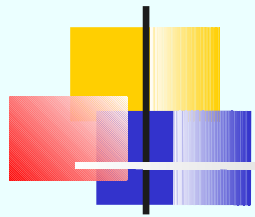




Lập trình mạng

Thiết kế theo mô hình MVC

Giảng viên: TS. Nguyễn Mạnh Hùng
Học viện Công nghệ Bưu chính Viễn thông (PTIT)



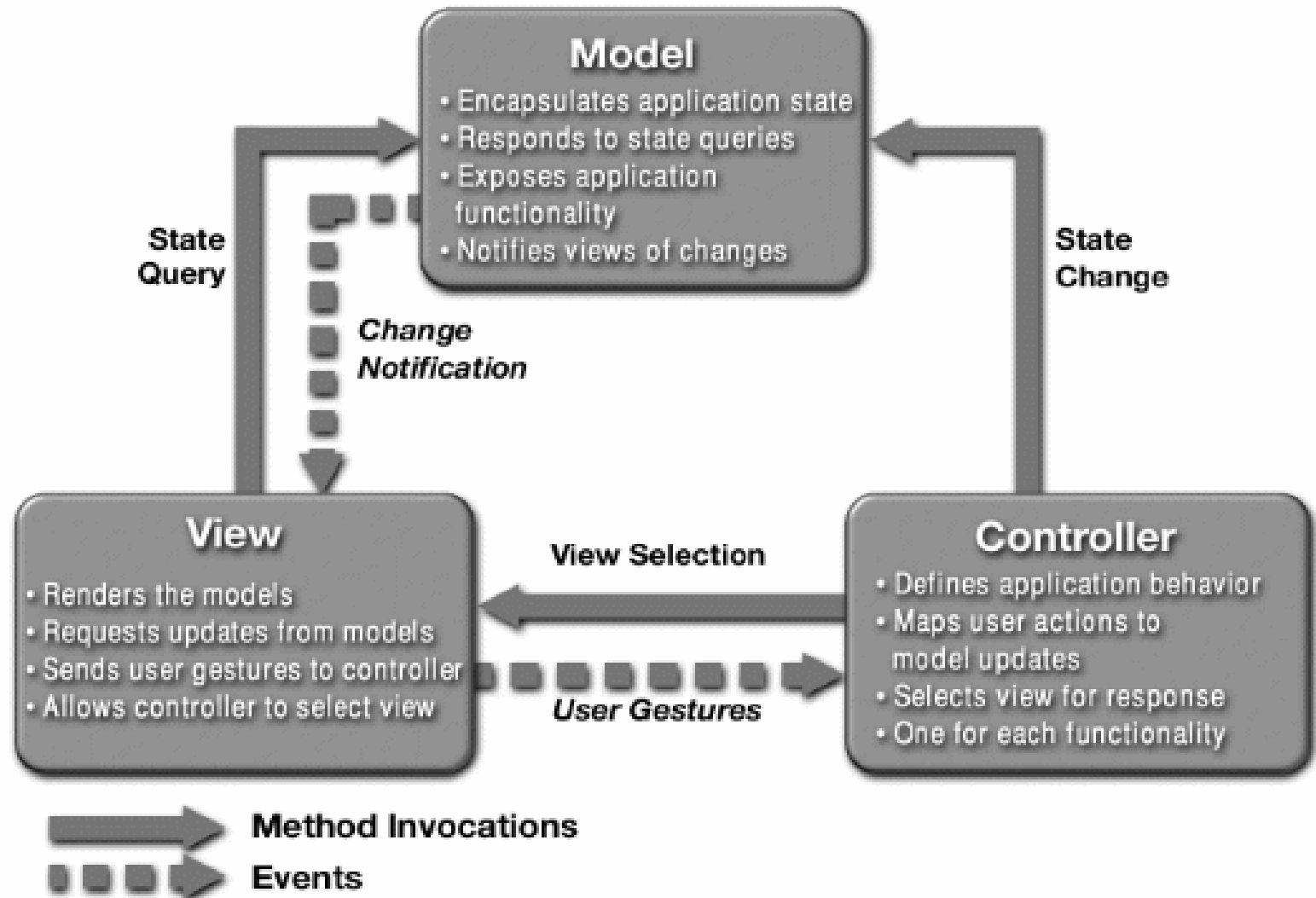
Nội dung

- Mô hình MVC tổng quan
- Mô hình MVC cải tiến
- Ví dụ
- Bài tập



Mô hình MVC

Mô hình MVC (1)





Mô hình MVC (2)

M - model:

- Đóng gói dữ liệu, thông tin
- Chức năng biểu diễn, vận chuyển thông tin để trình diễn (view) và xử lý (control)



Mô hình MVC (3)

C - control:

- Định nghĩa các hành vi, hoạt động, xử lý của hệ thống
- Đối chiếu hành động của user (nhận từ view), vào tập chức năng để xử lý, đồng thời chọn hành động đưa view ra để show

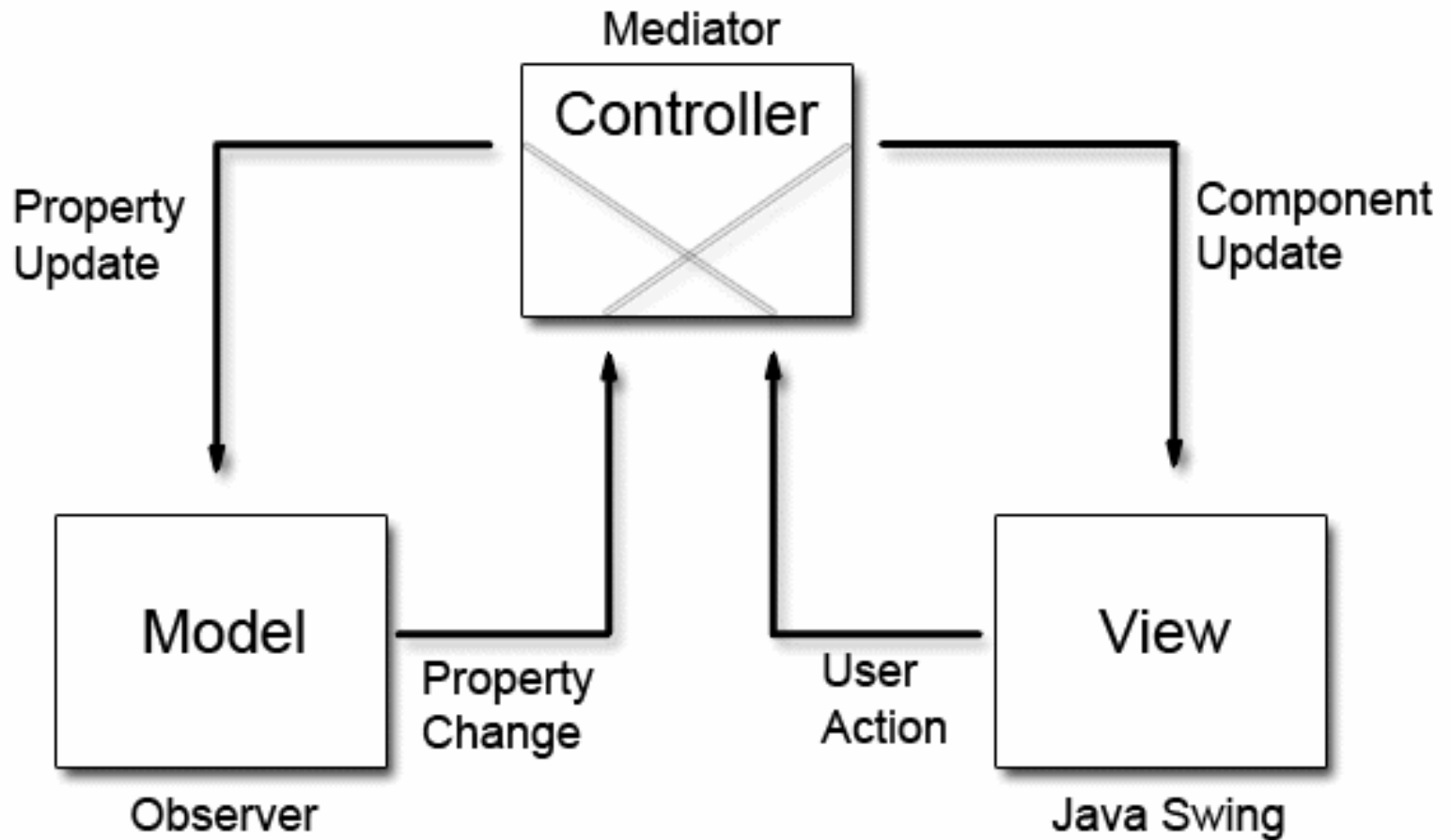


Mô hình MVC (4)

