Welcome!

- We'll start in a moment :)
- We may record tonight's event and plan to take screenshots for social media.
  - ***If you want to remain anonymous***, change your name & keep video off.
- We'll introduce the hosts and break in-between for Q/A.
- We will make some time for Q&A at the end of the presentation as well.
- Online event best practices:
  - Mute yourself when you aren't talking.
  - We want the session to be interactive.
  - Feel free to unmute and ask questions in the middle of the presentation.
  - Turn on your video if you feel comfortable

Check out:

- Technical Tracks
- Check our Digital Events
- Get updates – join the Digital mailing list
- Give us your feedback – take the Survey

WOMEN WHO CODE

WWCode Digital + **Backend**

Backend Study Group

June 3, 2021

WOMEN WHO CODE

Introduction & Agenda

- Welcome from WWCode!
- Our mission: Inspiring women to excel in technology careers.
- Our vision: A world where women are representative as technical executives, founders, VCs, board members and software engineers.

Prachi Shah
Senior Software Engineer @ Metromile

- **Software Design Patterns [Part 3 of 5]**
  - Creational Design Patterns [4/22]
  - Structural Design Patterns [5/20]
  - **Behavioral Design Patterns [6/3]**
  - Anti-patterns [6/17]
  - Interview Questions [7/1]
- What is Backend Engineering?
- Software Design

WOMEN WHO CODE

Backend Engineering

- What is Backend Engineering?
- Design, build and maintain server-side web applications.
- Concepts: Client-server architecture, API, micro-service, database engineering, distributed systems, storage, performance, deployment, availability, monitoring, etc.

Software Design

- Defining the architecture, modules, interfaces and data.
- Solve a problem or build a product.
- Define the input, output, business rules, data schema.
- Design patterns solve common problems.
- 3 Types:
  - UI design: Data visualization and presentation.
  - Data design: Data representation and storage.
  - Process design: Validation, manipulation and storage of data.

WOMEN WHO CODE

Backend Engineering

Design Patterns

- Set of template solutions that can be reused.
- Improved code maintainability, reusability and scaling.
- Leverages Object-oriented programming (OOP) principles for flexible and maintainable designs.
- Shared pattern vocabulary.
- Not a library or framework, but recommendations for code structuring and problem solving.
- Adapt a pattern and improve upon it to fit application needs.
- Defines relationship between objects, loosely coupled objects, secure code.

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

WOMEN WHO CODE

Backend Engineering

Types of Design Patterns

- **Creational**:
  - Initialize a class and instantiate the objects.
  - Decoupled from implementing system.
  - Singleton, Factory, Builder.
  - Abstract Factory, Prototype.
- **Structural**:
  - Class structure and composition.
  - Increase code reusability and functionality.
  - Create large objects relationships.
  - Adapter, Facade, Decorator, Bridge, Composite, Flyweight, Proxy.
- **Behavioural**:
  - Relationship and communication between different classes.
  - Observer, Strategy, Iterator, etc.

Creational	Structural	Behavioral
• Factory Method	• Adapter	• Interperter
• Abstract Factory • Builder • Prototype • Singleton	• Adapter • Bridge • Composite • Decorator • Facade • Flyweight • Proxy	• Chain of Responsibility • Command • Iterator • Mediator • Momento • Observer • State • Strategy • Visitor

WOMEN WHO CODE

Backend Engineering

- What are design patterns?
- What is a creational design pattern?
- What is a structural design pattern?
- Can you give examples?

You can unmute and talk or use the chat.

WOMEN WHO CODE

Backend Engineering

Object-Oriented Programming Concepts:

- **Inheritance and *super()***: subclass inherits methods and attributes of superclass.
  super() inside subclass constructor to pass attributes.
- **Polymorphism**: same method but different behavior.
  - Overloading: Same name method but different parameters.
  - Overriding: Method signature (name & parameters) same in subclass and superclass.
- **Abstract class**:
  - Has *abstract* and concrete methods. Declare vs. Define variable and method.
  - Objects that extend (one) class have same/default behavior & can be overridden.
- **Interface**:
  - Has abstract methods only. Uses *implements* keyword.
  - Polymorphic behavior for objects that implement Interface(s).

Backend Engineering

- What are the OOP concepts?
- Can you give examples?

