SOFTWARE ENGINEERING

Class 8 & 9

amrut@iiit.ac.in

TOPICS

- Overview
- Creational Design Patterns
- Structural Design Patterns
- Behavioral Design Patterns

OVERVIEW

OVERVIEW

Object Oriented Design

OVERVIEW

Object Oriented Design

- Identify the objects and their responsibilities

OVERVIEW

Object Oriented Design

- Identify the objects and their responsibilities
- Identify how they are structured

OVERVIEW

Object Oriented Design

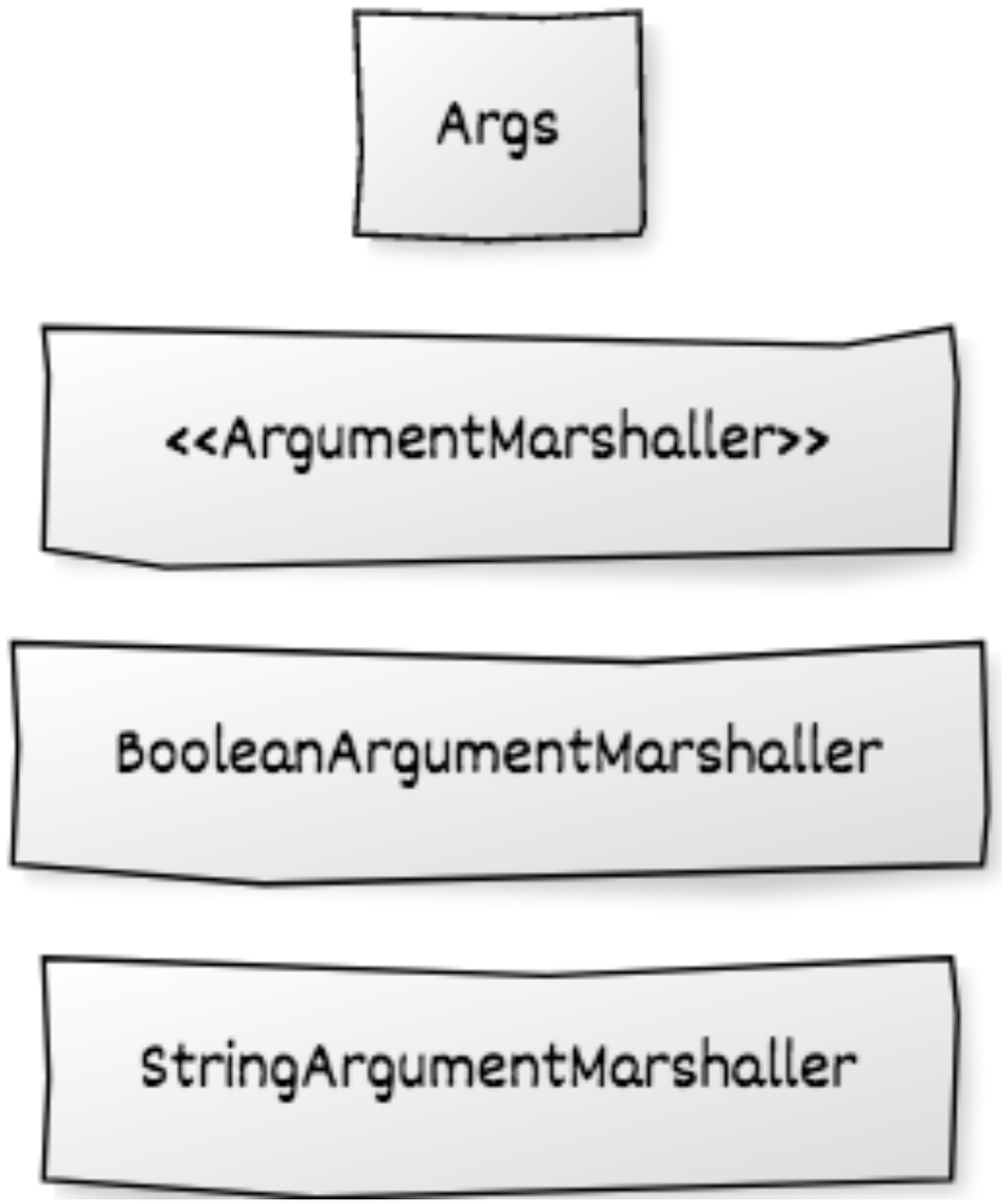
- Identify the objects and their responsibilities
- Identify how they are structured
- Identify how they interact

