**Exercise 6: Implementing the Proxy Pattern**

**Scenario:**

You are developing an image viewer application that loads images from a remote server. Use the Proxy Pattern to add lazy initialization and caching.

package proxy;

import java.util.HashMap;

import java.util.Map;

interface Image {

void display();

}

class RealImage implements Image {

private String imagename;

public RealImage(String imagename) {

this.imagename = imagename;

loadfromserver();

}

private void loadfromserver() {

System.***out***.println("loading froms server: " + imagename);

}

public void display() {

System.***out***.println("Displaying: " + imagename);

}

}

class ProxyImage implements Image {

private String imagename;

private static Map<String, RealImage> *cache* = new HashMap<>();

public ProxyImage(String imagename) {

this.imagename = imagename;

}

public void display() {

if (*cache*.containsKey(imagename)) {

System.***out***.println("Retrieved from cache of proxy: " + imagename);

*cache*.get(imagename).display();

} else {

System.***out***.println("Loading new image from server: " + imagename);

RealImage realImage = new RealImage(imagename);

*cache*.put(imagename, realImage);

realImage.display();

}

}

}

public class Tester {

public static void main(String[] args) {

Image img1 = new ProxyImage("ball.jpg");

Image img2 = new ProxyImage("bat.jpg");

Image img3 = new ProxyImage("ball.jpg");

img1.display();

img2.display();

img3.display();

}

}

A screenshot of a computer

AI-generated content may be incorrect.