Chain of Responsibility ID

Student:

Izabela Kuźniar

Teacher:

Andrea Corradini

Course:

Software Design Patterns

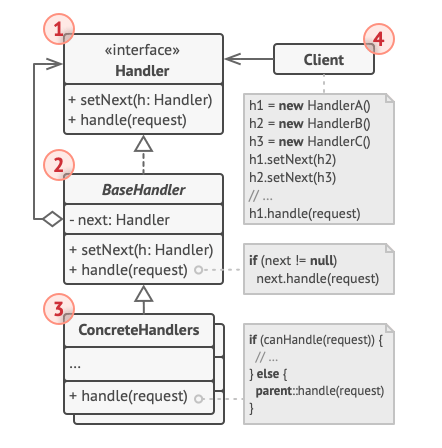
# Name and category

Chain of Responsibility, also known as Chain of Command, is a behavioral pattern.

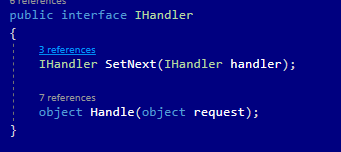
# Intent:

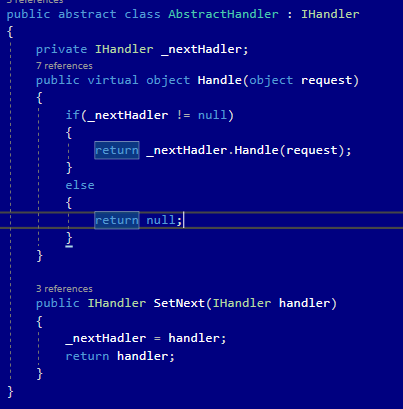
Chain of Responsibility gives the ability to pass the request along a chain of handlers.

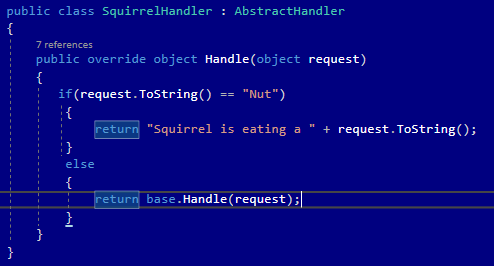
# Structure as a UML class diagram

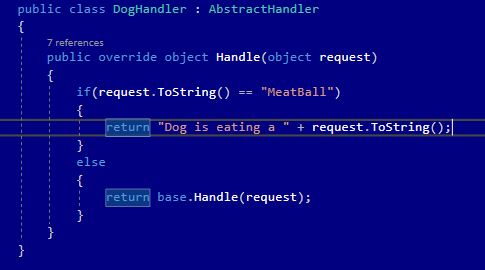


# Implementation:











# Consequences:

Benefits:

* Ability to control the order of request handling.
* Single Responsibility Principle. Ability to decouple classes that invoke operation s from classes that perform operations.
* Open/Closed Principle. Ability to introduce new handlers into the app without breaking the existing client code.

Drawbacks:

* Some request may end up unhandled.

# Known uses

* Used in Servlet Filters.

# Related patterns

1. Chain of Responsibility, Command, Mediator and Observer address various ways of connecting senders and receivers of requests:
   1. Chain of Responsibility passes a request sequentially along a dynamic chain of potential receivers until one of them handles it.
   2. Command establishes unidirectional connections between senders and receivers.
   3. Mediator eliminates direct connections between senders and receivers, forcing them to communicate indirectly via a mediator object.
   4. Observer lets receivers dynamically subscribe to and unsubscribe from receiving requests.
2. Handlers in Chain of Responsibility can be implemented as Command. In this case, you can execute a lot of different operations over the same context object, represented by a request.
3. Chain of Responsibility is often used in conjunction with Composite. When a leaf gets a request, it may pass it further to the parent components down to the root.
4. Chain of Responsibility and Decorator have very similar structures. Both patterns rely on recursive composition to pass the execution through a series of objects. However, there are several crucial differences.