INTEGRATION MODULE FOR AZURE DEVOPS

For Kelverion Runbook Studio and Azure Automation

Release Notes

Version 1.2

February 2021

Introduction

The Kelverion Integration Module for Azure DevOps is a new integration module for Runbook Studio and Azure Automation that lets you integrate your runbooks with Azure DevOps. The Kelverion Integration Module for Azure DevOps is Microsoft Azure Certified.

The integration module includes the following activities:

Activity	Description
Add-DevOpsAttachmentContent	Add attachment content on work item
Get-DevOpsAttachmentContent	Get attachment content
Get-DevOpsAttachmentInfo	Get attachment information
Get-DevOpsWorkItem	Get work item
New-DevOpsWorkItem	Create a new work item
Remove-DevOpsAttachment	Remove attachment from a work item
Remove-DevOpsWorkItem	Remove work item
Set-DevOpsWorkItem	Update a work item

System Requirements

The Integration Module for Azure DevOps requires the following software to be installed and configured prior to implementing the integration. For more information on installing Kelverion Runbook Studio, please refer to the Kelverion Runbook Studio User Guide.

- Kelverion Runbook Studio 3.4
- Microsoft .NET Framework 4.6.2

Azure DevOps or Microsoft TFS:

- Azure DevOps Services
- Azure DevOps Server 2020
- TFS 2017 Update 3, or TFS 2018 Update 3.2

Installing the Integration Module

The easiest way to install and deploy the Integration Module for Azure DevOps is from the PowerShell Gallery, but you can also download the module from Kelverion and perform the steps manually.

You must install and deploy the integration module to each Azure Automation Account and Hybrid Worker host system that you plan to use to run your runbooks. You must also install the integration module on any Runbook Studio host systems that you will be using to build and manage your runbooks.

Using the PowerShell Gallery

Using the commands in the **PowerShellGet** module you can download the Kelverion Integration Module for Azure DevOps from the PowerShell Gallery and install it on your local computer. You can also deploy the module directly from the PowerShell Gallery to any of your Azure Automation Accounts.

Install the integration module on your local computer or hybrid worker:

1. Confirm that the latest PowerShellGet module is installed.
2. Start a PowerShell window as Administrator and run the command:

```
Install-Module -Name Kelverion.Azure.DevOps -Scope AllUsers
```

Upload the integration module to an Azure Automation account:

1. Go to the PowerShell Gallery.
2. Click the **Azure Automation** tab.
3. Click **Deploy to Azure Automation**. You will be directed to Microsoft Azure.
4. Select the **Azure Automation account** that you want to deploy the module to.
5. Click **OK**.

Manual Installation

Alternatively, you can download the integration module package from Kelverion and deploy it manually to your local computer, hybrid workers and Automation Accounts.

The download package from Kelverion includes a **.zip** file containing the integration module as well as the User Guide and Release Notes. The following instructions assume that you have unzipped the download package and have access to the **.zip** file containing the integration module.

Install the integration module on your local computer or hybrid worker:

1. Copy the **Kelverion.Azure.DevOps.zip** file to your local computer.
2. Right-click on the file and select **Properties**.

3. Click the **General** tab. If necessary, click **Unblock**, and then click **OK**.
4. Unzip the **Kelverion.Azure.DevOps.zip** file.
5. Copy the **Kelverion.Azure.DevOps** folder to a location in the `%PsModulePath%` path.

Important: When installing the integration module on a Hybrid Worker, you must use a location that is accessible to all users of the computer.

Upload the integration module to an Azure Automation account:

1. Sign into Microsoft Azure.
2. Open the Automation Account that you want to upload the module to.
3. Click **Modules** under Shared Resources. The list of installed modules is displayed.
4. Click **Add a module** at the top of the list.
5. In the **Upload File** box, select the **Kelverion.Azure.DevOps.zip** file that you downloaded.
6. Click **OK**. Importing the module may take several minutes.

Licensing the Integration Module

Licenses for Kelverion integration modules are managed and deployed using the **Kelverion Runbook Studio** and Azure Automation connection assets.

Register an integration module license with Runbook Studio:

1. Open **Kelverion Runbook Studio**.
2. On the **File** tab, click **About**.
3. Click **License Information**.
4. Click the **Integration Modules** tab, and then click **Add License.**
5. Select the integration module license file (.kaml) and click **Open**.
6. You should see your entitlements displayed in the list.
7. Click **OK**.

Important: Entitlements will not display until after the integration module has been installed on the Runbook Studio computer.

Create an Azure Automation connection asset with a license key and upload it to Azure:

1. On the **Home** tab, click **Sign In**. The Sign In dialog appears.
2. Sign into your account.
3. In the **Active Azure Automation Account** box, select the account that you want to add the connection asset to.

4. Click **New Asset** and then click **Connection**. The New Connection dialog appears.
5. In the **Name** field, enter a name to identify the connection.
6. In the **Connection Type** field, select the desired connection type.
7. Enter the appropriate connection information in the provided fields.
8. Click **OK**.

Update all connection asset license keys and upload them to an Azure Automation account:

1. On the **Home** tab, click **Sign In**. The Sign In dialog appears.
2. Sign into your account.
3. In the Explorer panel, click the **Azure (Online)** group.
4. Right-click the Azure Automation account that contains the connection assets you want to update, and then click **Update License Keys**. A summary is displayed.

Change History

1.2 (Current)

- Added support for Azure DevOps 2020

1.1

- Improved the error message that is given when invalid licenses are used.
- Added new **ID Parameter Set** to the **Get-DevOpsWorkItem** cmdlet that lets you retrieve work items by their unique identifier.
- Renamed the **Wiql** parameter in the **Get-DevOpsWorkItem** cmdlet to **Query**.
- In the data returned from the **Get-DevOpsAttachmentInfo** cmdlet, the **Guid** property was renamed to **AttachmentId** to make it more suitable for pipelining.
- In the data returned from the **Get-DevOpsAttachmentInfo** cmdlet, the **ResourceCreatedDate** property was renamed to **CreatedDate**.
- In the data returned from the **Get-DevOpsAttachmentInfo** cmdlet, the **ResourceModifiedDate** property was renamed to **ModifiedDate**.
- In the data returned from the **Get-DevOpsAttachmentInfo** cmdlet, the **ResourceSize** property was renamed to **ContentLength**.
- In the data returned from the **Get-DevOpsAttachmentInfo** cmdlet the **Id**, **AuthorizedDate** and **RevisedDate** properties were removed.
- In the output returned from the **Get-DevOpsAttachmentInfo** cmdlet, renamed **Delete-DevOpsWorkItem** to **RemoveDevOpsWorkItem**

- Added **ID** alias to **WorkItemId** parameters to make them more suitable for pipelining.
- Made the **Connection** parameter in each cmdlet a positional parameter with position 0.
- Made all parameters, other than the Connection parameter, named parameters.