Pátron Memento en
Repositorio Spring Web Flow

Link repositorio:
https://github.com/spring-projects/spring-webflow/tree/main/spring-webflow/src/main/java/org/springframework/webflow/execution/repository/support

?QUÉ ES ?

Es un framework para aplicaciones web Java que ayuda a crear flujos de trabajo para la interfaz de usuario.

Tambien sirve para agregar una capa de abstraccion para manejar la navegacion entre las vistas, validacion de entradas y gestion del estado de la aplicacion.

ESTRUCTURA DE DISEÑO

esta basado en el Spring MVC y tambien utiliza el diseño Front Controller. En el modelo esta una parte de acceso a datos y otra que contiene la lógica y funciones del servicio, en el controlador se encuentra el patrón Front Controller y Flow Controller para controlar la navegación y la transición de vistas

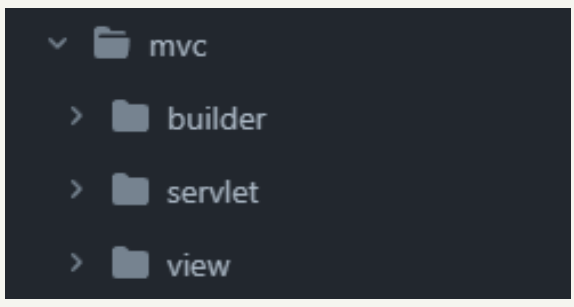
Retos

La aplicación al manejar tantas cosas para el desarrollo web cuenta con varios retos, entre los cuales destacan:

-Manejo de estados: Usa manejos de flujos complejos y dinámicos, por lo cual maneja estados , por lo cual,hay que escalar.

'**Integración:** El framework debe funcionar con otros marcos como Spring MVC y proveer funcionalidades adicionales. La integración puede ser un desafio ya que requiere conocimiento.

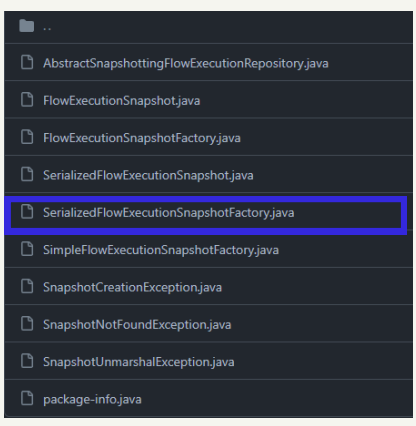
Diseño de las vistas: Las vistas de Spring Web Flow deben diseñarse para mostrar la información relevante en cada paso del flujo de trabajo. El diseño de las vistas puede ser un desafío, especialmente para flujos de trabajo complejos con múltiples pasos y transiciones.



Patrón Memento

El uso del patron dentro del repositorio se encuentra en el archivo **org.springframework.webflow.execution.repository.snapshot** en **SerializedExecutionSnapShotFactory.**

Tambien es posible encontrarlo en otras interfaces como **DefaultExecutionSnapshotFactory, CompressingExecutionSnapshotFactory ,** entre otras pero usaremos este a modo de ejemplo.



Función
SerializedExecutionSnapShotFactory.

Dentro de esta clase se encuentra parte de la implementación Memento. En este objeto se crean objetos snapshots de la ejecución de onjetos en el flujo de Spring Web Flow

