John Yoakum
RECAST SOFTWARE

1/3/2025

Application Workspace

Packaging Utility

Contents

Date	Who	Comment
01/03/2025	John	Initial document

Application Workspace Packaging Utility

While working on a POC for Application Workspace recently, the customer asked me some questions about potential onboarding and what that would look like and how can we quickly come online. A couple questions they asked were:

- How can I find out what all software is installed in my environment?
- How can I quickly enable applications so that I can roll them out to my users?

Well, this process that I am going to describe can help with that.

While trying to figure out how to accomplish this goal, a couple of questions came to mind.

- What is the best way to inventory software on devices?
- Where to store that information?
- How am I going to report on that information?
- How can I take that information and assist in creating those applications that I could easily in Application Workspace?

From those, this process came about. It is designed in two parts.

I have a PowerShell script that can be run as a remediation item from Intune that inventories all device-based and user-based installed software based on what is stored in the registry. This script will then create and store that information in a Log Analytics table. I then created a workbook tied to that table that provides some reporting on what is in your environment.

I then wrote a second PowerShell script that reaches out to that Log Analytics table and brings in all the installed software and compares it to your connector in Application Workspace to determine all the possible matches in the environment. Included in that list is if you have already enabled that application. From that list, you can select the packages you wish to create, and the script will create those packages for you, all from the one form. Now, there is one downside… I don't have a way to bring in dependencies at this time, so you may have to manually bring those in after the fact.

This article will guide you through all the setup that is required to make this process work efficiently.

This article and all the files needed can be found on my Github.

Log Analytics Setup

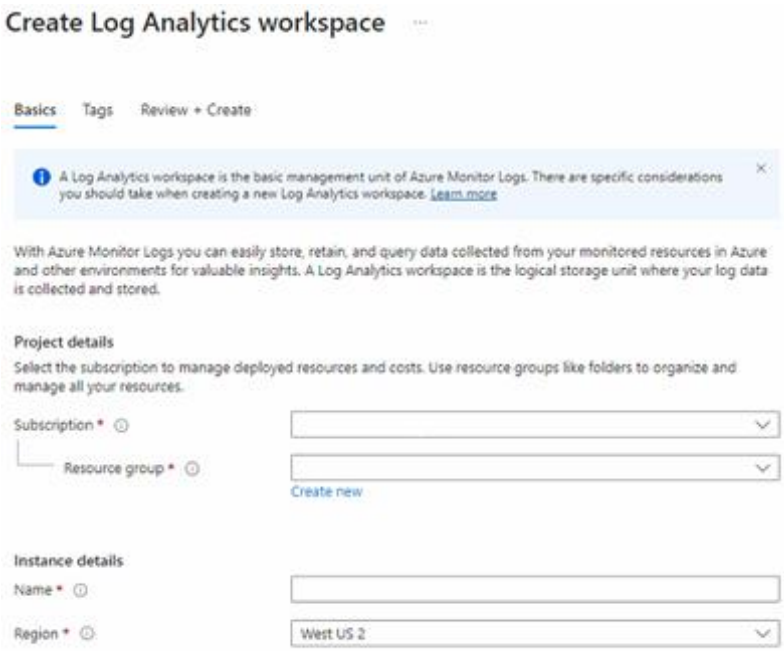
Log Analytics Workspace Setup

We will first set up the Log Analytics Workspace and App Registration to support the solution

1. Create the Log Analytics Workspace
    a. Log into the Entra Portal at https://portal.azure.com
    b. Navigate to the section for Log Analytics