OVERVIEW

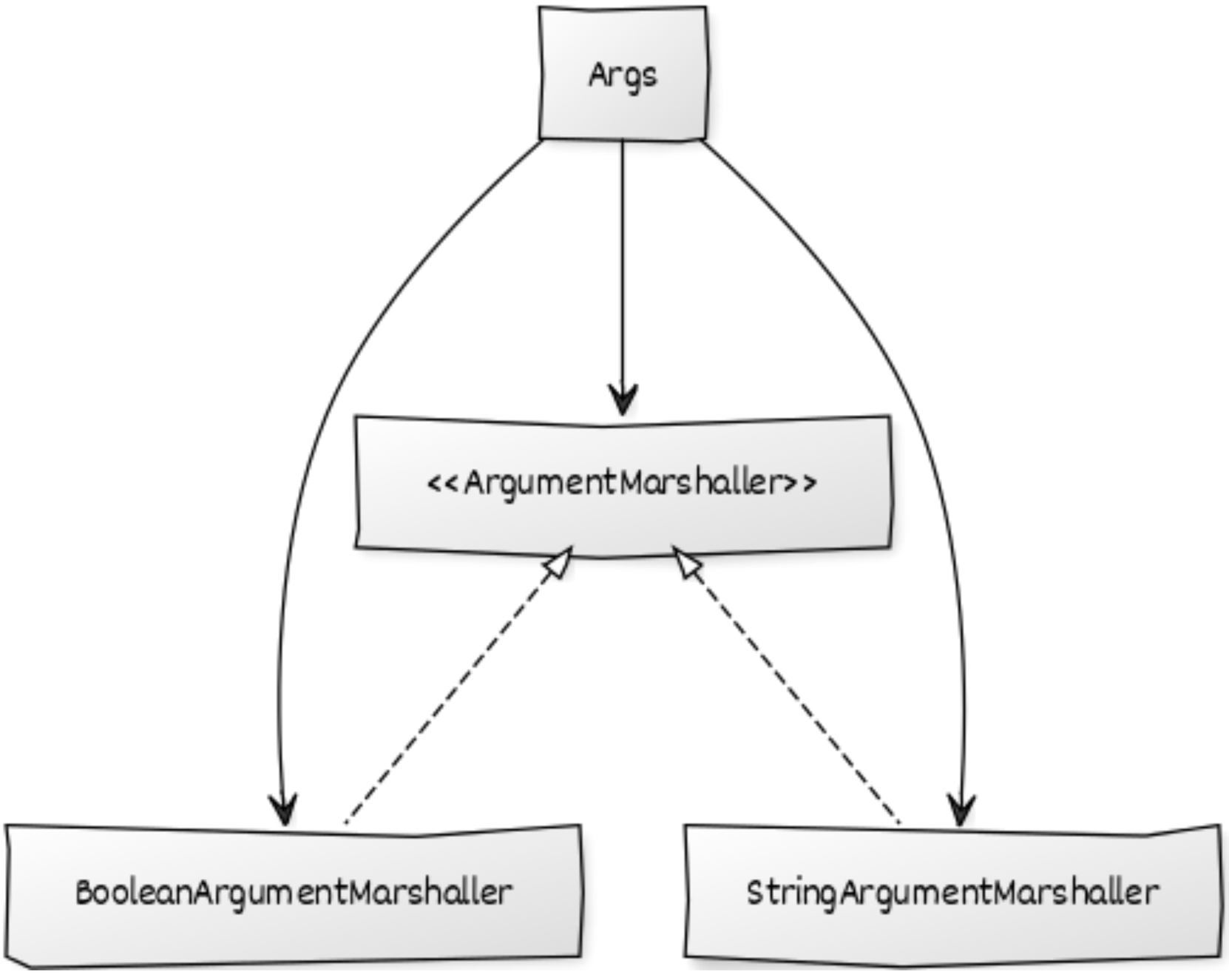
Object Oriented Design

- Identify the objects and their responsibilities
- Identify how they are structured
- Identify how they interact
- Identify how they behave

OVERVIEW


Args
<<ArgumentMarshaller>>
BooleanArgumentMarshaller
StringArgumentMarshaller

OVERVIEW


Args
<<ArgumentMarshaller>>
BooleanArgumentMarshaller
StringArgumentMarshaller

OVERVIEW

A **design pattern** is a
general, reusable solution to a commonly
occurring design problems

OVERVIEW

A **design pattern** is a
general, reusable solution to a commonly
occurring design problems

OVERVIEW

A **design pattern** is a **general**, **reusable** solution to a commonly occurring design problems

OVERVIEW

A **design pattern** is a **general**, **reusable** solution to a **commonly occurring** design problems

OVERVIEW

Design Pattern Categories

(Based on purpose)