En el objeto podemos encontrar las funciones CreateSnapshot y restoreExecution que por medio de la creacion de snapshots que contienen la información actual de una ejecución, les permite guardar y restaurar el estado anterior en las sesión con un usuario- Permitiendo que recuperen estados previos del trabajo .

```
public FlowExecutionSnapshot createSnapshot(FlowExecution flowExecution) throws SnapshotCreationException {
        return new SerializedFlowExecutionSnapshot(flowExecution, compress);
}

public FlowExecution restoreExecution(FlowExecutionSnapshot snapshot, String flowId, FlowExecutionKey key,
                MutableAttributeMap<Object> conversationScope, FlowExecutionKeyFactory keyFactory)
                throws FlowExecutionRestorationFailureException {
        SerializedFlowExecutionSnapshot snapshotImpl = (SerializedFlowExecutionSnapshot) snapshot;
        FlowDefinition def = flowDefinitionLocator.getFlowDefinition(flowId);
        FlowExecution execution;
        try {
                execution = snapshotImpl.unmarshal(def.getClassLoader());
        } catch (SnapshotUnmarshalException e) {
                throw new FlowExecutionRestorationFailureException(key, e);
        }
        flowExecutionFactory.restoreFlowExecution(execution, def, key, conversationScope, flowDefinitionLocator);
        return execution;
}
```

En que se parece al patrón...

Dentro del patrón Memento se permite guardar y restaurar el estado de un objeto por medio de un objeto de Memento que captura un estado interno de un objeto y luego restaurar el estado a una fecha posterior, algo asi como una foto o un recuerdo. Tiene la caracteristica de que no rompe el principio de la encapsulacion, entonces el objeto independientemente de estar guardado por otro objeto solo puede ser accedido y modificada por los métodos definidos.

Un ejemplo de esto es cuando cierras word sin guardar, y puedes recuperar la ultima version capturada del archivo.

Usos comunes

CTRL + Z

Se huce para implementar el deshacer. Al poner los mementos en una pila es posible restaurar cambios en orden inverso.

OH OH

Para deshacer operaciones realizaadas por el usuario dentro de las aplicaciones.

CHECKPOINT

Se usa como checkpoint en videojuegos. Cuando mueren pueden restaurar a estados anteriores

Pátron Memento

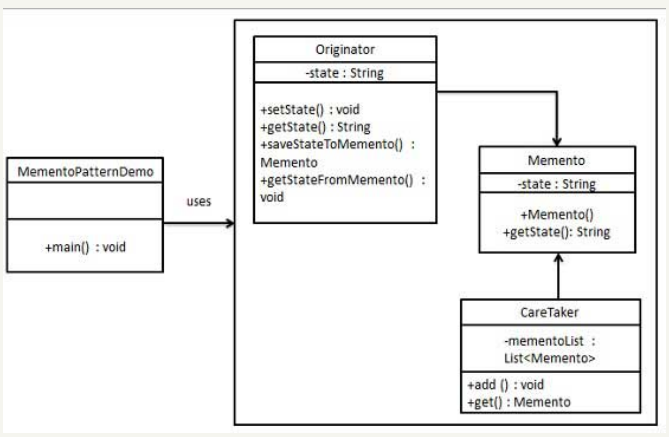
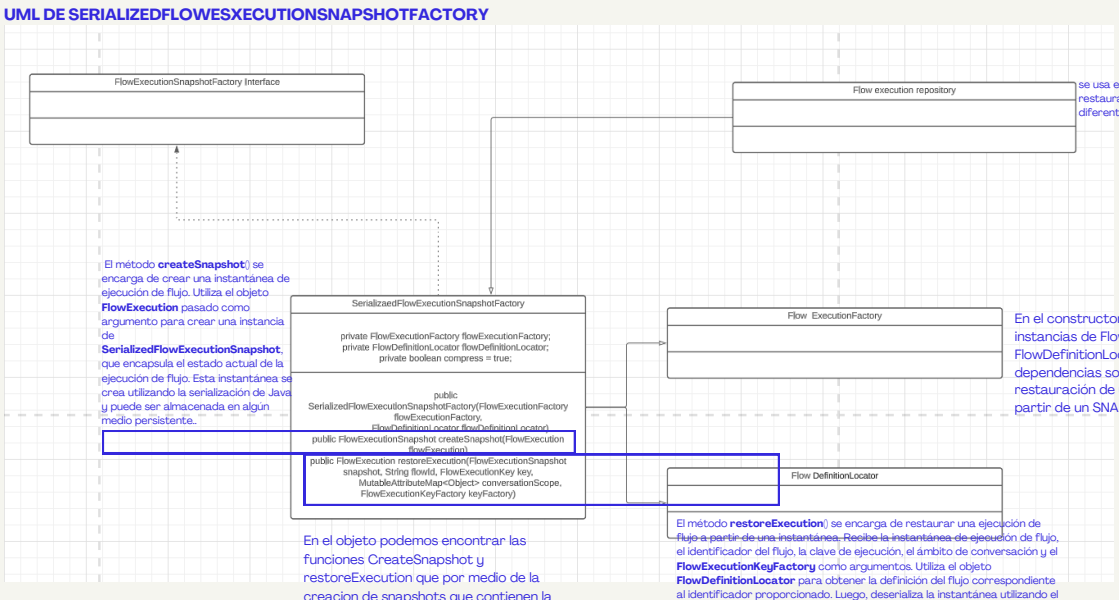


