Platform Administration Reference guide

Introduction

This document is a reference manual for common administrative and management tasks on the SnapLogic platform.

This document is for SnapLogic *Environment Administrators* (Org Administrators) and users involved in supporting or managing the platform components.

Authors: SnapLogic Enterprise Architecture team

Environment Administrator (*known as Org Admin in the Classic Manager*) permissions

There are two reserved groups in SnapLogic:

admins: Users in this group have full access to all projects in the Org.

members: Users in this group have access to projects that they create, or to which they are granted access. Users are automatically added to this group when you create them, and they must be a part of the members group to have any privileges within that Org.

There are two user roles:

Environment admins are Org users who can manage the Org. Environment admins are part of the *admins* group, and this role is named “Org Admin” in the classic Manager.

Basic user. All non-admin users. Within an Org, basic users can create projects and work with assets in the Project spaces to which they have been granted permission. To gain Org administrator privileges, a *Basic* user can be added to the *admins* group.

The below table lists the various tasks under the different categories that an ***Environment admin*** user can perform:

Task	Comments
USER MANAGEMENT	
• Create and delete users. • Update user profiles. • Create and delete groups. • Add users to a group. • Configure password expiration policies. • Enable users’ access to applications (AutoSync, IIP)	When a user is removed from an Org, the administrator that removes the user becomes the owner of that user's assets. Reference: User Management
MANAGER	

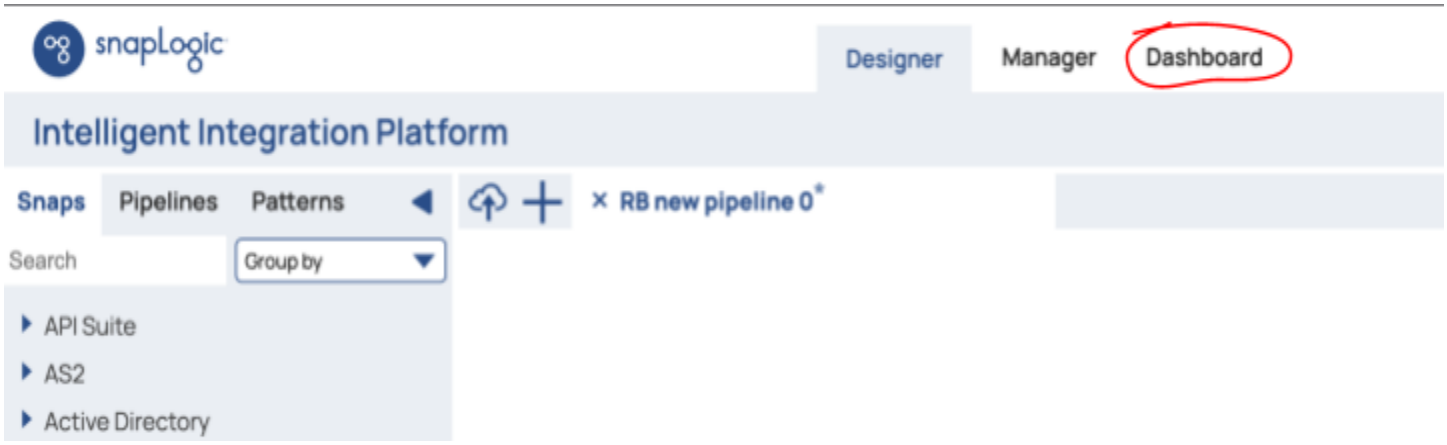
• Create and manage Project Spaces. • Update permissions (R, W, X) on an individual Project space and projects. • Delete a Project space. • Restore Project spaces, projects, and assets from the Recycle bin. • Permanently delete Project spaces, projects, and assets from the Recycle bin. • Configure Git integration and integration with tools such as Azure Repos, GitLab, and GHES. • View Account Statistics, and generate reports for accounts, projects, and pipelines within the project that use an account. • Upgrade/downgrade Snap Pack versions.	
ALERTS and NOTIFICATIONS	
• Set up alerts and notifications. • Set up Slack channels and recipients for notifications.	Reference: Alerts
SNAPLEX and ORG	
• Create Groundplexes. • Manage Snaplex versions. • Update Snaplex settings. • Enable feature flags on the org. An example for setting the feature flag: Alerts	Update or revert a Snaplex version.
APIM	
• Publish, unpublish, and deprecate APIs on the Developer portal. • Approve API subscriptions and manage/approve user accounts.	Reference: API Management
AutoSync	