- Creational Design Patterns
- Structural Design Patterns
- Behavioral Design Patterns

Design Pattern Categories

(Based on scope)

- Class based Design Patterns
- Object based Design Patterns

OVERVIEW

Object Oriented Basics

OVERVIEW

Object Oriented Basics

- Concrete Classes

OVERVIEW

Object Oriented Basics

- Concrete Classes
- Abstract Classes

OVERVIEW

Object Oriented Basics

- Concrete Classes
- Abstract Classes
- Interfaces

OVERVIEW

Object Oriented Basics

- Concrete Classes
- Abstract Classes
- Interfaces
- Inheritance

OVERVIEW

Object Oriented Basics

- Concrete Classes
- Abstract Classes
- Interfaces
- Inheritance
- Composition

CREATIONAL DESIGN PATTERNS

Creational Design Patterns

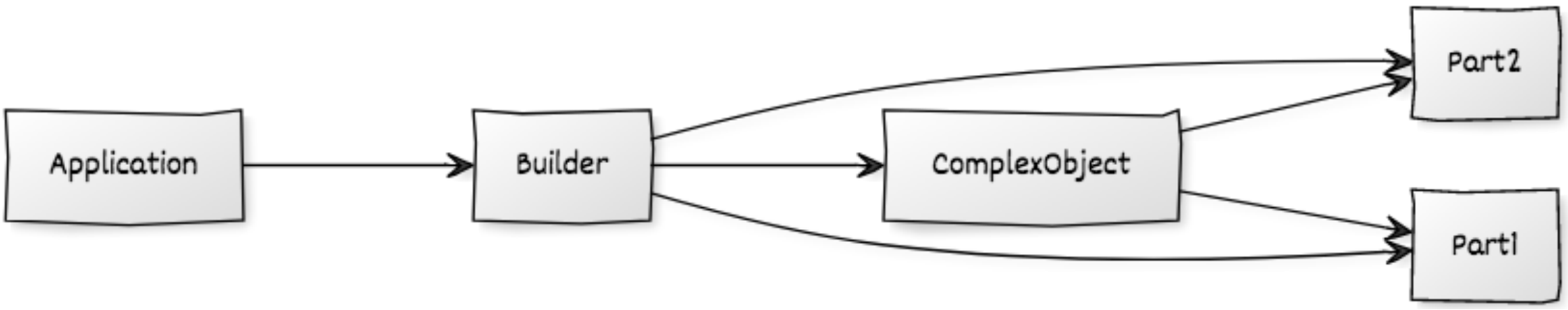
- Encapsulate knowledge about ***which*** concrete classes are being used
- Encapsulate knowledge about ***how*** concrete classes are created.