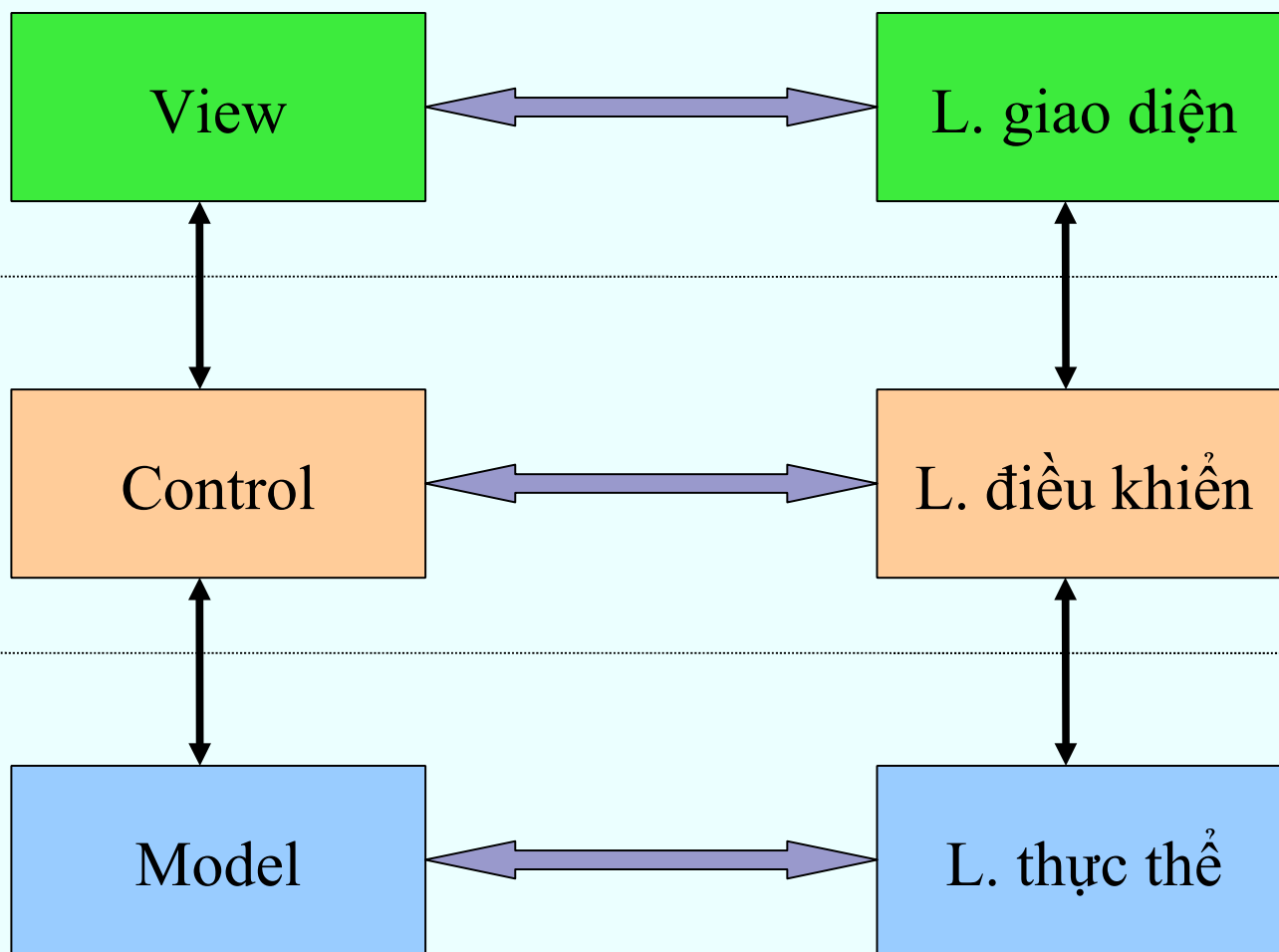
V - view:

- Giao diện với người sử dụng
- Show các kết quả xử lý của tầng control
- Thu nhận các hoạt động, yêu cầu của người sử dụng và chuyển cho tầng control xử lý

MVC cải tiến (1)



MVC cải tiến (2)





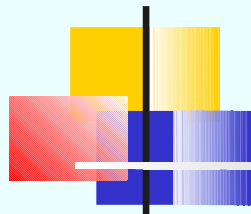
Các lớp thực thể

- Đóng gói dữ liệu, thông tin
- Chỉ chứa các thuộc tính và các phương thức truy cập các thuộc tính (javaBean)
- Chức năng biểu diễn, vận chuyển thông tin để trình diễn (view) và xử lý (control)



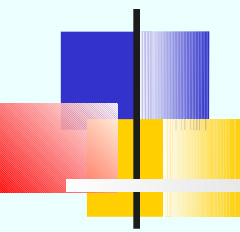
Các lớp điều khiển

- Cập nhật thông tin vào DB (thông tin chứa trong các thực thể)
- Thực hiện các tính toán, xử lý trung gian
- Đối chiếu hành động của user (nhận từ view), vào tập chức năng để xử lý, đồng thời chọn hành động đưa view ra để show



Các lớp giao diện

- Các frame, cửa sổ của ứng dụng (javaSwing)
- Các trang giao diện web: html, jsp
- Các bảng, mẫu biểu, báo cáo in ra

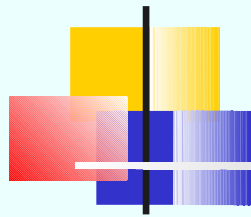


Ví dụ:
điều khiển đăng nhập từ dòng lệnh



Login: Model

```
public class LoginModel {  
    String userName;  
    String password;  
  
    public LoginModel() {}  
  
    public String getPassword() {  
        return password;  
    }  
  
    public void setPassword(String password) {  
        this.password = password;  
    }  
  
    public String getUserName() {  
        return userName;  
    }  
  
    public void setUserName(String userName) {  
        this.userName = userName;  
    }  
}
```



Login: View (1)

```
import java.io.DataInputStream;
import java.io.IOException;

public class LoginView {
    LoginModel user;

    public LoginView(LoginModel user) {
        this.user = user;
    }

    public void showMessage(String smg) {
        System.out.println(smg);
    }
}
```



Login: View (2)

```
public void getUserInfo() {  
    try{  
        Scanner input = new Scanner(System.in);  
  
        System.out.print("Username: ");  
        user.setUserName(input.nextLine());  
        System.out.print("Password: ");  
        user.setPassword(input.nextLine());  
  
        input.close();  
    } catch(IOException e) {  
        System.out.println(e);  
    }  
}
```

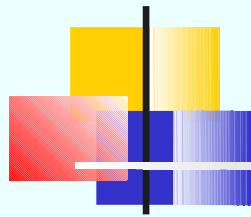



Login: Control (1)

```
public class LoginControl {
    LoginModel user;
    LoginView view;

    public LoginControl(LoginModel user, LoginView view){
        this.user = user;
        this.view = view;

        while(true){
            view.getUserInfo();
            if(checkLogin()){
                view.showMessage("success!");
                break;
            }else{
                view.showMessage("wrong username or password!");
            }
        }
    }
}
```



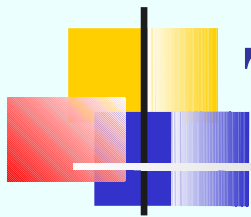
Login: Control (2)

```
private boolean checkLogin() {  
    if ((user.getUserName().equals("sa"))  
        &&(user.getPassword().equals("sa"))){  
        return true;  
    }  
    return false;  
}  
}
```



Login: main

```
public class LoginMVC {  
  
    public static void main(String[] args){  
        LoginModel user = new LoginModel();  
        LoginView view = new LoginView(user);  
        LoginControl control = new LoginControl(user, view);  
    }  
}
```



Thiết kế hệ thống theo MVC

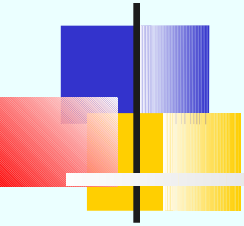
Có thể áp dụng một số dạng mô hình MVC phổ biến:

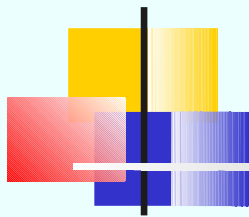
- Mô hình MVC dùng thực thể thuần (cổ điển)
- Mô hình MVC dùng bean
- Mô hình MVC cải tiến (hiện đại)

Lưu ý:

Với bài tập lớn, các nhóm nên chọn thiết kế theo 1 trong 3 dạng kiến trúc trên.

Thiết kế theo mô hình MVC với thực thể thuần





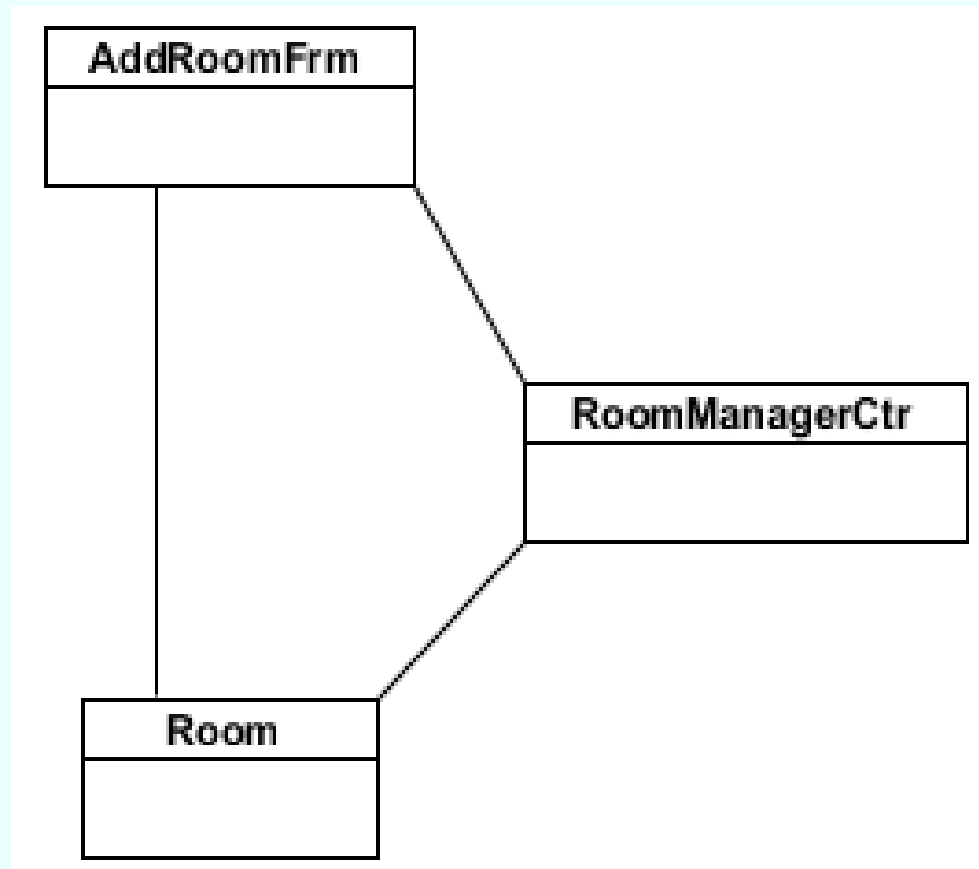
MVC với thực thể thuần (1)

