# Log 8/6/2024

Agregada clase cortadora

Clase cortadora se convierte en extensión de estación

En clase Cortadora agregado método operar en donde ya puede esperar x cantidad de segundos

Agregada clase Listener, implementa thread y es para escuchar cuando la cortadora está ocupada

# Log 9/6/2024

1. **nbproject/private/private.xml**
   1. Adiciones: 13 líneas
      1. Agregado un nuevo archivo XML para la configuración privada del proyecto en NetBeans.
      2. Registro de los archivos abiertos: Pieza.java, Transportador.java, Estacion.java, Gestor.java, y App.java.
   2. Se añadió una línea: <file>file:/D:/Projects/FMS\_Remaster/src/Simulador/Estacion.java</file>
2. **src/Simulador/Estacion.java**
   1. Adiciones: 7 líneas
      1. Nueva variable estado añadida a la clase Estación.
      2. Inicialización de estado a true en el constructor.
      3. Método operar añadido para cambiar el estado de la estación.
3. **src/Simulador/EstacionesPanel.form**
   1. Adiciones: 129 líneas
      1. Archivo XML generado por NetBeans para la interfaz gráfica del panel de estaciones.
      2. Definición de propiedades de la ventana, colores, y disposición de componentes.
      3. Creación de paneles y componentes como canvas, estCortadoraPanel, jLabel1, y btnCortando.
4. **src/Simulador/EstacionesPanel.java**
   1. Adiciones: 141 líneas
      1. Nueva clase EstacionesPanel creada como un JFrame para la interfaz gráfica de estaciones.
      2. Métodos para inicializar componentes y manejar eventos de la interfaz.
      3. Métodos setEstadoCortadora para actualizar el estado visual de la cortadora.
5. **src/Simulador/Gestor.java**
   1. Adiciones: 32 líneas
   2. Eliminaciones: 7 líneas
      1. Actualización en la clase Gestor para integrar EstacionesPanel.
      2. Inicialización de EstacionesPanel en el método empezar.
      3. Llamada al método operar de Estacion y actualización del estado de la cortadora en el panel.
   3. Se añadieron 27 líneas y se eliminaron 5 líneas.
      1. Se añadió una variable estática cortadoraEstado para rastrear el estado de la máquina de corte.
      2. Nuevos métodos getCortadoraEstado y setCortadoraEstado para acceder y modificar el cortadoraEstado.
      3. Se actualizó el método empezar para llamar a generarListener.
      4. Se añadió un método generarListener para iniciar un nuevo hilo Listener que monitorea el cortadoraEstado.
6. **src/Simulador/Cortadora.java**
   1. Adiciones: 34 líneas
      1. Esta nueva clase Java Cortadora extiende la clase Estacion.
      2. Contiene métodos para operar el proceso de corte, incluyendo el inicio de la operación, la espera por una duración establecida y el registro del inicio y fin de la operación.
      3. Se añadió un método privado waitInSeconds para manejar la simulación de la espera.
7. **src/Simulador/Listener.java**
   1. Adiciones: 150 líneas
      1. Esta nueva clase Java Listener implementa la interfaz Runnable.
      2. El método run verifica continuamente el cortadoraEstado y registra mensajes en consecuencia.
      3. Contiene un bloque de código comentado que parece ser parte de una simulación más extensa que implica piezas procesadas y movidas entre diferentes estaciones.
8. **Plan.docx**
   1. Se añadió un archivo binario. No se muestran cambios específicos, pero normalmente esto indica un documento actualizado o recién añadido.

