


Send Azure Databricks application logs to Azure Monitor

03/26/2019 • 2 minutes to read • Contributors 

In this article

[Prerequisites](#)

[Send application metrics using Dropwizard](#)

[Send application logs using Log4j](#)

[Run the sample application](#)

[Next steps](#)

This article shows how to send application logs and metrics from Azure Databricks to a [Log Analytics workspace](#). It uses the [Azure Databricks Monitoring Library](#), which is available on GitHub.

Prerequisites

Configure your Azure Databricks cluster to use the monitoring library, as described in [Configure Azure Databricks to send metrics to Azure Monitor](#).

⚠ Note

The monitoring library streams Apache Spark level events and Spark Structured Streaming metrics from your jobs to Azure Monitor. You don't need to make any changes to your application code for these events and metrics.

Send application metrics using Dropwizard

Spark uses a configurable metrics system based on the Dropwizard Metrics Library. For more information, see [Metrics](#) in the Spark documentation.

To send application metrics from Azure Databricks application code to Azure Monitor, follow these steps:

1. Build the `spark-listeners-loganalytics-1.0-SNAPSHOT.jar` JAR file as described in [Build the Azure Databricks Monitoring Library](#).
2. Create Dropwizard [gauges or counters](#) in your application code. You can use the `UserMetricsSystem` class defined in the monitoring library. The following example creates a counter named `counter1`.

Scala

 Copy

```
import org.apache.spark.metrics.UserMetricsSystems
import org.apache.spark.sql.SparkSession

object StreamingQueryListenerSampleJob {

  private final val METRICS_NAMESPACE = "samplejob"
  private final val COUNTER_NAME = "counter1"

  def main(args: Array[String]): Unit = {

    val spark = SparkSession
      .builder
```

```
.getOrCreate
```

```
val driverMetricsSystem = UserMetricsSystems
    .getMetricSystem(METRICS_NAMESPACE, builder => {
        builder.registerCounter(COUNTER_NAME)
    })

driverMetricsSystem.counter(COUNTER_NAME).inc(5)
}
```

The monitoring library includes a [sample application](#) that demonstrates how to use the `UserMetricsSystem` class.

Send application logs using Log4j

To send your Azure Databricks application logs to Azure Log Analytics using the [Log4j appender](#) in the library, follow these steps:

1. Build the `spark-listeners-loganalytics-1.0-SNAPSHOT.jar` JAR file as described in [Build the Azure Databricks Monitoring Library](#).
2. Create a `log4j.properties` [configuration file](#) for your application. Include the following configuration properties. Substitute your application package name and log level where indicated:

YAML

 Copy

```
log4j.appender.A1=com.microsoft.pnp.logging.loganalytics.LogAnalyticsAppender
log4j.appender.A1.layout=com.microsoft.pnp.logging.JSONLayout
log4j.appender.A1.layout.LocationInfo=false
log4j.additivity.<your application package name>=false
log4j.logger.<your application package name>=<log level>, A1
```

You can find a sample configuration file [here](#).

3. In your application code, include the `spark-listeners-loganalytics` project, and import `com.microsoft.pnp.logging.Log4jConfiguration` to your application code.

Scala

 Copy

```
import com.microsoft.pnp.logging.Log4jConfiguration
```

4. Configure Log4j using the `log4j.properties` file you created in step 3:

Scala

 Copy

```
getClass.getResourceAsStream("<path to file in your JAR file>/log4j.properties")) {
    stream => {
        Log4jConfiguration.configure(stream)
    }
}
```

5. Add Apache Spark log messages at the appropriate level in your code as required. For example, use the `logDebug` method to send a debug log message. For more information, see [Logging](#) in the Spark documentation.

Scala

 Copy

```
logTrace("Trace message")
logDebug("Debug message")
logInfo("Info message")
```

```
logWarning("Warning message")
logError("Error message")
```


Run the sample application

The monitoring library includes a [sample application](#) that demonstrates how to send both application metrics and application logs to Azure Monitor. To run the sample:


1. Build the **spark-jobs** project in the monitoring library, as described in [Build the Azure Databricks Monitoring Library](#).
2. Navigate to your Databricks workspace and create a new job, as described [here](#).
3. In the job detail page, select **Set JAR**.
4. Upload the JAR file from `/src/spark-jobs/target/spark-jobs-1.0-SNAPSHOT.jar`.
5. For **Main class**, enter `com.microsoft.pnp.samplejob.StreamingQueryListenerSampleJob`.
6. Select a cluster that is already configured to use the monitoring library. See [Configure Azure Databricks to send metrics to Azure Monitor](#).

When the job runs, you can view the application logs and metrics in your Log Analytics workspace.

Application logs appear under SparkLoggingEvent_CL:

Kusto	 Copy
SparkLoggingEvent_CL where logger_name_s contains "com.microsoft.pnp"	

Application metrics appear under SparkMetric_CL:

Kusto	 Copy
SparkMetric_CL where name_s contains "rowcounter" limit 50	

Next steps

Deploy the performance monitoring dashboard that accompanies this code library to troubleshoot performance issues in your production Azure Databricks workloads.

Use dashboards to visualize Azure Databricks metrics