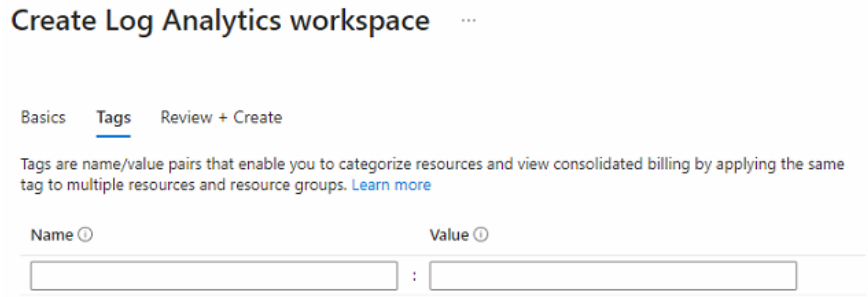
    c. Once here, you have two options:
        i. Use an existing Workspace
            1. If using an existing Workspace, then go to step 1.e
        ii. Create a new Workspace
            1. If creating new, continue to Step C
    d. Click Create
        i. Click Create

        ii. Enter or Choose the Basic Details



        iii. Click Next : Tags >

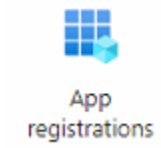
iv. Enter in any tags you want to apply



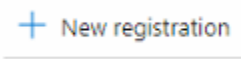
v. Click Review + Create

e. After the Log Analytics Workspace is created, copy and store the connection information
   i. Navigate to the Agents section and expand the Log Analytics agent Instructions
   ii. Copy the following values and save for later
      1. Workspace ID
      2. Primary Key
      3. Secondary Key

2. Create the App Registration so that you can gain API Access
   a. Navigate Home in Entra and then navigate to the section for App Registrations

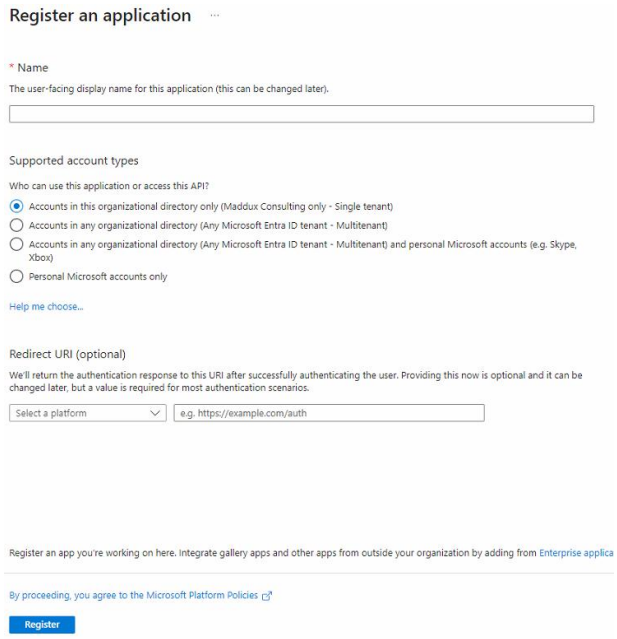


   b. Create a new App Registration, Click New Registration

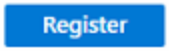


   c. Enter the details for the App Registration

i. Name is the only thing required

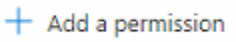


d. Click Register



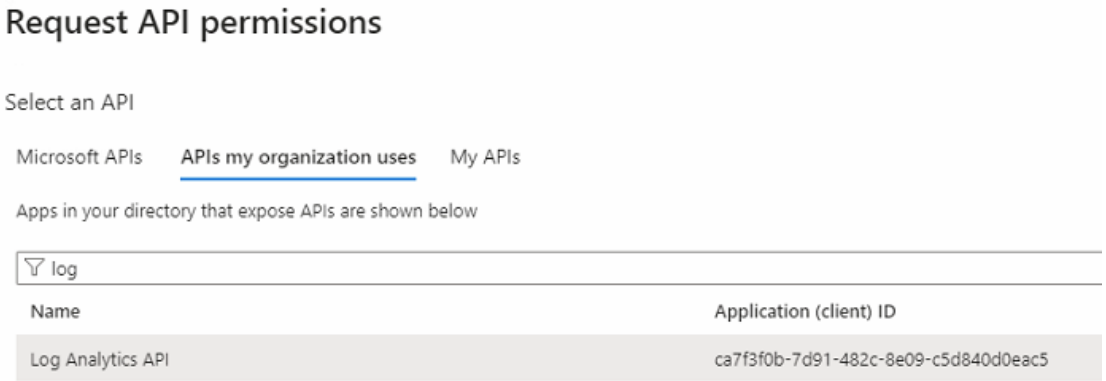
e. Once Created, we need to assign some permissions