• Configure AutoSync user permissions. • Configure connections for data pipeline endpoints. • Create user groups to share connection configuration. • View information on all data pipelines in the Org.	Reference: AutoSync Administration

Table 1.0 Org admin tasks

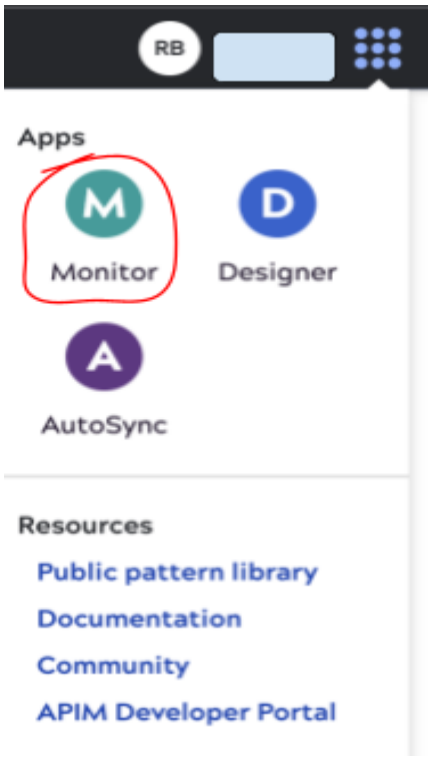
SnapLogic Monitoring Dashboards

The classic IIP monitoring Dashboard interface can be launched from the *Dashboard* tab after logging on to SnapLogic IIP. The Dashboard has multiple tabs that displays information about various Snaplogic assets.



The new, enhanced *Monitor* interface can be launched from the Apps (Waffle) menu located on the top right corner of the page.

Note: *Presently, only some of the features of the classic Dashboard are available in the new Monitor App.*



The following table lists some common administration and monitoring tasks (grouped by category) for both the classic Dashboard and the Monitor App.

Task	Classic Dashboard	Monitor App
ORG		
• View statistical data for all assets in an Org for a specific time period. (Example: Number of Pipeline and task executions, Snap executions, documents processed, etc.)	Dashboard -> Insights Wall *Note that the Insights tab is only accessible to Org admins.*	N/A
PIPELINES		

• View Pipeline run history, run status, duration, and documents processed.	Dashboard -> Health wall -> Pipeline Health wall	Monitor -> Execution tab -> Execution Overview -> Pipeline
• View and download Pipeline execution runtime logs.	Dashboard -> Health wall -> Pipeline Health wall Dashboard -> Pipeline wall	Monitor -> Execution tab -> Execution Overview -> Pipeline
• Stop currently executing or suspended Pipelines. • Resume suspended Pipelines.	Dashboard -> Pipeline wall	N/A
• Insights about Pipeline counts and Pipeline executions executed in the Org for a specific time period.	Dashboard -> Insights wall *Note that the Insights wall is only accessible to Org admins.*	N/A
TASKS		
• View the execution status of Scheduled, Triggered, and Ultra Tasks for a specific time period.	Dashboard -> Task	Monitor -> Execution tab -> Execution Overview -> Task
• View Task execution metrics and average execution time for ipelines in the tasks. • View interactive graphs for the Task execution statistics.	Dashboard -> Task	N/A
SNAPLEX		
• Monitor resource utilization over a time period for Snaplex nodes. • View Snaplex metric charts and data points. • Interactive widgets to correlate Pipelines with data spikes.	Dashboard -> Snaplex	Monitor -> Infrastructure -> <node> -> Metrics *and* Monitor -> <node> -> Metrics

• View any alerts sent by the Snaplex.	Dashboard -> Health wall	Monitor -> Infrastructure -> <node> -> Alerts
• View node configurations and download node logs.	Dashboard -> Health wall -> Snaplex -> <node>	Monitor -> Infrastructure -> <node>
API		
• View and download data for various metrics such as consumption, errors, latency, etc. for the published APIs.	Dashboard -> API Insights wall	Monitor -> Execution -> *Filter by Invocation Type = API calls* *The Monitor only allows to view the API execution details for a specific time period.*

Table 2.0 Monitoring Dashboard features

Additional reference:

Insights

Health

Pipeline
Snaplex

API
Monitor Interface

Java Component Container (jcc) command line utility (for Groundplexes)

The jcc script is a command-line tool that provides a set of commands to manage the Snaplex nodes. This utility is installed in the /opt/snaplogic/bin directory of the Groundplex node. The below table lists the commonly used arguments for the jcc script (jcc.sh on Linux and jcc.bat on Windows). Note that the command would list other arguments (for example, try-restart). However, those are mainly included for backward compatibility and not frequently used.

