Mediator Pattern

The **Mediator Pattern** is a **behavioural pattern** that is very similar to the **observer pattern**, except that it deals with **many-to-many relationships**. For cases where multiple observers and multiple subjects need to communicate with each other, we create an intermediary **mediator class** which handles the communication.

A good example of a use case for the mediator pattern is a chatroom where user’s can both send and receive messages.

import java.util.ArrayList;  
import java.util.*List*;  
  
public class ChatMediator {  
 private *List*<User> users = new ArrayList<>();  
  
 public void addUser(User user){  
 this.users.add(user);  
 }  
  
 public void sendMessage(String msg, User receiver) {  
 for(User user : this.users){  
 if(user == receiver){  
 user.receive(msg);  
 }  
 }  
 }  
  
}

public class User {  
 protected ChatMediator mediator;  
  
 public User(ChatMediator mediator){  
 this.mediator = mediator;  
 }  
  
 public void send(String msg, User receiver){  
 mediator.sendMessage(msg, receiver);  
 }

public void receive(String msg) {  
 System.*out*.println("Received Message:" + msg);  
 }  
}  
  
public class Main {  
 public static void main(String[] args) {  
 ChatMediator mediator = new ChatMediator();  
 User user1 = new User(mediator);  
 User user2 = new User(mediator);  
 User user3 = new User(mediator);  
 User user4 = new User(mediator);  
 mediator.addUser(user1);  
 mediator.addUser(user2);  
 mediator.addUser(user3);  
 mediator.addUser(user4);  
  
 user1.send("Hello!", user2);  
 }  
}

JAVA