PATRÓN DE DISEÑO CALLBACK

Andreu orenga ramon

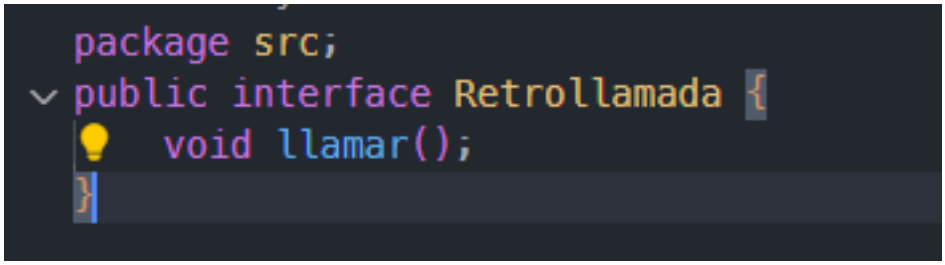
ENTORNOS DE DESARROLLO [Dirección de la compañía]

índice

Ejemplo Patrón Callback

1. Definir la interfaz de Callback

Primero, se define una interfaz Retrollamada que contendrá el método llamar(). Este método será llamado una vez que una operación específica haya terminado.



```
package src;
public interface Retrollamada {
    void llamar();
}
```

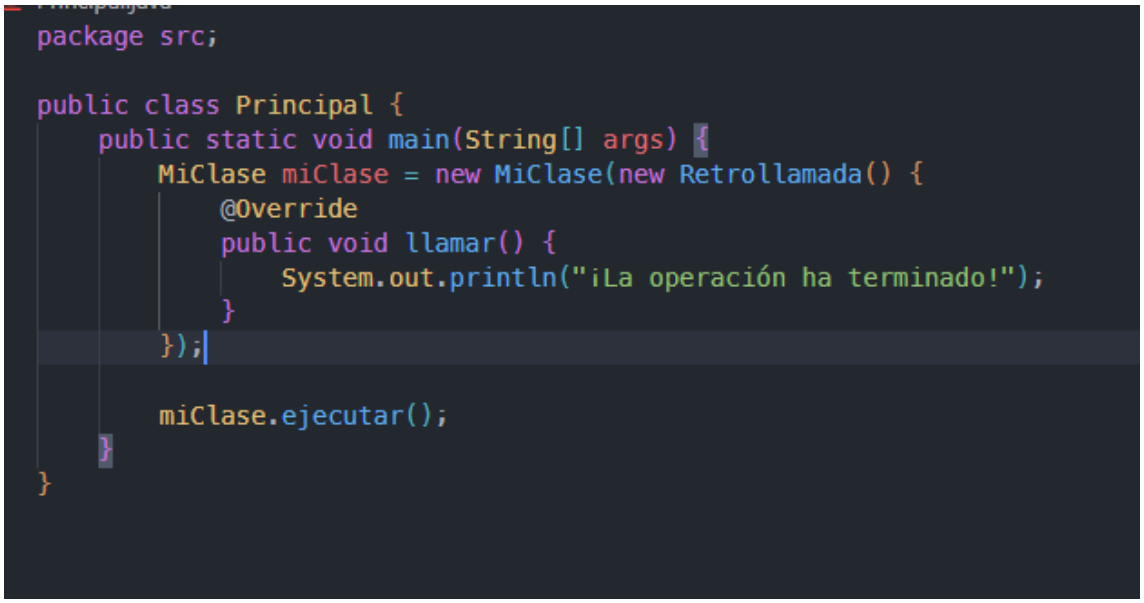
2. Crear una clase que usa la interfaz de Callback

Luego, se crea una clase que utilizará esta interfaz. MiClase realiza alguna operación que puede llevar tiempo y utiliza el método llamar de la interfaz Retrollamada para notificar cuando ha finalizado la operación.

```
package src;
public class MiClase {
    private Retrollamada retrollamada;

    public MiClase(Retrollamada retrollamada) {
        this.retrollamada = retrollamada;
    }

    public void ejecutar() {
        // Simula una operación que toma tiempo
        try {
            Thread.sleep(millis:1000); // Simula tiempo de espera
        } catch (InterruptedException e) {
            Thread.currentThread().interrupt();
        }

        // Llama al método de callback una vez completada la operación
        retrollamada.llamar();
    }
}
```

3. Uso del patrón Callback en la clase principal

Finalmente, en la clase principal, se instancia 'MiClase' pasándole una implementación de la interfaz 'Retrollamada', que en este caso, imprime un mensaje al ser llamada.



```java
package src;

public class Principal {
    public static void main(String[] args) {
        MiClase miClase = new MiClase(new Retrollamada() {
            @Override
            public void llamar() {
                System.out.println("¡La operación ha terminado!");
            }
        });

        miClase.ejecutar();
    }
}
```

Funcionamiento del ejemplo

Definición de Retrollamada: Esta interfaz es crucial porque define cómo se comunicarán las clases de manera asíncrona.

MiClase implementa la operación: Esta clase realiza la operación que puede demorar y luego llama al método llamar() del objeto Retrollamada que se le pasa en el constructor. Esto desacopla el tiempo de ejecución de la operación de la respuesta a esa operación.

Uso en Principal: Aquí se muestra cómo instanciar MiClase y cómo manejar el callback sin bloquear otras operaciones, permitiendo que el hilo principal continúe con otras tareas o maneje otros eventos.

Este ejemplo ilustra cómo el patrón Callback permite una mayor flexibilidad y mejor respuesta en aplicaciones donde no se desea bloquear el flujo principal de ejecución, como en interfaces de usuario o en operaciones de I/O.

Código del ejemplo adjunto en la carpeta src del proyecto