoft.PowerShell.Utility\Write-Error : This is an Error Line
At Start-LoggingLab:16 char:16
+
    + CategoryInfo          : NotSpecified: (:) [Write-Error], WriteErrorException
    + FullyQualifiedErrorId :
Microsoft.PowerShell.Commands.WriteErrorException,Microsoft.PowerShell.Commands.WriteErrorCommand
```

{400}

14. [] We can also do this within an InlineScript by using the following line:

```
$VerbosePreference =
[System.Management.Automation.ActionPreference]$Using:VerbosePreference
```

15. [] Close the **Test** blade.

16. [] Add the following at line 5 and then move the code down to line 6:

```
InlineScript{
    $VerbosePreference =
[System.Management.Automation.ActionPreference]$Using:VerbosePreference
    Write-Verbose "This is a Verbose Line Within an InlineScript"
}
```

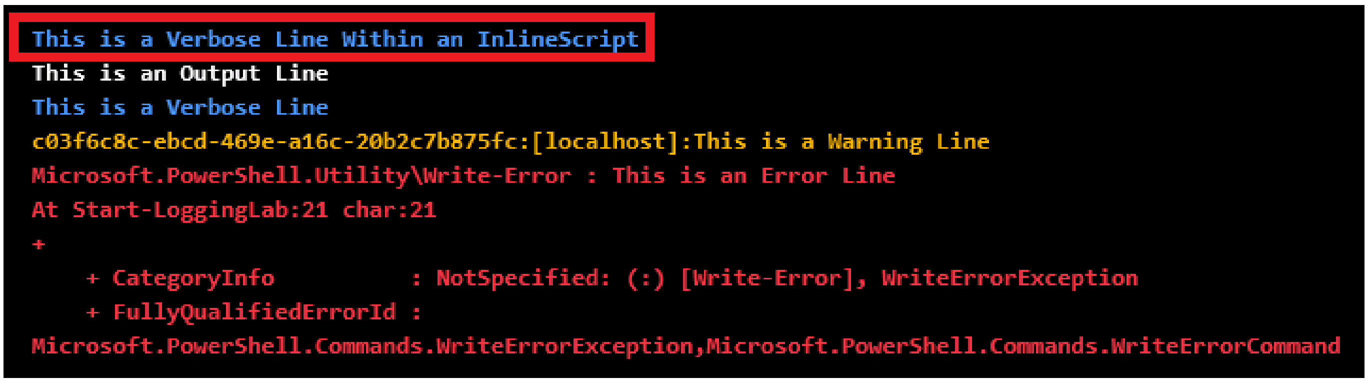
Your code should now look like this:

```
workflow Start-LoggingLab
{

    $VerbosePreference = "Continue"

    InlineScript {
        $VerbosePreference =
[System.Management.Automation.ActionPreference]$Using:VerbosePreference
        Write-Verbose "This is a Verbose Line Within an InlineScript"
    }

    Write-Output "This is an Output Line"

    Write-Debug "This is a Debug Line"

    Write-Verbose "This is a Verbose Line"

    Write-Progress "This is a Progress Line"

    Write-Warning "This is a Warning Line"

    Write-Error "This is an Error Line"
}
```

17. [] Click **Save** > **Test pane**.

18. [] Click **Start**.

Your output shows the Verbose line within the InlineScript.



```
This is a Verbose Line Within an InlineScript
This is an Output Line
This is a Verbose Line
c03f6c8c-ebcd-469e-a16c-20b2c7b875fc:[localhost]:This is a Warning Line
Microsoft.PowerShell.Utility\Write-Error : This is an Error Line
At Start-LoggingLab:21 char:21
+
    + CategoryInfo          : NotSpecified: (:) [Write-Error], WriteErrorException
    + FullyQualifiedErrorId :
Microsoft.PowerShell.Commands.WriteErrorException,Microsoft.PowerShell.Commands.WriteErrorCommand
```

{400}

19. [] Close the **Test** blade.
20. [] Now, let's publish our runbook and see how our Streams and Outputs look.
21. [] Click **Publish** > **Yes**.
22. [] Click **Start** > **Yes**.
23. [] A job blade will expand. The **Output, Warning and Error** streams will all look the same as before. Click **All Logs** to see the difference now that we enabled.
24. [] We can see that now we have all streams available for review except for Debug and Progress.

> `Write-Debug` and `Write-Progress` are strictly for console use and do not appears in Azure Automation logs. It is best practice to utilize `Write-Verbose` for general comments you want to appear in the job output for testing and troubleshooting.