Builder Pattern

Builder pattern delegates creation of a complex object to another object.

Builder Pattern

Builder pattern delegates creation of a complex object to another object.



Builder Pattern Example

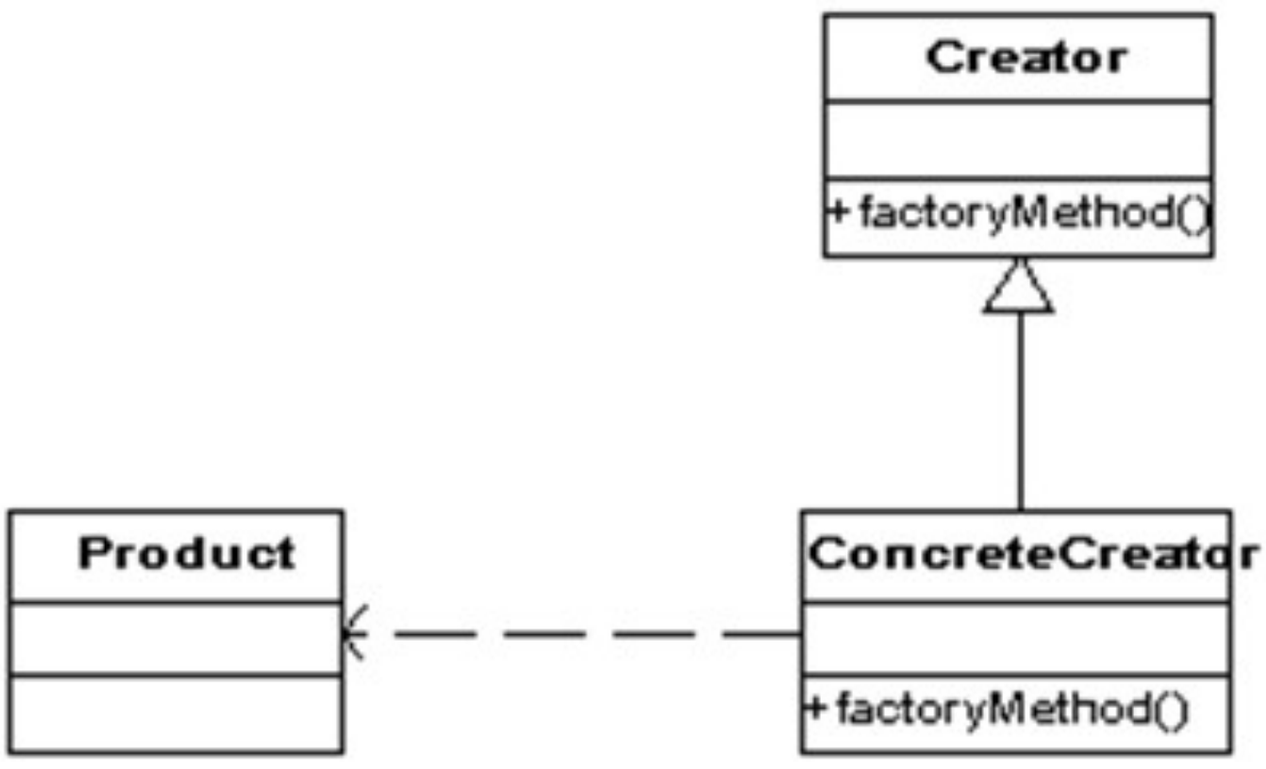
```
Locale aLocale = new Locale.Builder()
        .setLanguage(“sr")
        .setScript(“Latn")
        .setRegion(“RS")
        .build();
```

Factory Method Pattern

Factory method pattern delegates creation of a complex object to subclasses.

