ENSF 614 – Fall 2021

Lab 7 – Tuesday, November 23

Student Name: Aastha Patel and Bhavyai Gupta

Submission date: November 23, 2021

# Exercise A and B – Source file DemoDecoratorPattern.java

/\*\*

 \* File Name:               DemoDecoratorPattern.java

 \* Course:                  ENSF 614 - Fall 2021

 \* Lab # and Assignment #:  Lab 7 Exercise A and B

 \* Lab section:             B01

 \* Completed by:            Aastha Patel, Bhavyai Gupta

 \* Submission Date:         November 23, 2021

 \*/

import java.awt.Font;

import java.awt.Graphics;

import javax.swing.JFrame;

import javax.swing.JPanel;

public class DemoDecoratorPattern extends JPanel {

    Component t;

    public DemoDecoratorPattern() {

        t = new Text("Hello World", 60, 80);

    }

    public void paintComponent(Graphics g) {

        int fontSize = 10;

        g.setFont(new Font("TimesRoman", Font.PLAIN, fontSize));

        // Now lets decorate t with BorderDecorator: x = 30, y = 30, width = 100,

        // and height 100

        t = new BorderDecorator(t, 30, 30, 100, 100);

        // Now lets add a ColouredFrameDecorator with x = 25, y = 25, width = 110,

        // height = 110, and thickness = 10.

        t = new ColouredFrameDecorator(t, 25, 25, 110, 110, 10);

        // GlassFrameDecorator info: x = 25, y = 25, width = 110, and height = 110

        t = new ColouredGlassDecorator(t, 25, 25, 110, 110);

        // Now lets draw the product on the screen

        t.draw(g);

    }

    public static void main(String[] args) {

        DemoDecoratorPattern panel = new DemoDecoratorPattern();

        JFrame frame = new JFrame("Learning Decorator Pattern");

        frame.getContentPane().add(panel);

        frame.setSize(400, 400);

        frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        frame.setLocationRelativeTo(null);

        frame.setVisible(true);

    }

}

# Exercise A and B – Source file Component.java

/\*\*

 \* File Name:               Component.java

 \* Course:                  ENSF 614 - Fall 2021

 \* Lab # and Assignment #:  Lab 7 Exercise A and B

 \* Lab section:             B01

 \* Completed by:            Aastha Patel, Bhavyai Gupta

 \* Submission Date:         November 23, 2021

 \*/

import java.awt.Graphics;

public interface Component {

    public void draw(Graphics g);

}

# Exercise A and B – Source file Text.java

/\*\*

 \* File Name:               Text.java

 \* Course:                  ENSF 614 - Fall 2021

 \* Lab # and Assignment #:  Lab 7 Exercise A and B

 \* Lab section:             B01

 \* Completed by:            Aastha Patel, Bhavyai Gupta

 \* Submission Date:         November 23, 2021

 \*/

import java.awt.\*;

public class Text implements Component {

    protected int x;

    protected int y;

    protected String text;

    public Text(String text, int x, int y) {

        this.text = text;

        this.x = x;

        this.y = y;

    }

    @Override

    public void draw(Graphics g) {

        Graphics2D g2d = (Graphics2D) g;

        int fontSize = 8;

        g2d.setFont(new Font("Lucida Console", Font.PLAIN, fontSize));

        g2d.setColor(Color.GREEN);

        g2d.drawString(this.text, this.x, this.y);

    }

}

# Exercise A and B – Source file Decorator.java

/\*\*

 \* File Name:               Decorator.java

 \* Course:                  ENSF 614 - Fall 2021

 \* Lab # and Assignment #:  Lab 7 Exercise A and B

 \* Lab section:             B01

 \* Completed by:            Aastha Patel, Bhavyai Gupta

 \* Submission Date:         November 23, 2021

 \*/

public abstract class Decorator implements Component {

    protected Component cmp;

    protected int x;

    protected int y;

    protected int width;

    public int height;

