SDET Course

Design Patterns - Proxy

3 Types of Design Patterns

- Creational
  - Singleton
  - Builder
  - Prototype
  - Factory Method
  - Abstract Factory
- Structural
  - Adapter
  - Composite
  - **Proxy**
  - Flyweight
  - Bridge
  - Facade
  - Decorator
- Behavioral
  - Strategy
  - Observer
  - Command
  - Memento
  - State
  - Template Method
  - Mediator
  - Chain of Responsibility
  - Interpreter
  - Visitor
  - Iterator

Agenda

- Description
- Diagram
- Code sample (Java)
- Use cases

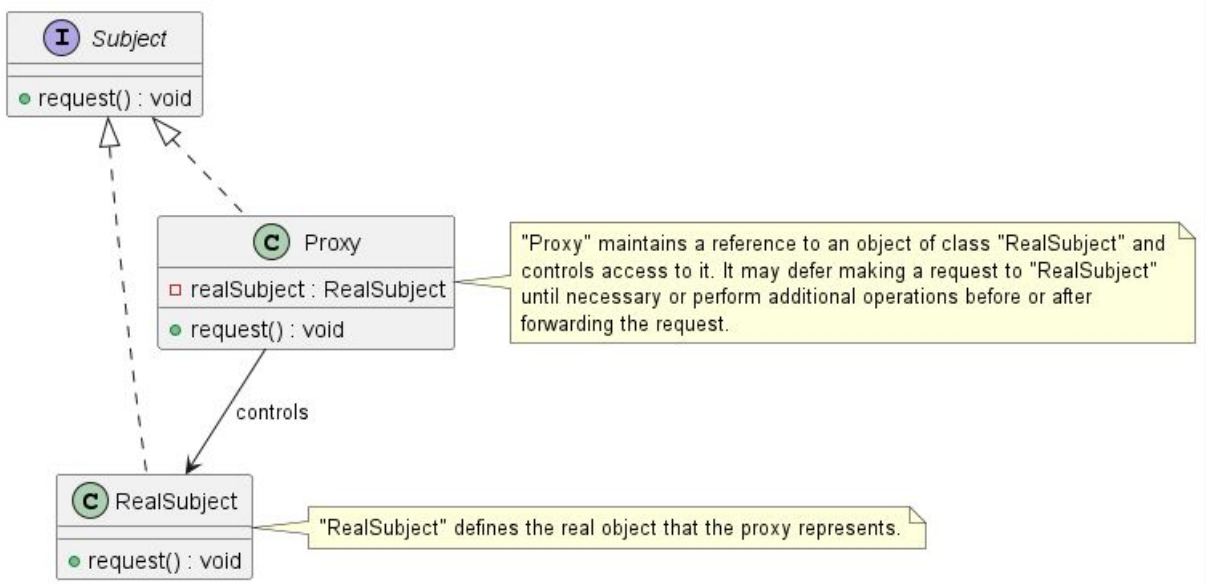
VERISOFT

Descript
on

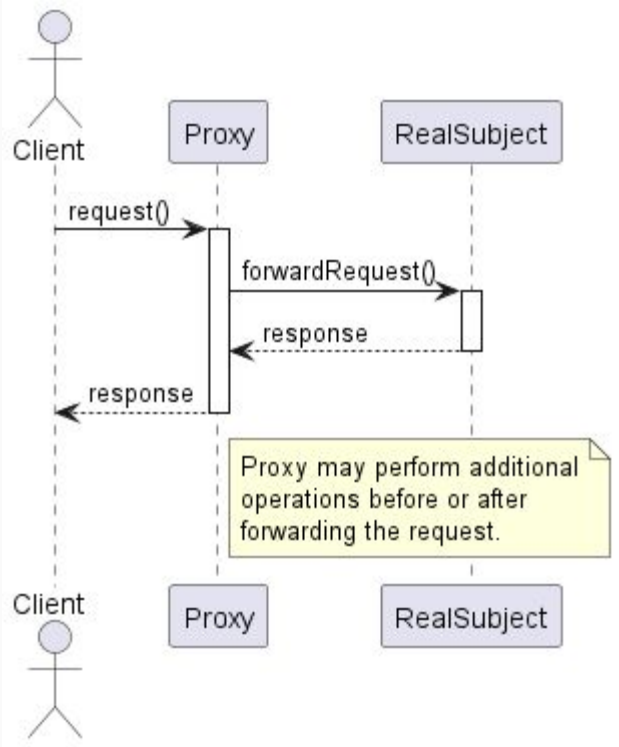
Description

The Proxy design pattern is a structural pattern that provides a surrogate or placeholder for another object to control access to it. Acting as an intermediary, the Proxy pattern allows for the implementation of additional functionality such as logging, caching, or access control without altering the original object's code. This pattern is beneficial in scenarios where the creation or usage of the real object is expensive or resource-intensive, as the proxy can defer the object's creation until it is actually needed. Additionally, the Proxy pattern can be used for implementing lazy initialization, access control, or providing a level of indirection to facilitate remote communication or asynchronous behavior.

