DePaul University

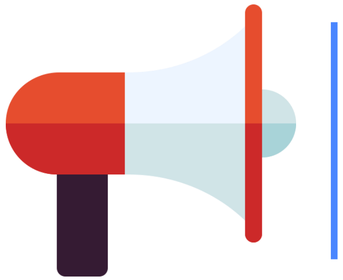
Design Patterns:

Decorator

Object-oriented Software Development
SE 350– Spring 2021

Vahid Alizadeh

Week 9.1
May 25, 2021



Announcements

Future Schedule

Assignment 3 solutions on D2L

Assignment 4 Due date: May 28

- ~~Assignment 1~~
- ~~Assignment 2~~
- ~~Mid Term Exam~~
- ~~Assignment 3:~~
  - ~~Release: Week 7~~
  - ~~Due: Week 8~~
- **Assignment 4:**
  - ~~Release: Week 8~~
  - Due: Week 9
- **Bonus Research Project:**
  - Presentation Due: Week 10
  - Report Due: Week 11
- **Final Exam:**
  - Week 11

SE 350: OO Software Development

Assignment 4: Design Patterns (2)

Instructor: Vahid Alizadeh
Email: v.alizadeh@depaul.edu
Quarter: Spring 2021

DEPAUL UNIVERSITY
COLLEGE OF COMPUTING AND DIGITAL MEDIA

Last update: May 19, 2021

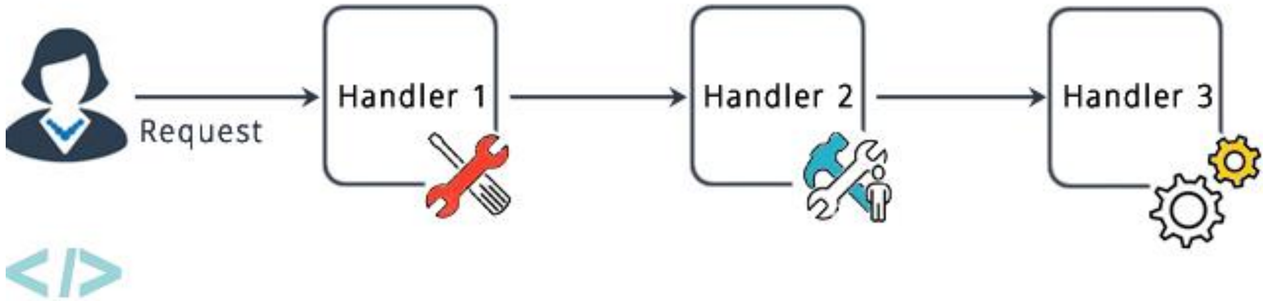
Chain of Responsibility Pattern

CoR / Chain of Commands

BEHAVIORAL

CoR Pattern Introduction

Chain of Responsibility is a behavioral design pattern that can be used when we want to give more than one object a chance to handle a request.



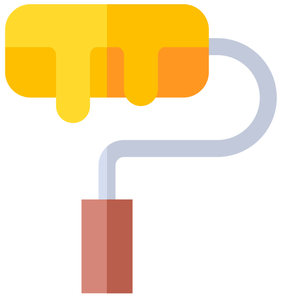
CoR Pattern Pros & Cons

Pros

- Decouple sender and receiver.
- Dynamically add or remove responsibilities.
- Control over the order of handlers.
- Single responsibility and Open/closed principles.

Cons

- The request may fail to be executed by the end of the chain.
- Many implementation classes leads to maintenance and debugging difficulties.