Đặc trưng:

- Lớp thực thể chỉ chứa các thuộc tính và các phương thức get/set cho mỗi thuộc tính (còn gọi là các lớp thực thể thuần)
- Các thao tác liên quan đến CSDL đều đặt trong lớp điều khiển (dạng lớp DAO – Data Access Object)

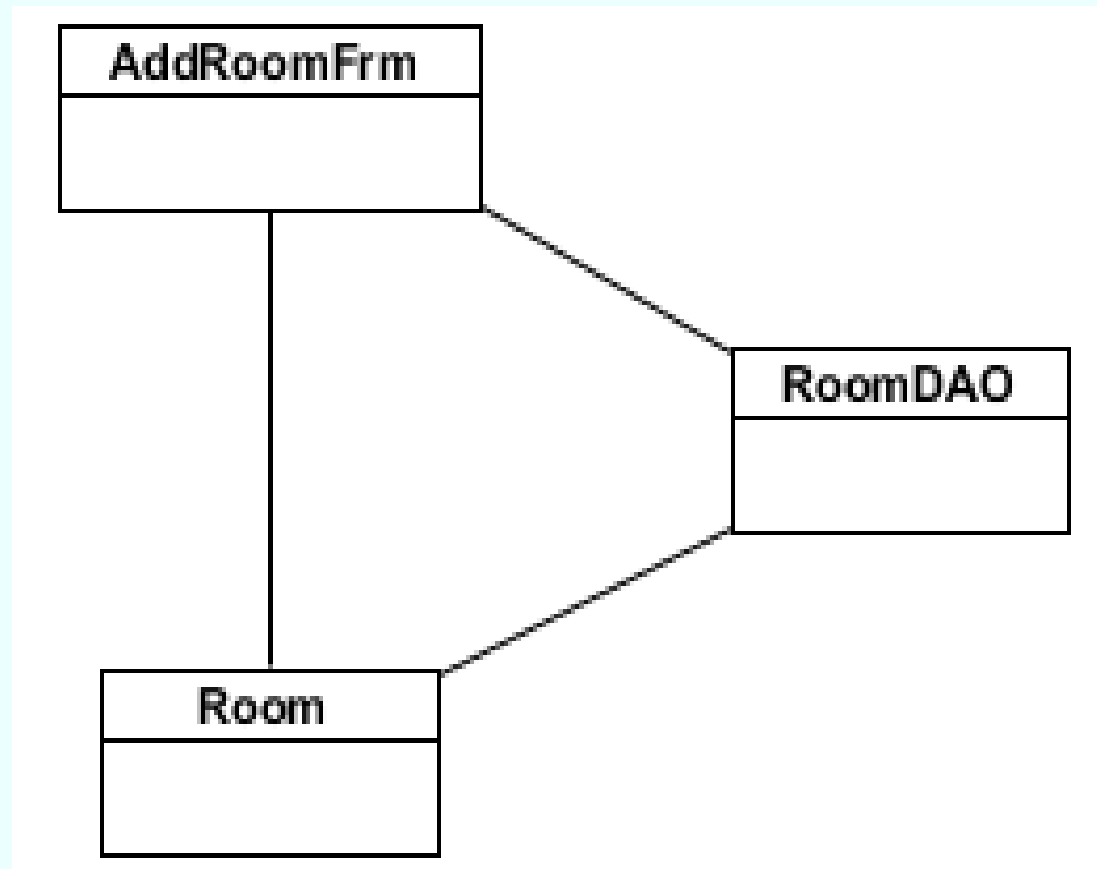
MVC với thực thể thuần (2)

Ví dụ modul quản lí phòng của Manager, sơ đồ lớp cuối pha phân tích của chức năng thêm phòng:



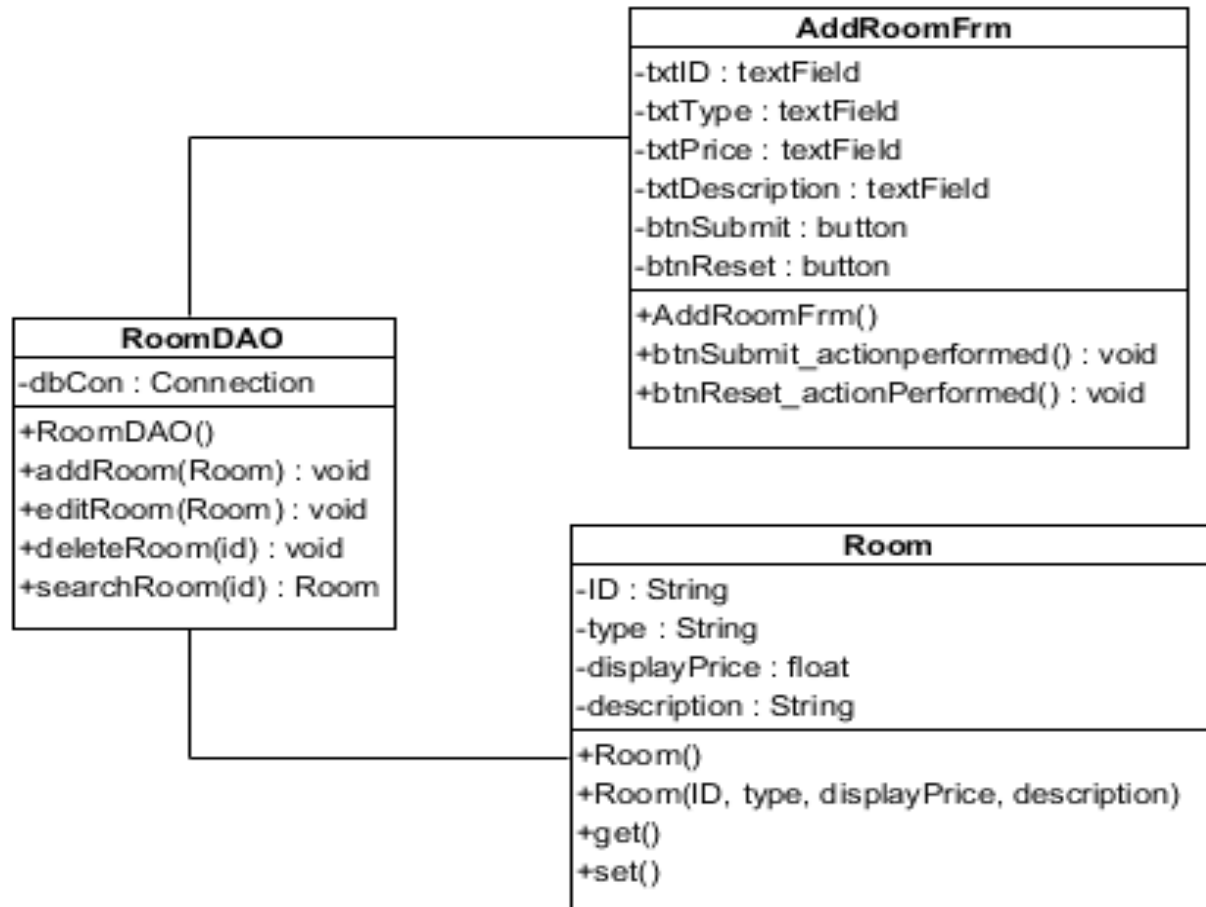
MVC với thực thể thuần (3)

Ví dụ chức năng thêm phòng của Manager, sơ đồ lớp theo MVC dùng thực thể thuần:



MVC với thực thể thuần (4)

Kết quả thu được sơ đồ lớp như sau:





Lớp Room (1)

```
package mvcPure;
```

```
public class Room {  
    private String id;  
    private String name;  
    private String type;  
    private float displayPrice;  
    private String description;
```

```
    public Room() {  
        super();  
    }
```

```
    public Room(String id, String name, String type,  
        float displayPrice, String description) {  
        super();  
        this.id = id;  
        this.name = name;  
        this.type = type;  
        this.displayPrice = displayPrice;  
        this.description = description;  
    }
```



Lớp Room (2)

```
public String getId() {
    return id;
}
public void setId(String id) {
    this.id = id;
}
public String getName() {
    return name;
}
public void setName(String name) {
    this.name = name;
}
public String getType() {
    return type;
}
public void setType(String type) {
    this.type = type;
}
public float getDisplayPrice() {
    return displayPrice;
}
public void setDisplayPrice(float displayPrice) {
    this.displayPrice = displayPrice;
}
public String getDescription() {
    return description;
}
public void setDescription(String description) {
    this.description = description;
}
```



Lớp RoomDAO (1)

```
package mvcPure;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

public class RoomDAO {
    private Connection con;

    public RoomDAO(){
        String dbUrl = "jdbc:mysql://localhost:3306/hotel";
        String dbClass = "com.mysql.jdbc.Driver";

        try {
            Class.forName(dbClass);
            con = DriverManager.getConnection (dbUrl,
                                                "root", "12345678");
        }catch(Exception e) {
            e.printStackTrace();
        }
    }
}
```



Lớp RoomDAO (2)

```
public void addRoom(Room room){
    String sql = "INSERT INTO tblRoom(id, name, type, displayPrice,
        description) VALUES(?,?,?,?,?)";
    try{
        PreparedStatement ps = con.prepareStatement(sql);
        ps.setString(1, room.getId());
        ps.setString(2, room.getName());
        ps.setString(3, room.getType());
        ps.setFloat(4, room.getDisplayPrice());
        ps.setString(5, room.getDescription());

        ps.executeUpdate();
    }catch(Exception e){
        e.printStackTrace();
    }
}
```