Imagen sacada de https://www.tutorialspoint.com/design_pattern/memento_pattern.htm

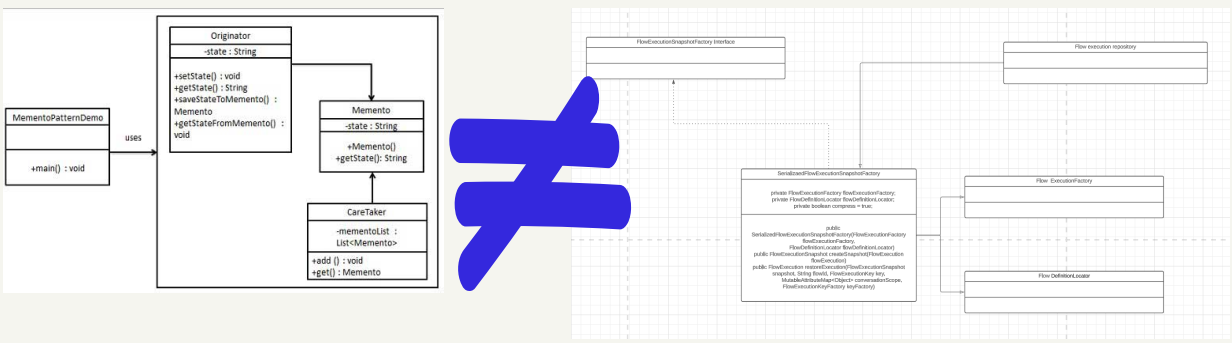
En el objeto podemos encontrar las funciones CreateSnapshot y restoreExecution que por medio de la creacion de snapshots que contienen la información actual de una ejecución, les permite guardar y restaurar el estado anterior en las sesión con un usuario- Permitiendo que recuperen estados previos del trabajo .

UML DE SERIALIZEDFLOWESXECUTIONSNAPSHOTFACTORY



En el objeto podemos encontrar las funciones CreateSnapshot y restoreExecution que por medio de la creacion de snapshots que contienen la información actual de una ejecución, les permite guardar y restaurar el estado anterior en las sesión con un usuario- Permitiendo que recuperen estados previos del trabajo .

Los UML no se parecen



pero...

El UML de **SerializedFlowExecutionSnapshotFactory** no se parece al UML del patrón Memento. Esto se debe a que el archivo **SerializedFlowExecutionSnapshotFactory** en el repositorio de Spring Web Flow no sigue estrictamente la estructura clásica del patrón Memento.

El patr'on memento tienen los objetos del Originator, el Memento y el Caretaker . El originador crea y restaura el objeto el cual seria el papel que cumple el SERIALIZEDFLOWEXECUTIONSNAPSHOTFACTORY, pero las otras clases no se encuentran tan definidas, por lo cual, SERIALIZEDFLOWEXECUTIONSNAPSHOTFACTORY se puede considerar una implementación personalizada de un enfoque similar al patrón Memento, donde se crea una instantánea de la ejecución del flujo y se puede restaurar posteriormente. Pero la estructura y dependencias no cumplen las reglas del patr'om.

