PRUEBAS MEZCLADOR DE PINTURA

David Gómez Rivas

PROGRAMACIÓN DE SERVICIOS Y PROCESOS 2ºDAM
Las Naves Salesianos Alcalá de Henares

Contenido

Explicación de las clases 2
MezcladorDePintura: 2
Persona: 3
ColorDeposito: 4
Pruebas: 5
Utilizando los tiempos indicados: 5
Poniendo los tiempos al mínimo: 5

Explicación de las clases

MezcladorDePintura:

Actúa como el punto de entrada principal y coordina la simulación del mezclado de colores. Aquí está lo que hace, de manera resumida:

Define Depósitos de Colores: Crea tres instancias de la clase ColorDeposito, representando los colores primarios cian, magenta y amarillo.

Crea Hilos para Personas: Inicializa tres hilos, cada uno asociado con una instancia de la clase Persona. Cada Persona tiene asignada la tarea de mezclar dos colores primarios para crear un color secundario (rojo, azul, verde).

Inicia los Hilos: Pone en marcha los tres hilos. Cada hilo entra en un bucle donde intenta acceder a los depósitos de color asignados para mezclar un color secundario, lo hace durante un tiempo aleatorio, y luego espera un tiempo antes de intentar mezclar de nuevo.

La clase MezcladorDePintura es esencialmente la estructura que configura y lanza el proceso de simulación de mezcla de pinturas, utilizando la concurrencia (hilos) en Java.

Persona:

representa a un individuo que realiza la tarea de mezclar colores en la simulación. Aquí está lo que hace, de manera resumida:

Implementa Runnable: Al implementar la interfaz Runnable, esta clase se puede utilizar para crear hilos. Cada instancia de Persona representa un hilo diferente en la simulación.

Almacena Colores Primarios y Secundario: Cada Persona tiene dos colores primarios (primerColor y segundoColor) y un color secundario (colorSecundario) asignados. Estos colores representan los depósitos de pintura que necesita mezclar y el resultado de esa mezcla, respectivamente.

Define un Bucle Infinito en run(): Dentro del método run(), que es la entrada del hilo, se ejecuta un bucle infinito donde la persona intenta mezclar los colores.

Sincronización para Acceso a los Depósitos de Color: Usa bloques synchronized para asegurar acceso exclusivo a los depósitos de color mientras mezcla.

Simula la Mezcla de Pinturas: Dentro de los bloques sincronizados, la Persona "usa" los depósitos de color (simulado por una espera aleatoria) para mezclar el color secundario.

Espera Aleatoria entre Mezclas: Después de mezclar los colores, el hilo espera un tiempo aleatorio antes de intentar mezclar nuevamente, simulando un descanso entre mezclas.

En resumen, la clase Persona modela a un trabajador en la simulación que mezcla colores, gestionando el acceso concurrente a recursos compartidos (los depósitos de pintura) y realizando tareas en un ciclo continuo.

ColorDeposito:

representa un depósito individual de pintura de un color específico. He aquí un resumen de sus funciones principales:

Almacena el Nombre del Color: Tiene una variable nombre que almacena el nombre del color de pintura en el depósito, como "Cian", "Magenta" o "Amarillo".

Constructor para Inicializar el Nombre: El constructor de la clase toma un String como parámetro y lo asigna a la variable nombre, inicializando así el depósito con un color específico.

Método para Obtener el Nombre del Color: Proporciona un método getNombre() para acceder al nombre del color del depósito.