Lớp AddRoomFrm (1)

```
package mvcPure;
```

```
import java.awt.GridLayout;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.awt.event.WindowAdapter;  
import java.awt.event.WindowEvent;
```

```
import javax.swing.JButton;  
import javax.swing.JFrame;  
import javax.swing.JLabel;  
import javax.swing.JOptionPane;  
import javax.swing.JPanel;  
import javax.swing.JTextField;
```

```
public class AddRoomFrm extends JFrame implements ActionListener{  
    private JTextField txtID;  
    private JTextField txtName;  
    private JTextField txtType;  
    private JTextField txtDisplayPrice;  
    private JTextField txtDescription;  
    private JButton btnSubmit;  
    private JButton btnReset;
```



Lớp AddRoomFrm (2)

```
public AddRoomFrm(){
    super("Room management pure-MVC");
    txtID = new JTextField(15);
    txtName = new JTextField(15);
    txtType = new JTextField(15);
    txtDisplayPrice = new JTextField(15);
    txtDescription = new JTextField(15);
    btnSubmit = new JButton("Submit");
    btnReset = new JButton("Reset");

    JPanel content = new JPanel();
    content.setLayout(new GridLayout(6,2));
    content.add(new JLabel("ID:")); content.add(txtID);
    content.add(new JLabel("Name:")); content.add(txtName);
    content.add(new JLabel("Type:")); content.add(txtType);
    content.add(new JLabel("Display price:")); content.add(txtDisplayPrice);
    content.add(new JLabel("Description:")); content.add(txtDescription);
    content.add(btnReset); content.add(btnSubmit);
    btnSubmit.addActionListener(this);
    btnReset.addActionListener(this);
    this.setContentPane(content);
    this.pack();

    this.addWindowListener(new WindowAdapter(){
        public void windowClosing(WindowEvent e){
            System.exit(0);
        }
    });
}
```



Lớp AddRoomFrm (3)

```
public void actionPerformed(ActionEvent e) {
    JButton btn = (JButton) e.getSource();
    if(btn.equals(btnSubmit)){
        btnSubmit_actionPerformed();
    }else if(btn.equals(btnReset)){
        btnReset_actionPerformed();
    }
}

public void btnSubmit_actionPerformed() {
    Room room = new Room();
    room.setId(txtID.getText());
    room.setName(txtName.getText());
    room.setType(txtType.getText());
    room.setDisplayPrice(Float.parseFloat(txtDisplayPrice.getText()));
    room.setDescription(txtDescription.getText());

    RoomDAO rd = new RoomDAO();
    rd.addRoom(room);
    JOptionPane.showMessageDialog(this, "Add room successfullly!");
}

public void btnReset_actionPerformed() {
    txtID.setText(""); txtName.setText("");
    txtType.setText(""); txtDisplayPrice.setText("");
    txtDescription.setText("");
}
}
```




Lớp Test

```
package mvcPure;

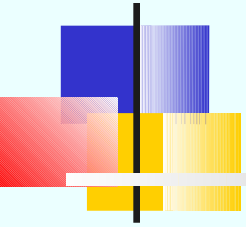
public class Test {

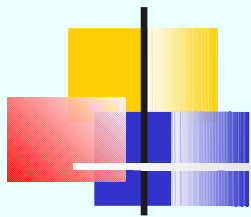
    public static void main(String[] args) {
        AddRoomFrm arf = new AddRoomFrm();
        arf.setVisible(true);
    }
}
```

Lưu ý trước khi chạy, phải:

- Cài đặt CSDL và bật server MySQL
- Add driver của Jdbc mysql vào library của project

Thiết kế theo mô hình MVC với thực thể bean





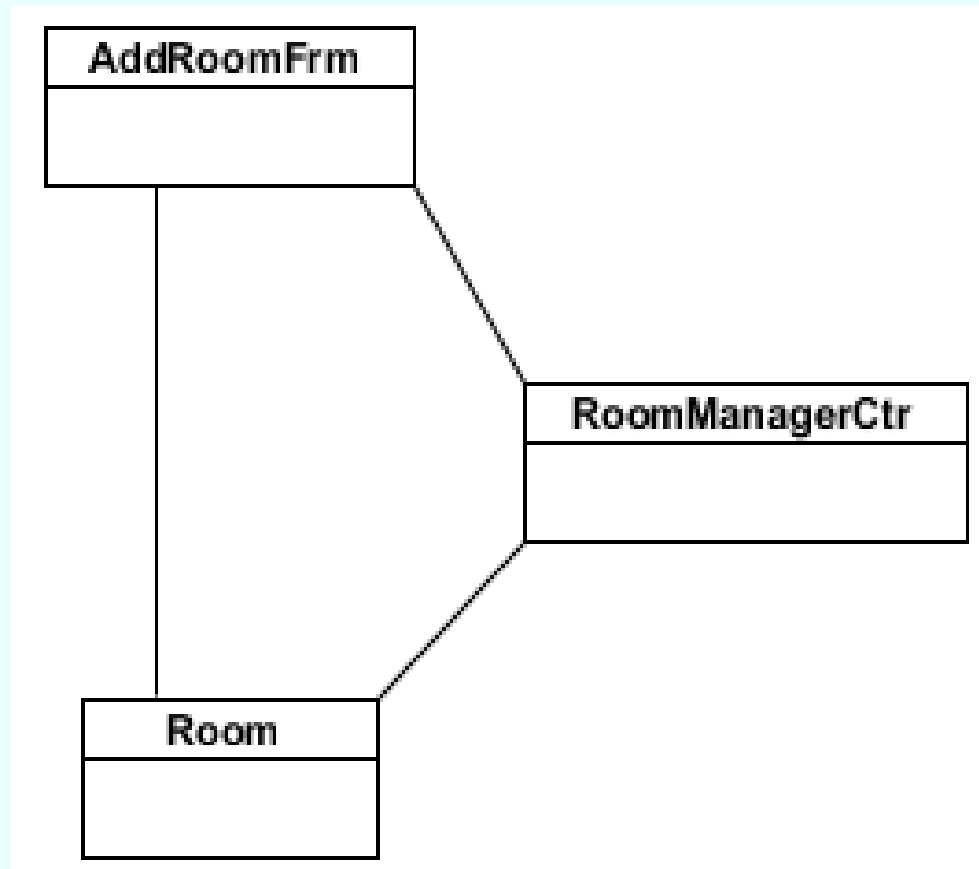
MVC với thực thể bean (1)

Đặc trưng:

- Lớp thực thể chứa các thuộc tính và các phương thức get/set cho mỗi thuộc tính, và
- Các thao tác liên quan đến CSDL mà liên quan đến lớp thực thể nào thì đều đặt trong lớp điều khiển đó
- Các lớp thực thể kiểu này được gọi là lớp bean
- Trong nhiều trường hợp, không còn cần đến lớp điều khiển nữa vì lớp bean đã kiêm luôn vai trò của lớp điều khiển

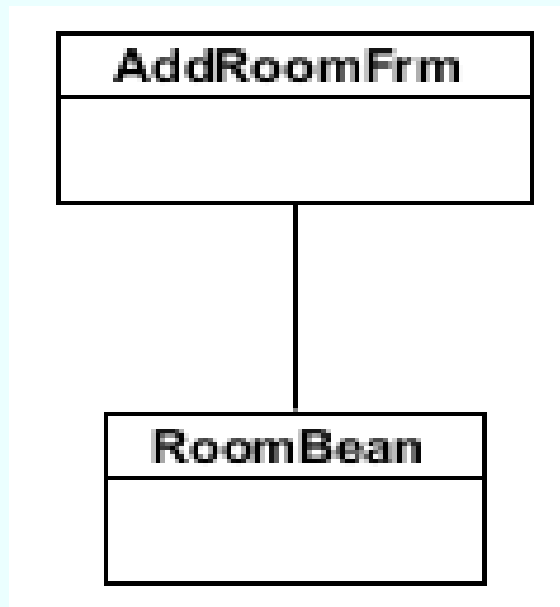
MVC với thực thể bean (2)

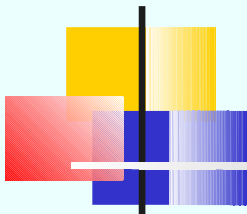
Ví dụ modul quản lí phòng của Manager, sơ đồ lớp cuối pha phân tích của chức năng thêm phòng:



MVC với thực thể bean (3)

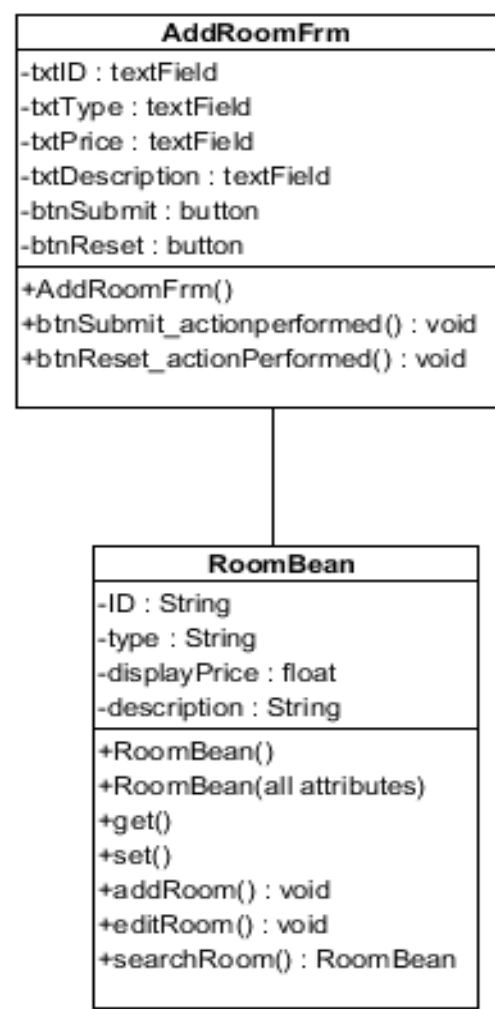
Ví dụ chức năng thêm phòng của Manager, sơ đồ lớp theo MVC dùng thực thể bean:





