Byteman Workshop

Brno Devconf
7th February, 2014

Andrew Dinn

Workshop Agenda

- Example application
  - process dataflow pipeline
  - run/debug in eclipse
- Using Byteman to trace
  - mvn or ant with JUnit
- Using Byteman to inject faults
  - mvn or ant with JUnit
- Using Byteman to order threads
  - mvn or ant with TestNG

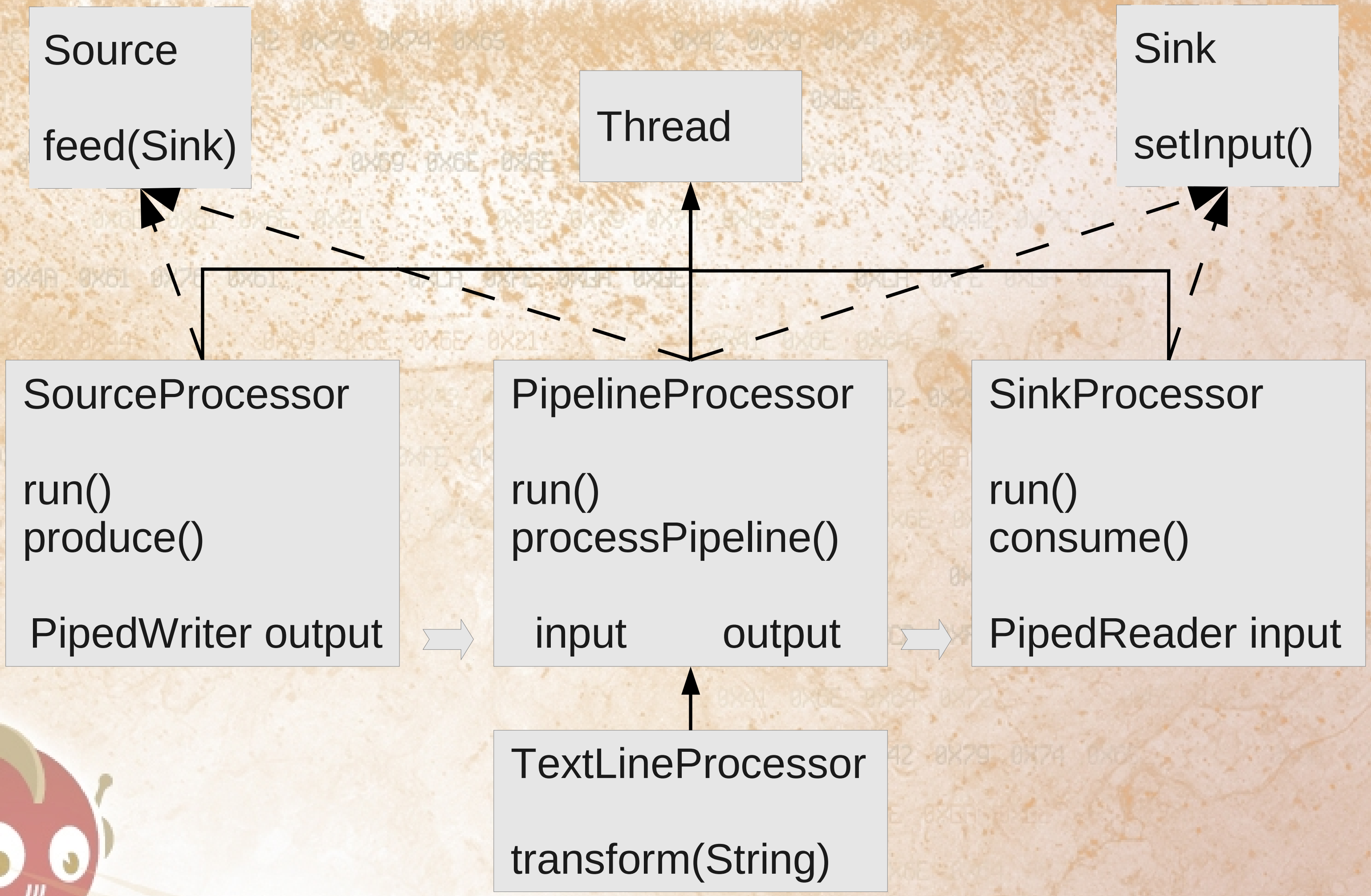
Introduction: Pipeline Apps

Brno DevConf 2014

Pipeline Core

- Process Dataflow Model
  - Stream of data
  - Transformed by sequence of processes
  - Processes execute in parallel
- Pipeline elements extend `Thread`
  - `SourceProcessor` implements `Source`
  - `SinkProcessor` implements `Sink`
  - `PipelineProcessor` implements `Source, Sink`

Pipeline Core



Brno DevConf 2014

Pipeline Impl

- **Source data from disk or memory**
  - `FileSource`, `CharSequenceSource`
- **Sink data to disk or memory**
  - `FileSink`, `CharSequenceSink`
- **Transform data in to data out**
  - `TraceProcessor`, `TeeProcessor`
  - `PatternReplacer`, `Binder`, `BindingInserter`, `BindingReplacer`
    - utility class `BindingMap`

Session 1: Run & Debug Apps

Brno DevConf 2014

Pipeline App

- “Run As” and “Debug As” from Eclipse
  - step through pipeline setup
  - step through pipeline execution(?)
- PipelineAppMain
  - see package org.my.app
  - copies a file
  - FileSource --> TraceProcessor --> FileSink