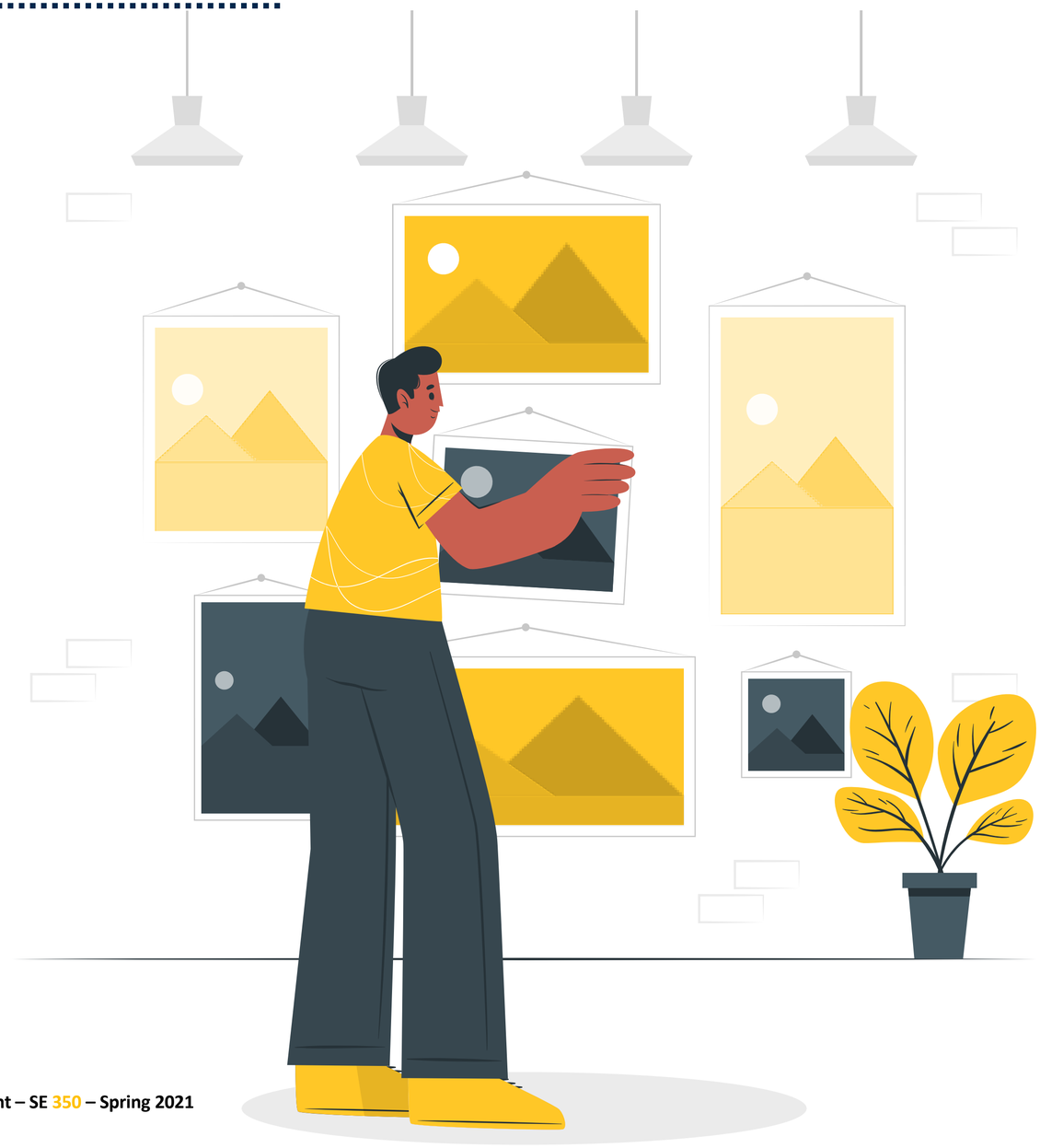


Decorator Pattern

STRUCTURAL

Decorator Pattern Introduction

Decorator is a structural design pattern that allows for an object's behavior to be extended dynamically at run time.



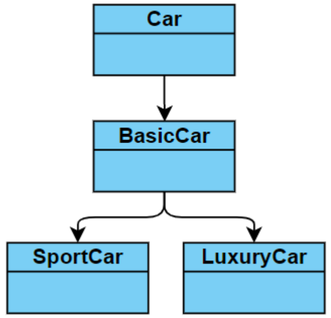
Decorator Design Pattern

INTENT

- Add additional responsibilities to individual objects dynamically.
- withdraw responsibilities from an object.