$SNAPLOGIC refers to the /opt/snaplogic directory on Linux or the <Windows drive>:\opt\snaplogic directory on Windows servers. Run these commands as the root user on Linux and as an Administrator on Windows.

Example:

sudo /opt/snaplogic/bin/jcc.sh restart or c:\snaplogic\bin\jcc.bat restart

Argument	Description	Comments
`status`	Returns the Snaplex status. The response string would indicate if the Snaplex Java process is running.	
`start`	Starts the Snaplex process on the node.	
`stop`	Stops the Snaplex process on the node.	
`restart`	Stops and restarts the Snaplex process on the node.	Restarts both the *monitor and the Snaplex processes.*
`diagnostic`	Generates the diagnostic report for the Snaplex node. The HTML output file is generated in the $SNAPLOGIC/run/log directory.	Resolve any warnings from the report to ensure normal operations.
`clearcache`	Clears the cache files from the node. This command must be executed when the JCC is stopped.	
`addDataKey`	Generates a new key pair and appends it to the keystore in the /etc/snaplogic folder with the specified alias. This command is used to rotate the private keys for *Enhanced Account Encryption.*	Doc reference: Enhanced Account Encryption
The following options are available for a Groundplex on Windows server.		
install_service *remove_service*	The jcc.bat install_service command installs the Snaplex as a Windows service. The jcc.bat remove_service command removes the installed Windows service.	Run these commands as an Administrator user.

Table 3.0 jcc script arguments

Example of custom log configuration for a Snaplex node (Groundplex)

Custom log file configuration is occasionally required due to internal logging specifications or to troubleshoot problems with specific Snaps.

In the following example, we illustrate the steps to configure the log level of 'Debug' for the Azure SQL Snap pack. The log level can be customized for each node of the Groundplex where the related pipelines are executed, and will be effective for all pipelines that use any of the Azure SQL Snaps (for example, *Azure SQL - Execute, Azure SQL - Update*, etc.). *Note that Debug logging can affect pipeline performance so this configuration must only be used for debugging purposes.*