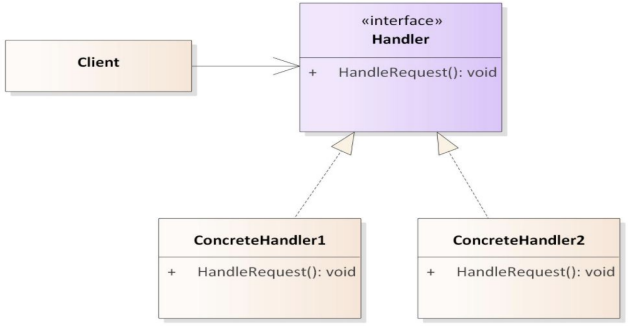
You can unmute and talk or use the chat.

WOMEN WHO CODE

Backend Engineering

Behavioral Design Patterns:

- **Chain of Responsibility:**
  - Sender object sends request to a chain of receiving objects to eventually reach the receiver object. This avoids coupling between sender object and receiver.
  - Links receiving objects together in a chain of objects.
  - Once an object independently handles the request, it is sent to the next object in the chain.
  - These “handler” objects are determined at runtime.
  - *Handler*: Interface that receives a request and sends it to the next handler object.
  - *Concrete Handler*: Sequential handler objects in the chain.
  - *Sender*: Originates the request and references the *Handler*.
- Example: Shipment delivery. Package --> shipping state 1 --> state 2 --> Receiver
- Code example.

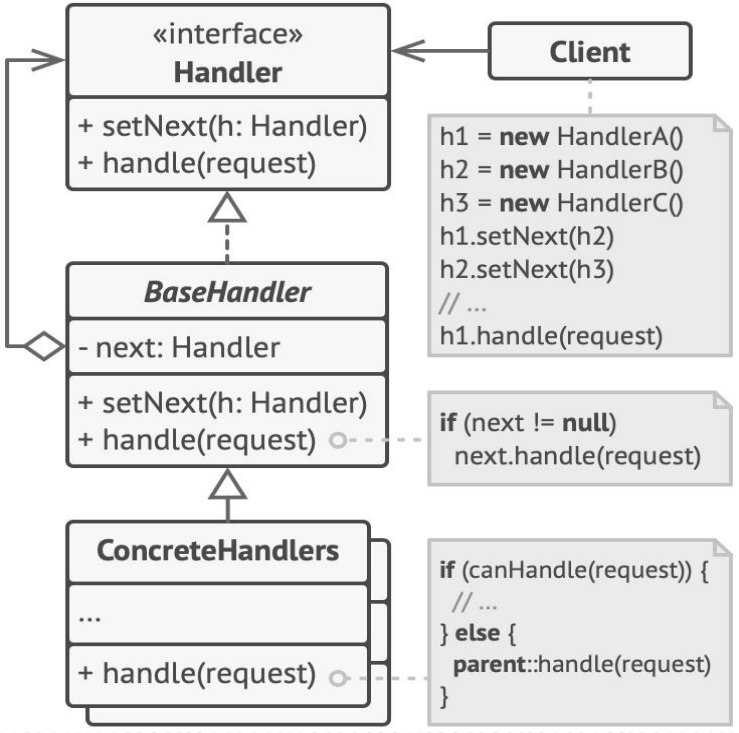


WOMEN WHO CODE

Backend Engineering

- What is a Chain of Responsibility design pattern?
- Can you give examples?

You can unmute and talk or use the chat.

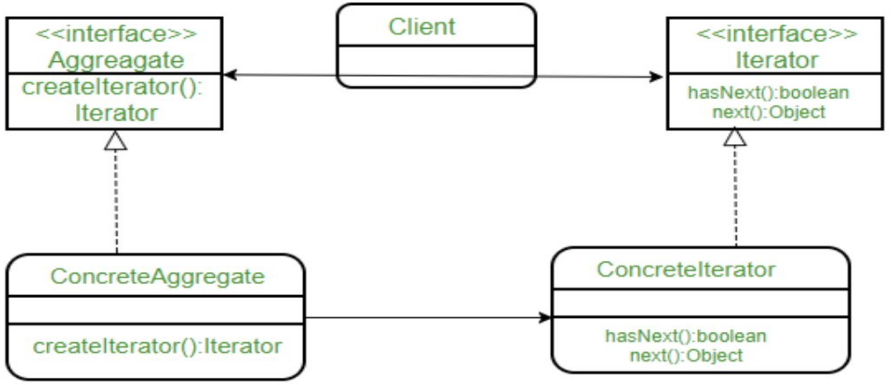


WOMEN WHO CODE

Backend Engineering

Behavioral Design Patterns:

