WENDY JULIETH GUIO QUINTERO

uninpahu  educación

singleton

**1. Introducción**

Este documento describe el proyecto denominado **Ejerciciospatronessingleton**, proporcionando detalles sobre su propósito, características y estructura. Se incluyen aspectos técnicos sobre su implementación y configuración.

**2. Descripción del Proyecto**

El proyecto **Ejercicios patrones singleton** está enfocado en la implementación del patrón de diseño Singleton en Java. Su objetivo es demostrar el uso y la utilidad de este patrón en la gestión de instancias únicas dentro de una aplicación.

**3. Características**

* Implementación del patrón de diseño **Singleton**.
* Uso de **Java** como lenguaje de programación principal.
* Uso de **Ant** como herramienta de construcción, con un archivo build.xml configurado.
* Estructura basada en **NetBeans** para facilitar la compilación y ejecución.
* Posible integración con pruebas unitarias mediante JUnit.

│-- manifest.mf # Archivo de manifiesto del proyecto

**4. Archivo build.xml**

Este archivo define el proceso de compilación, ejecución y generación de JAR del proyecto. Contiene tareas predefinidas para la limpieza, construcción y prueba del código.

**5. Archivo manifest.mf**

El archivo de manifiesto contiene información sobre el paquete, versión y configuración del JAR generado.

**6. Requisitos Técnicos**

**7. Software Requerido**

* **JDK 8 o superior**
* **NetBeans o cualquier IDE compatible con Ant**
* **Apache Ant** (para compilación automatizada)

**8. Configuración**

Para ejecutar el proyecto, se deben seguir los siguientes pasos:

1. Abrir el proyecto en NetBeans o en un IDE compatible.
2. Asegurarse de que **Ant** está instalado y configurado en la máquina.
3. Utilizar el comando:
4. ant clean build

para compilar el proyecto.

1. Para ejecutar la aplicación, usar:
2. ant run

**9. Patrones de Diseño Utilizados**

Este proyecto está basado en el **patrón Singleton**, que se utiliza para restringir la creación de instancias de una clase a una única instancia. Se aplica en casos donde se requiere una única fuente de acceso a recursos compartidos.

**10. Conclusiones**

Este proyecto permite comprender la implementación y uso del patrón Singleton en Java. También proporciona una estructura modular que facilita su mantenimiento y escalabilidad. La configuración con Ant permite una construcción eficiente y automatizada del código.