Pipeline App 2

- PipelineAppMain1
  - class PatternReplacer
    - matches patterns in each input line
    - substitutes replacement (may include matched text)
  - main pipeline
    - FileSource --> PatternReplacer 1 --> TeeProcessor 1 --> PatternReplacer 2 --> TeeProcessor 2 --> PatternReplacer 3 --> FileSink 3
  - tee branches dump intermediate stream
    - TeeProccessor 1 -->FileSink 1
    - TeeProccessor 2 -->FileSink 2

Pipeline App 3

- PipelineAppMain2
  - class BindingInserter
    - matches input text “`the ([A-Za-z0-9]+)`”
    - creates binding [`X1 --> boy`]
    - substitutes variable reference “`the ${X1}`”
    - existing bindings reused
  - main pipeline
    - CharSequenceSource --> BindingInserter 1 --> BindingInserter 2 --> BindingInserter 3 --> CharSequenceSink

Pipeline App 4

- PipelineAppMain2
  - class Binder
    - binds like BindingInserter but does not replace
  - class BindingReplacer
    - matches bound variable “`the ${X4}`”
    - replaces with binding “`the stick`”
  - main pipeline
    - CharSequenceSource --> Binder --> BindingReplacer 2 --> CharSequenceSink

Session 2: Tracing with Byteman

Brno DevConf 2014

Tracing with Byteman

- example in junit mvn submodule
  - run with `mvn -P junit test` or `ant junit`
  - class BytemanJUnitTests
    - `@RunWith` installs Byteman agent, injects rules
    - `@BMScript` says load rules from script file
  - file trace.btm
    - code to be injected into app/runtime classes before tests
    - code gets removed after tests
    - annotation on class or single test method

Tracing with Byteman 2

- inject into app or JVM class/interface
  - n.b. `^Thread` injects into subclasses too
- target method and location
  - AT ENTRY, AT EXIT, etc
- rule firing is always conditional
  - `IF TRUE`, IF NOT `$1.equals($!)`
- conditions/actions are Java expressions
  - multiple actions separated by `;`

Tracing with Byteman 3

- **method variables can be read and written**
  - `$0` for this, `$1` etc for params, `$i` for locals, `$!` for return value (`AT EXIT` or `AFTER CALL`)
- **expressions built from usual Java syntax**
  - `$! = $1.substring(0, $len)`
- can also use Byteman builtin methods
  - look like function calls
  - e.g. `traceln(String)` prints to `System.out`
  - many other builtins available (as we will see)

mvn and ant configuration

- mvn requires 4 jars as test dependencies
  - byteman.jar, bminstall.jar. bmsubmit.jar, bmunit.jar
  - also depend on JDK tools.jar
  - in surefire config `useManifestOnlyJar = true`
- ant requires same 4 jars as download
  - download zip from `jboss.org/byteman`
  - export `BYTEMAN_HOME=<unzip_root_dir>`
  - add jars to classpath
    - junit task -- all 4 byteman jars plus tools.jar
    - compile task – just bmunit.jar

Session 3: Fault Injection

Brno DevConf 2014

Fault Injection with Byteman

- example in junit2 mvn submodule
  - run with `mvn -P junit2 test` or `ant junit2`
  - class BytemanJUnitTests2
    - `@BMRule` provides rule text inline
    - `@BMRules` groups multiple `@BMRule` annotations
  - throw new java.io.IOException()
    - thrown from trigger method `processPipeline`
    - bypasses internal control structure (try/catch)
    - exception must be declared by trigger method
      - or unchecked exception

Fault Injection with Byteman 2

- location `AT CALL transform(String)`
  - injects rule just before call to `transform`
- countdown builtins
  - use in condition to trigger rule at Nth firing
  - createCountDown(Object label, int count)
    - Object is to identify new countdown instance
  - countDown(Object label) --> boolean
    - decrements countDown identified byObject
    - returns false if pre-decrement count > 0
    - returns true if pre-decrement count == 0

Session 4: Thread Control

Brno DevConf 2014

Thread Control with Byteman

- example in testng mvn submodule
  - run with `mvn -P testng test` or `ant testng`
  - different ordering exposes race condition
    - upstream `Binder` *binds* `X4` when it transforms line 3
    - downstream `Replacer` *uses* `X4` when it transforms line 2
    - can the *use* happen before the *bind*?
  - class BytemanNGTests
    - inject rendezvous (barrier) code into app *and* test code
    - test thread meets both pipeline threads at a rendezvous
    - different tests meet in different orders

Thread Control with Byteman

- ordering rules in timing.btm
  - rendezvous each for Binder/BindingReplacer
    - create labels rendezvous with pipeline processor
  - counting rule tracks number of lines processed
    - Counter labelled with pipeline processor
    - initial value is zero
  - rendezvous in transform AT ENTRY EXIT
    - thread *cannot start* transform until 1st rendezvous exited
    - thread *must have completed* transform after 2nd exit
    - condition ensures rendezvous is for correct line

Byteman Resources

- **http://jboss.org/byteman/documentation**
  - command line tutorial
  - fault injection tutorial
    - older version of this material
  - other online resources
- **http://jboss.org/byteman/downloads**
  - access to current and older releases
- **http://bytemanblog.blogspot.com/**
  - release announcements

Byteman Resources

- **https://issues.jboss.org/browse/BYTEMAN**
  - JIRA issue management
  - report a bug/request a feature
- **http://community.jboss.org/en/byteman**
  - user forum
    - ask for help
    - describe bugs here (before raising a JIRA)
- **https://community.jboss.org/en/byteman/dev**
  - dev forum
    - suggest code fixes
    - suggest new features here (before raising JIRA)