Uso del Patrón

El patrón Memento se utiliza en el repositorio de ejecución de Spring Web Flow para permitir que los objetos de flujo se guarden y se restauren en diferentes puntos del flujo de trabajo. Esto se hace a través de la interfaz **FlowExecutionSnapshotFactory**, que define los métodos para crear y restaurar instantáneas de ejecución de flujo. En el repositorio, se pueden encontrar varias implementaciones de **FlowExecutionSnapshotFactory**.

Por ejemplo, la clase **SerializedFlowExecutionSnapshotFactory** utiliza la serialización de Java para crear instantáneas de ejecución de flujo que se pueden almacenar en algún medio persistente. Por otro lado, la clase **FlowExecutionSnapshotFactoryDecorator** proporciona una decoración de instantánea de ejecución de flujo personalizada.

A que se debe el uso

El uso del patrón Memento en el repositorio de ejecución de Spring Web Flow proporciona una forma efectiva y encapsulada de guardar y restaurar el estado de una ejecución de flujo, lo que permite implementar funcionalidades como deshacer/rehacer y checkpoints en el flujo de trabajo.

GUARDAR Y RESTAURAR

Permite guardar el estado actual de una ejecución de flujo y luego restaurarla a ese estado en un momento posterior. Esto es útil en situaciones en las que es necesario deshacer o revertir un cambio en el flujo de trabajo.

ENCAPSULACIÓN

Permite guardar y restaurar el estado de un objeto sin violar el principio de encapsulación. En el contexto de Spring Web Flow, esto significa que el estado interno de una ejecución de flujo se puede guardar y restaurar sin que los objetos externos accedan directamente a su estructura interna. Esto mejora la seguridad y la modularidad del código.

REHACER/DESHACER

Es especialmente útil cuando se necesita implementar la funcionalidad de deshacer/rehacer en un sistema. Al guardar y restaurar estados previos de una ejecución de flujo, se puede permitir que los usuarios deshagan una serie de acciones y luego las vuelvan a realizar si así lo desean.

Desventajas

MEMORIA

El patrón Memento implica almacenar y mantener múltiples instantáneas de ejecución de flujo en memoria. Esto puede resultar en un mayor consumo de memoria, especialmente si se están manejando flujos de trabajo complejos con grandes cantidades de datos.

COMPLEHIDAD Y RENDIMIENTO

Puede aumentar la complejidad del código, ya que requiere la definición de interfaces, clases y métodos adicionales para la creación y restauración de instantáneas de ejecución. Esto puede afectar el rendimiento y la legibilidad del código

GESTIÓN DE CAMBIO

Si se realizan cambios en la estructura interna de los objetos de flujo, puede ser necesario gestionar la compatibilidad hacia atrás de las instantáneas de ejecución existentes. Esto puede generar complicaciones al manejar actualizaciones y versiones futuras de la aplicación.

Otras soluciones...

Otras soluciones que podrían considerarse en el repositorio de ejecución de Spring Web Flow podrian ser:

Serialización directa

En lugar de crear y restaurar instantáneas de ejecución de flujo utilizando el patrón Memento, se podría utilizar la serialización directa de objetos de flujo.

Almacenamiento de cambios incrementales:

se podria para reducir la cantidad de datos que se guardan almacenar solo los cambios incrementales realizados en el flujo a medida que avanza. Esto implica registrar y almacenar solo los eventos o acciones que se producen en el flujo, en lugar de guardar el estado completo del objeto de flujo en cada punto.

Historial de cambios:

Mantener un registro de las acciones que realiza el usuario para navegar adelante o hacia atras.

Valentina Perea Marquez
202013095