}

# Exercise A and B – Source file BorderDecorator.java

/\*\*

 \* File Name:               BorderDecorator.java

 \* Course:                  ENSF 614 - Fall 2021

 \* Lab # and Assignment #:  Lab 7 Exercise A and B

 \* Lab section:             B01

 \* Completed by:            Aastha Patel, Bhavyai Gupta

 \* Submission Date:         November 23, 2021

 \*/

import java.awt.\*;

public class BorderDecorator extends Decorator {

    public BorderDecorator(Component cmp, int x, int y, int width, int height) {

        this.cmp = cmp;

        this.x = x;

        this.y = y;

        this.width = width;

        this.height = height;

    }

    @Override

    public void draw(Graphics g) {

        this.cmp.draw(g);

        Stroke dashed = new BasicStroke(3, BasicStroke.CAP\_BUTT, BasicStroke.JOIN\_BEVEL, 0, new float[] { 9 }, 0);

        Graphics2D g2d = (Graphics2D) g;

        g2d.setColor(Color.BLACK);

        g2d.setStroke(dashed);

        g2d.drawRect(this.x, this.y, this.width, this.height);

    }

}

# Exercise A and B – Source file ColouredFrameDecorator.java

/\*\*

 \* File Name:               ColouredFrameDecorator.java

 \* Course:                  ENSF 614 - Fall 2021

 \* Lab # and Assignment #:  Lab 7 Exercise A and B

 \* Lab section:             B01

 \* Completed by:            Aastha Patel, Bhavyai Gupta

 \* Submission Date:         November 23, 2021

 \*/

import java.awt.\*;

public class ColouredFrameDecorator extends Decorator {

    private int thickness;

    public ColouredFrameDecorator(Component cmp, int x, int y, int width, int height, int thickness) {

        this.cmp = cmp;

        this.x = x;

        this.y = y;

        this.width = width;

        this.height = height;

        this.thickness = thickness;

    }

    @Override

    public void draw(Graphics g) {

        this.cmp.draw(g);

        Graphics2D g2d = (Graphics2D) g;

        Stroke oldStroke = g2d.getStroke();

        Color oldColor = g2d.getColor();

        g2d.setStroke(new BasicStroke(this.thickness));

        g2d.setColor(Color.red);

        g2d.drawRect(this.x, this.y, this.width, this.height);

        g2d.setStroke(oldStroke);

        g2d.setColor(oldColor);

    }

}

# Exercise A and B – Source file ColouredGlassDecorator.java

/\*\*

 \* File Name:               ColouredGlassDecorator.java

 \* Course:                  ENSF 614 - Fall 2021

 \* Lab # and Assignment #:  Lab 7 Exercise A and B

 \* Lab section:             B01

 \* Completed by:            Aastha Patel, Bhavyai Gupta

 \* Submission Date:         November 23, 2021

 \*/

import java.awt.\*;

public class ColouredGlassDecorator extends Decorator {

    public ColouredGlassDecorator(Component cmp, int x, int y, int width, int height) {

        this.cmp = cmp;

        this.x = x;

        this.y = y;

        this.width = width;

        this.height = height;

    }

    @Override

    public void draw(Graphics g) {

        this.cmp.draw(g);

        Graphics2D g2d = (Graphics2D) g;

        g2d.setColor(Color.yellow);

        g2d.setComposite(AlphaComposite.getInstance(AlphaComposite.SRC\_OVER, 1 \* 0.1f));

        g2d.fillRect(25, 25, 110, 110);

    }

}

# Exercise A and B – Program output

Graphical user interface, text

Description automatically generated

# Exercise C Part One – Source file User.hpp

/\*

 \* File Name:               User.hpp

 \* Course:                  ENSF 614 - Fall 2021

 \* Lab # and Assignment #:  Lab 7 Exercise C

 \* Lab section:             B01

 \* Completed by:            Aastha Patel, Bhavyai Gupta

 \* Submission Date:         November 23, 2021

 \*/

#include <string>

using namespace std;

#ifndef USER\_H

#define USER\_H

struct User

{

    string username;

    string password;

};