Factory Method Pattern

Factory method pattern delegates creation of a complex object to subclasses.



Factory Method Pattern Example

```
DecimalFormat numberFormat =
        (DecimalFormat) NumberFormat.getInstance(locale);
```

STRUCTURAL DESIGN PATTERNS

STRUCTURAL

Structural Design Patterns

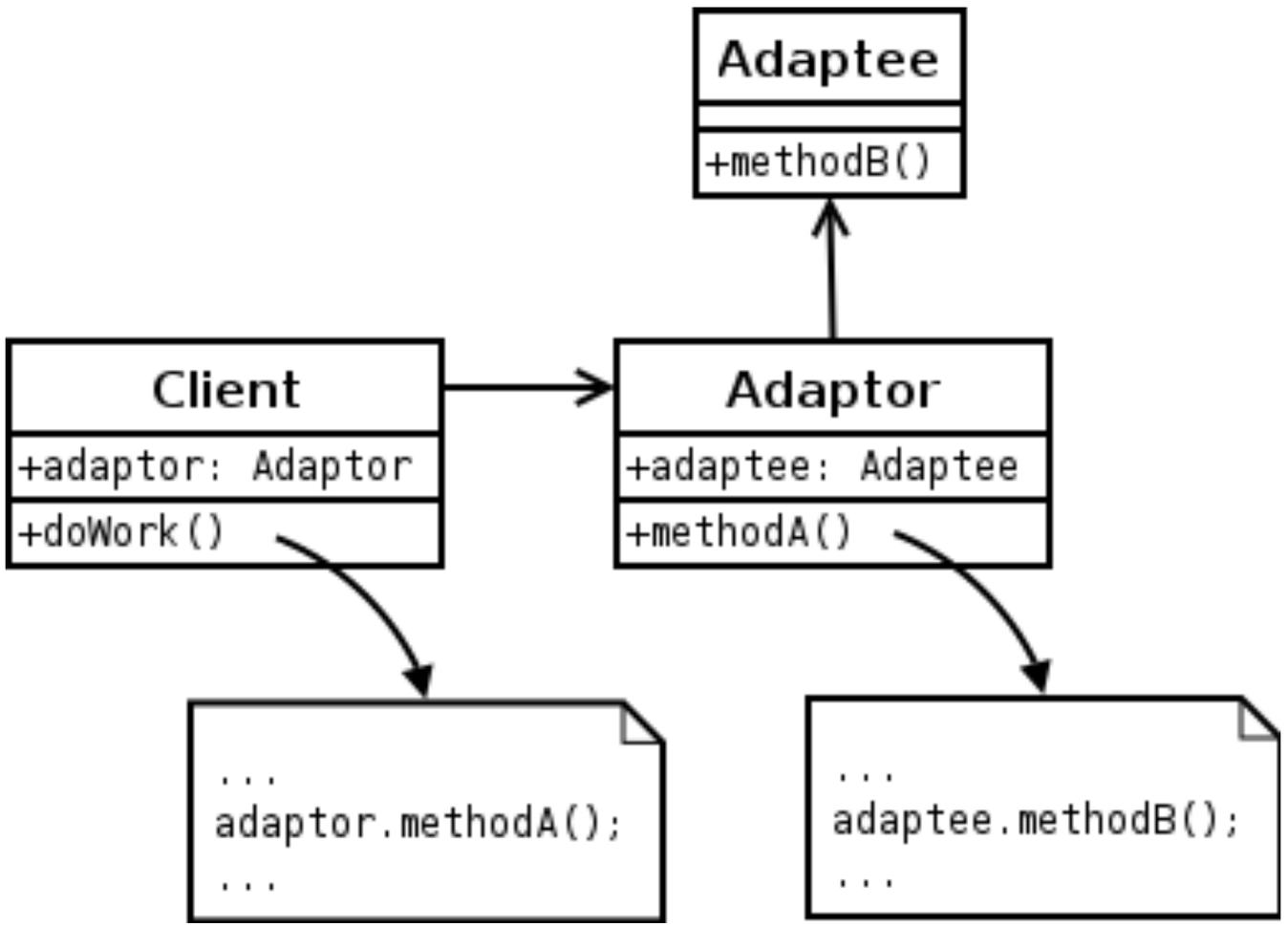
- Solve problems by structuring object relationships using class inheritance
- Solve problems by structuring object relationships using object composition