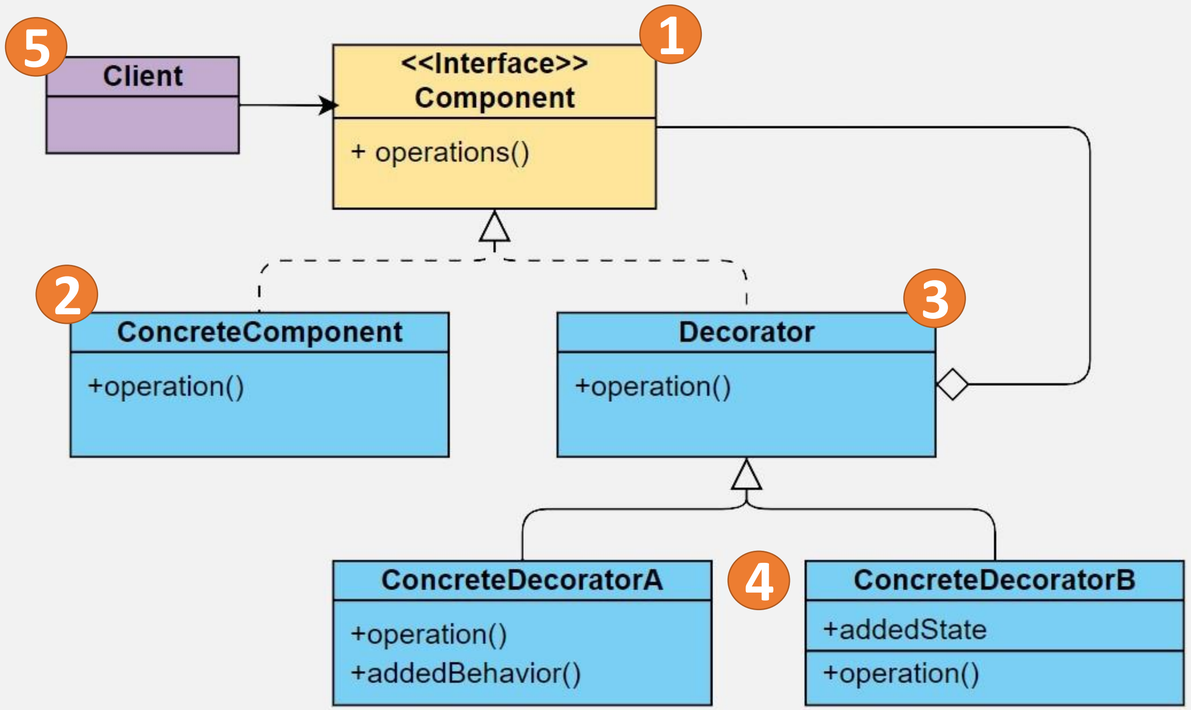
PROBLEM

- Adding a behavior to an object at run-time in not possible by inheritance.
- Car example

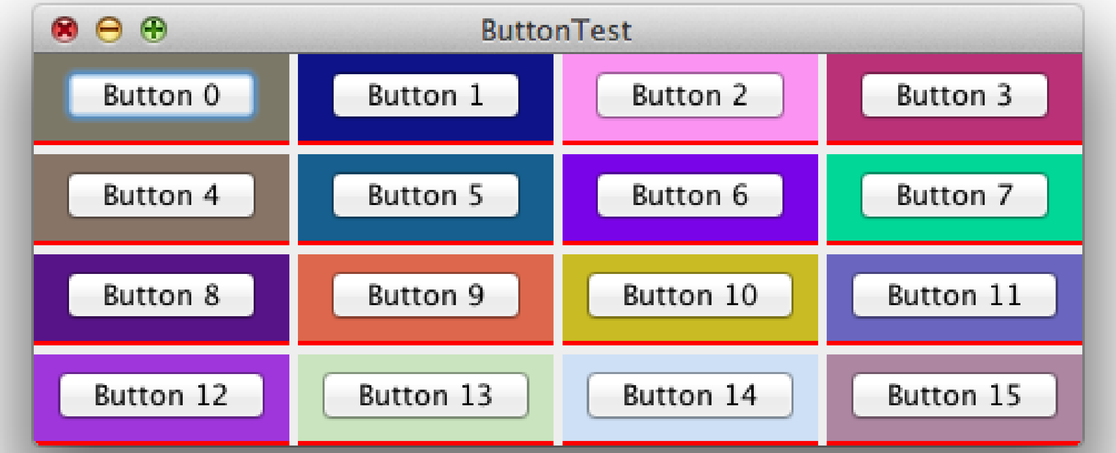


STRUCTURE

- 1- Component Interface
- 2- Concrete Component
- 3- Decorator
- 4- Concrete Decorators