i. Navigate to API Permissions and click Add a Permission



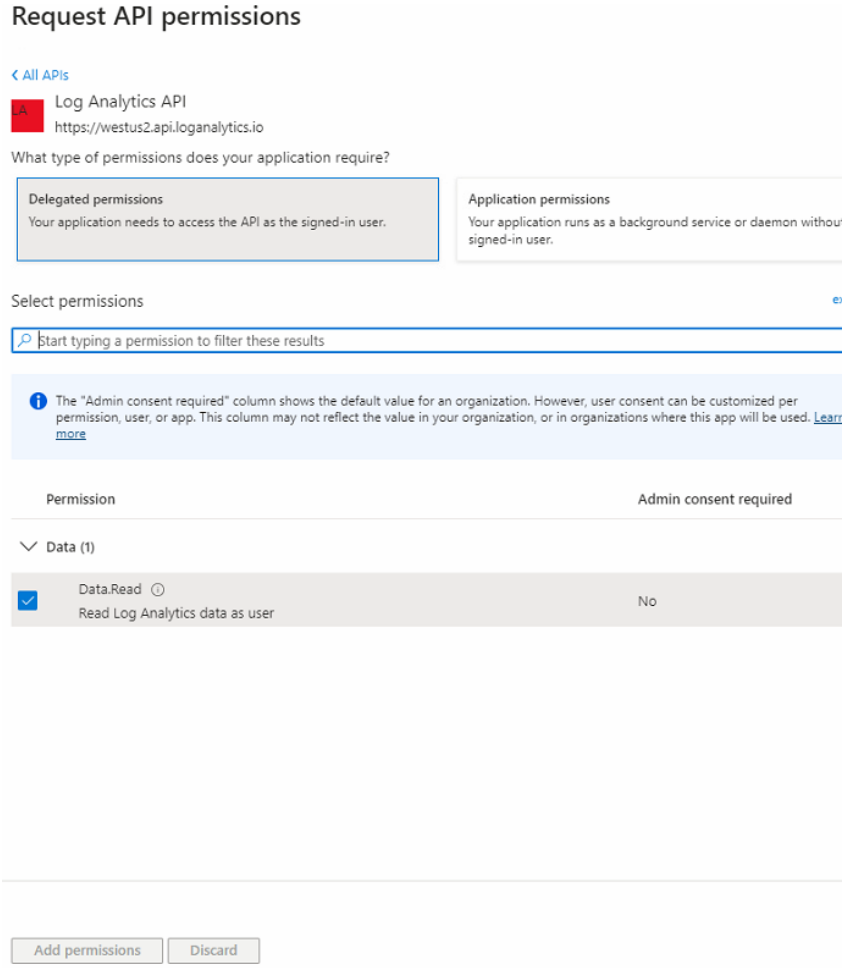
ii. Choose APIs my organization uses at the top

iii. Choose Log Analytics API



iv. Choose Delegated permissions

v. Only one is available, Data.Read, make sure that it is selected and then Choose Add Permission



