|  |  |  |  |
| --- | --- | --- | --- |
|  | **Course Name: Design Patterns/Thinking LAB** | **EXPERIMENT NO. 14** | |
| **Course Code: 20CP210P**  **Faculty: Dr. Ketan Sabale** | **Branch: CSE** | **Semester: IV** |
| **(To be filled by Student)**  **Submitted by: Jangle Parth**  **Roll no: 22BCP083** | | | |

Objective: To familiarize students with standard Behavioral design patterns.

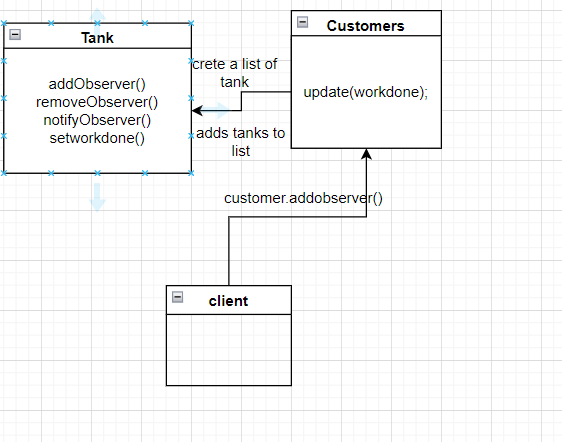
Experiment: Explain the Observer design pattern and write a program using any object-oriented programming language to demonstrate the working of Observer design pattern.

Theory: Imagine a Scenario where you have n no of customers and each customer give n no of orders it is very hard to keep all the customers keep update about their product progress so we use observer patter it updates all the customer of a particular product if there is some change in work

**Problem Statement Explanation:**

We have a class Customer which is the subject here it has property like add observer , notify observer etc. and we have object tank which has a property update in which we can update the work done on tank. The subject has a list of tanks and for each customer we can add a tank. And notify them when a change occurs

**Flowchart Explanation:**

****

**Code:**

package Observer;

import java.util.ArrayList;

import java.util.List;

class Customers {

    List<Tank> tanks = new ArrayList<>();

    String workdone;

    public void addObserver(Tank tank) {

        tanks.add(tank);

    }

    public void removeObserver(Tank tank) {

        tanks.remove(tank);

    }

    public void notifyObservers() {

        for (Tank tank : tanks) {

            tank.update(workdone);

        }

    }

    public void setworkdone(String workdone) {

        this.workdone = workdone;

        notifyObservers();

    }

}

class Tank {

    String workdone;

    public void update(String workdone) {

        this.workdone = workdone;

        System.out.println("Work Done: " + workdone);

    }

}

public class observer {

    public static void main(String[] args) {

        Customers tetrapack = new Customers();

        Tank t = new Tank();

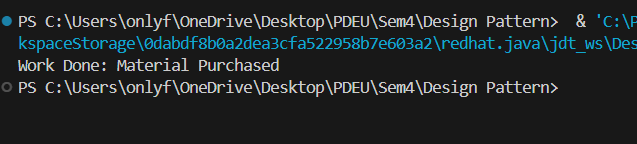
        tetrapack.addObserver(t);

        tetrapack.setworkdone("Material Purchased");

    }

}

**Output:**

****