Decorator Pattern: Real-world Example

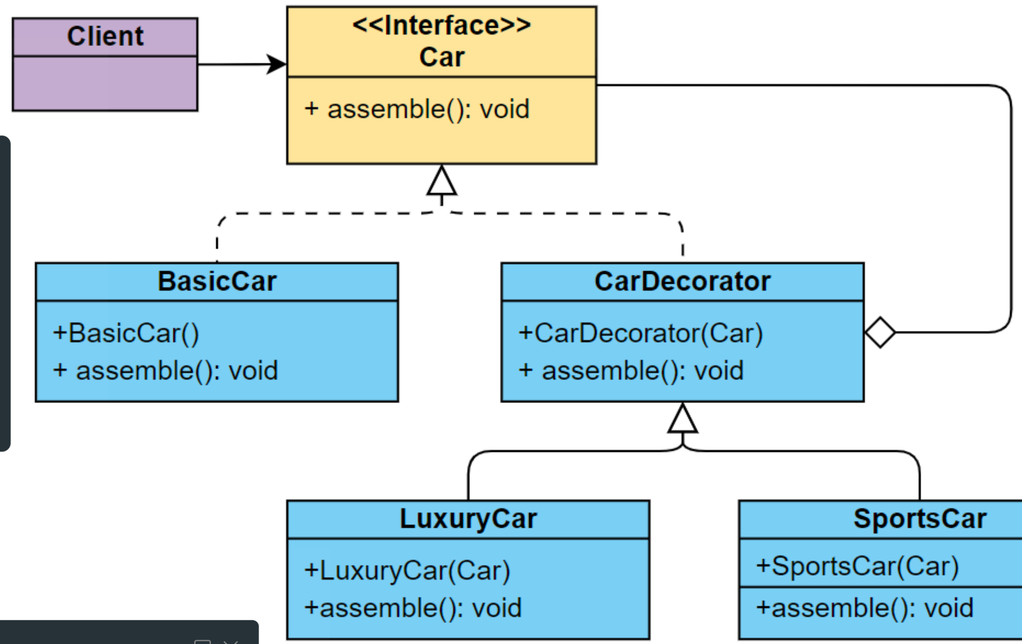


Find the source codes in the course GitHub repository.

Decorator Use Case Example: Car Manufacturing

```java
public class CarManufacturingDriver {
    public static void main(String[] args) {
        Car sportsCar = new SportsCar(new BasicCar());
        sportsCar.assemble();
        System.out.println("\n*****");
        Car sportsLuxuryCar = new SportsCar(new LuxuryCar(new BasicCar()));
        sportsLuxuryCar.assemble();
    }
}
```

```java
public interface Car {
    public void assemble();
}
```



