

Monitoring and Logging Reactor Applications

Exercise Objectives

In this exercise, you will:

- Add metrics and logging to the example project
 - Build, deploy and run the application
 - View the resulting metrics and logs in the Reactor Dashboard Metrics and Log Explorers
-

Exercise Steps

- Add imports
 - Add metrics and logging to Flowlets and Procedures
 - Build, deploy and run sentences through the Flow system
 - View the results in the Metrics and Log Explorers
-

Normalization Flowlet

Add to the Normalization Flowlet a handler to emit metrics:

```
/**
 * Handlers to emit logs and metrics.
 */
Metrics metrics;
private static final Logger LOG = LoggerFactory.getLogger(Normalization.class);
```

Then, modify the `process` method to emit the metrics:

```
if (text != null) {
    metrics.count("data.processed.size", text.length());
    out.emit(text);
} else {
    metrics.count("data.ignored.text", 1);
}
```

Analyze **Flowlet**

To the `Analyze` Flowlet, add a `LOG` variable:

```
private static final Logger LOG = LoggerFactory.getLogger(Analyze.class);
```

and add to its `process` method a Log statement:

```
LOG.info("Sentence = {}", sentence);
```

Update **Flowlet**

To the `Update` Flowlet, add a similar set of variables:

```
/**
 * Handlers to emit logs and metrics.
 */
Metrics metrics;
private static final Logger LOG = LoggerFactory.getLogger(Update.class);
```

and to its `process` method, add where `if Iterables.size(parts) == 2,:`

```
metrics.count("sentiment." + sentiment, 1);
LOG.info("Sentence = {}, Sentiment = {}", sentence, sentiment);
```

Add an `else` clause if not:

```
metrics.count("data.ignored.sentiments", 1);
```

Procedure Logging

To the Procedure, add a LOG variable:

```
private static final Logger LOG =  
    LoggerFactory.getLogger(SentimentAnalysisProcedure.class);
```

and after the `List<SimpleTimeseriesTable.Entry> entries`, add a Log statement:

```
LOG.info("sentiment:{}, entries:{}", sentiment, entries.size());
```

Build, Deploy and Run

- After stopping the Flow and Procedure, rebuild and redeploy the App
- Re-run some sentences, and view the metrics and logs using the Metrics and Logs Explorers in the Continuity Reactor Dashboard

```
curl -o /dev/null -sL -w "%{http_code}\\n" -d  
"Continuity Reactor is awesome"  
http://localhost:10000/v2/streams/sentence
```

- Run the query `sentiments` with the parameters such as `{"sentiment": "neutral"}` and read the results in the Log Explorer
-

Exercise Summary

You should now be able to:

- Perform basic Reactor operations of Start and Stop
 - Taken the Quick Start tour
 - Use the Dashboard for basic operations
 - Inject and Query Data through the Dashboard
 - Modify an Application
 - Redeploy and restart an Application
-

Exercise Completed

[Chapter Index](#)