SDET Course

Design Patterns - Iterator

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

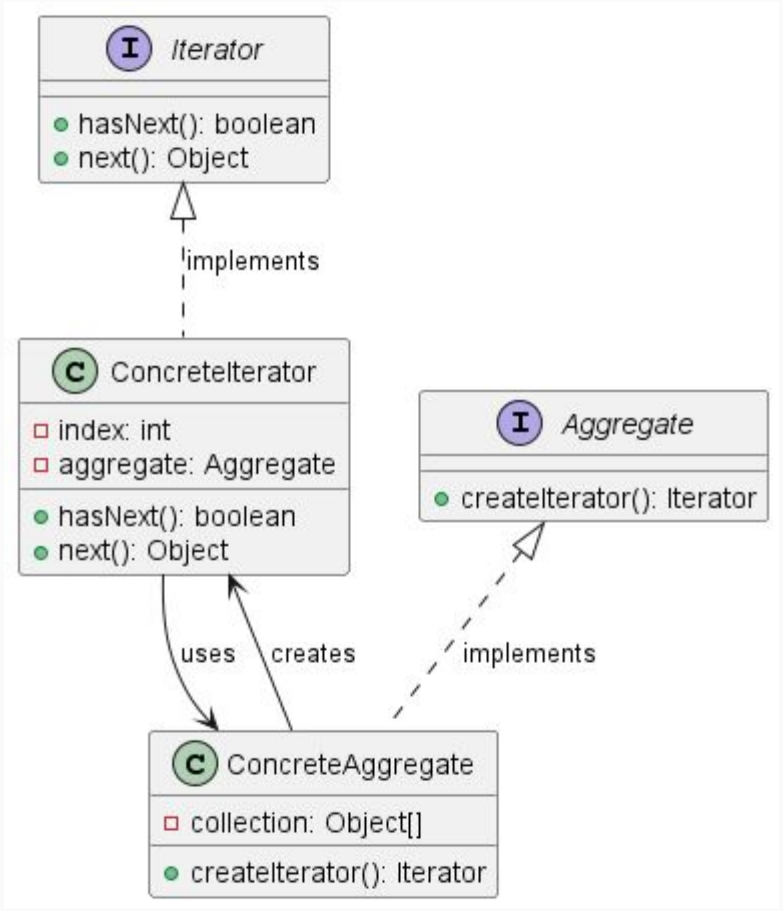
Descript
on

Introduction - Read Blog Posts

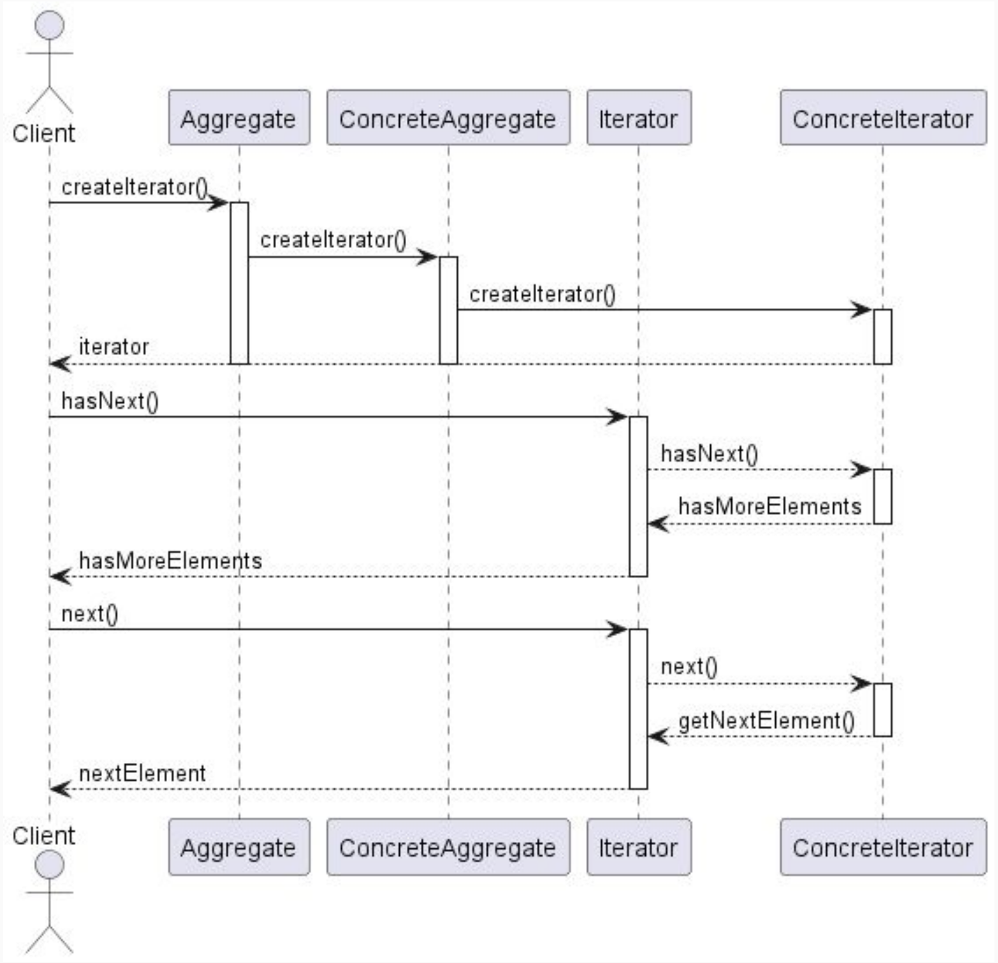
Description

The Iterator design pattern is a behavioral design pattern that provides a way to access elements of an aggregate object sequentially without exposing its underlying representation. It defines a common interface for iterating over various types of collections, such as lists, arrays, trees, or other complex data structures, without needing to know the specific details of each structure. This pattern decouples the iteration logic from the aggregate object, promoting code reusability and flexibility. By encapsulating iteration within a separate object, it allows for multiple iterators to traverse the same collection concurrently or in different ways, facilitating efficient and versatile traversal of data structures.

Class Diagram



Sequence Diagram



Code Sample

Use cases

- General
  - Menu Iteration in GUIs
  - Database Query Results
  - Tree Traversal Algorithms
  - Collections Frameworks

- In Test Automation
  - Iterate through users
  - Iterate through table entries
  - Iterate through SQL results

Verisoft

Happy Coding