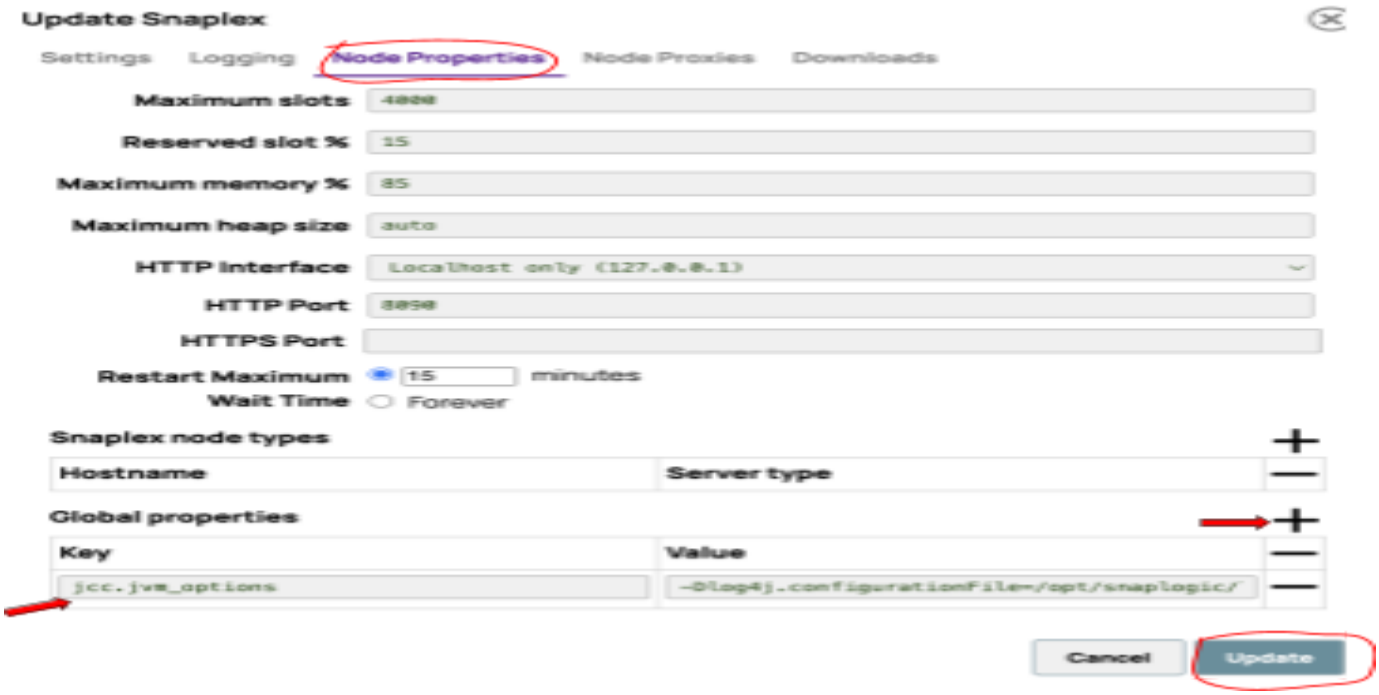
Configuration Steps

a) Follow steps 1 and 2 from this document:

Custom log configuration

Note: You can perform Step 2 by adding the property key and value under the *Global Properties* section. Example:

Key: *jcc.jvm_options*
Value: *-Dlog4j.configurationFile=/opt/snaplogic/logconfig/log4j2-jcc.xml*



The Snaplex node must be restarted for the change to take effect. Refer to the commands in *Table 3.0*.

b) Edit the *log4j2-jcc.xml* file configured in Step a.

c) Add a new `RollingRandomAccessFile` element under <Appenders>. In this example, the element is referenced with a unique name *JCC_AZURE.* It also has a log size and rollover policy defined. The policy would enable generation of up to 10 log files of 1 MB each. These values can be adjusted depending on your requirements.

