PacIOOS FileSource Driver Manual

December 30, 2012

1. Overview

This document describes how to use the Java-based FileSource driver for streaming ASCII-based sample data from instruments such as Seabird CTDs, etc. to the PacIOOS DataTurbine streaming server. This guide assumes that the Java code has been compiled according to the Realtime Operations Guide.

2. Running from the command line

The FileSource java class can be run on the comand line using the ‘java’ command. It requires that all of the libraries that the class depends on are available to the JVM via the CLASSPATH environment variable. The class also takes command line options to configure it to read from a specific incoming data file, and to output to a specific DataTurbine server’s IP address, TCP port, etc.

2.1 Setting the CLASSPATH variable

The CLASSPATH variable should be set using the command. This can also be set in a bash script. We first set the location of the BBL software, and then use that as a base directory to point to all of the library dependencies.

```
export BBL_HOME=/usr/local/bbl/trunk;
export CLASSPATH=$BBL_HOME/build/classes/:\
$BBL_HOME/lib/rbnb.jar:\
$BBL_HOME/lib/commons-codec-1.3.jar:\
$BBL_HOME/lib/commons-cli-1.0.jar:\
$BBL_HOME/lib/commons-logging-1.0.4.jar:\
$BBL_HOME/lib/log4j-1.2.8.jar:\
$BBL_HOME/lib/log4j.properties;
```

Note that the backslashes above denote a continuation of the command line, escaping the newline character that makes it easier to read each line.

2.2 Understanding the command line options

The FileSource class can be called using the ‘java’ command with the -h option to show a listing of what all command line options are available. Use the fully-qualified java class name:

```
cd $BBL_HOME
java edu.hawaii.soest.kilonalu.utilities.FileSource -h

(results in)
usage: edu.hawaii.soest.kilonalu.utilities.FileSource
-e regular expression for data line matching, e.g “*[0-9][0-9]”
-d Date formats as a comma separated list(YYYY-MM-DD,HH:MM:SS
-f Date fields as a one-based comma separated list(1,2)
-C RBNB source channel name e.g. DecimalASCIISampleData
-D Delimiter between variables as a regular expression (“,”, “\s+”)
```

```
-F Data source file name e.g. /tmp/data.txt
-S RBNB Source Name *RBNBClient
-Z archive size *0
-h Print help
-p RBNB Server Port Number
-s RBNB Server Hostname
-t Timezone indicator (UTC, HST, EDT, etc.)
-v Print Version information
-z cache size *1024
```

2.3 Using the command line options

Once the CLASSPATH environment variable is set, call the fully-qualified class name, and pass in the command line options, with each option value configured for you particular setup. An example is shown below:

```
java edu.hawaii.soest.kilonalu.utilities.FileSource\ -F “/incoming-data/PIAS01-
data.txt”\
-e “# ., ., ., ., ., ., \d{2} [A-Z][a-z][a-z] *\d{4} *\d{2}:\d{2}:\d{2}\s”\
-S PIAS01_001CTDXXXXR00\
-C DecimalASCIISampleData\
-D “\s+”\
-d “dd MMM yyyy HH:mm:ss”\
-t “HST”\
-f “7”\
-s bbl.ancl.hawaii.edu\
-p 3333\
-z 126000\
-Z 31536000
```

Note that the ‘-d’ flag only has a single datetime format, followed by the ‘-f’ flag with a single column number (7). Other instruments may define the date in one column, and the time in another, both separated by a comma, such as:

```
-d “dd MMM yyyy, HH:mm:ss” -f “6,7”
```

The ‘-D’ flag is used to set the sample delimiter in cases where sample variables aren’t delimited by the default comma (,). This flag uses a regular expression, so in the example above (‘\s+’) we’re saying that each field is delimited by one or more whitespace characters. Also note the ‘-e’ flag. This tells the driver what to expect as a data sample, expressed as a regular expression pattern. Any ASCII text lines that don’t match this pattern will not be sent to the DataTurbine, such as lines that say “!!! LOW BATT !!!”. This first-pass quality control keeps that data channels pretty clean in the DataTurbine.

Lastly, note that time zone is set in the example using the ‘-t’ flag, and is not encoded in the date format.

3. Conclusion

This is a quick guide to the FileSource driver command line options. Although this doesn’t cover each option in detail, most are straight foward. An example of a bash script used to run the driver can be found here.