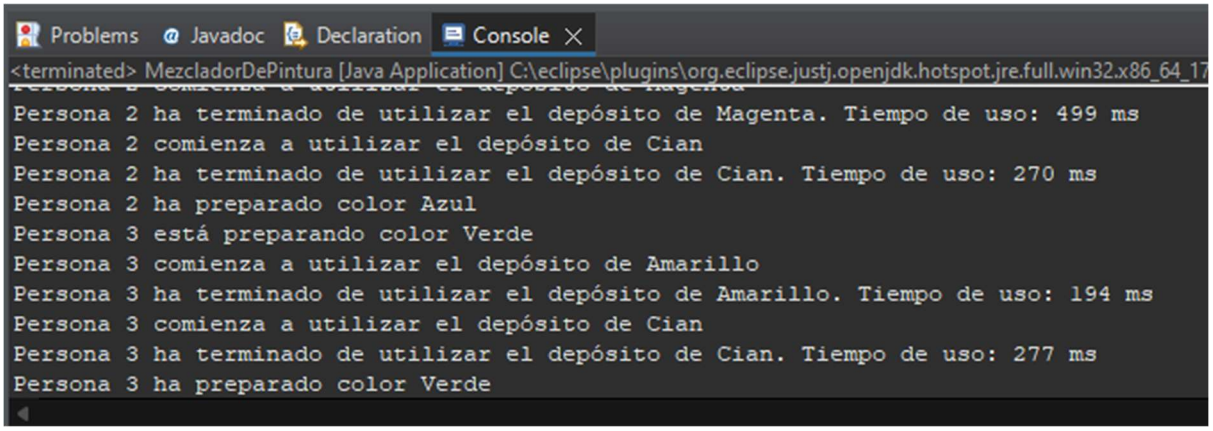
Método Sincronizado para Usar el Depósito: Incluye un método sincronizado usarDeposito() que simula el uso del depósito de pintura. Este método es sincronizado para asegurar que solo un hilo (o Persona) pueda usar el depósito a la vez, evitando conflictos en el acceso concurrente.

Simula el Tiempo de Uso del Depósito: Dentro de usarDeposito(), se usa Thread.sleep() para simular el tiempo que una persona tarda en usar el depósito. Este tiempo es aleatorio entre 100 y 500 milisegundos.

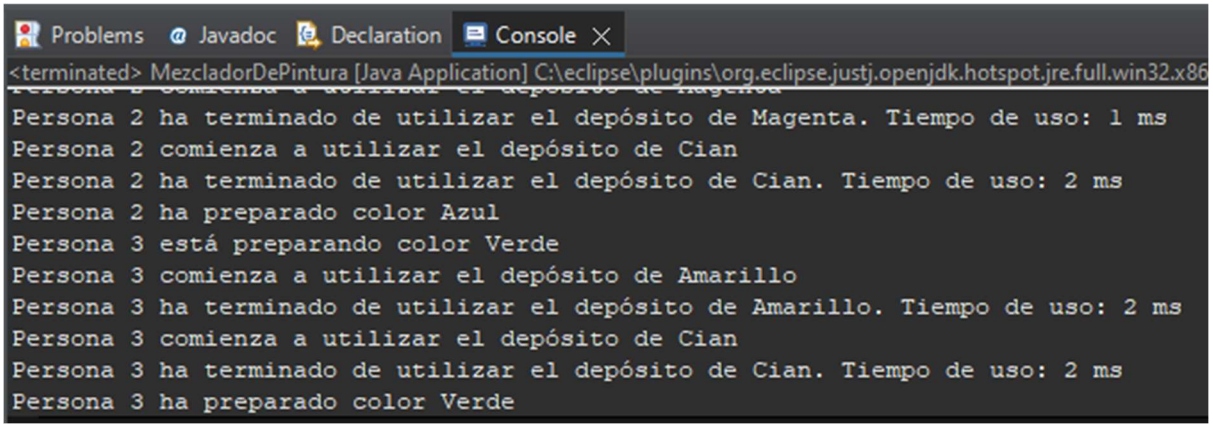
En resumen, la clase ColorDeposito modela un depósito de un color específico de pintura, controlando el acceso a este recurso compartido de manera sincronizada para evitar conflictos en una simulación de mezcla de colores concurrente.

Pruebas:

Utilizando los tiempos indicados:



Poniendo los tiempos al mínimo:



Funcionan perfectamente durante el tiempo que quieras