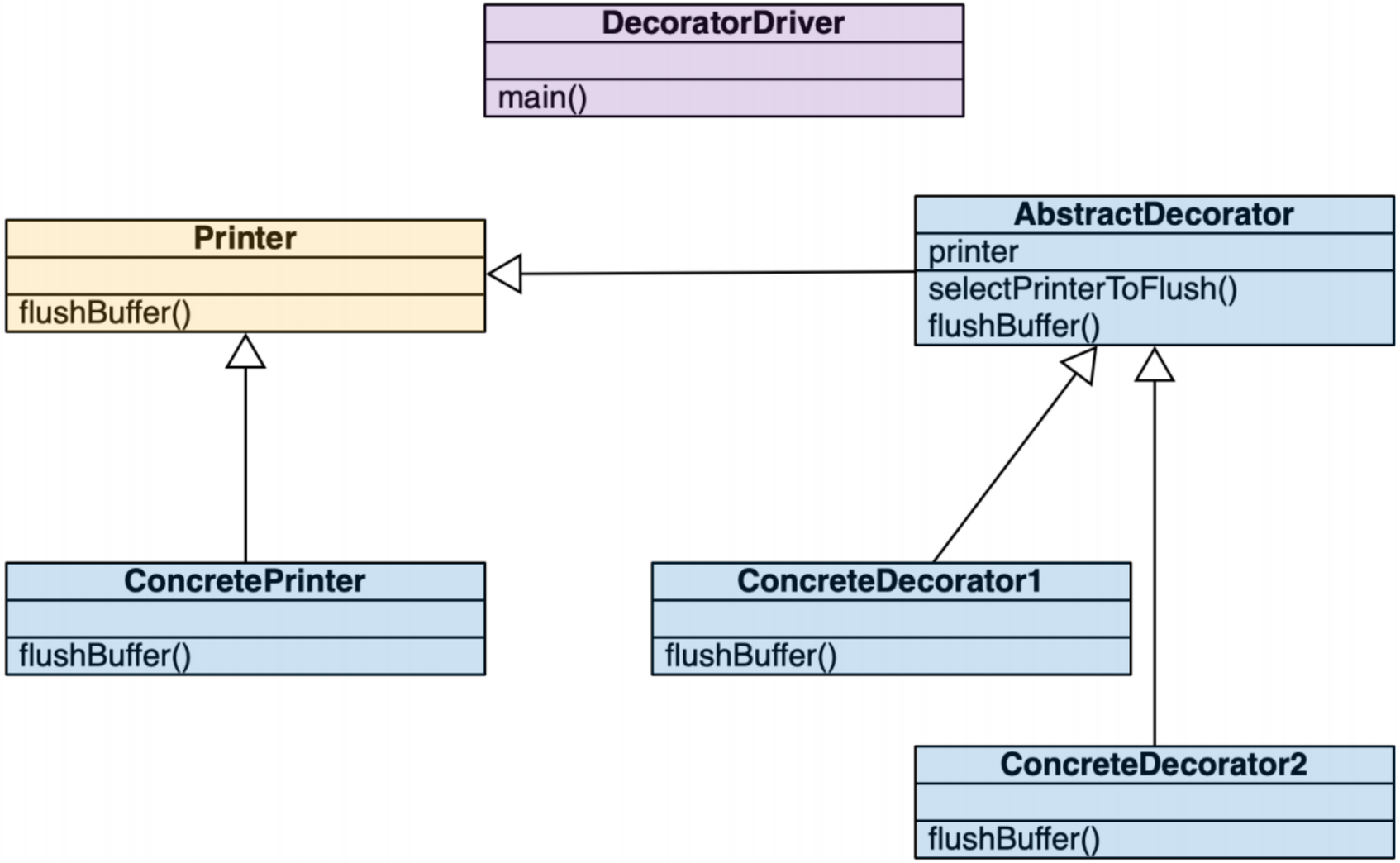

```java
public class BasicCar implements Car {
    @Override
    public void assemble() {
        System.out.print("Basic Car.");
    }
}
```

```java
public class CarDecorator implements Car {
    protected Car car;
    public CarDecorator(Car c){
        this.car=c;
    }
    @Override
    public void assemble() {
        this.car.assemble();
    }
}
```

```java
public class LuxuryCar extends CarDecorator {
    public LuxuryCar(Car c) {
        super(c);
    }
    @Override
    public void assemble(){
        super.assemble();
        System.out.print(" Adding features of Luxury Car.");
    }
}
```

```java
public class SportsCar extends CarDecorator {
    public SportsCar(Car c) {
        super(c);
    }
    @Override
    public void assemble(){
        super.assemble();
        System.out.print(" Adding features of Sports Car.");
    }
}
```

