**MODUL 7 DESIGN PATTERN**

**PEMROGRAMAN BERORIENTASI OBJEK**

Seluruh code pada praktikum ini dapat dibuka di: <https://github.com/afrzl/Pemrograman-Berorientasi-Objek/tree/main/Week%207/Praktikum>

1. **Observer Pattern**
2. Interface Observer

*/\*\**

*\* Observer*

*\*/*

public interface Observer {

  public void **update**();

}

1. Interface Observable

*/\*\**

*\* Observable*

*\*/*

public interface Observable {

  void **addObserver**(Observer *o*);

  void **removeObserver**(Observer *o*);

  void **notifyObserver**();

}

1. Class PinkBook

*/\*\**

*\* PinkBook*

*\*/*

import java.util.\*;

public class PinkBook implements Observable {

  private boolean inStock = true;

  private ArrayList<Observer> customers;

  public **PinkBook**(boolean *inStock*) {

    this.inStock = *inStock*;

    customers = new **ArrayList**<Observer>();

  }

  public boolean **isInStock**() {

    return inStock;

  }

  public void **setInStock**(boolean *inStock*) {

    this.inStock = *inStock*;

    if (**isInStock**()) {

**notifyObserver**();

    }

  }

  @Override

  public void **addObserver**(Observer *o*) {

    customers.**add**(*o*);

  }

  @Override

  public void **removeObserver**(Observer *o*) {

    customers.**remove**(*o*);

  }

  @Override

  public void **notifyObserver**() {

    for (int i = 0; i < customers.**size**(); i++) {

      customers.**get**(i).**update**();

    }

  }

}

1. Class Customer

*/\*\**

*\* Customer*

*\*/*

public class Customer implements Observer {

  private Observable observable;

  private String username;

  public **Customer**(Observable *observable*, String *username*) {

    this.observable = *observable*;

    this.username = *username*;

  }

  @Override

  public void **update**() {

    System.out.**println**("Buku Pink Tersedia");

**buyDress**();

  }

  private void **buyDress**() {

    System.out.**println**(username + " mendapatkan Buku Pink.");

  }

  public void **unsubscribe**() {

    observable.**removeObserver**(this);

  }

}

1. Class ObserverPatternMain

*/\*\**

*\* ObserverPatternMain*

*\*/*

public class ObserverPatternMain {

  public static void **main**(String[] *args*) {

    PinkBook pinkbook = new **PinkBook**(true);

    Customer customer1 = new **Customer**(pinkbook, "Lutfi");

    pinkbook.**addObserver**(customer1);

    Customer customer2 = new **Customer**(pinkbook, "Tuti");

    pinkbook.**addObserver**(customer2);

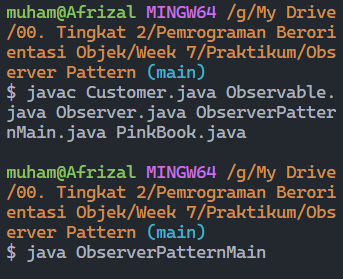
*// pinkbook.setInStock(true);*

  }

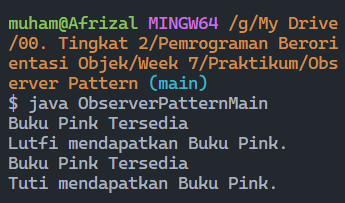
}

1. Output

* Ketika pinkbook tidak diset stock true



* Ketika pinkbook diset stock true



1. **Decorator Pattern**
2. Interface Pakaian

*/\*\**

*\* Pakaian*

*\*/*

public interface Pakaian {

  public void **pakai**();

}

1. Class Kaos

*/\*\**

*\* Kaos*

*\*/*

public class Kaos implements Pakaian {

  @Override

  public void **pakai**() {

    System.out.**println**("Jenis : Kaos");

  }

}

1. Class Celana

*/\*\**

*\* Celana*

*\*/*

public class Celana implements Pakaian {

  @Override

  public void **pakai**() {

    System.out.**println**("Jenis : Celana");

  }

}

1. Class WarnaiPakaian

*/\*\**

*\* WarnaiPakaian*

*\*/*

public abstract class WarnaiPakaian implements Pakaian {

  protected Pakaian warnai;

  public **WarnaiPakaian**(Pakaian *warnai*) {

    this.warnai = *warnai*;