Class Diagram



Sequence Diagram



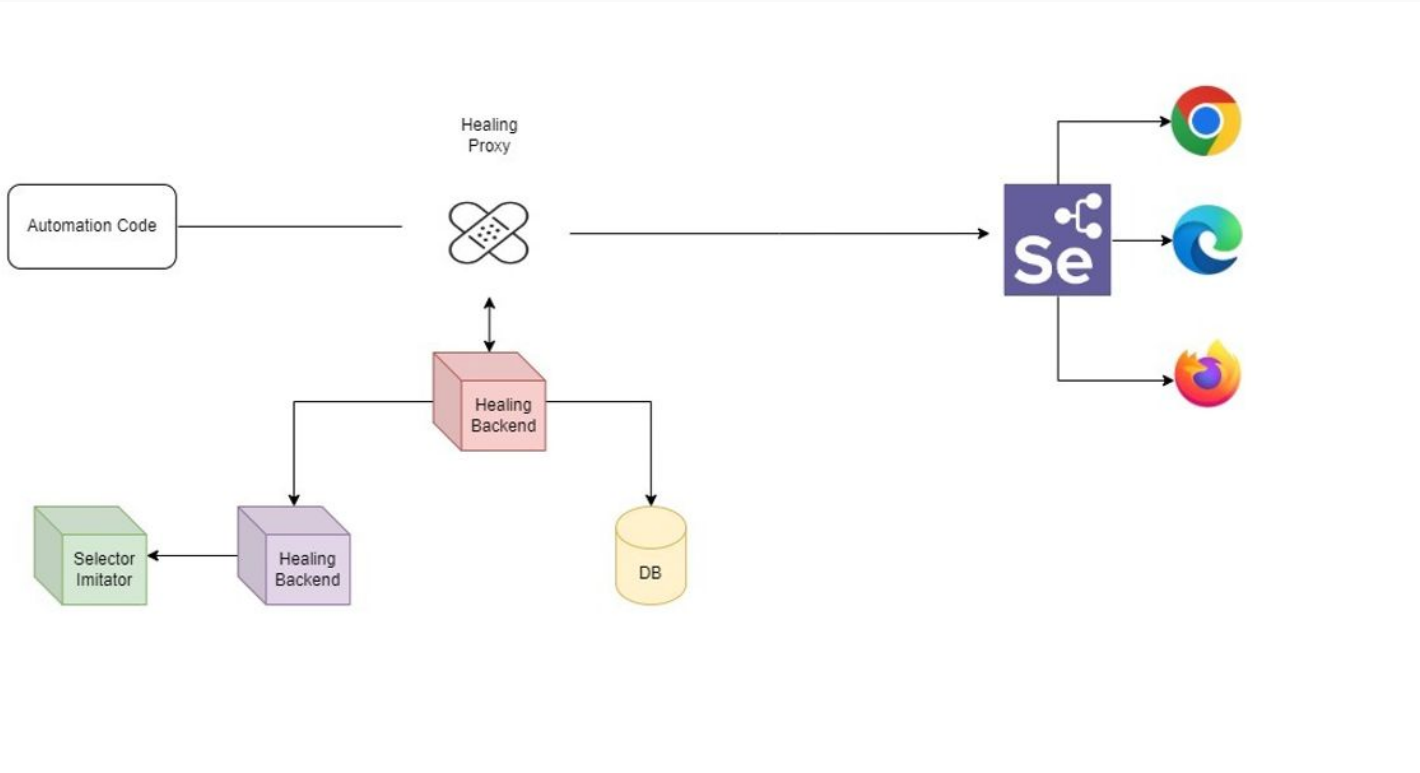
Code Sample

General use cases

- Access Control
- Remote Object Access
- Content Filtering
- API Gateway (APIGEE / WSO2)

Use Cases in Test Automation

- Healenium



Happy Coding