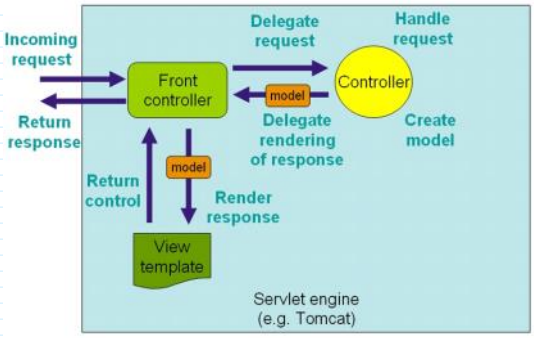
Diagrama de Spring MVC y Spring Batch

miércoles, 1 de febrero de 2023 06:33 a. m.

Procesamiento de una petición en Spring MVC



- Todas las peticiones HTTP se canalizan a través del *front controller*. En casi todos los frameworks MVC que siguen este patrón, el *front controller* no es más que un servlet cuya implementación es propia del framework. En el caso de Spring, la clase DispatcherServlet.
- El *front controller* averigua, normalmente a partir de la URL, a qué Controller hay que llamar para servir la petición. Para esto se usa un HandlerMapping.
- Se llama al Controller, que ejecuta la lógica de negocio, obtiene los resultados y los devuelve al servlet, encapsulados en un objeto del tipo Model. Además se devolverá el nombre lógico de la vista a mostrar (normalmente devolviendo un String, como en JSF).
- Un ViewResolver se encarga de averiguar el nombre físico de la vista que se corresponde con el nombre lógico del paso anterior.
- Finalmente, el *front controller* (el DispatcherServlet) redirige la petición hacia la vista, que muestra los resultados de la operación realizada.

En realidad, el procesamiento es más complejo. Nos hemos saltado algunos pasos en aras de una mayor claridad. Por ejemplo, en Spring se pueden usar interceptores, que son como los filtros del API de servlets, pero adaptados a Spring MVC. Estos interceptores pueden pre y postprocesar la petición alrededor de la ejecución del Controller. No obstante, todas estas cuestiones deben quedar por fuerza fuera de una breve introduccion a Spring MVC como la de estas páginas.

¿Qué es Spring Batch?

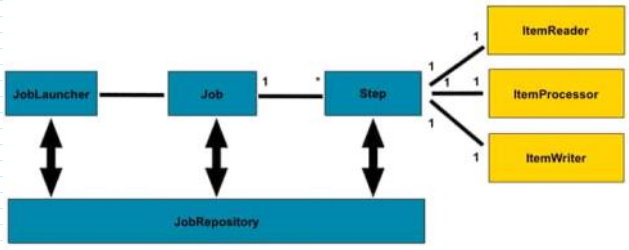
Spring Batch es un **framework ligero open source para procesamientos batch o procesamientos por lotes.** Este framework es un módulo de Spring y fue desarrollado como fruto de una colaboración entre **SpringSource** (ahora Pivotal) y **Accenture**.
Dicho framework, además de estar enfocado al procesamiento batch, incluye más herramientas que nos permiten monitorizar estos procesos, disponer de logs, configuraciones, transaccionalidad, estadísticas, alertas, etc.

¿Qué es un proceso batch?

Un **proceso batch o proceso por lote** es un proceso pensado para trabajar con volúmenes muy grandes de datos y generalmente de una forma programada. Es decir, sin intervención humana.
Imaginemos, por ejemplo, la carga de un fichero enorme con millones de registros, o bien un proceso nocturno que, a partir de una serie de consultas, envía una gran cantidad de e-mails, sms, etc. Esto sería un proceso batch.
Hay que aclarar que **Spring Batch no es un planificador de tareas (*scheduller*)**, aunque puede incluirse un planificador en el proceso batch. Existen muchos planificadores que pueden integrarse con Spring Batch. Por ejemplo, Quartz y Control-M entre otros.

Componentes de Spring Batch

Este framework está compuesto por los siguientes **componentes**



Componentes de Spring Batch.

Job y Step

Un **job** es un bloque de trabajo y está compuesto por uno o varios pasos o **steps**. Una vez se han llevado a cabo todos estos pasos, se considera el job como completado.
Cada uno de estos **steps** suele constar de **tres partes**:

1. **ItemReader**: se encarga de la lectura del procesamiento por lotes. Esta lectura puede ser, por ejemplo, de una base datos, o también podría ser de un broker de mensajes o bien un fichero csv, xml, json, etc.
2. **ItemProccessor**: se encarga de transformar items previamente leídos. Esta transformación además de incluir cambios en el formato puede incluir filtrado de datos o lógica de negocio.
3. **ItemWriter**: este elemento es lo opuesto al itemReader. Se encarga de la escritura de los ítems. Esta puede ser inserciones en una base de datos, en un fichero csv, en un broker de mensajes, etc.

Si observamos, está centrado a **trabajar con los items de manera unitaria**. Para el procesamiento por lotes podemos definir de qué tamaño será el número de ítems en el que se organizará el procesamiento por lotes. Si cogemos un tamaño 20, leerá, procesará y escribirá de 20 en 20. Este número de ítems que se procesarán en cada uno de los commits que realice el step se denomina **chunk**.

Tasklet

Un step no tiene que estar compuesto por un reader, processor y writer. También puede tener únicamente una lógica de negocio. Es el caso del tasklet con el código que se desea ejecutar en el step.

Consejos y buenas prácticas en Spring Batch

A raíz de los tres proyectos en los que he sido partícipe con Spring Batch, hay una serie de **lecciones aprendidas** que quiero dejar reflejadas para quienes vayan a aplicar esta solución en un proceso y también para mí mismo, a modo de recordatorio. Recalcar que es una visión subjetiva mía propia y que no es una norma. En muchos casos una opinión o preferencia propia basada en mi experiencia.

Principios a la hora de definir un proceso batch

- **Simplificar** todo lo posible la **lógica**: de forma que quede fragmentada en procesos muy pequeños de lectura, procesamiento y escritura.
- Utilizar los **mínimos recursos posibles**: ya que se va a procesar un enorme volumen de datos.
- Revisar y **optimizar** sentencias **sql**: es esencial que las consultas estén optimizadas ya que se verá tanto en el redimiendo de la base de datos como en los tiempos de ejecución del job.
- Utilizar comprobaciones **checksum**: esto es muy útil a la hora de generar ficheros de gran tamaño, incluir en el pie del fichero un contador con el número de registros e información del procesamiento realizado que puede ayudar a comprobar la integridad del fichero.
- Utilizar pruebas de **stress** con datos lo más realistas posibles.