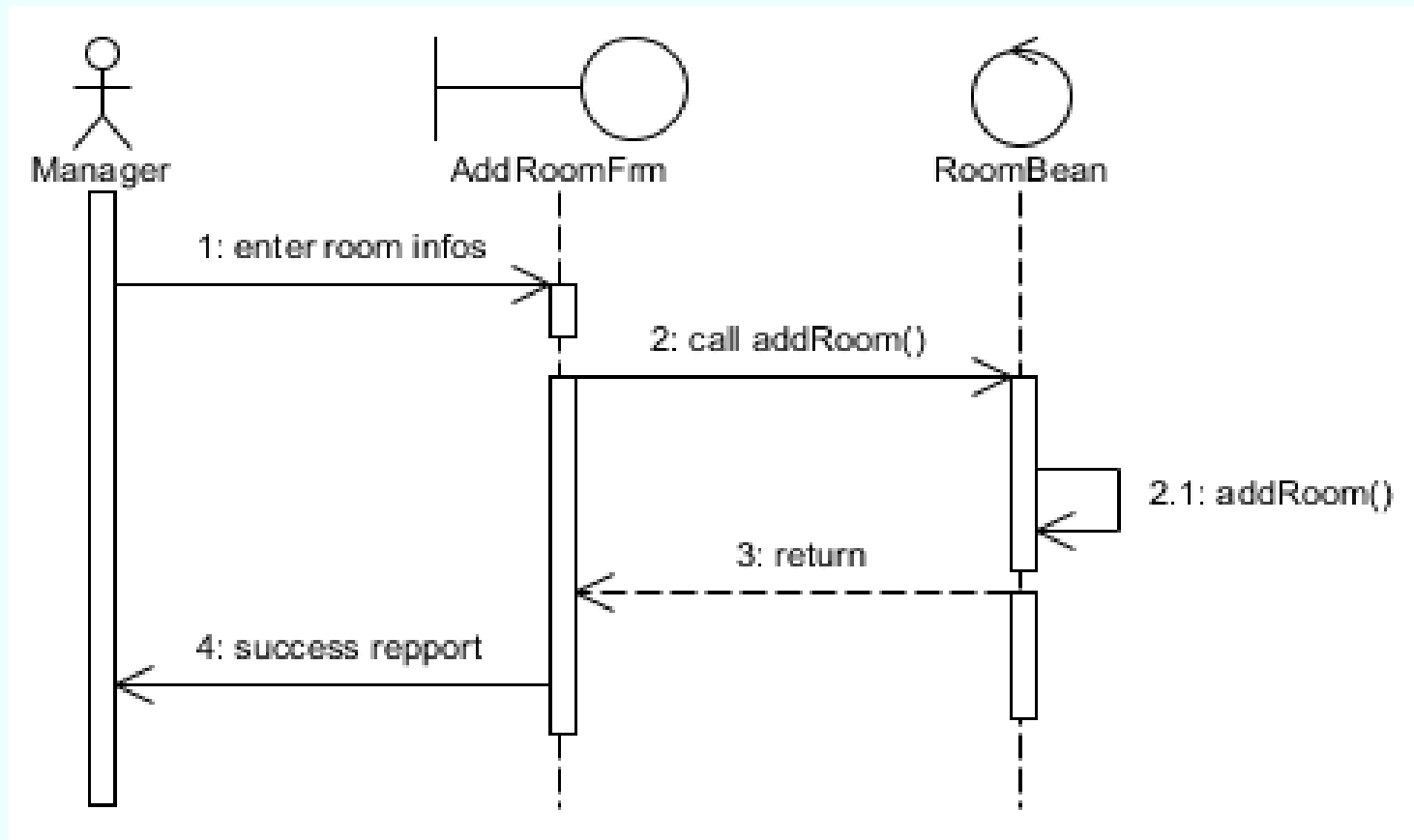
Hoàn thiện sơ đồ lớp

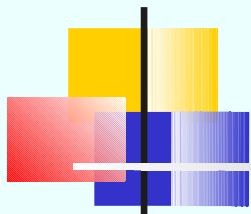
Kết quả thu được sơ đồ lớp sau khi áp dụng các nguyên lí A, B, và C như sau:



Sơ đồ tuần tự pha thiết kế

Sơ đồ tuần tự cho cách thiết kế dùng bean:





Lớp RoomBean (1)

```
package mvcBean;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

public class RoomBean {
    private String id;
    private String name;
    private String type;
    private float displayPrice;
    private String description;

    public RoomBean() {
        super();
    }

    public RoomBean(String id, String name, String type, float
        DisplayPrice, String description) {
        super();
        this.id = id;
        this.name = name;
        this.type = type;
        this.displayPrice = displayPrice;
        this.description = description;
    }
}
```




Lớp RoomBean (2)

```
public String getId() {  
    return id;  
}  
public void setId(String id) {  
    this.id = id;  
}  
public String getName() {  
    return name;  
}  
public void setName(String name) {  
    this.name = name;  
}  
public String getType() {  
    return type;  
}  
public void setType(String type) {  
    this.type = type;  
}  
public float getDisplayPrice() {  
    return displayPrice;  
}  
public void setDisplayPrice(float displayPrice) {  
    this.displayPrice = displayPrice;  
}  
public String getDescription() {  
    return description;  
}  
public void setDescription(String description) {  
    this.description = description;  
}
```



Lớp RoomBean (3)

```
public void addRoom(){
    String dbUrl = "jdbc:mysql://localhost:3306/hotel";
    String dbClass = "com.mysql.jdbc.Driver";
    String sql = "INSERT INTO tblRoom(id, name, type, displayPrice,
        description) VALUES(?,?,?,?,?)";
    try{
        Class.forName(dbClass);
        Connection con = DriverManager.getConnection (
            dbUrl, "root", "12345678");
        PreparedStatement ps = con.prepareStatement(sql);
        ps.setString(1, getId());
        ps.setString(2, getName());
        ps.setString(3, getType());
        ps.setFloat(4, getDisplayPrice());
        ps.setString(5, getDescription());

        ps.executeUpdate();
        con.close();
    }catch(Exception e){
        e.printStackTrace();
    }
}
```



Lớp AddRoomFrm (1)

```
package mvcBean;
```

```
import java.awt.GridLayout;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.awt.event.WindowAdapter;  
import java.awt.event.WindowEvent;
```

```
import javax.swing.JButton;  
import javax.swing.JFrame;  
import javax.swing.JLabel;  
import javax.swing.JOptionPane;  
import javax.swing.JPanel;  
import javax.swing.JTextField;
```

```
public class AddRoomFrm extends JFrame implements ActionListener{  
    private JTextField txtID;  
    private JTextField txtName;  
    private JTextField txtType;  
    private JTextField txtDisplayPrice;  
    private JTextField txtDescription;  
    private JButton btnSubmit;  
    private JButton btnReset;
```



Lớp AddRoomFrm (2)

```
public AddRoomFrm(){
    super("Room management pure-MVC");
    txtID = new JTextField(15);
    txtName = new JTextField(15);
    txtType = new JTextField(15);
    txtDisplayPrice = new JTextField(15);
    txtDescription = new JTextField(15);
    btnSubmit = new JButton("Submit");
    btnReset = new JButton("Reset");

    JPanel content = new JPanel();
    content.setLayout(new GridLayout(6,2));
    content.add(new JLabel("ID:")); content.add(txtID);
    content.add(new JLabel("Name:")); content.add(txtName);
    content.add(new JLabel("Type:")); content.add(txtType);
    content.add(new JLabel("Display price:")); content.add(txtDisplayPrice);
    content.add(new JLabel("Description:")); content.add(txtDescription);
    content.add(btnReset); content.add(btnSubmit);
    btnSubmit.addActionListener(this);
    btnReset.addActionListener(this);
    this.setContentPane(content);
    this.pack();

    this.addWindowListener(new WindowAdapter(){
        public void windowClosing(WindowEvent e){
            System.exit(0);
        }
    });
}
```



Lớp AddRoomFrm (3)

```
public void actionPerformed(ActionEvent e) {
    JButton btn = (JButton) e.getSource();
    if(btn.equals(btnSubmit)){
        btnSubmit_actionPerformed();
    }else if(btn.equals(btnReset)){
        btnReset_actionPerformed();
    }
}

public void btnSubmit_actionPerformed() {
    RoomBean room = new RoomBean();
    room.setId(txtID.getText());
    room.setName(txtName.getText());
    room.setType(txtType.getText());
    room.setDisplayPrice(Float.parseFloat(txtDisplayPrice.getText()));
    room.setDescription(txtDescription.getText());

    room.addRoom();
    JOptionPane.showMessageDialog(this, "Add room successfullly!");
}

public void btnReset_actionPerformed() {
    txtID.setText(""); txtName.setText("");
    txtType.setText(""); txtDisplayPrice.setText("");
    txtDescription.setText("");
}
}
```



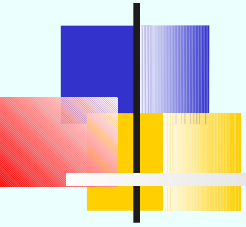
Lớp Test

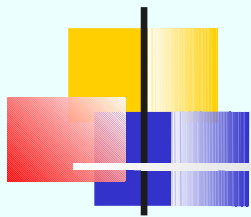
```
package mvcBean;  
  
public class Test {  
  
    public static void main(String[] args) {  
        AddRoomFrm arf = new AddRoomFrm();  
        arf.setVisible(true);  
    }  
}
```

Lưu ý trước khi chạy, phải:

- Cài đặt CSDL và bật server MySQL
- Add driver của Jdbc mysql vào library của project

Thiết kế theo mô hình MVC cải tiến (hiện đại)