- **Iterator:**
  - Traverse a collection of objects in a specific manner. AKA *cursor.*
  - Access elements without revealing the implementation.
  - Iterator:
    - Interface with methods to iterate over a collection (of any type).
    - Different simultaneous iterations: one-way and bi-directional.
  - Concrete Iterator: Implements the *Iterator*. Tracks current position in traversal.
  - Aggregator: Collection *Interface* to create an *Iterator.*
  - Concrete Aggregator: Returns an instance of *Concrete Iterator.*
- Example: Directory of names: Search alphabetically, search from start or from end.
- Code example.

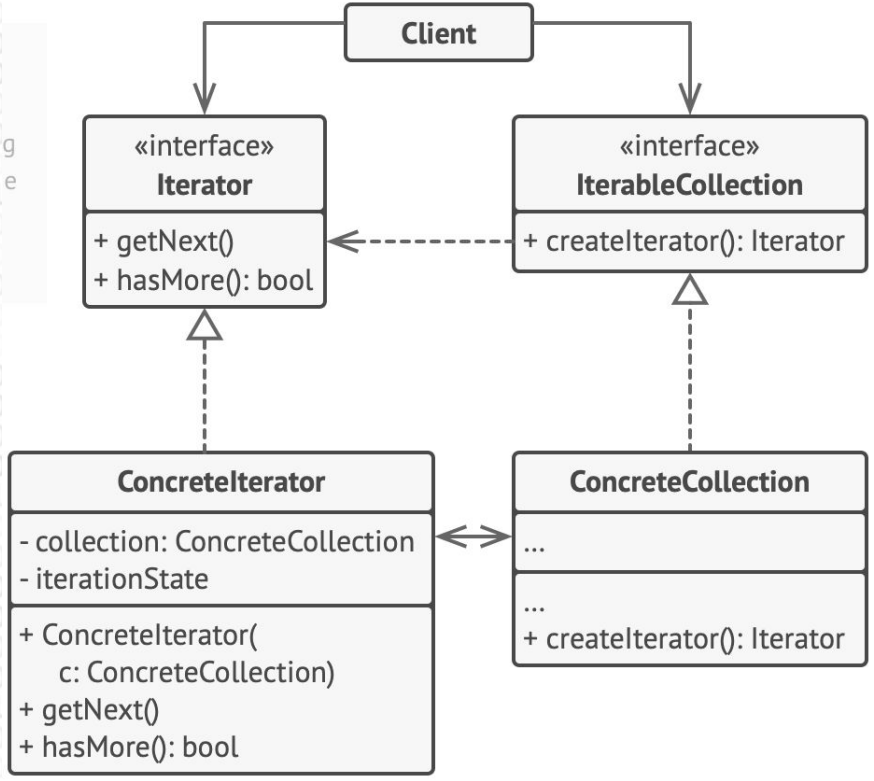


WOMEN WHO CODE

Backend Engineering

- What is a Iterator design pattern?
- Can you give examples?

You can unmute and talk or use the chat.

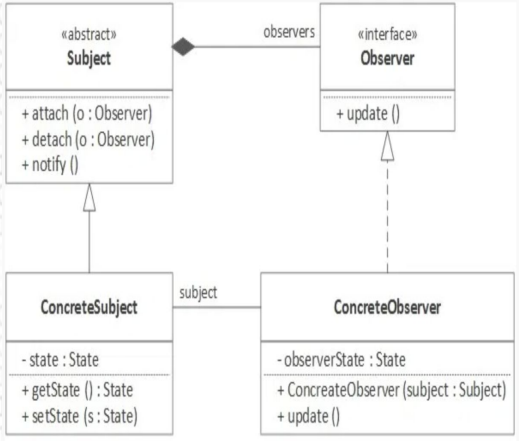


WOMEN WHO CODE

Backend Engineering

Behavioral Design Patterns:

- **Observer:**
  - Define 1-1 dependency between objects.
  - On change of state in one object, dependant objects are notified and updated.
  - AKA broadcast communication or subscribe-publish.
  - Objects can be same/different but implement the same Observer interface.
  - Observable: Objects state change is of interest.
  - Observer: Registered objects that are notified on Observable' state change.
  - Common functions: add(), remove(), hasChanged().
- Example: Marketing & new products notifications. Kafka Pub/Sub.
- Code example.