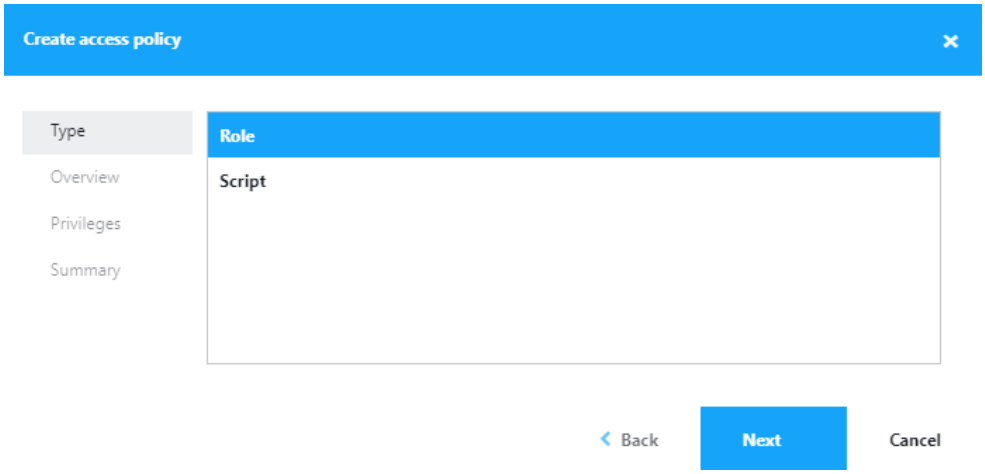
vi. Click on Certificates & Secrets
vii. Create a New client secret and name it and give it a time span for validity
viii. Copy the Value for the Secret Key and store it for later
ix. Navigate to the Overview section and copy the following values as well for later
    1. Application ID (Client ID)
    2. Tenant ID

Application Workspace Configuration

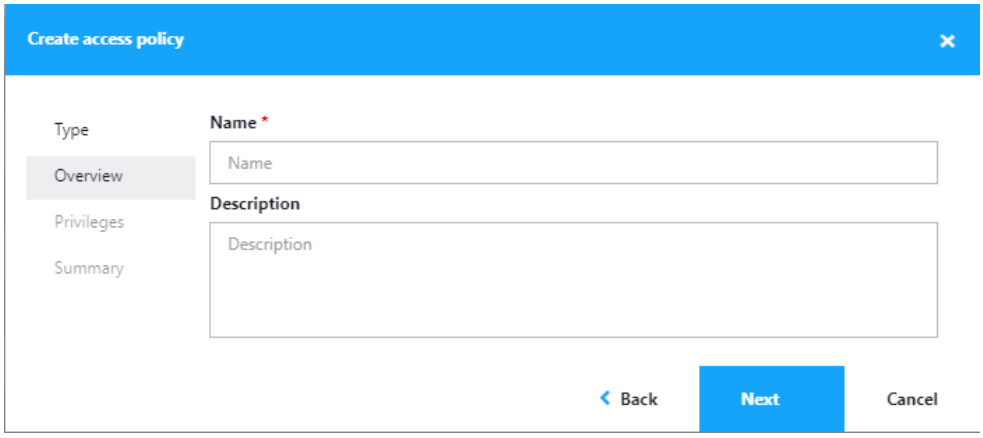
Now we will configure your Application Workspace environment to support this solution. This guide assumes that you already have a connector to the Liquit Setup Store already configured and you have a prefix already assigned

1. Connector Setup
    a. We don't need to do anything here except save the connector prefix you are using for later
2. Account Setup
    In order for the solution to work correctly, we must have a "service" account setup in Application Workspace with limited permissions.
    a. Create the Access Policy for script-based access
        i. Navigate to Access Policies
        ii. Click Create