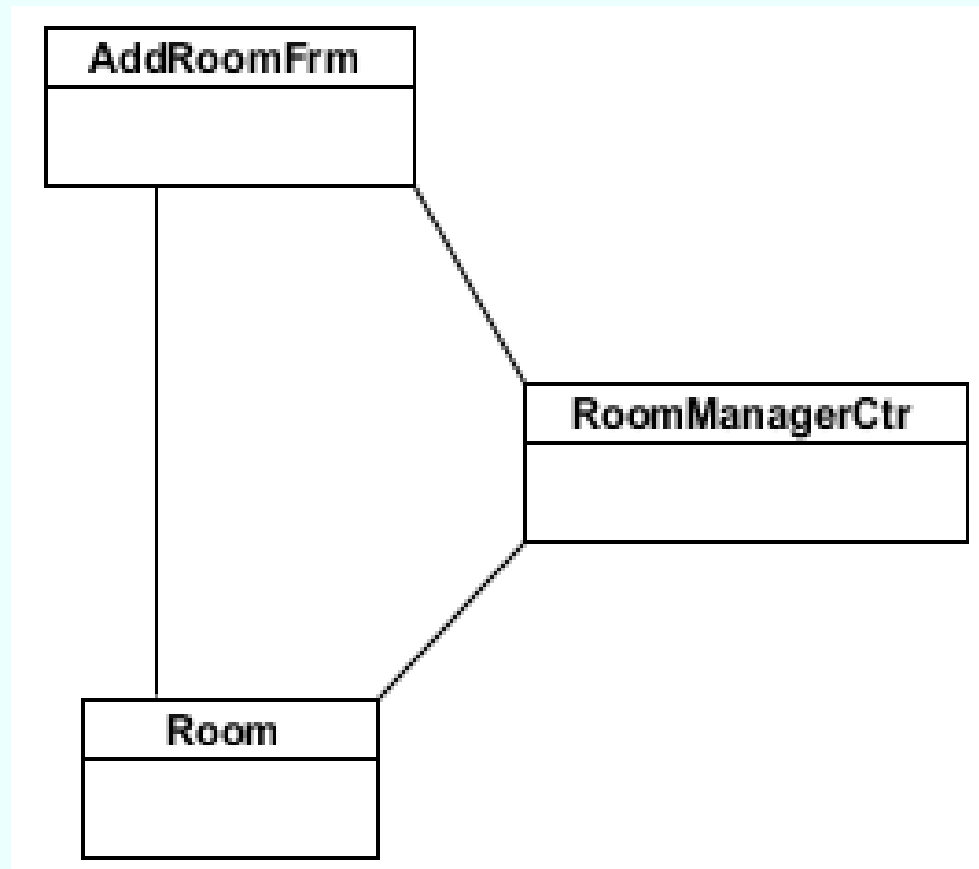
MVC cải tiến (1)

Đặc trưng:

- Lớp thực thể thuần
- Các thao tác liên quan đến CSDL đều đặt trong lớp điều khiển
- Các lớp điều khiển giành quyền điều khiển toàn bộ các lớp view và control

MVC cải tiến (2)

Ví dụ modul quản lí phòng của Manager, sơ đồ lớp cuối pha phân tích của chức năng thêm phòng:

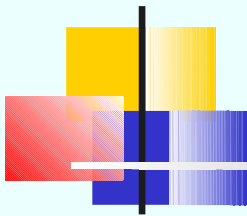




MVC cải tiến (3)

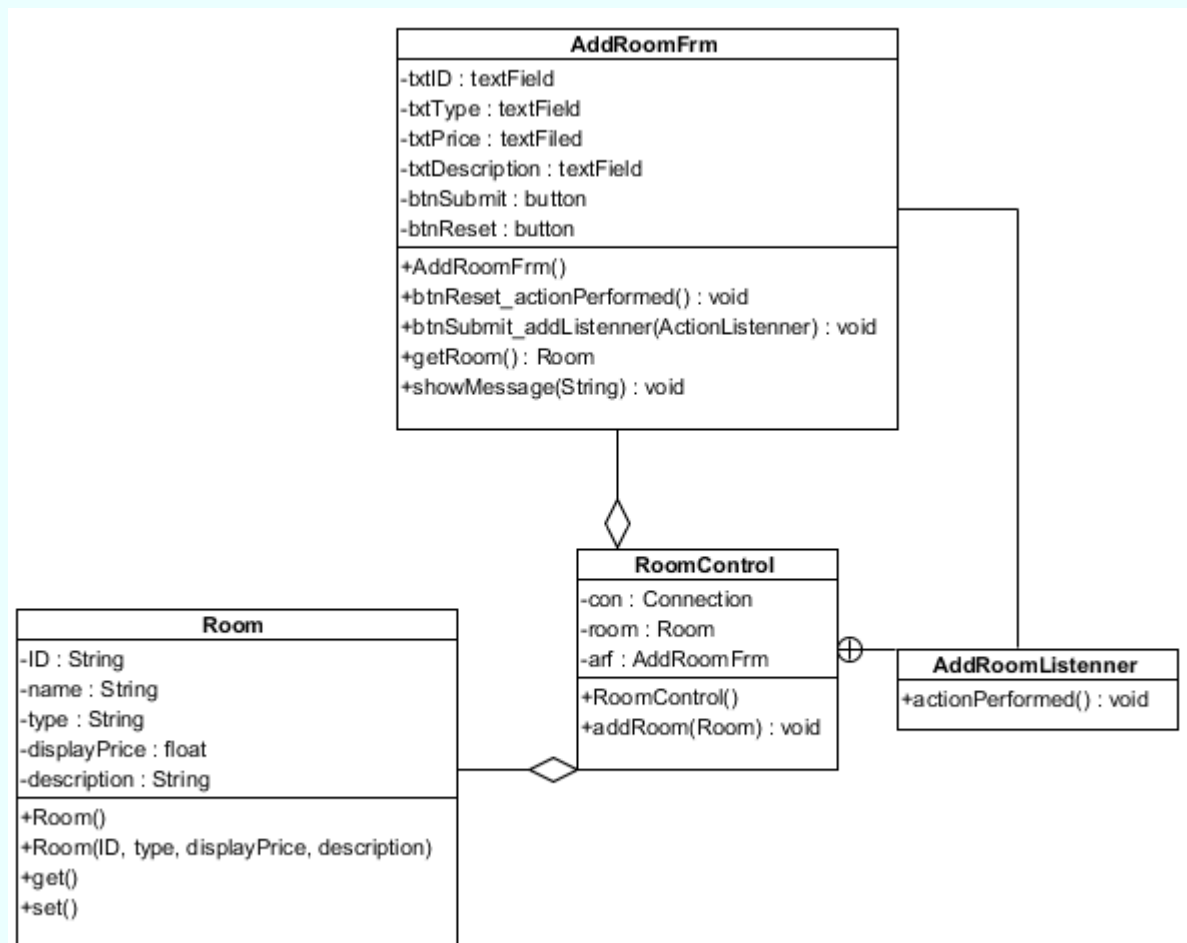
Ví dụ chức năng thêm phòng của Manager, sơ đồ lớp theo MVC cải tiến:

- Khi nút Add trên form của lớp biên AddRoomFrm bị click thì lớp biên này không được xử lí gì mà phải truyền sự kiện này xuống cho lớp điều khiển
- Để làm được việc này, trên lớp biên có phương thức truyền quyền điều khiển cho lớp điều khiển, và
- Trong lớp điều khiển có một lớp nội tại (inner class) dùng để nhận sự kiện do lớp biên truyền xuống để xử lí