# Log 10/6/2024

1. **App.java**:
   * El archivo App.java ahora incluye una instancia de Gestor, inicializándola y comenzándola en el método main.
2. **Cortadora.java**:
   * La clase Cortadora ahora extiende Estacion y anula los métodos operar() y run().
   * waitInSeconds se ha movido a una clase de utilidades separada (Herramientas).
   * La clase implementa un comportamiento similar a un hilo con run().
3. **Dobladora.java**:
   * Se ha creado una nueva clase Dobladora que extiende Estacion.
   * Implementa el método run() para un comportamiento similar a un hilo y anula el método operar().
4. **Estacion.java**:
   * Estacion ahora implementa Runnable y tiene un atributo flag para controlar el estado.
   * Los métodos run() y operar() se han actualizado en consecuencia.
5. **EstacionesPanel.form** y **EstacionesPanel.java**:
   * Actualizaciones de interfaz de usuario para agregar un panel y un botón para Dobladora.
   * Se agregaron nuevos métodos setEstadoDobladora y btnDoblarActionPerformed para manejar el estado y la acción del botón de Dobladora.
6. **Gestor.java**:
   * Actualizaciones para administrar múltiples instancias de Estacion (Cortadora y Dobladora).
   * Nuevos métodos generarEstacionesHilos y set/getDobladora.
   * Inicia hilos para cada Estacion.
7. **Herramientas.java**:
   * Se creó una nueva clase de utilidades Herramientas con un método estático waitInSeconds.
8. **nbproject/private/private.xml**
   * Actualización de rutas de archivos en el archivo XML, probablemente para reflejar cambios en la estructura del proyecto.
9. **src/App/App.java**
   * Ajustes para iniciar y administrar las estaciones y piezas en el simulador.
10. **src/Simulador/Cortadora.java**
    * Mejoras en la lógica de operación de la cortadora, incluyendo el manejo de la bandera de estado y la sincronización de hilos.
11. **src/Simulador/Dobladora.java**
    * Creación de una nueva clase para la dobladora, similar a la cortadora pero con operaciones específicas.
12. **src/Simulador/Estacion.java**
    * Ajustes en la clase base de las estaciones para mejorar la gestión de hilos y la operación de las piezas.
13. **src/Simulador/EstacionesPanel.java**
    * Cambios en la interfaz de usuario para reflejar el estado de las estaciones y permitir la interacción con las mismas.
14. **src/Simulador/Gestor.java**
    * Actualización de métodos para crear y gestionar las estaciones, así como para iniciar los hilos correspondientes.
15. **src/Simulador/Herramientas.java**
    * Adición de un método para esperar un número específico de segundos, probablemente utilizado para simular el tiempo de operación de las estaciones.
16. **src/Simulador/Pieza.java**
    * Agregado de un método toString para facilitar la visualización de información sobre las piezas.

# Log 10/6/2024

1. **src/App/App.java**:
   1. Estas adiciones incluyen la creación de un nuevo hilo para el gestor, la espera de 100 milisegundos antes de continuar, y la impresión de un mensaje indicando que el gestor ha sido iniciado. Finalmente, se llama al método empezar() del gestor para comenzar la simulación.**src/Simulador/Cortadora.java**:
2. **src/Simulador/EstacionesPanel.form**:
   1. Se realizaron ajustes en el diseño del panel de estaciones, con 31 adiciones y 6 eliminaciones de líneas de código para reflejar estos cambios.
3. **src/Simulador/EstacionesPanel.java**:
   1. Se realizaron modificaciones significativas en esta clase, con 74 adiciones y 26 eliminaciones de líneas de código. Los cambios incluyen mejoras en la funcionalidad y el diseño del panel de estaciones.
4. **src/Simulador/Gestor.java**:
   1. Se realizaron cambios importantes en esta clase, con 100 adiciones y ninguna eliminación de líneas de código. Los cambios incluyen la adición de funcionalidad para operar la siguiente pieza en el almacén, así como la creación de las estaciones y la generación de las piezas para las pruebas.

Interfaz de usuario gráfica

Descripción generada automáticamente

 **Thread Management**:

* You are implementing a multi-threaded environment where each station (Cortadora, Dobladora, etc.) runs in its own thread.
* Ensure thread-safety when accessing shared resources or flags.

 **UI Integration**:

* Buttons and panels in EstacionesPanel reflect the state and control actions of each station.
* Handle UI updates on the Event Dispatch Thread (EDT) to prevent UI freezes.

 **Error Handling**:

* Ensure robust error handling around thread sleeps and interruption to maintain the application's stability.

 **Testing**:

* Thoroughly test the multi-threaded interactions to avoid race conditions and deadlocks.
* Validate UI changes and interactions through integration tests.