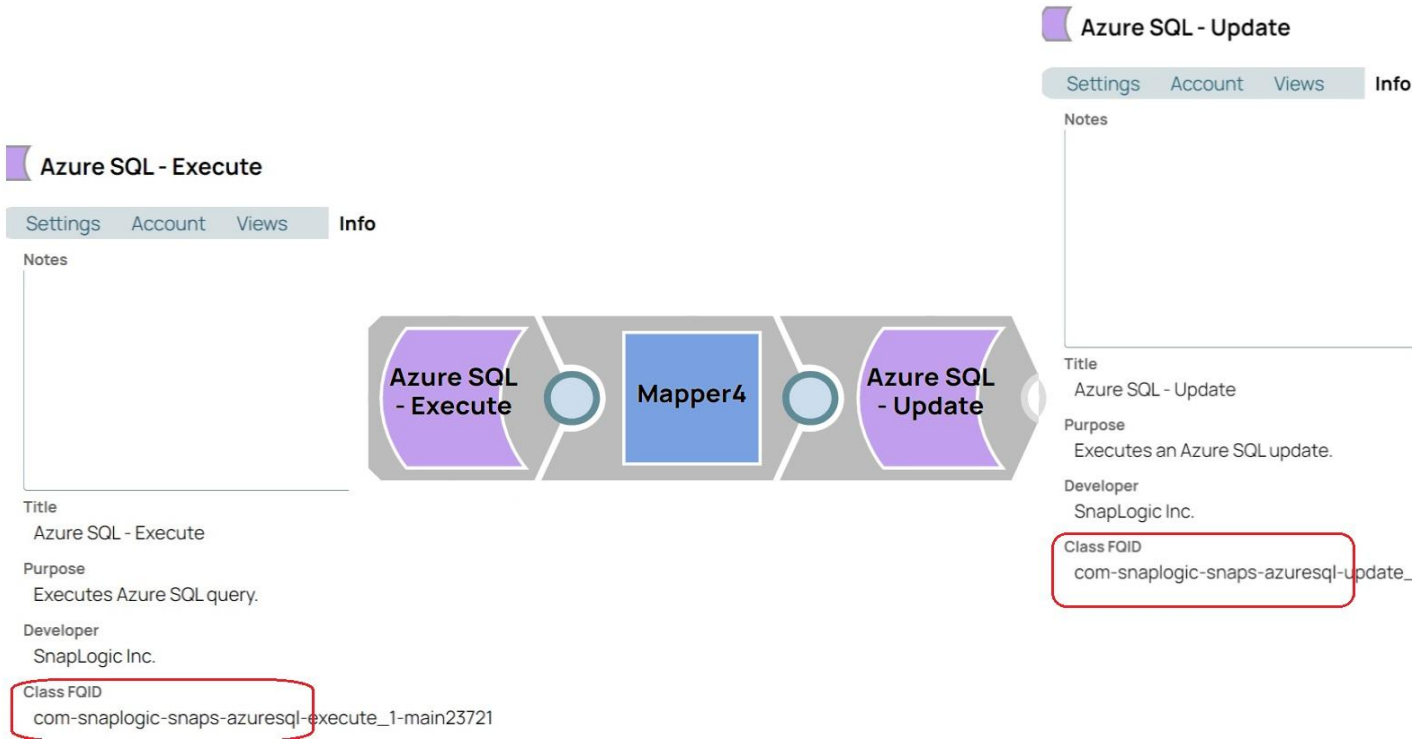
```
<RollingRandomAccessFile name="JCC_AZURE"

fileName="${env:SL_ROOT}/run/log/${sys:log.file_prefix}jcc_azure.json"
immediateFlush="true" append="true"

filePattern="${env:SL_ROOT}/run/log/jcc_azure-log-%d{yyyy-MM-dd-HH-mm}.json”
ignoreExceptions="false">

<JsonLogLayout properties="true"/>

<Policies>
            <SizeBasedTriggeringPolicy size="1 MB"/>
</Policies>

            <DefaultRolloverStrategy max="10"/>
</RollingRandomAccessFile>

…

…

</Appenders>
```

d) The next step is to configure a Logger that references the *Appender* defined in step #c. This is done by adding a new <Logger> element. In this example, the Logger is defined with log level = *Debug*.

```
 <Logger name="com.snaplogic.snaps.azuresql" level="debug"
includeLocation="true" additivity="false">

            <AppenderRef ref="JCC_AZURE" />

 </Logger>

..

..

<Root>
…
```

```
 </Root
</Loggers>
</Configuration>
```

The value for the name attribute is derived from the Class FQID value of the associated Snap.



The changes to log4j2-jcc.xml are marked by the highlighted text in steps c and d. The complete XML file is also attached for reference. You can refer to the Log4j documentation for more details on the attributes or for additional customization.

Log4j reference

Debug log messages and log files

Additional debug log messages will be printed to the pipeline execution logs for any pipeline with Azure SQL Snaps. These logs can be retrieved from Dashboard.

Example:

{"ts": "2023-11-30T20:21:33.490Z", "lvl": "DEBUG", "fi": "JdbcDataSourceRegistryImpl.java:369", "msg": "JDBC URL: jdbc:sqlserver://sltapdb.database.windows.net:1433;database=SL.TAP;encrypt=true;trustServerCertificate=false;hostNam

eInCertificate=.database.windows.net;authentication=sqlPassword;loginTimeout=30;connectRetryCount=3;connectRetryInterval=5;applicationName=SnapLogic (main23721) - pid-113e3955-1969-4541-9c9c-e3e0c897cccd, database server: Microsoft SQL Server(12.00.2531), driver: Microsoft JDBC Driver 11.2 for SQL Server(11.2.0.0)", "snlb": "Azure+SQL+-+Update", "snrd": "5c06e157-81c7-497f-babb-edc7274fa4f6", "plrd": "5410a1bdc8c71346894494a2_f319696c-6053-46af-9251-b50a8a874ff9", "prc": "Azure SQL -*