  }

  @Override

  public void **pakai**() {

    warnai.**pakai**();

  }

}

1. Class WarnaiMerah

*/\*\**

*\* WarnaiMerah*

*\*/*

public class WarnaiMerah extends WarnaiPakaian {

  public **WarnaiMerah**(Pakaian *warnai*) {

    super(*warnai*);

  }

  @Override

  public void **pakai**() {

    warnai.**pakai**();

**setWarnaPakaian**(warnai);

  }

  private void **setWarnaPakaian**(Pakaian *warnai*) {

    System.out.**println**("Warna Border : Merah");

  }

}

1. Class DecoratorPatternMain

*/\*\**

*\* DecoratorPatternMain*

*\*/*

public class DecoratorPatternMain {

  public static void **main**(String[] *args*) {

    Pakaian kaos = new **Kaos**();

    Pakaian kaosMerah = new **WarnaiMerah**(new **Kaos**());

    Pakaian celanaMerah = new **WarnaiMerah**(new **Celana**());

    System.out.**println**("Kaos belum diwarnai");

    kaos.**pakai**();

    System.out.**println**("\nCelana warna merah");

    celanaMerah.**pakai**();

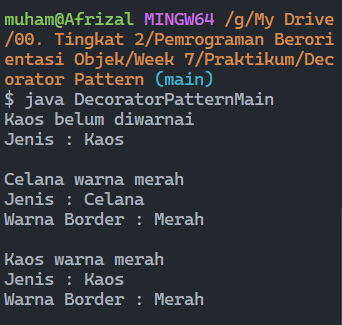
    System.out.**println**("\nKaos warna merah");

    kaosMerah.**pakai**();

  }

}

1. Output



1. **Factory Pattern**
2. Class Pegawai

*/\*\**

*\* Pegawai*

*\*/*

public class Pegawai {

  private String nama;

  private String tipe;

  private String pembayarangaji;

  public void **setNama**(String *nama*) {

    this.nama = *nama*;

  }

  public String **getNama**() {

    return nama;

  }

  public void **setTipe**(String *tipe*) {

    this.tipe = *tipe*;

  }

  public String **getTipe**() {

    return tipe;

  }

  public void **setPembayarangaji**(String *pembayarangaji*) {

    this.pembayarangaji = *pembayarangaji*;

  }

  public String **getPembayarangaji**() {

    return pembayarangaji;

  }

  @Override

  public String **toString**() {

    return (

      "Nama          : " +

      this.nama +

      "\nTipe Pegawai     : " +

      this.tipe +

      "\nPembayarangaji     : " +

      this.pembayarangaji

    );

  }

}

1. Class PegawaiTetap

*/\*\**

*\* PegawaiTetap*

*\*/*

public class PegawaiTetap extends Pegawai {

  public **PegawaiTetap**(String *nama*) {

**setNama**(*nama*);

**setTipe**("Permanen");

**setPembayarangaji**("Perbulan");

  }

}

1. Class PegawaiKontrak

*/\*\**

*\* PegawaiKontrak*

*\*/*

public class PegawaiKontrak extends Pegawai {

  public **PegawaiKontrak**(String *nama*) {

**setNama**(*nama*);

**setTipe**("Kontrak");

**setPembayarangaji**("Perjam");

  }

}

1. Class PegawaiFactory

*/\*\**

*\* PegawaiFactory*

*\*/*

public class PegawaiFactory {

  public Pegawai **buatPegawai**(String *nama*, String *tipe*) {

    switch (*tipe*) {

      case "tetap":

        return new **PegawaiTetap**(*nama*);

      case "kontrak":

        return new **PegawaiKontrak**(*nama*);

      default:

        return null;

    }

  }

}

1. Class FactoryPatternMain

*/\*\**

*\* FactoryPatternMain*

*\*/*

public class FactoryPatternMain {

  public static void **main**(String[] *args*) {

    PegawaiFactory factory = new **PegawaiFactory**();

    System.out.**println**(factory.**buatPegawai**("Lutfi", "tetap").**toString**());

    System.out.**println**(factory.**buatPegawai**("Dani", "kontrak").**toString**());

  }

}

1. Output