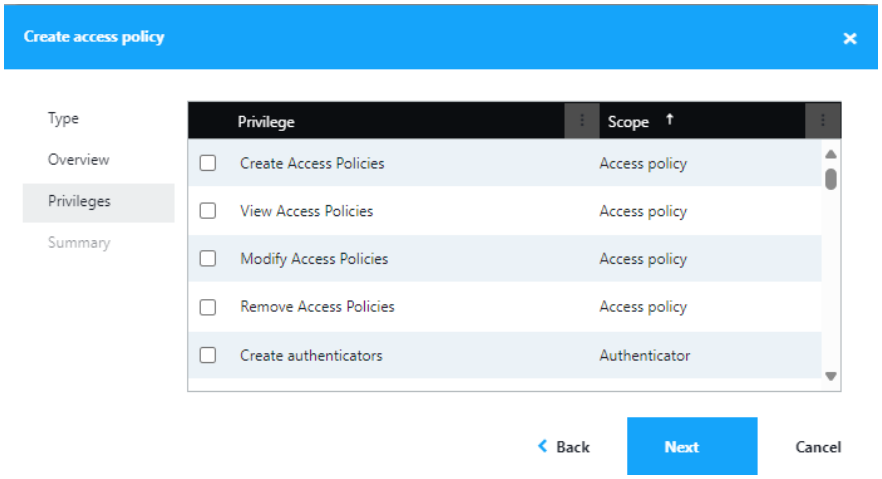
iii. In the Create Access Policy box, choose Role as the type and click Next



iv. Enter a Name and Description for the Role and Click Next



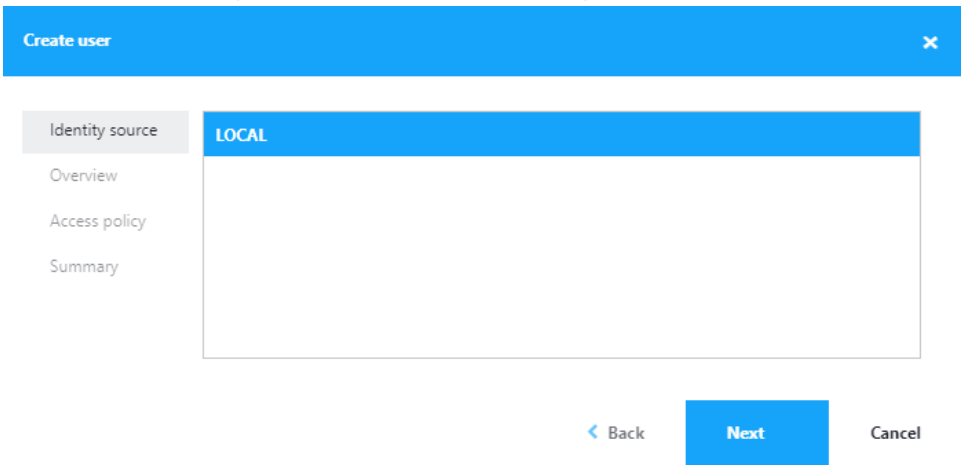
v. Choose the permissions you want for this role. The following permissions should be all you need. This should limit that account to only being able to run the commands through the PowerShell.