The updated log configuration would also write the custom JCC logs (for all pipelines that have executed the Azure SQL Snaps) to disk under the /opt/snaplogic/run/log directory. The file size for each log file and the number of files would depend on the configuration in the *log4j2-jcc.xml* file.

```
/opt/snaplogic/run/log$ ls -rlt

r snapuser        0 Nov 30 18:26 metrics_logs.json
r snapuser   130548 Nov 30 20:06 monitor_error.json
r snapuser        0 Nov 30 20:07 jcc_azure.json
r snapuser    26051 Nov 30 20:19 monitor.log
r snapuser     1726 Nov 30 20:19 jcc_state_logs.json
r snapuser       59 Nov 30 20:19 nodeState.json
r snapuser    45056 Nov 30 20:21 threadDump.tdp
r snapuser     3795 Nov 30 20:22 runtime-log.json
r snapuser   237492 Nov 30 20:37 jcc_error.json
r snapuser  2034659 Nov 30 21:34 jcc_status.json
r snapuser     4096 Nov 30 21:36 log_service
r snapuser  5264172 Nov 30 21:36 jcc_output.log
r snapuser   100221 Nov 30 21:36 jcc_access.log
r snapuser 26665584 Nov 30 21:36 jcc.json
r snapuser 16555161 Nov 30 21:36 monitor.json
/opt/snaplogic/run/log$
```

The changes to *log4j2-jcc.xml* can be reverted if the additional custom logging is no longer required.

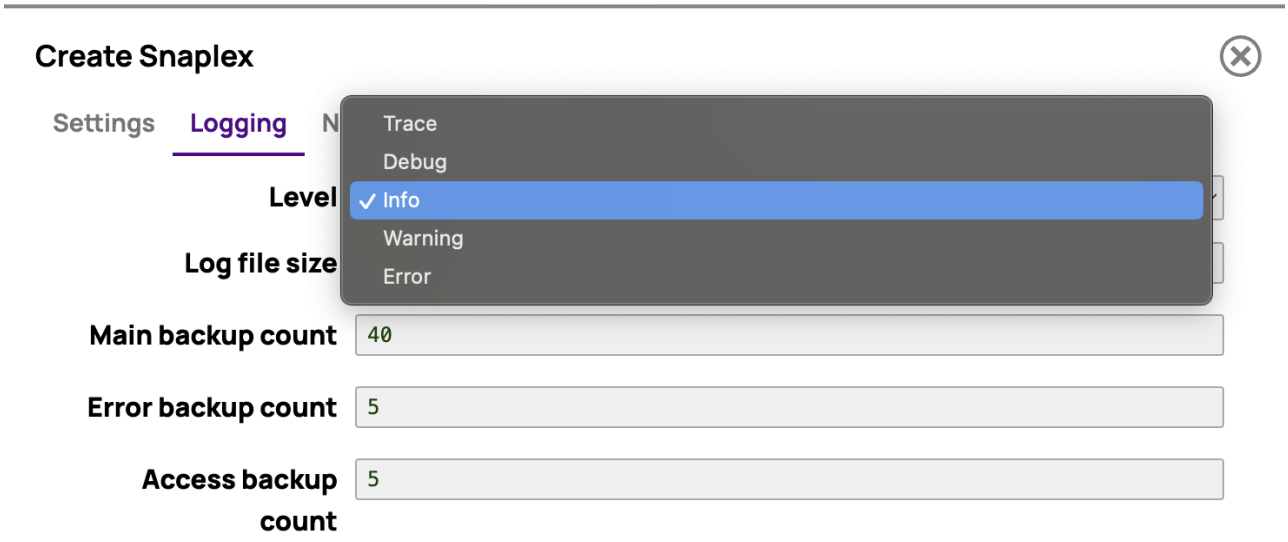
Log level configuration for a Snaplex in Production Orgs

The default log level for a new Snaplex is '*Debug*.' This value can be updated to *'Info'* in Production Orgs as a best practice.

The available values are:

- *Trace*: Records details of all events associated with the Snaplex.
- *Debug*: Records all events associated with the Snaplex.
- *Info*: Records messages that outline the status of the Snaplex and the completed Tasks.
- *Warning*: Records all warning messages associated with the Snaplex.
- *Error*: Records all error messages associated with the Snaplex.

Reference: Snaplex logging


Create Snaplex
Settings
Logging
N
Trace
Debug
Info
Warning
Error
Level
Log file size
Main backup count
40
Error backup count
5
Access backup count
5