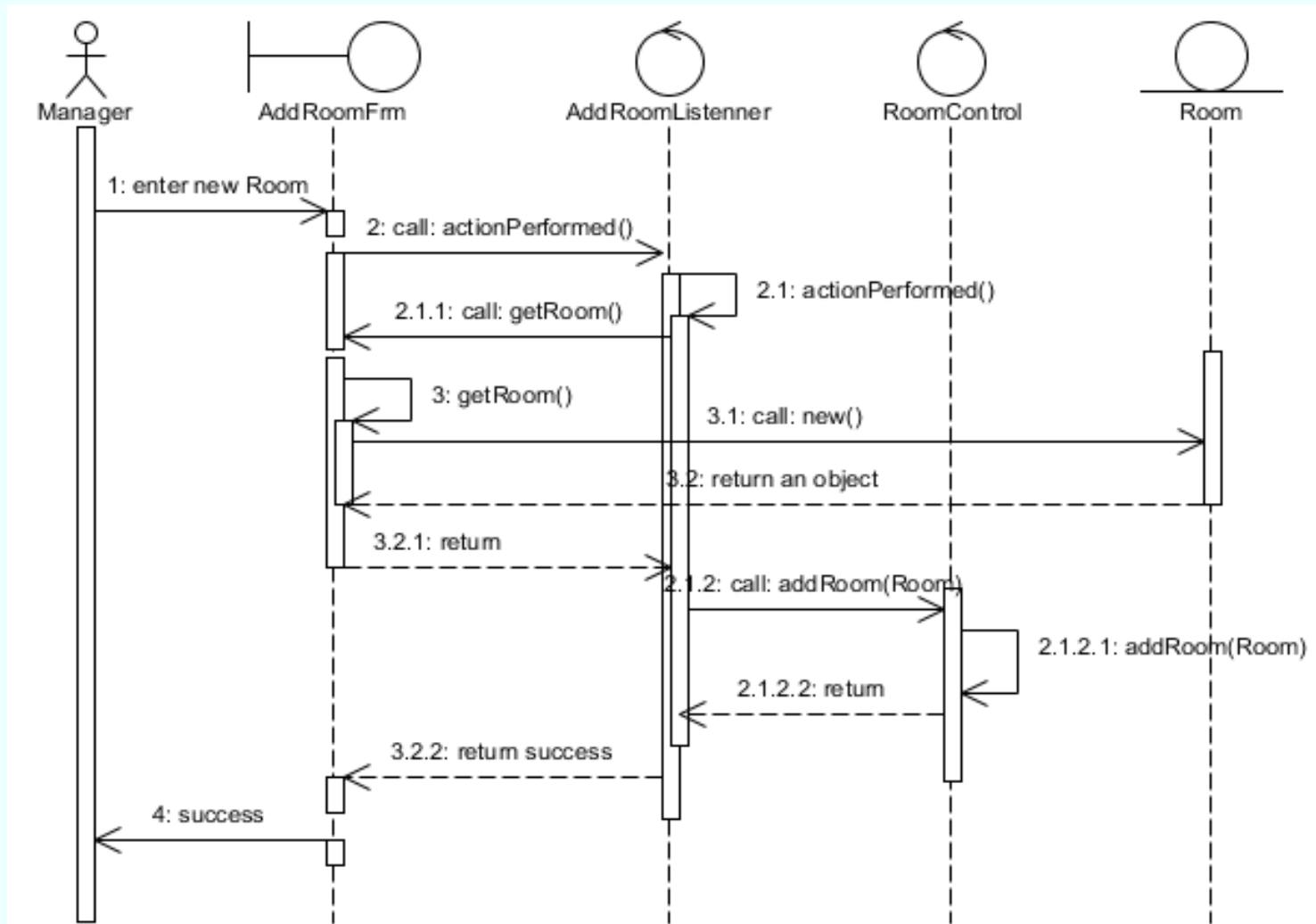
Hoàn thiện sơ đồ lớp

Kết quả thu được sơ đồ lớp sau khi áp dụng các nguyên lý A, B, và C như sau:



Sơ đồ tuần tự pha thiết kế

Sơ đồ tuần tự cho cách thiết kế dùng MVC cải tiến:





Lớp Room (1)

```
package mvcNew;
```

```
public class Room {  
    private String id;  
    private String name;  
    private String type;  
    private float displayPrice;  
    private String description;
```

```
    public Room() {  
        super();  
    }
```

```
    public Room(String id, String name, String type,  
        float displayPrice, String description) {  
        super();  
        this.id = id;  
        this.name = name;  
        this.type = type;  
        this.displayPrice = displayPrice;  
        this.description = description;  
    }
```



Lớp Room (2)

```
public String getId() {
    return id;
}
public void setId(String id) {
    this.id = id;
}
public String getName() {
    return name;
}
public void setName(String name) {
    this.name = name;
}
public String getType() {
    return type;
}
public void setType(String type) {
    this.type = type;
}
public float getDisplayPrice() {
    return displayPrice;
}
public void setDisplayPrice(float displayPrice) {
    this.displayPrice = displayPrice;
}
public String getDescription() {
    return description;
}
public void setDescription(String description) {
    this.description = description;
}
```



Lớp AddRoomFrm (1)

```
package mvcNew;

import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextField;

public class AddRoomFrm extends JFrame implements ActionListener{
    private JTextField txtID;
    private JTextField txtName;
    private JTextField txtType;
    private JTextField txtDisplayPrice;
    private JTextField txtDescription;
    private JButton btnSubmit;
    private JButton btnReset;
```



Lớp AddRoomFrm (2)

```
public AddRoomFrm(){
    super("Room management pure-MVC");
    txtID = new JTextField(15);
    txtName = new JTextField(15);
    txtType = new JTextField(15);
    txtDisplayPrice = new JTextField(15);
    txtDescription = new JTextField(15);
    btnSubmit = new JButton("Submit");
    btnReset = new JButton("Reset");

    JPanel content = new JPanel();
    content.setLayout(new GridLayout(6,2));
    content.add(new JLabel("ID:")); content.add(txtID);
    content.add(new JLabel("Name:")); content.add(txtName);
    content.add(new JLabel("Type:")); content.add(txtType);
    content.add(new JLabel("Display price:")); content.add(txtDisplayPrice);
    content.add(new JLabel("Description:")); content.add(txtDescription);
    content.add(btnReset); content.add(btnSubmit);
    btnSubmit.addActionListener(this);
    btnReset.addActionListener(this);
    this.setContentPane(content);
    this.pack();

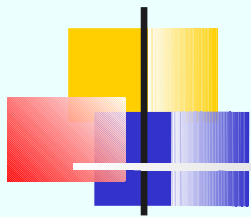
    this.addWindowListener(new WindowAdapter(){
        public void windowClosing(WindowEvent e){
            System.exit(0);
        }
    });
}
```




Lớp AddRoomFrm (3)

```
public void actionPerformed(ActionEvent e) {  
    JButton btn = (JButton) e.getSource();  
    if(btn.equals(btnReset)){  
        btnReset_actionPerformed();  
    }  
}
```

```
public void btnReset_actionPerformed() {  
    txtID.setText("");  
    txtName.setText("");  
    txtType.setText("");  
    txtDisplayPrice.setText("");  
    txtDescription.setText("");  
}
```



Lớp AddRoomFrm (4)

```
public Room getRoom(){
    Room room = new Room();
    room.setId(txtID.getText());
    room.setName(txtName.getText());
    room.setType(txtType.getText());
    room.setDisplayPrice(
        Float.parseFloat(txtDisplayPrice.getText()));
    room.setDescription(txtDescription.getText());
    return room;
}

public void showMessage(String msg){
    JOptionPane.showMessageDialog(this, msg);
}

public void addSubmitListener(ActionListener log) {
    btnSubmit.addActionListener(log);
}
```



Lớp RoomControl (1)

```
package mvcNew;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;

public class RoomControl {
    private Connection con;
    private Room room;
    private AddRoomFrm arf;

    public RoomControl(){
        String dbUrl = "jdbc:mysql://localhost:3306/hotel";
        String dbClass = "com.mysql.jdbc.Driver";
        try {
            Class.forName(dbClass);
            con = DriverManager.getConnection (dbUrl, "root", "12345678");
        }catch(Exception e) {
            e.printStackTrace();
        }

        arf = new AddRoomFrm();
        arf.addSubmitListener(new AddRoomListener());
        arf.setVisible(true);
    }
}
```



Lớp RoomControl (2)

```
public void addRoom(Room room){
    String sql = "INSERT INTO tblRoom(id, name, type, displayPrice,
        description) VALUES(?,?,?,?,?)";
    try{
        PreparedStatement ps = con.prepareStatement(sql);
        ps.setString(1, room.getId());
        ps.setString(2, room.getName());
        ps.setString(3, room.getType());
        ps.setFloat(4, room.getDisplayPrice());
        ps.setString(5, room.getDescription());

        ps.executeUpdate();
    }catch(Exception e){
        e.printStackTrace();
    }
}

class AddRoomListener implements ActionListener {
    public void actionPerformed(ActionEvent e) {
        try {
            room = arf.getRoom();
            addRoom(room);
            arf.showMessageDialog("Add room successfullly!");
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }
}
```

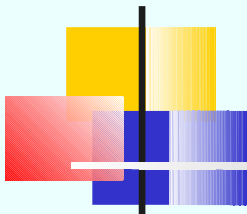


Lớp Test

```
package mvcNew;  
  
public class Test {  
  
    public static void main(String[] args) {  
        RoomControl rc = new RoomControl();  
    }  
}
```

Lưu ý trước khi chạy, phải:

- Cài đặt CSDL và bật server MySQL
- Add driver của Jdbc mysql vào library của project



Bài tập

Cài đặt theo kiến trúc đã thiết kế các modul sau:

- Chức năng sửa thông tin phòng
- Chức năng xóa thông tin phòng
- Chức năng đặt phòng
- Chức năng checkin
- Chức năng trả phòng và thanh toán

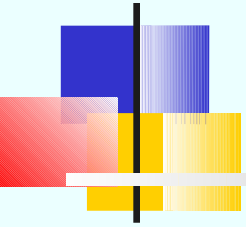


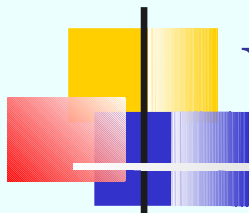
Bài tập về nhà

Cài đặt modul cá nhân theo kiến trúc đã thiết kế:

- Trình bày sơ đồ lớp đã thiết kế
- Trình bày sơ đồ cơ sở dữ liệu đã thiết kế
- Cài đặt các lớp theo đúng thiết kế
- Demo chương trình