Decorator Use Case Example: Printer Demo

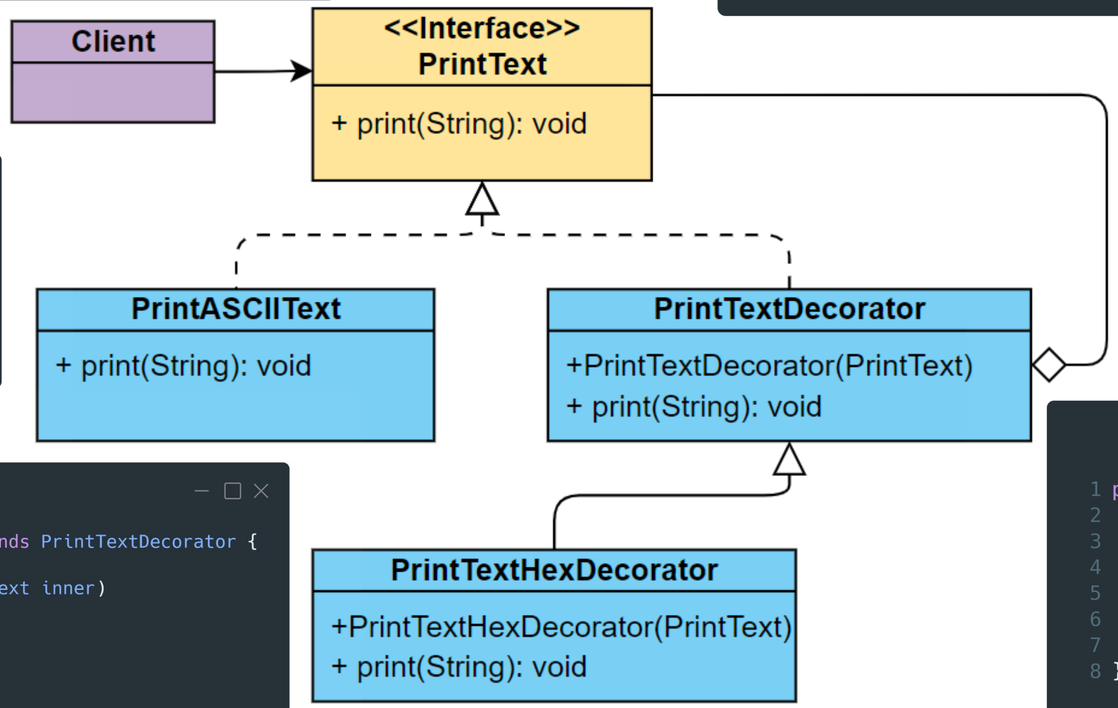


Find the source codes in the course GitHub repository.

Decorator Use Case Example: Print Text Demo

```java
public class PrintTextDriver {
    public static void main (String[] args) throws java.lang.Exception
        {
            final String text = "SE 450 - Winter 2021 - Decorator Pattern Example";
            final PrintText object = new PrintAsciiText();
            final PrintText printer = new PrintTextHexDecorator(object);
            object.print(text);
            printer.print(text);
        }
}
```

```java
public interface PrintText {
    public void print(String text);
}
```




```java
public class PrintAsciiText implements PrintText {
    public void print(String text)
    {
        System.out.println("Print ASCII: " + text);
    }
}
```

```java
public abstract class PrintTextDecorator implements PrintText {
    private PrintText inner;
    public PrintTextDecorator(PrintText inner)
    {
        this.inner = inner;
    }
    public abstract void print(String text);
}
```

```java
public class PrintTextHexDecorator extends PrintTextDecorator {
    private PrintText inner;
    public PrintTextHexDecorator(PrintText inner)
    {
        super(inner);
    }
    public void print(String text)
    {
        String hex = text.chars()
                .boxed()
                .map(x -> "0x" + Integer.toHexString(x))
                .collect(Collectors.joining(" "));
        inner.print(text + " -> HEX: " + hex);
    }
}
```


Find the source codes in the course GitHub repository.

Assignments

Assignment 3 Solution

Assignment 4 – Q 2

Assignment 4

Due date: May 28, Friday, 11:59PM

SE 350: OO Software Development

Assignment 4: Design Patterns (2)

