http://www.egovframe.go.kr/wiki/doku.php?id=egovframework:rte2:brte:batch_example:skip_mgmt

Example of Skip

Outline

The Skip Example demonstrates how to skip the error data to move on to the ensuing data. In EgovSkipSampleFunctionalTests, you can check out the pre-configured skips are implemented for exceptions thrown.

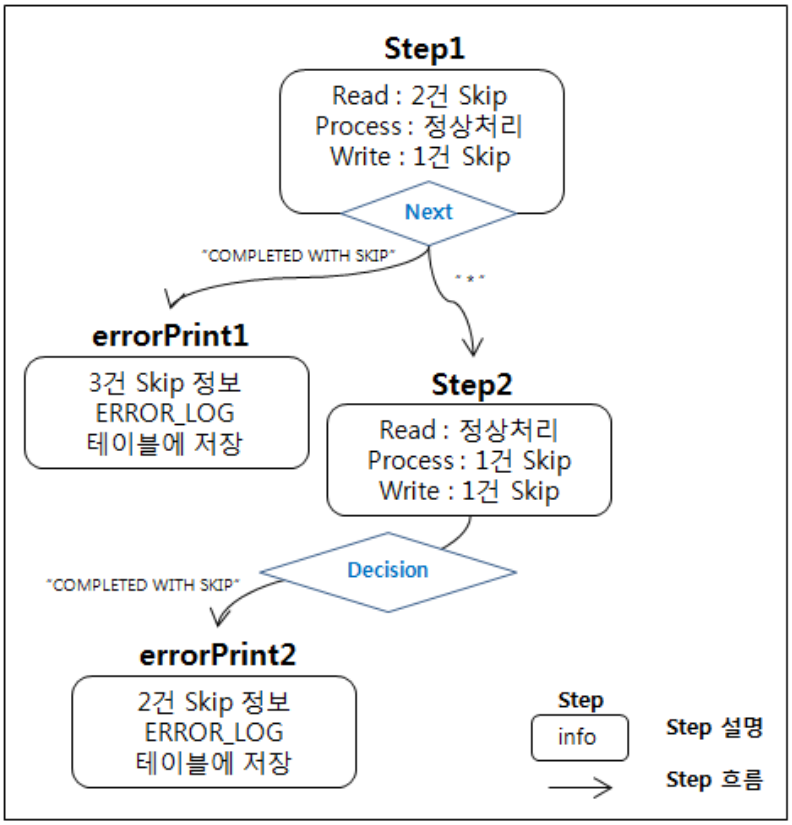
Description

Settings

Configuring Jobs

Check skipSample.xml, the Job configuration file for the Skip Example.

✔ See the flow of step for skipSample



Refer to the following for chunk configuration in Job composition:

- skip-limit : The maximum skip count
- <skippable-exception-classes> : Scope of Exception Thrown.

<include> : Designate the exception to skip.
<exclude>: The exceptions that won't actuate Skip among sub-exceptions.

```
<job id="skipJob" incrementer="incrementer" xmlns="http://www.springframework.org/schema/batch">
        <step id="step1" parent="baseStep">
                <tasklet>
                        <chunk reader="fileItemReader" processor="tradeProcessor" writer="tradeWriter"
                                commit-interval="3" skip-limit="10">
```

```
                                        <skippable-exception-classes>
                                                <include
class="org.springframework.batch.item.file.FlatFileParseException" />
                                                <include
class="org.springframework.batch.item.WriteFailedException" />
                                        </skippable-exception-classes>
                                </chunk>
                        </tasklet>
                        <next on="*" to="step2" />
                        <next on="COMPLETED WITH SKIPS" to="errorPrint1" />
                        <fail on="FAILED" exit-code="FAILED" />
                </step>
                <step id="errorPrint1" next="step2">
                        <tasklet ref="errorLogTasklet" />
                </step>
                <step id="step2" parent="baseStep" next="skipCheckingDecision">
                        <tasklet>
                                <chunk reader="tradeSqlItemReader" processor="tradeProcessorFailure"
writer="itemTrackingWriter"
                                        commit-interval="2" skip-limit="10">
                                        <skippable-exception-classes merge="true">
                                                <include
class="org.springframework.batch.item.validator.ValidationException" />
                                                <include class="java.io.IOException" />
                                        </skippable-exception-classes>
                                </chunk>
                                <no-rollback-exception-classes>
                                        <include
class="org.springframework.batch.item.validator.ValidationException" />
                                </no-rollback-exception-classes>
                        </tasklet>
                </step>
                <decision id="skipCheckingDecision" decider="skipCheckingDecider">
                        <end on="*" />
                        <next on="COMPLETED WITH SKIPS" to="errorPrint2" />
                        <fail on="FAILED" exit-code="FAILED" />
                </decision>
                <step id="errorPrint2">
                        <tasklet ref="errorLogTasklet" />
                </step>
</job>
```

