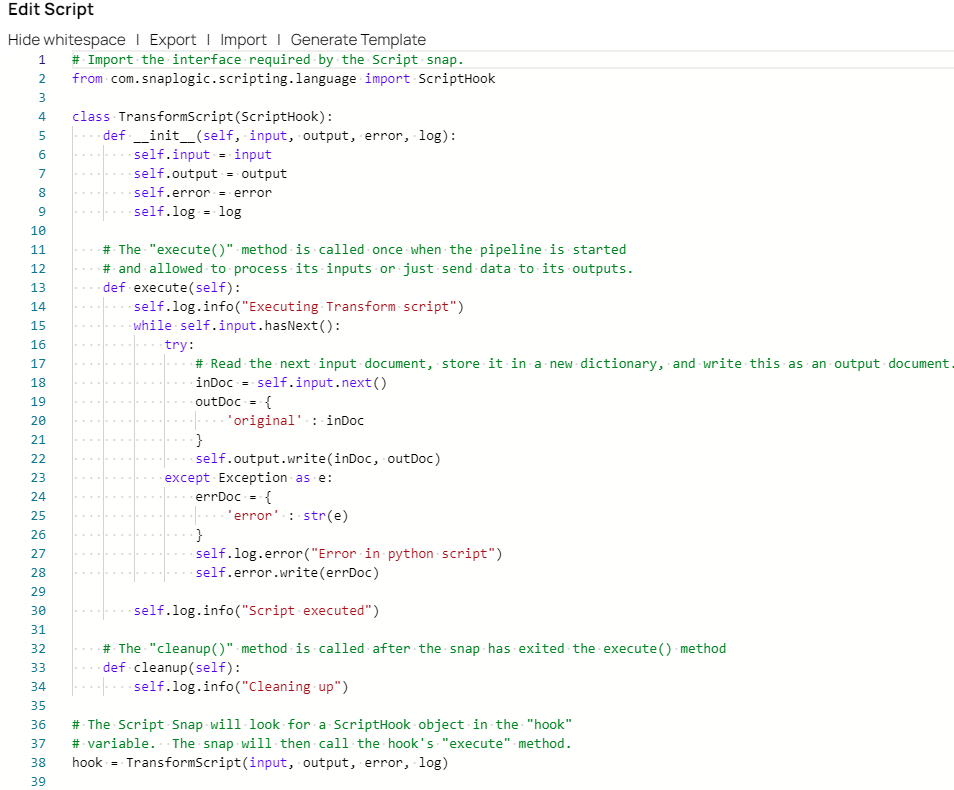
SCRIPT SNAP

This script snap allows ruby, java script & python language to execute.

When it comes to **python**, there is a template where we write code and execute

The picture below is boilerplate template of python code



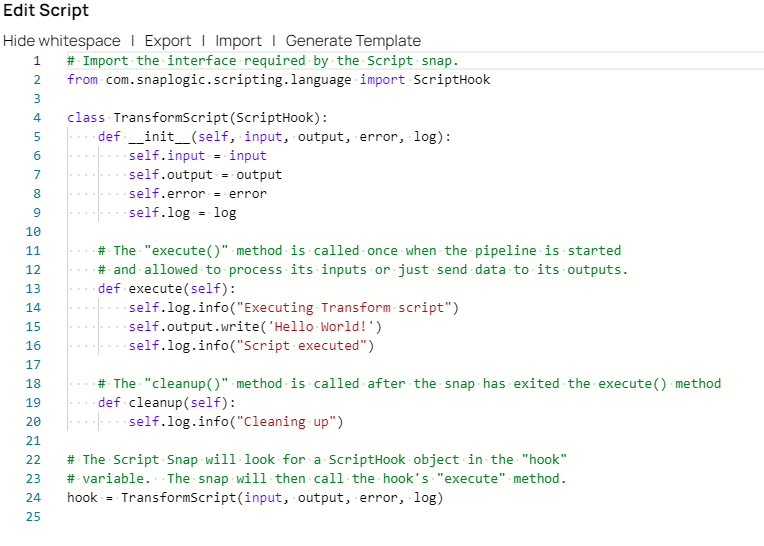
**ScriptHook** is an interface provided by SnapLogic to enable scripts to interact with the data flow in a pipeline. It provides script access to the input, output, error, and log objects.