WOMEN WHO CODE

Backend Engineering

- What is a Observer design pattern?
- Can you give examples?

You can unmute and talk or use the chat.

WOMEN WHO CODE

Backend Engineering

Behavioral Design Patterns:

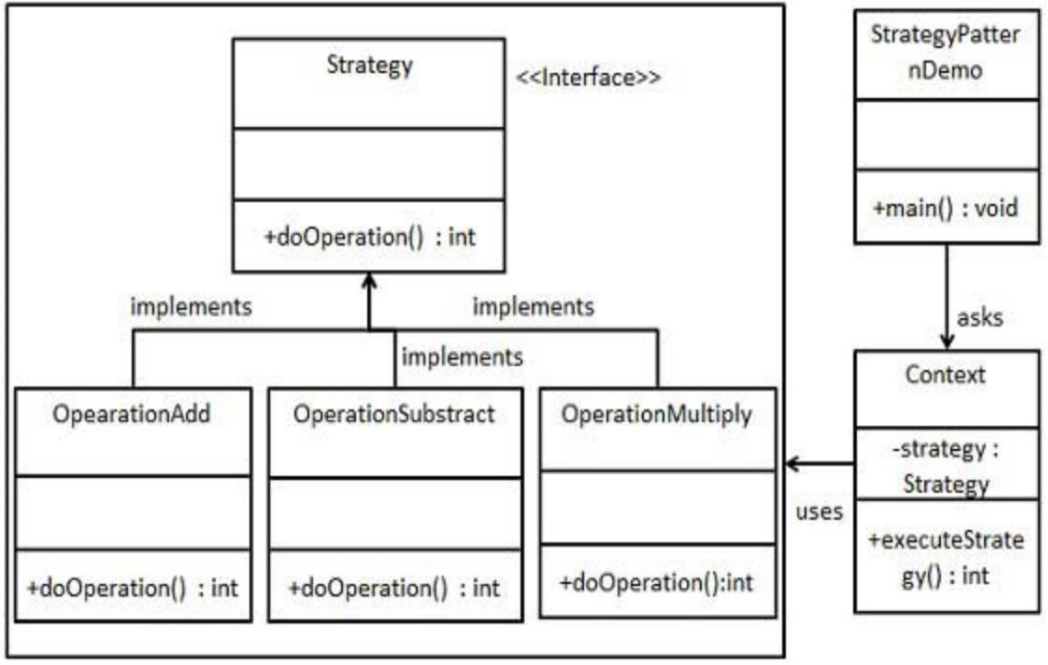
- **Strategy:**
  - Select one out of different strategies/ algorithms/ implementations at runtime.
  - AKA *Policy.*
  - Add strategies in separate classes that the client references w.r.t. the context.
  - Strategy: *Interface* with methods to implement the strategy (Example: Sorting).
  - Context: Determines what strategy and respective implementation to select.
  - Implementation: Various *Strategy* implementations (Example: Merge, Quick, etc.).
  - Open/Closed principle:
    - Open: Objects extended to select strategies/implementations.
    - Closed: Context and objects not modified.
- Example: Sort (algorithms) a collection of objects (List, Set, etc.).
- Code example.

WOMEN WHO CODE

Backend Engineering

- What is a Strategy design pattern?
- Can you give examples?

You can unmute and talk or use the chat.



WOMEN WHO CODE

Backend Engineering

Summary:

- Behavioral design patterns: Focus on relationships and communications between different classes.
- Chain of Responsibility: Sender object sends request to a chain of receiving objects to eventually reach the receiver object.
- Iterator: Traverse a collection of objects in a specific manner.
- Observer: Define 1-1 dependency between objects. Subscribe to the Observable.
- Strategy: Select one out of different strategies at runtime.

NEXT SESSION on 6-17-2021: Anti-patterns

You can unmute and talk or use the chat.

WOMEN WHO CODE

Backend Study Group

- WWCode Presentation and Demo
- WWCode YouTube channel:
  - March 25, 2021 session recording: Backend Engineering
  - April 8, 2021 session recording: Java Microservice and REST API Demo
  - April 22, 2021 session recording: Creational Design Patterns
  - May 20, 2021 session recording: **Structural Design Patterns**
- **Resources**:
  - Software design pattern
  - Design Patterns in Java
  - Design patterns
  - Design Patterns
  - Head First Design Patterns book

You can unmute and talk or use the chat.

WOMEN WHO CODE