

# JobLauncher

## Outline

JobLauncher is used to execute the batches using jobs and job parameters and returning JobExecutions.

## Description

In JobLauncher you can find out the method Run is defined to execute the batches using jobs and job parameters and returning JobExecutions.

```
public interface JobLauncher {  
  
    public JobExecution run(Job job, JobParameters jobParameters) throws  
        JobExecutionAlreadyRunningException,  
        JobRestartException, JobInstanceAlreadyCompleteException,  
        JobParametersInvalidException;  
  
}
```

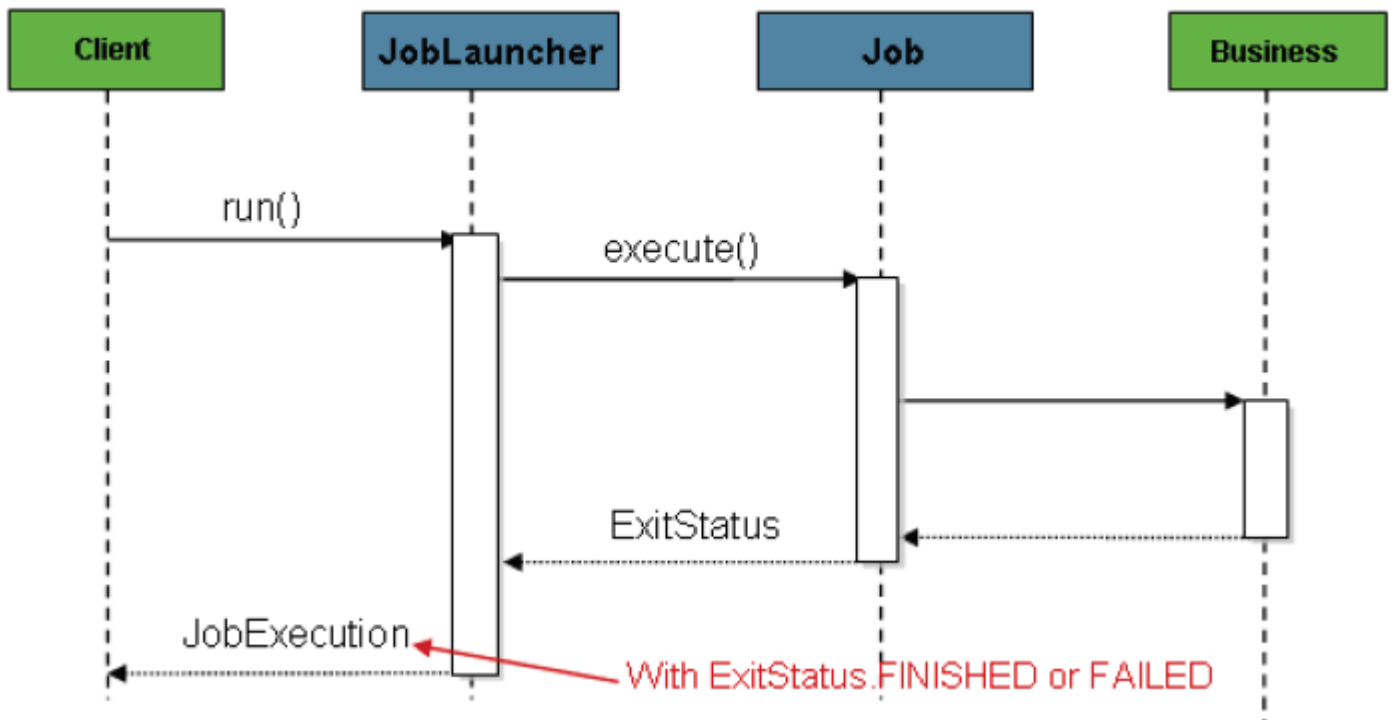
In the JobLauncher interface, SimpleJobLauncher is provided to implement the class using JobName and JobParameter to acquire JobExecution information in [JobRepository](#).