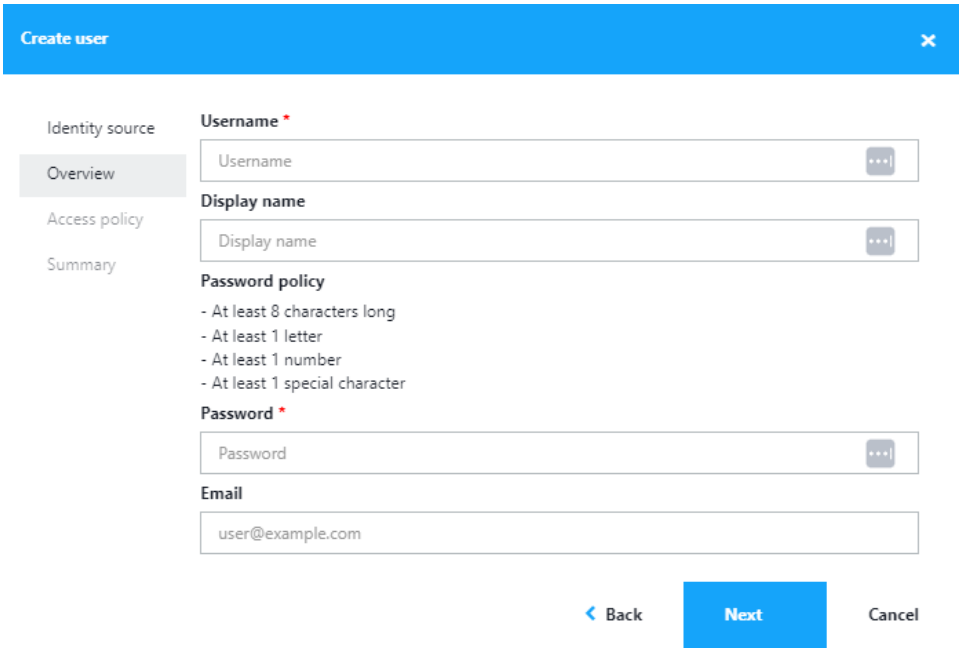
1. API Access
2. View Connectors
3. Create Packages
4. View Packages
5. Modify Packages

6. Remove Packages
7. View Resources

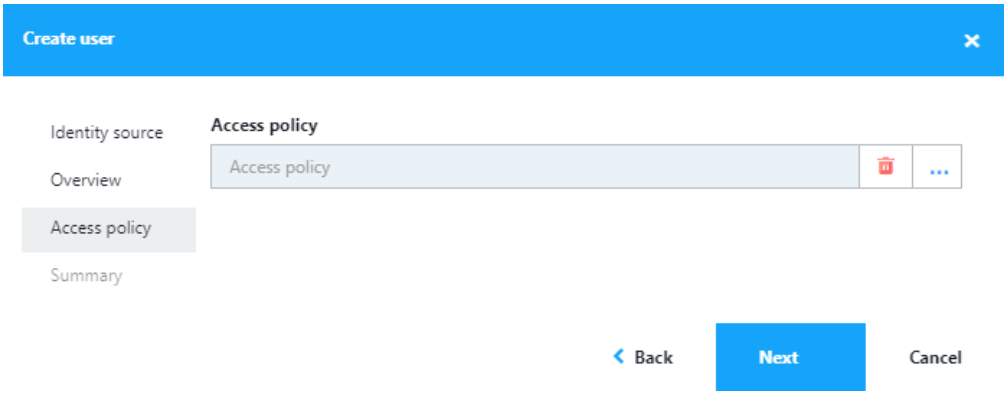
b. Create the service account in your Application Workspace environment
   i. Navigate to Users
      1. Click Create
      2. Choose the Identity Source, LOCAL should be perfect and click Next



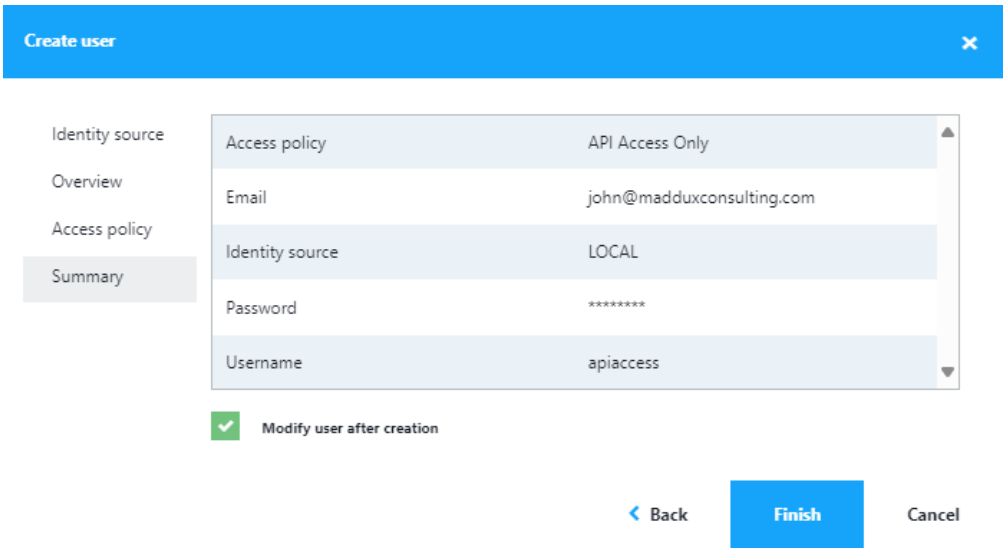
      3. Give them a Username, Display Name, Password and an Email and click Next.



