Desarrollo e implantación de sistemas de software

Activity 2 - Patrones Estructurales Proxy y Decorator

Andrés Daniel Martínez Bermúdez - A00227463

Grupo 103

10 Apr 2025

Instrucciones

Visita la página oficial de Refactoring.Guru:

- Patrón Decorator
- Patrón Proxy

Lee la **intención**, **estructura**, **ejemplo del mundo real**, y **ejemplo de código** de ambos patrones.

- ¿Cuál es la principal diferencia entre el patrón **Decorator** y el patrón **Proxy**?
- ¿En qué tipo de escenarios usarías cada uno?

En tus propias palabras, responde brevemente:

Actividad

- **¿Cuál es la principal diferencia entre el patrón Decorator y el patrón Proxy?**

El patrón Decorator se encarga de modificar un objeto en el runtime sin modificar su lógica detrás. Es decir, es capaz de modificar su comportamiento sin directamente tener que entrar a su código y nos ayuda a generar proyectos más concisos que evitan la redundancia de dependencias entre las diferentes clases en un sistema.

Por otra parte, el patrón Proxy se encarga de controlar el funcionamiento de un objeto ya que logra que el usuario interactúe con el Proxy en lugar del Objeto base y de esta manera podemos implementar diferentes medidas que nos ayudan a controlar el acceso al mismo sin tener que modificar la lógica del objeto en cuestión.

En otras palabras, el patrón Decorador agrega funcionalidades a un objeto, mientras que el Proxy lo controla y limita su acceso en ciertas situaciones.

- **¿En qué tipo de escenarios usarías cada uno?**

El patrón Decorator lo utilizaría para implementar funcionalidades nuevas que me permitan manejar la lógica de un objeto sin alterar su lógica base. En ese sentido, lo utilizaría para testear funciones bajo diferentes circunstancias con el objetivo de crear código con una mayor resistencia a bugs y que sea capaz de adaptarse a múltiples circunstancias.

Por otra parte, utilizará el patrón Proxy para poder implementar medidas de seguridad hacia un sistema en común, esto con el fin de aligerar la carga y/o evitar modificaciones inesperadas al objeto original y proteger los datos que pudiese tener en un principio.

Referencias Bibliográficas

- Decorator Pattern Explained: Basics to Advanced. (2025, 4 febrero). https://daily.dev/blog/decorator-pattern-explained-basics-to-advanced
- Differences between Proxy and Decorator Pattern. (s. f.). Stack Overflow. https://stackoverflow.com/questions/18618779/differences-between-proxy-and-decorator-pattern#:~:text=A%20decorator%20adds%20one%20or,controls%20access%20to%20an%20object.
- Derek Banas. (2012, 19 octubre). Proxy design pattern tutorial [Vídeo]. YouTube. https://www.youtube.com/watch?v=cHg5bWW4nUI
- ByteVigor. (2024, 16 julio). Decorator Design Pattern: Easy Guide for Beginners [Vídeo]. YouTube. https://www.youtube.com/watch?v=P-9fXUbQIYw
- Refactoring.Guru. (2025, 1 enero). Proxy. https://refactoring.guru/design-patterns/proxy
- Refactoring.Guru. (2025a, enero 1). Decorator. https://refactoring.guru/design-patterns/decorator