Composition and Implementation of JunitTest

Composition of JunitTest

Work Junit Test using the simple-job-launcher-context.xml and skipSample.xml configurations, where batch implementation is involved.

✔ See Junit Test Description for Batch Execution for more information.
✔ assertEquals("COMPLETED", jobExecution.getExitStatus().getExitCode()) : Make sure you check the batch execution result is COMPLETED.
✔ Check out th annotation for the following test classes to see how skip takes place.

```
@ContextConfiguration(locations = {"/egovframework/batch/simple-job-launcher-context.xml",
                        "/egovframework/batch/jobs/skipSampleJob.xml",
                        "/egovframework/batch/job-runner-context.xml" })
public class EgovSkipSampleFunctionalTests {        ...
                @Test
                public void testJobIncrementing() {
```

```
...
JobExecution execution1 = jobExplorer.getJobExecution(id1);
assertEquals(BatchStatus.COMPLETED, execution1.getStatus());

validateLaunchWithSkips(execution1);
}

private void validateLaunchWithSkips(JobExecution jobExecution) {

// Step1: 10 input records, a total of 2 skips took place in READ (ROW:5,10), a skip took place in WRITE, a total of seven SKIP => OUTPUT took place
assertEquals(7, SimpleJdbcTestUtils.countRowsInTable(simpleJdbcTemplate, "TRADE"));

// Step2: 7 input records, a skip took place in PROCESS, a skip took place in WRITE, a total of five SKIP => OUTPUT took place
assertEquals(5, simpleJdbcTemplate.queryForInt("SELECT COUNT(*) from TRADE where VERSION=?", 1));

// A skip took place in Step2
assertEquals(1, EgovSkipCheckingListener.getProcessSkips());

// Information recorded in the table ERROR_LOG for both steps
assertEquals(2, SimpleJdbcTestUtils.countRowsInTable(simpleJdbcTemplate, "ERROR_LOG"));

// A total of three skips took place in Step1
assertEquals("3 records were skipped!", simpleJdbcTemplate.queryForObject(
"SELECT MESSAGE from ERROR_LOG where JOB_NAME = ? and STEP_NAME = ?", String.class, "skipJob", "step1"));

// A total of two skips took place in Step2
assertEquals("2 records were skipped!", simpleJdbcTemplate.queryForObject(
"SELECT MESSAGE from ERROR_LOG where JOB_NAME = ? and STEP_NAME = ?", String.class, "skipJob", "step2"));

}
}
```

Implementation of JunitTest

See Implementation of JunitTest for more information.

Verify Result

You can check out the list of skips taken place in EgovErrorLogTasklet, in the table ERROR_LOG.

EgovSkipSampleFunctionalTests.java | skipSample-job-launcher-context.xml | skipSampleJob.xml | ERROR_LOG

JOB_NAME [CHAR(20)]	STEP_NAME [CHAR(20)]	MESSAGE [VARCHAR(300)]
skipJob	step1	3 records were skipped!
skipJob	step2	2 records were skipped!
<new row>		

References

- Skip