Adapter Pattern

Adapter Pattern allows the interface of an existing class to be used as another interface.

Adapter Pattern

Adapter Pattern allows the interface of an existing class to be used as another interface.



Adapter Pattern Example

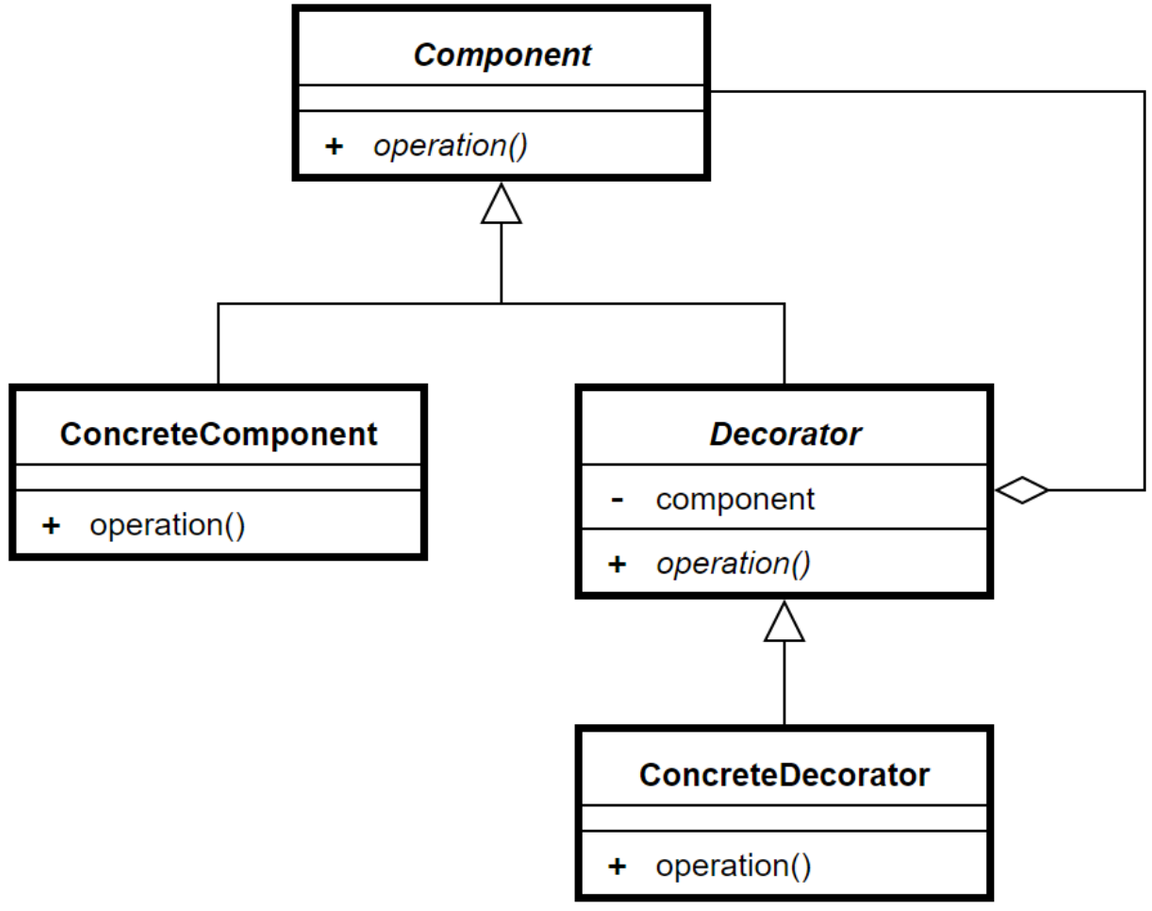
```
List<String> stooges = Arrays.asList("Larry", "Moe", "Curly");
```