The key things to know about scriptHook are:

* It is a global variable that must be declared in the script. The name is case-sensitive.
* It allows the script to access the following objects:
  + input - provides access to read input documents
  + output - allows writing output documents
  + error - allows writing error documents
  + log - provides access to logging
* The script defines a class that implements ScriptHook. An instance of this class is assigned to the global hook variable.
* The Script snap will call the **execute()** method on the hook instance during pipeline execution. This is where the **main script logic** goes.
* The cleanup() method can be used for any cleanup tasks after execution.

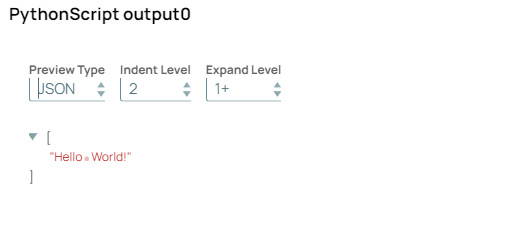
So in summary, scriptHook enables interaction with the pipeline data flow and logging from within a script snap. It is responsible to execute the script

Executing script snap - Hello world program

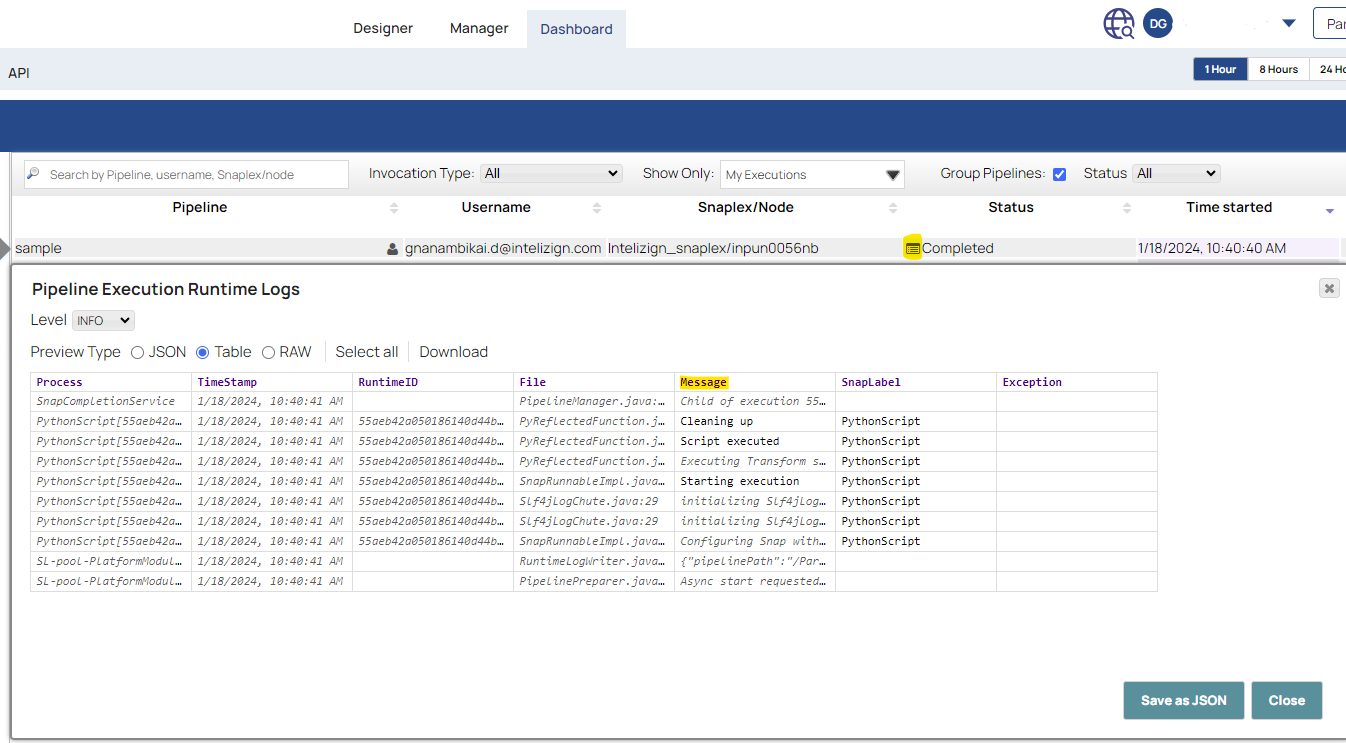


Whatever the code we implement should be inside the **def execute(self)** method

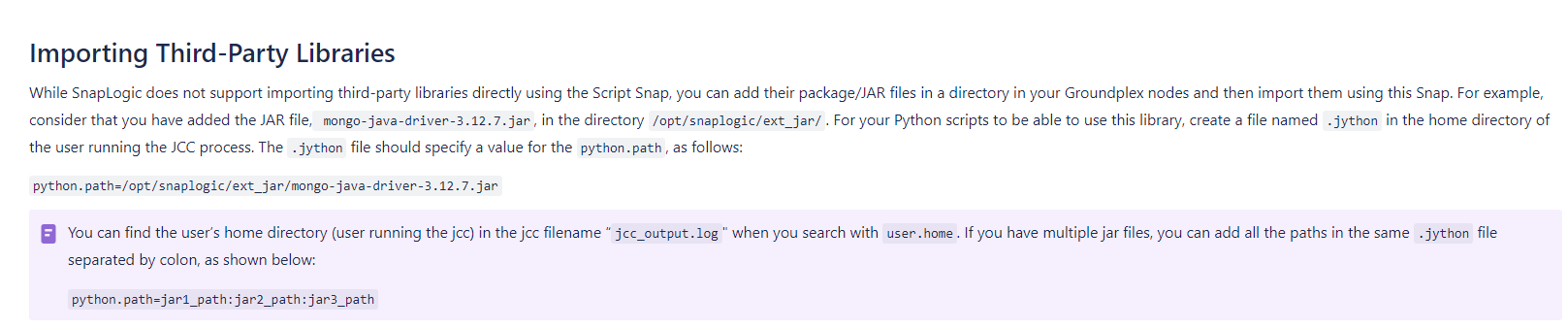
Self.output.write() is the output statement that can be previewed after validating the pipeline



We can also check logs that we written in the script after executing the pipeline