#endif

# Exercise C Part One – Source file LoginServer.hpp

/\*

 \* File Name:               LoginServer.hpp

 \* Course:                  ENSF 614 - Fall 2021

 \* Lab # and Assignment #:  Lab 7 Exercise C

 \* Lab section:             B01

 \* Completed by:            Aastha Patel, Bhavyai Gupta

 \* Submission Date:         November 23, 2021

 \*/

#include <vector>

#include "User.hpp"

#ifndef LOGIN\_SERVER\_H

#define LOGIN\_SERVER\_H

class LoginServer

{

public:

    static LoginServer \*getInstance();

    //  PROMISES: returns single instance of LoginServer

    void add(string username, string password);

    //  REQUIRES: username and password

    //  PROMISES: add new users as per the given arguments

    User \*validate(string username, string password);

    //  REQUIRES: username and password

    //  PROMISES: returns pointer as per the arguments

private:

    LoginServer();

    //  PROMISES: constructor to create new LoginServer object

    LoginServer(const LoginServer &src);

    //  REQUIRES: source refrence to other object

    //  PROMISES: create copy of object

    LoginServer &operator=(const LoginServer &rhs);

    //  REQUIRES: rhs reference to refer object of LoginServer

    //  PROMISES: copy and assign the data members of the rhs object to the LoginServer

    vector<User> users;

    static LoginServer \*instance;

};

#endif

# Exercise C Part One – Source file LoginServer.cpp

/\*

 \* File Name:               LoginServer.cpp

 \* Course:                  ENSF 614 - Fall 2021

 \* Lab # and Assignment #:  Lab 7 Exercise C

 \* Lab section:             B01

 \* Completed by:            Aastha Patel, Bhavyai Gupta

 \* Submission Date:         November 23, 2021

 \*/

#include "LoginServer.hpp"

#include "User.hpp"

#include <iostream>

#include <string>

using namespace std;

LoginServer \*LoginServer::instance = 0;

LoginServer::LoginServer() {}

LoginServer::LoginServer(const LoginServer &src)

{

    instance = LoginServer::getInstance();

    users = vector<User>(users);

}

LoginServer &LoginServer::operator=(const LoginServer &rhs)

{

    if (this != &rhs)

    {

        instance = LoginServer::getInstance();

        users = vector<User>(users);

    }

    return \*this;

}

LoginServer \*LoginServer::getInstance()

{

    if (instance == NULL)

    {

        instance = new LoginServer;

    }

    return instance;

}

void LoginServer::add(string username, string password)

{

    struct User user;

    user.password = password;

    user.username = username;

    for (int i = 0; i < (int)users.size(); i++)

    {

        if (users.at(i).username.compare(username) == 0)

        {

            cout << "Username already exists, unable to add user!" << endl;

            return;

        }

    }

    users.push\_back(user);

    cout << "User successfully added!" << endl;

}

User \*LoginServer::validate(string username, string password)

{

    for (int i = 0; i < (int)users.size(); i++)

    {

        if (users.at(i).username.compare(username) == 0 && users.at(i).password.compare(password) == 0)

        {

            return &users.at(i);

        }

    }

    return 0;

}

# Exercise C Part One – Source file Client\_A.hpp

/\*

 \* File Name:               Client\_A.hpp

 \* Course:                  ENSF 614 - Fall 2021

 \* Lab # and Assignment #:  Lab 7 Exercise C

 \* Lab section:             B01

 \* Completed by:            Aastha Patel, Bhavyai Gupta

 \* Submission Date:         November 23, 2021

 \*/

#include "User.hpp"

#include "LoginServer.hpp"

#ifndef CLIENT\_A\_H

#define CLIENT\_A\_H

class Client\_A

{

public:

    Client\_A();

    //  PROMISES: constructor of Client\_A object and initializes its data member

    Client\_A(const Client\_A &src);

    //  REQUIRES: source to refer Client\_A object

    //  PROMISES: constructor of new Client\_A object and sets its instance data member to LoginServer

    Client\_A &operator=(const Client\_A &rhs);