Decorator Pattern

Decorator Pattern allows behavior to be added to an individual object, dynamically, without affecting the behavior of other objects from the same class

Decorator Pattern

Decorator Pattern allows behavior to be added to an individual object, dynamically, without affecting the behavior of other objects from the same class



Decorator Pattern Example

```
Collection set = new HashSet();
Collection stringSet =
    Collections.checkedCollection(set, String.class);
```

BEHAVIORAL DESIGN PATTERNS

Behavioral Design Patterns

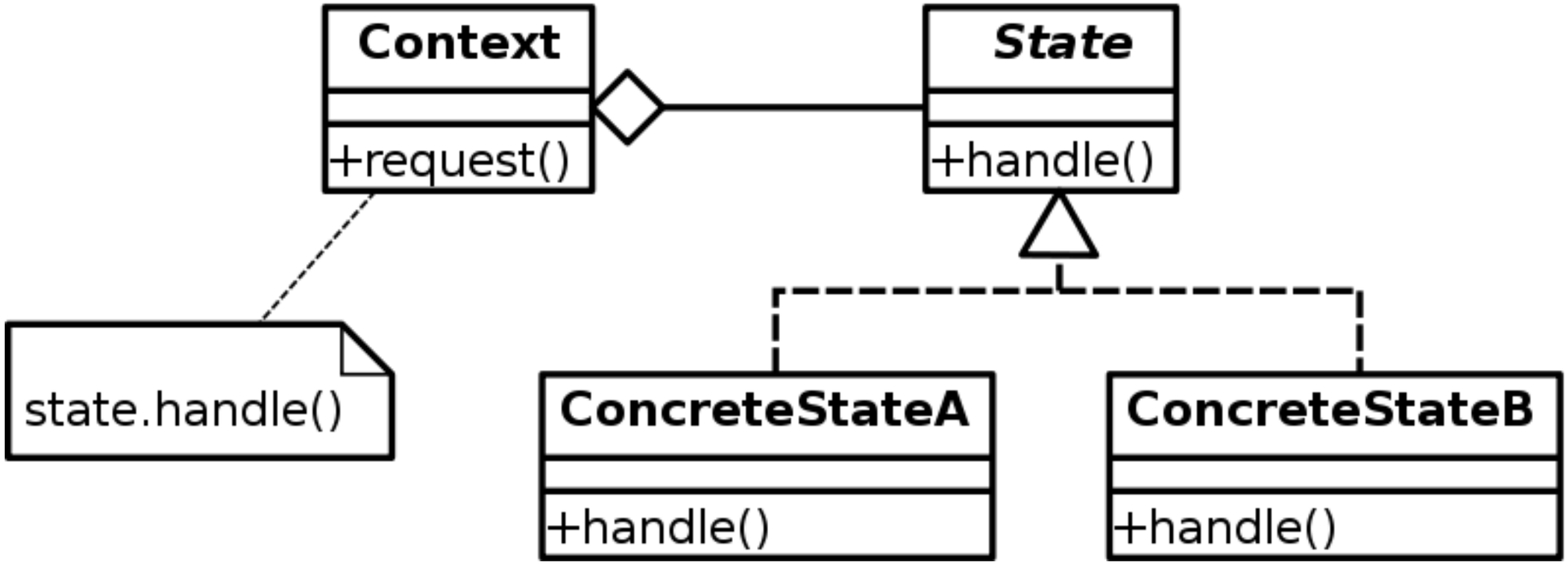
- Behavioral design patterns are concerned with algorithms and assignment of responsibilities between objects.
- Behavioral design patterns are related to control flow between objects.