Case study: MVC với GUI

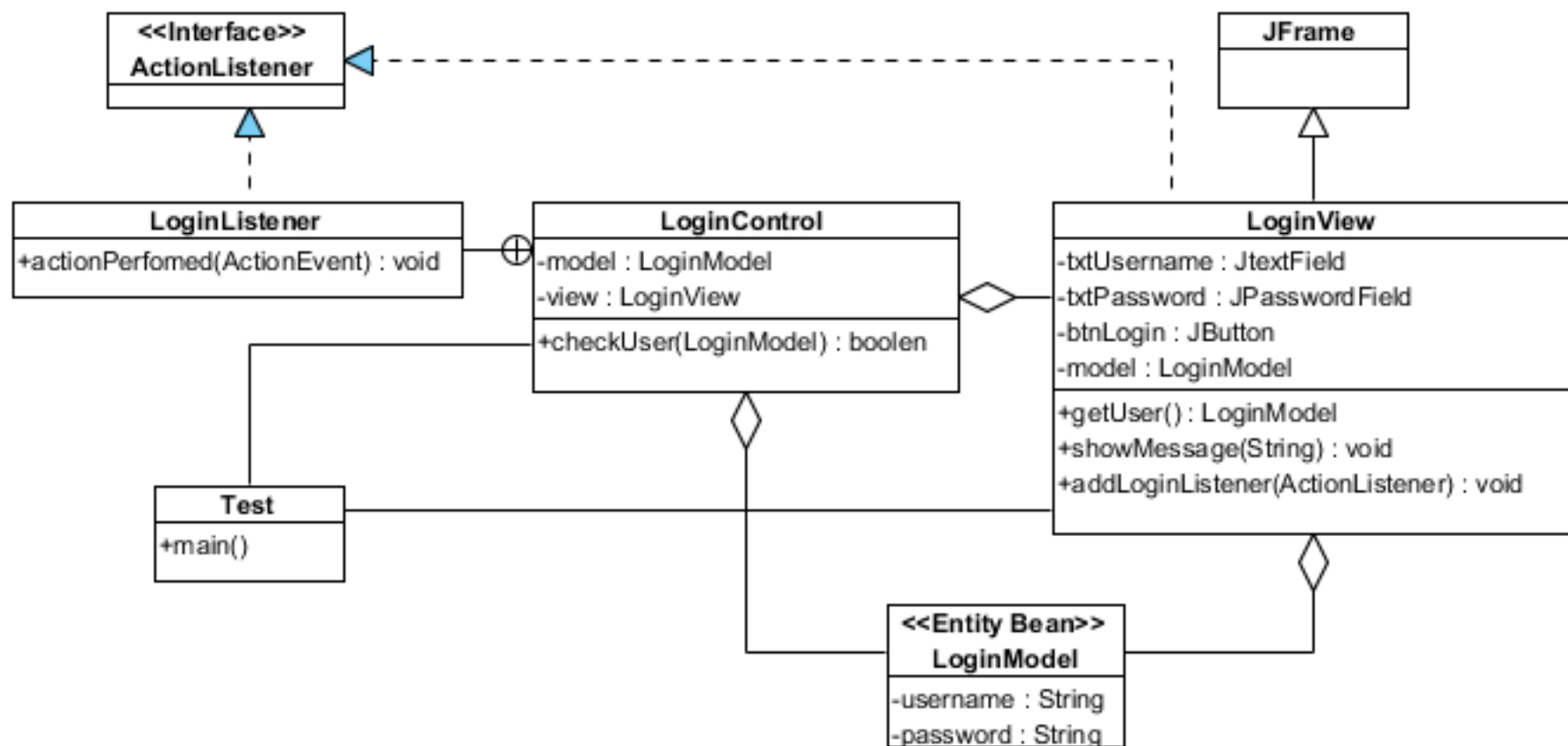




Yêu cầu bài toán

- Tạo một form đăng nhập gồm username, password và một nút login
- Thông tin của người dùng được lưu trong CSDL, bảng users có ít nhất 2 cột username và password
- Mỗi khi click vào nút login, chương trình phải kiểm tra thông tin đăng nhập có đúng không, nếu đúng thông báo thành công, nếu sai thông báo đăng nhập sai!
- Xây dựng chương trình theo mô hình MVC

Sơ đồ các lớp





LoginModel

```
public class LoginModel {  
    private String userName;  
    private String password;  
  
    public LoginModel(){  
    }  
  
    public LoginModel(String username, String password){  
        this.userName = username;  
        this.password = password;  
    }  
  
    public String getPassword() {  
        return password;  
    }  
  
    public void setPassword(String password) {  
        this.password = password;  
    }  
  
    public String getUserName() {  
        return userName;  
    }  
  
    public void setUserName(String userName) {  
        this.userName = userName;  
    }  
}
```



LoginView (1)

```
import java.awt.FlowLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JPasswordField;
import javax.swing.JTextField;

public class LoginView extends JFrame implements ActionListener{
    private JTextField txtUsername;
    private JPasswordField txtPassword;
    private JButton btnLogin;
    private LoginModel model;
```



LoginView (2)

```
public LoginView(){
    super("Login MVC");

    txtUsername = new JTextField(15);
    txtPassword = new JPasswordField(15);
    txtPassword.setEchoChar('*');
    btnLogin = new JButton("Login");

    JPanel content = new JPanel();
    content.setLayout(new FlowLayout());
    content.add(new JLabel("Username:"));
    content.add(txtUsername);
    content.add(new JLabel("Password:"));
    content.add(txtPassword);
    content.add(btnLogin);

    this.setContentPane(content);
    this.pack();

    this.addWindowListener(new WindowAdapter(){
        public void windowClosing(WindowEvent e){
            System.exit(0);
        }
    });
}
```



LoginView (3)

```
public void actionPerformed(ActionEvent e) {  
}  
  
public LoginModel getUser(){  
    model = new LoginModel(txtUsername.getText(),  
                             txtPassword.getText());  
    return model;  
}  
  
public void showMessage(String msg){  
    JOptionPane.showMessageDialog(this, msg);  
}  
  
public void addLoginListener(ActionListener log) {  
    btnLogin.addActionListener(log);  
}  
}
```



LoginControl (1)

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;

public class LoginControl {
    private LoginModel model;
    private LoginView view;

    public LoginControl(LoginView view){
        this.view = view;

        view.addLoginListener(new LoginListener());
    }
}
```



LoginControl (2)

```
class LoginListener implements ActionListener {  
    public void actionPerformed(ActionEvent e) {  
        try {  
            model = view.getUser();  
            if(checkUser(model)){  
                view.showMessageDialog("Login succesfully!");  
            }else{  
                view.showMessageDialog("Invalid username and/or password!");  
            }  
        } catch (Exception ex) {  
            view.showMessageDialog(ex.getStackTrace().toString());  
        }  
    }  
}
```




LoginControl (3)

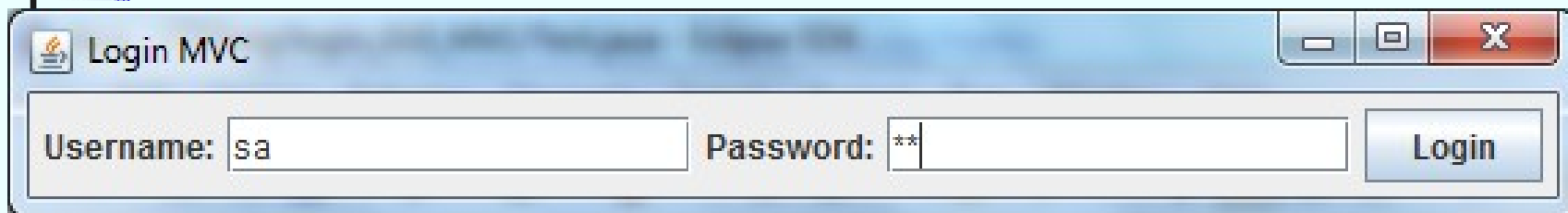
```
public boolean checkUser(LoginModel user) throws Exception {  
  
    String dbUrl = "jdbc:mysql://your.database.domain/yourDBname";  
    String dbClass = "com.mysql.jdbc.Driver";  
    String query = "Select * FROM users WHERE username ="  
        + user.getUserName() + "' AND password ="  
        + user.getPassword() + "'";  
  
    try {  
        Class.forName(dbClass);  
        Connection con = DriverManager.getConnection (dbUrl);  
        Statement stmt = con.createStatement();  
        ResultSet rs = stmt.executeQuery(query);  
  
        if (rs.next()) {  
            return true;  
        }  
  
        con.close();  
    } catch (Exception e) {  
        throw e;  
    }  
    return false;  
}
```



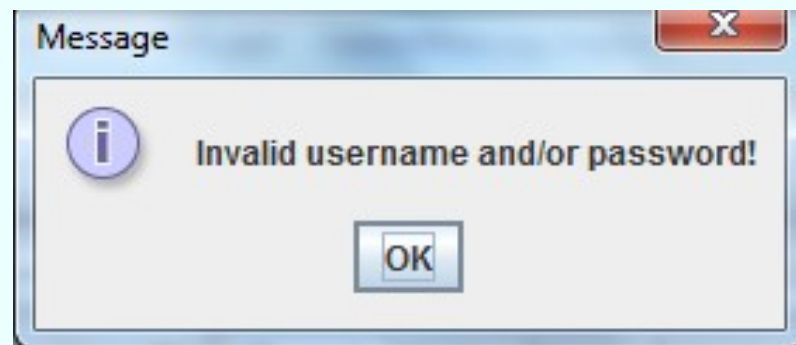
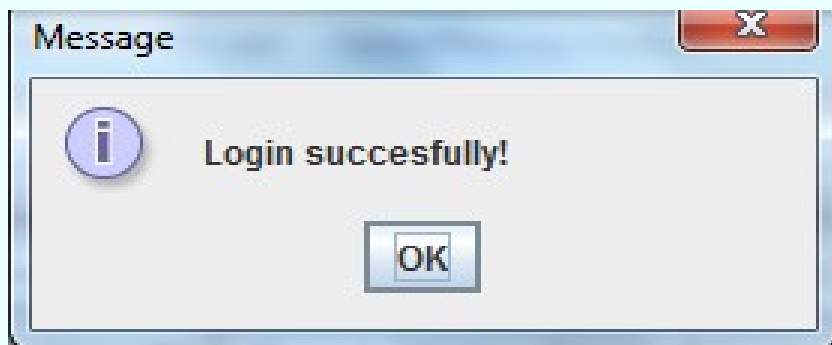
Test

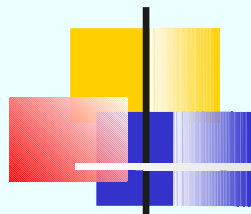
```
public class Test {  
    public static void main(String[] args) {  
        LoginView view      = new LoginView();  
        LoginControl controller = new LoginControl(view);  
        view.setVisible(true);  
    }  
}
```

Kết quả



A screenshot of a Windows-style application window titled "Login MVC". It features a standard title bar with minimize, maximize, and close buttons. The main area contains two text input fields: "Username:" with the value "sa" and "Password:" with masked characters "**". To the right of these fields is a "Login" button.





Bài tập

- Viết thêm modul đăng kí, chỉnh sửa thông tin của user vào ví dụ trong bài
- Viết modul tìm kiếm user có username chứa một đoạn text nào đó, show kết quả lên GUI
- Viết một ứng dụng hoàn chỉnh quản lí người dùng của một hệ thống nào đấy



Questions?
