**Spring Batch**

--Created by David Song

Contents

[Simple batch job without DB 1](#_Toc121063237)

[With xml configuration 1](#_Toc121063238)

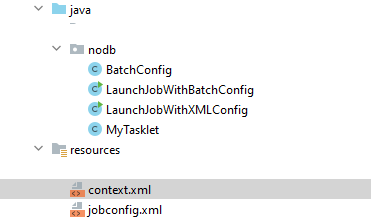
[with java bean configuration 3](#_Toc121063239)

[References 4](#_Toc121063240)

# Simple batch job without DB

This section will show a simple example how to create and launch a spring batch job without using database. The example is twisted from [1].

Code structure:



## With xml configuration

Context.xml

<?xml version="1.0" encoding="UTF-8"?>  
<beans xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xmlns="http://www.springframework.org/schema/beans"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans-3.2.xsd">  
  
<bean id="jobRepository"  
 class="org.springframework.batch.core.repository.support.MapJobRepositoryFactoryBean">  
 <property name="transactionManager" ref="transactionManager"/>  
</bean>  
  
<bean id="transactionManager"  
 class="org.springframework.batch.support.transaction.ResourcelessTransactionManager"/>  
  
<bean id="jobLauncher"  
 class="org.springframework.batch.core.launch.support.SimpleJobLauncher">  
 <property name="jobRepository" ref="jobRepository"/>  
</bean>  
  
</beans>

Jobconfig.xml

<beans xmlns:batch="http://www.springframework.org/schema/batch"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xmlns="http://www.springframework.org/schema/beans"  
 xsi:schemaLocation="http://www.springframework.org/schema/batch  
 http://www.springframework.org/schema/batch/spring-batch-2.2.xsd  
 http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans-3.2.xsd ">  
  
 <import resource="context.xml"/>  
  
 <!-- Defining a bean -->  
 <bean id="tasklet" class="nodb.MyTasklet"/>  
  
 <!-- Defining a job-->  
 <batch:job id="helloWorldJob">  
 <!-- Defining a Step -->  
 <batch:step id="step1">  
 <tasklet ref="tasklet"/>  
 </batch:step>  
 </batch:job>  
</beans>

Launch Job

**package** nodb;  
**import** org.springframework.batch.core.Job;  
**import** org.springframework.batch.core.JobExecution;  
**import** org.springframework.batch.core.JobParameters;  
**import** org.springframework.batch.core.launch.JobLauncher;  
**import** org.springframework.context.ApplicationContext;  
**import** org.springframework.context.support.ClassPathXmlApplicationContext;  
  
**public class** LaunchJobWithXMLConfig {  
 **public static void** main(String[] args) **throws** Exception {  
  
 System.**out**.println(**"main started ..."**);  
  
 //  
 // load the configuration  
 //  
 String[] springConfig = {**"jobconfig.xml"**};  
 ApplicationContext context = **new** ClassPathXmlApplicationContext(springConfig);  
  
 JobLauncher jobLauncher = (JobLauncher) context.getBean(**"jobLauncher"**);  
 Job job = (Job) context.getBean(**"helloWorldJob"**);  
  
 //  
 // Executing the JOB with no parameters  
 //  
 JobExecution execution = jobLauncher.run(job, **new** JobParameters());  
 System.**out**.println(**"Exit Status : "** + execution.getStatus());  
 }  
}

## with java bean configuration

**package** nodb;  
  
**import** …  
  
@Configuration  
**public class** BatchConfig {  
  
 @Bean  
 ResourcelessTransactionManager transactionManager(){  
 **return new** ResourcelessTransactionManager();  
 }  
  
 @Bean  
 JobRepository jobRepository(){  
 **try** {  
 **return new** MapJobRepositoryFactoryBean(transactionManager()).getObject();  
 } **catch** (Exception e) {  
 **throw new** RuntimeException(e);  
 }  
 }  
  
 @Bean(name = **"jobLauncher"**)  
 SimpleJobLauncher jobLauncher(){  
 SimpleJobLauncher jobLauncher = **new** SimpleJobLauncher();  
 jobLauncher.setJobRepository(jobRepository());  
 **return** jobLauncher;  
 }  
  
 @Bean  
 Job helloWorldJob(){  
  
 JobBuilderFactory factory = **new** JobBuilderFactory(jobRepository());  
  
 JobBuilder builder = factory.get(**"helloWorldJob"**);  
 **return** builder.flow(step1()).end().build();  
  
 }  
 @Bean  
 **public** Step step1() {  
 StepBuilderFactory stepFactory = **new** StepBuilderFactory(jobRepository(), transactionManager());  
 **return** stepFactory.get(**"step1"**).tasklet(tasklet()).build();  
 }  
  
 @Bean  
 MyTasklet tasklet(){  
 **return new** MyTasklet();  
 }  
  
}

Launch the job

package nodb;  
import ……

public class LaunchJobWithBatchConfig {  
 public static void main(String[] args) throws Exception {  
  
 System.out.println("main started ...");  
  
 //  
 // load the configuration  
 //  
 ApplicationContext context = new AnnotationConfigApplicationContext(BatchConfig.class);  
 String[] beans = context.getBeanDefinitionNames();  
 for (String bean : beans) {  
 System.out.println(bean);  
 }  
  
  
 JobLauncher jobLauncher = (JobLauncher) context.getBean("jobLauncher");  
 Job job = (Job) context.getBean("helloWorldJob");  
  
 //  
 // Executing the JOB with no parameters  
 //  
 JobExecution execution = jobLauncher.run(job, new JobParameters());  
 System.out.println("Exit Status : " + execution.getStatus());  
 }  
}

From above examples, we can see that the xml configuration is easier and clearer than the java bean configuration in this case.

# References

1. https://www.tutorialspoint.com/spring\_batch/spring\_batch\_basic\_application.htm