Instructor: Vahid Alizadeh

Email: v.alizadeh@depaul.edu

Quarter: Spring 2021

DEPAUL UNIVERSITY
COLLEGE OF COMPUTING AND DIGITAL MEDIA

Last update: May 19, 2021

Assignment 3 Solutions

Assignment 3 Solutions on D2L

SE 350: OO Software Development

Assignment 3: Design Principles and Design Patterns

Instructor: Vahid Alizadeh

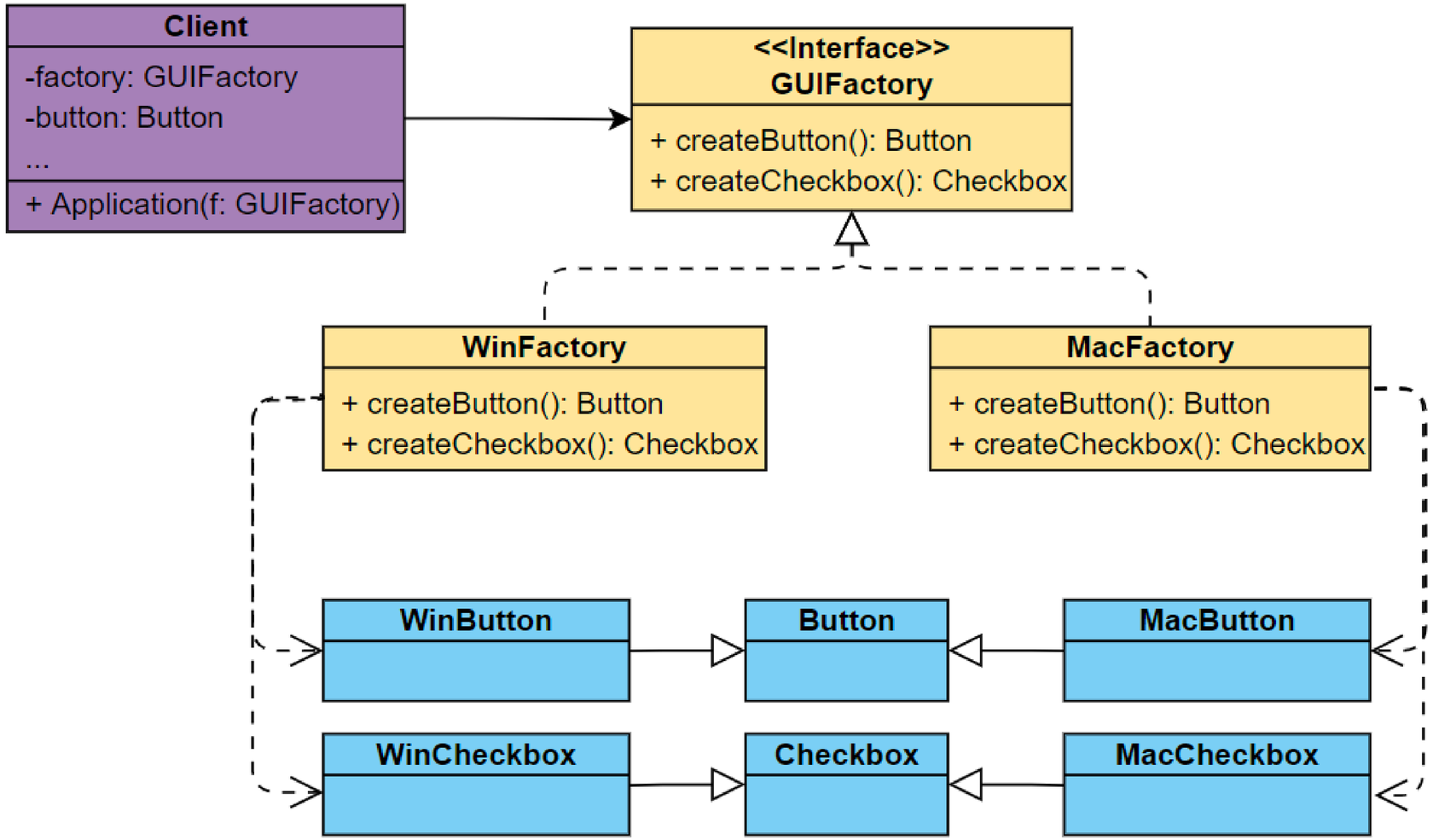
Email: v.alizadeh@depaul.edu

Quarter: Spring 2021

DEPAUL UNIVERSITY
COLLEGE OF COMPUTING AND DIGITAL MEDIA

Last update: May 19, 2021

Use Case Example: Cross-platform UI Elements



Any Question

??????????????

How do you feel about the course?

Powered by Poll Everywhere

Start the presentation to see live content. For screen share software, share the entire screen. Get help at **pollev.com/app**

Please Send Your Question or Feedback...

Top **New**

Powered by Poll Everywhere

Start the presentation to see live content. For screen share software, share the entire screen. Get help at **pollev.com/app**