State Pattern

State pattern is used to encapsulate varying behavior for the same object, based on its internal state.

State Pattern

State pattern is used to encapsulate varying behavior for the same object, based on its internal state.

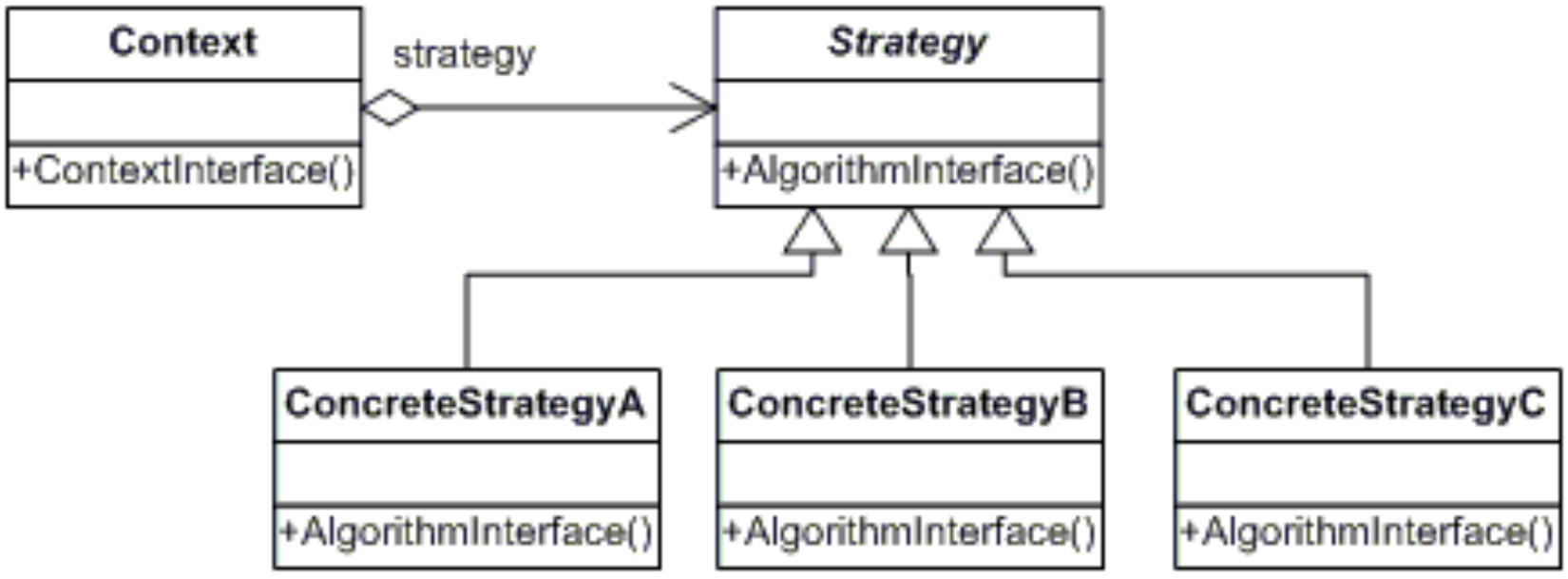


Strategy Pattern

Strategy pattern enables selecting an algorithm at runtime.

Strategy Pattern

Strategy pattern enables selecting an algorithm at runtime.



REFERENCE

		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 (195) Observer State Strategy Visitor

REFERENCE

- https://en.wikipedia.org/wiki/Software_design_pattern
- Design Patterns: Elements of Reusable Object-Oriented Software (Addison-Wesley Professional Computing Series)
- https://en.wikipedia.org/wiki/Object-oriented_design