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

# Synchronous / Asynchronous Processing Examples

## Outline

In this Section we are going to learn how uniform (batch) processing involves synchronous processing to stand by till the end of data processing and asynchronous processing receiving data using callback mechanism.

## Description

### Settings

#### Configuring Launcher

**Check out skipSample-job-launcher-context.xml, the launcher file for Synchronous / Asynchronous Processing.**

You can get either synchronous or asynchronous for data processing, with synchronous being default configuration in the following example. The configuration is to be made in the property taskExecutor of jobLauncher bin in the launcher configuration file, where you can configure for either 'sync' or 'async':

- sync : Configures classes for synchronous processing.
- async : Configures classes for asynchronous processing.

```
<bean id="jobLauncher" class="org.springframework.batch.core.launch.support.SimpleJobLauncher">
        <property name="jobRepository" ref="jobRepository" />
        <property name="taskExecutor" ref="sync"/>   <!-- for asynchronous ref="async" -->
</bean>

<!-- for sync -->
<bean id="sync" class="org.springframework.core.task.SyncTaskExecutor" />

<!-- for async -->
<bean id="async" class="org.springframework.core.task.SimpleAsyncTaskExecutor" />
```

#### Configuring Jobs

**Check delegatingJob.xml, the job configuration file for Synchronous / Asynchronous Processing.**

Configuring jobs for Synchronous / Asynchronous Processing is nothing special. Refer to Job Configuration Exampleformore information.

```
<job id="delegateJob" xmlns="http://www.springframework.org/schema/batch">
        <step id="delegateStep1">
                <tasklet>
                        <chunk reader="reader" writer="writer" commit-interval="3"/>
                </tasklet>
        </step>
</job>

<bean id="reader" class="org.springframework.batch.item.adapter.ItemReaderAdapter">
        <property name="targetObject" ref="delegateObject" />
        <property name="targetMethod" value="getData" />
</bean>

<bean id="writer" class="org.springframework.batch.item.adapter.PropertyExtractingDelegatingItemWriter">
        <property name="targetObject" ref="delegateObject" />
```

```
        <property name="targetMethod" value="processPerson" />
        <property name="fieldsUsedAsTargetMethodArguments">
                <list>
                        <value>firstName</value>
                        <value>address.city</value>
                </list>
        </property>
</bean>

<bean id="delegateObject" class="egovframework.brte.sample.common.domain.person.PersonService" />
```

### Composition and Implementation of JunitTest

#### Composition of JunitTest

**Work Junit Test that comprises sync-job-launcher-context and delegatingJob, where batch implementation is involved.**

✔For structure of the class JunitTest, see Junit Test Description for Batch Execution Environment.
✔ assertEquals("COMPLETED", jobExecution.getExitStatus().getExitCode()) : Check if the batch result is COMPLETED.
✔ Thread.sleep(4000) : As for asynchronous execution of batch, you won't find the proper status of job completed (COMPLETED,FAILED) although the status of batch (UNKNOWN) is set in DB. In this connection, the example involves temporary stop of thread to check for the result of job executed:

```
@ContextConfiguration(locations = { "/egovframework/batch/sync-job-launcher-context.xml",
                                    "/egovframework/batch/jobs/delegatingJob.xml",
                                    "/egovframework/batch/job-runner-context.xml" })
public class EgovSyncDelegatingJobFunctionalTests {
        ...
        @Test
        public void testLaunchJob() throws Exception {
                JobExecution jobExecution=null;
                try{
                        jobExecution =jobLauncherTestUtils.launchJob();
                        // Exit Status UNKNOWN in case of Async
                        assertEquals("UNKNOWN", jobExecution.getExitStatus().getExitCode());
                        Thread.sleep(4000);

                    }catch (InterruptedException ie){
                        ie.printStackTrace();
                    }
                  assertTrue(personService.getReturnedCount() > 0);
                  assertEquals(personService.getReturnedCount(), personService.getReceivedCount()) ;
                  assertEquals("COMPLETED", jobExecution.getExitStatus().getExitCode());
        }
}
```

#### Implementation of JunitTest

See Implementation of JunitTest for more information.

## References

- Synchronous/Asynchronous Examples
- JobLauncher