    //  REQUIRES: rhs will refer to a Client\_A's object

    //  PROMISES: copy and assign the data members of the rhs object to the Client\_A

    void add(string username, string password);

    //  REQUIRES: username and password of Client\_A

    //  PROMISES: add a new user of Client\_A to list

    User \*validate(string username, string password);

    //  PROMISES: returns a pointer based on passed arguments

private:

    LoginServer \*instance;

};

#endif

# Exercise C Part One – Source file Client\_A.cpp

/\*

 \* File Name:               Client\_A.cpp

 \* Course:                  ENSF 614 - Fall 2021

 \* Lab # and Assignment #:  Lab 7 Exercise C

 \* Lab section:             B01

 \* Completed by:            Aastha Patel, Bhavyai Gupta

 \* Submission Date:         November 23, 2021

 \*/

#include "User.hpp"

#include "Client\_A.hpp"

#include <iostream>

using namespace std;

Client\_A::Client\_A() {

    instance = LoginServer::getInstance();

}

Client\_A::Client\_A(const Client\_A &src) {

    instance = LoginServer::getInstance();

}

Client\_A &Client\_A::operator=(const Client\_A &rhs) {

    if (this != &rhs) {

        instance = LoginServer::getInstance();

    }

    return \*this;

}

void Client\_A::add(string username, string password) {

    instance->add(username, password);

}

User \*Client\_A::validate(string username, string password) {

    User \*foundUser = instance->validate(username, password);

    return foundUser;

}

# Exercise C Part One – Source file Client\_B.hpp

/\*

 \* File Name:               Client\_B.hpp

 \* Course:                  ENSF 614 - Fall 2021

 \* Lab # and Assignment #:  Lab 7 Exercise C

 \* Lab section:             B01

 \* Completed by:            Aastha Patel, Bhavyai Gupta

 \* Submission Date:         November 23, 2021

 \*/

#include "User.hpp"

#include "LoginServer.hpp"

#ifndef CLIENT\_B\_H

#define CLIENT\_B\_H

class Client\_B

{

public:

    Client\_B();

    //  PROMISES: constructor of Client\_B object and initializes its data member

    Client\_B(const Client\_B &src);

    //  REQUIRES: source to refer Client\_B object

    //  PROMISES: constructor of new Client\_B object and sets its instance data member to LoginServer

    Client\_B &operator=(const Client\_B &rhs);

    //  REQUIRES: rhs will refer to a Client\_B's object

    //  PROMISES: copy and assign the data members of the rhs object to the Client\_B

    void add(string username, string password);

    //  REQUIRES: username and password of Client\_B

    //  PROMISES: add a new user of Client\_B to list

    User \*validate(string username, string password);

    //  PROMISES: returns a pointer based on passed arguments

private:

    LoginServer \*instance;

};

#endif

# Exercise C Part One – Source file Client\_B.cpp

/\*

 \* File Name:               Client\_B.cpp

 \* Course:                  ENSF 614 - Fall 2021

 \* Lab # and Assignment #:  Lab 7 Exercise C

 \* Lab section:             B01

 \* Completed by:            Aastha Patel, Bhavyai Gupta

 \* Submission Date:         November 23, 2021

 \*/

#include "User.hpp"

#include "Client\_B.hpp"

#include <iostream>

using namespace std;

Client\_B::Client\_B() {

    instance = LoginServer::getInstance();

}

Client\_B::Client\_B(const Client\_B &src) {

    instance = LoginServer::getInstance();

}

Client\_B &Client\_B::operator=(const Client\_B &rhs) {

    if (this != &rhs) {

        instance = LoginServer::getInstance();

    }

    return \*this;

}

void Client\_B::add(string username, string password) {

    instance->add(username, password);

}

User \*Client\_B::validate(string username, string password) {

    User \*foundUser = instance->validate(username, password);

    return foundUser;

}

# Exercise C Part One – Program output

Text

Description automatically generated

# Exercise C Part Two – Answers

* Program was not able to create LoginServer object.
* The reason is singleton object requires constructor should be private. And the new object can only be created only if there is no previous object was created of the same class.