4. Assign the Access Policy you created in the previous step by click the triple ellipsis next to the box and selecting the policy and then Clicking Next



5. Review the Settings and click Finish



Software Inventory Script Setup

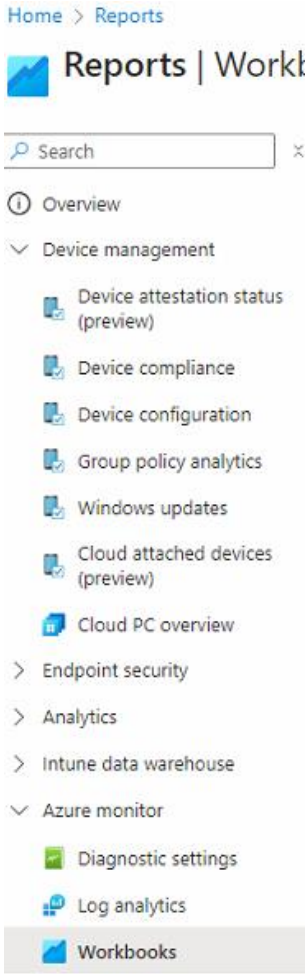
In this section we will modify the script with your values so that it can populate the Log Analytics table with the installed software on a client machine. I'm only going to cover modifying the script, not uploading and delivering it to your client computers.

1. Download the script from Github (Get-InstalledApplications.ps1)
2. In the following lines, enter in the values we copied from above instructions
    a. Line 230 – ClientID
    b. Line 231 – ClientSecret Value
    c. Line 232 – TenantID
    d. Line 233 – WorkspaceID
    e. Line 234 – SharedKey (Primary Key)
3. Deploy to Devices

Azure Workbook Setup

For this section, we will be creating a workbook for reporting on the data that is collected in the previous section. There isn't a lot to change here, just create the workbook and copy the json. You can access the json here.

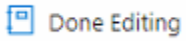
1. Log into Intune
2. Expand Azure Monitor and select Workbooks



3. On the right-hand side, choose New

4. In the editor, select the button that allows you to change the code

5. Copy the Json from the Workbook in Github and replace the code here and click Apply
6. Click the Done Editing Button



7. Click the Save Button and give it a name

8. If desired, you can add this workbook to your dashboard for easy access

Application Workspace Package Utility

For this section, we will be modifying the script that will assist in creating packages in your Application Workspace environment based on the Installed Applications collected from previous steps

1. Download the script from Github (Create-AWApplications.ps1)
2. Modify the following lines with the data that we set aside from previous steps
    a. Line 46 – ClientID (Application ID)
    b. Line 47 – ClientSecret Value
    c. Line 48 – TenantID
    d. Line 49 – WorkspaceID
    e. Line 50 – LiquitConnectorPrefix – this is that prefix for the connector that we want to pull resources from
    f. Line 51 – LiquitURI – your Application Workspace zone
    g. Line 52 – username – your API Username for your Application Workspace zone
    h. Line 53 – password – you API password for the api user you created
    i. Line 54 – ClosingTime – the number of seconds you want to wait after completion of the script before closing out

Now you can run this script and start bringing in new applications into your Application Workspace Environment