Modificación Patrón State

Objetivo

Usando el patrón State, tomando como base el ejemplo del servidor, implementar un apagado seguro. Esta funcionaria de la siguiente manera:

Una vez establecido no permite enviar más mensajes y el servidor no debe apagarse hasta que todos los mensajes que entraron (antes de establecer el estado) sean procesados. Una vez que el servidor se apague cambiará su estado al de Apagado.

Resolución

A continuación, se detallarán los pasos realizados para completar el objetivo de la tarea

1. Creación de la clase con el nuevo estado llamada SafeShutdownServerState la cual cumple con la condición de que no se puede apagar hasta haber procesado todos los mensajes y si esta en el estado SateShutdownServerState no se podrá enviar más mensajes

```
public class SafeShutdownServerState extends AbstractServerState{
    private Thread monitoringThread;

    public SafeShutdownServerState(final Server server) {
        monitoringThread = new Thread(new Runnable() {

            @Override
            public void run() {
                try {
                    System.out.println("Turning off");
                    while (true) {
                        Thread.sleep(1000 * 10);
                        if (server.getMessageProcess().countMessage() > 0) {
                                server.getMessageProcess().start();
                        }
                        else {
                            break;
                        }
                    }
                    System.out.println("Safe Shutdown");
                } catch (Exception e) {
                    System.out.println(e.getMessage());
                }
            }
        });
        monitoringThread.start();
    }
    @Override
    public void handleMessage(Server server, String message) {
        System.out.println("Cant´t process request, Server is state safe shutdown");
    }
}
```

2. Se procedió a modificar el ServerPanel en el source se modifico el método startAction (en donde en la primera condición se agregó que pueda volver a prenderse si está en el estado SafeShutdownServerState) y se agregó el método safeShutdownAction

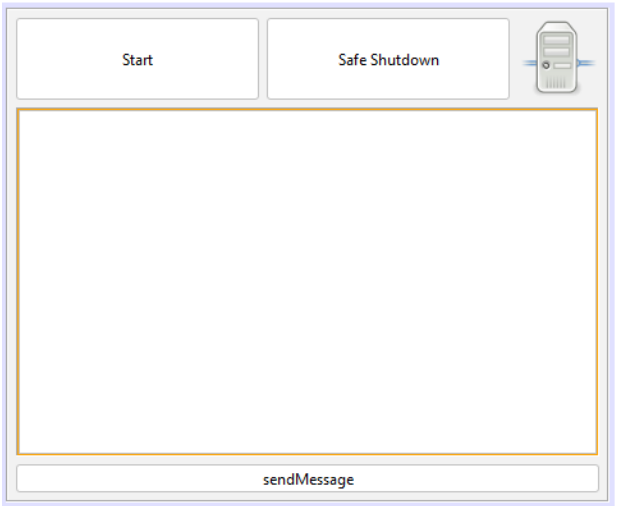
```
    private void startAction(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        AbstractServerState state = server.getState();
        if (state instanceof StopServerState ||
                state instanceof SafeShutdownServerState) {
            btnStart.setText("Stop");
            server.setState(new StartingServerState(server));
        } else {
            if (state instanceof StartingServerState) {
                server.setState(new StopServerState(server));
            } else {
                btnStart.setText("Start");
                server.setState(new StopServerState(server));
            }
        }

    }
```

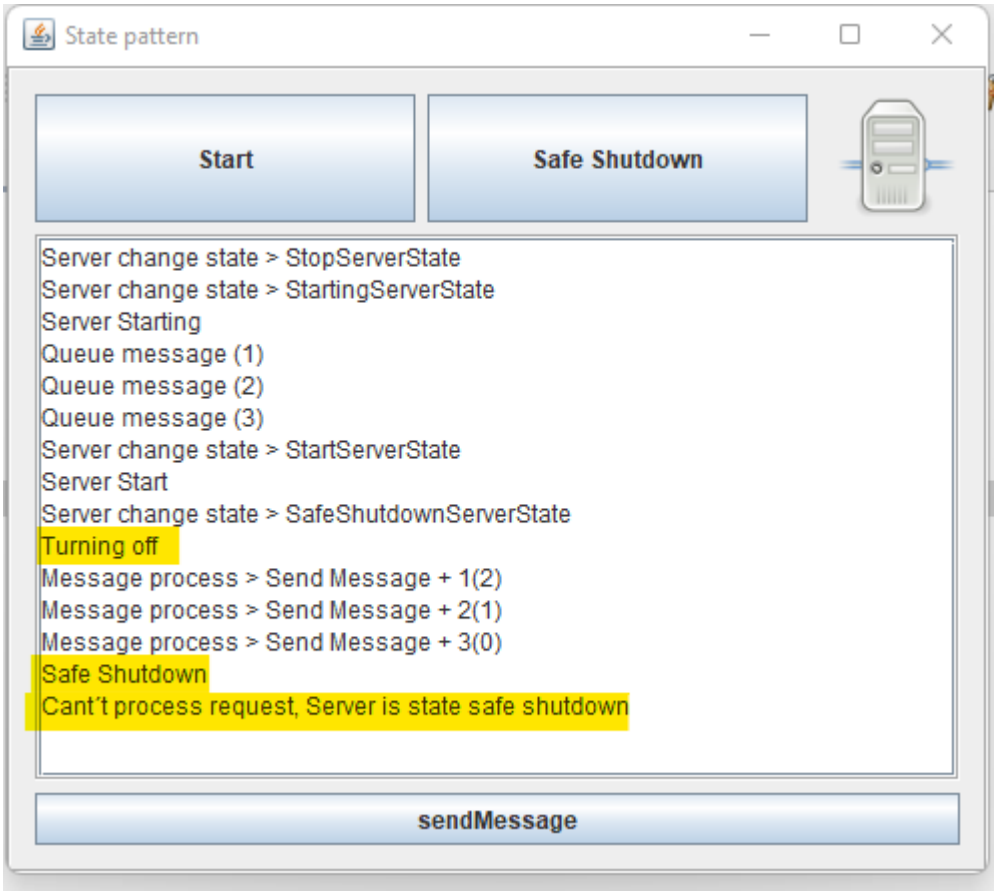
```
    private void safeShutdownAction(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        AbstractServerState state = server.getState();
        if (state instanceof StartingServerState) {
            System.out.println("Server is starting, "
                    + "cannot change state");
        }   else if (state instanceof StopServerState){
            System.out.println("Server is stop");
        }
        else{
            btnStart.setText("Start");
            server.setState(new SafeShutdownServerState(server));
        }

    }
```

3. Se modifico la pantalla y quedo de la siguiente manera



4. Al momento de correr el programa se muestra lo pedido en la tarea (como se puede observar no se apaga hasta completar los tres mensajes que tenía en cola y cuando ya entro al estado no se puede enviar más mensajes)



5. El código fuente se encuentra en la siguiente dirección https://github.com/cristianmoreno1986/patrones