Refer to the following for how jobLauncher can be configured. Note that configuration of JobRepository is required to acquire JobExecution information:

```
<bean id="jobLauncher" class="org.springframework.batch.core.launch.support.SimpleJobLauncher">  
    <property name="jobRepository" ref="jobRepository" />  
</bean>
```

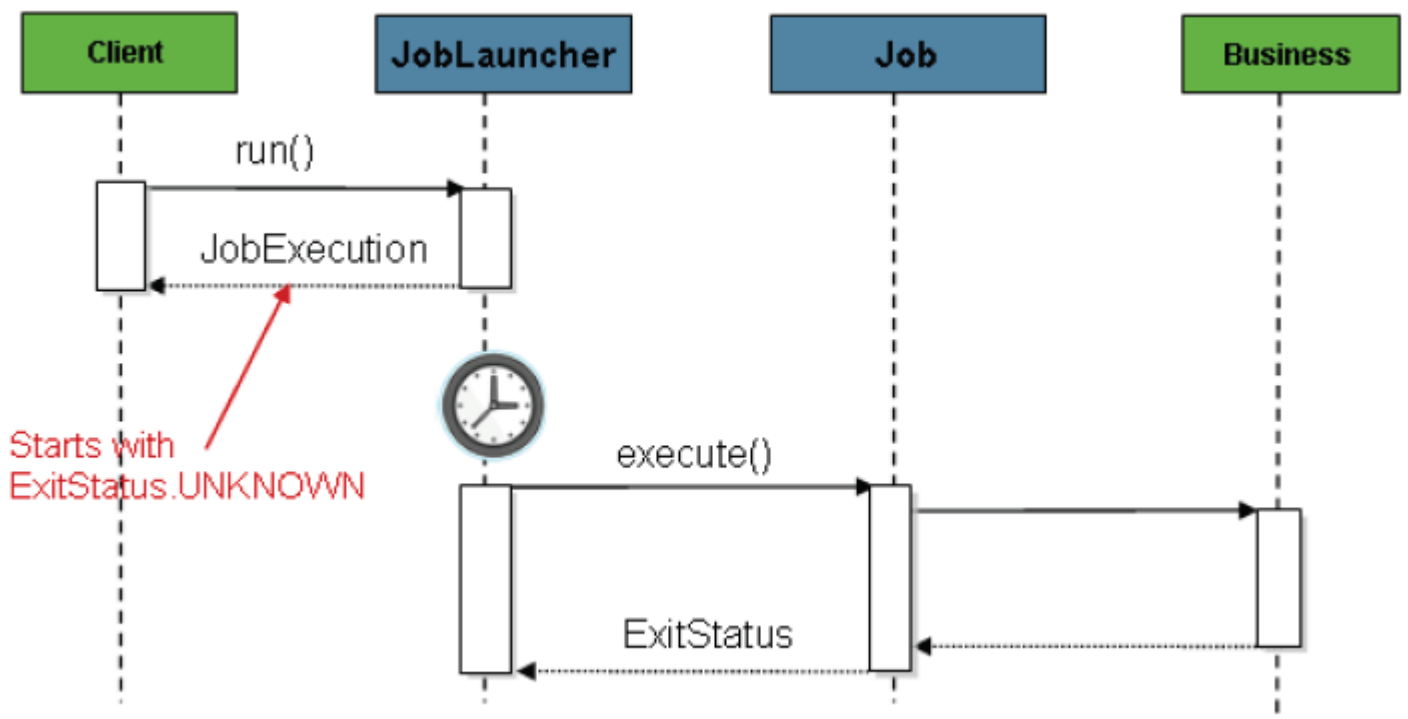
Both synchronous and asynchronous executions of job are available for JobLauncher, by wisely configuring taskExecutor. The default configuration for the class syncTaskExecutor synchronously actuates the following executions. When the client is requested to process batch task, JobLauncher acquires a single JobExecution transferred to the concerned method for execution of the task and returns JobExecution to the eventual client:

- Synchronous



Executing the foregoing workflow in the scheduler is a cakewalk, until you encounter HTTP request that involves latency. Instead of wasting your precious time until HTTP response is delivered, you are advised to work asynchronously to have SimpleJobLauncher return JobExecution to the eventual Client, as follows:

- Asynchronous



You can simply work the JobLauncher configuration for asynchronous configuration in the class SimpleAsyncTaskExecutor, as follows:

```

<bean id="jobLauncher" class="org.springframework.batch.core.launch.support.SimpleJobLauncher">
  <property name="jobRepository" ref="jobRepository" />
  <property name="taskExecutor">

```

```
        <bean class="org.springframework.core.task.SimpleAsyncTaskExecutor" />
    </property>
</bean>
```

Note that Spring's TaskExecutor interface is intended to control batch tasks executed on an asynchronous

## References

- [Synchronous/Asynchronous Examples](#)
- <http://static.springsource.org/spring-batch/reference/html/configureJob.html#configuringJobLauncher>