Executing script snap - Using Thirty party libraries

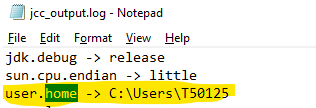


When it comes to third party libraries. We need ground plex and to add jars there. In script snap, we use **ONLY JAVA JARS** & write code in terms of language we select (python code Syntax) to execute the pipeline.

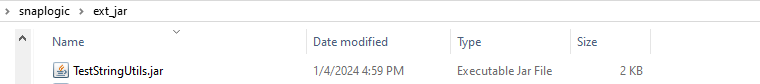
Step – 1: Add folder in the name of ext\_jar in your groundplex **/opt/snaplogic/ext\_jar**  ext\_jar is the place where to add external libraries/jars.

Step –2 : Add .jython file in the user node of Jcc. Add the path of your library in the jython file

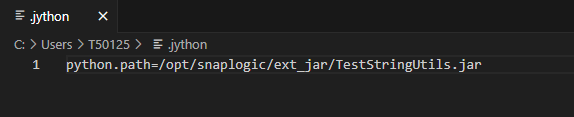
You can find the user node in **/opt/snaplogic/run/log/jcc\_output.log**



TestStringUtils.jar is my custom jar to convert string into lowercase



Add this path in the jython file



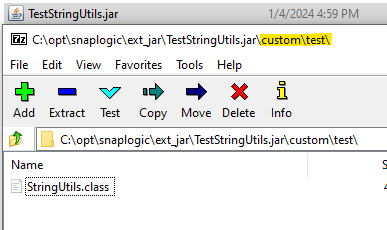
Start writing the code in the snap

To import library in the script

import sys

sys.path.append('/opt/snaplogic/ext\_jar/TestStringUtils.jar')

from custom.test import StringUtils



Import must be in the order how our class located in the jar

Once all these are done, start validating the pipeline



Reference: <https://docs-snaplogic.atlassian.net/wiki/spaces/SD/pages/1439321/Script>