

**实验报告**

|  |  |
| --- | --- |
| **课程名称**: | 设计模式 |
| **开课学期**: | 2020-2021学年 第2学期 |
| **专业**: | 软件工程 |
| **班级年级**: | 2018级 |
| **学生姓名**: | 宋行健 |
| **学号:** | 222018321062006 |
| **实验教师:** | 王晓蒙 |

**计算机与信息科学学院 软件学院**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 实验项目名称 | | 实验1 基础实验 | | | |
| 实验时间 | | 2021/3/16 | Type | □验证性 \*设计性 □综合性 | |
| 1. **实验1**   思路：  根据本题的描述，我认为这是一种工厂方法模式。首先定义一个抽象电脑类（Computer）。之后分别创建两个型号的电脑类（Computer\_A，Computer\_B）去实现抽象电脑类，这两个类和零件类（包括CPU、Memery、HardDesk）之间形成组合关系。  定义一个接口（TestComputer）进行测试电脑这一抽象方法，根据具体的电脑型号创建测试子类（TestComputerA，TestComputerB）来实现这个接口。这两个类依赖于其对应型号的电脑，决定哪个类被实例化，形成一个经典的多态应用。  类图：    总结：  该项目是一个工厂方法模式，满足开闭原则。定义了抽象类（Computer）和接口（TestComputer）。测试具体型号的电脑，只需要添加相应的实体类和测试类即可。对测试的电脑型号没有做限制，且添加型号时不需要改动原始模块的代码。  **2. 实验2**  思路：  首先是对讲义中定时器的代码进行研究，理解定时器的工作原理和实现方式，在这个基础上进行修改得到逐帧动画。  在绘制图片时，需要先创建逐帧窗格，在窗格上放置一个画布，然后调用一个绘图类（GraphicsContext）进行图片的绘制。通过对变量（int num）先“+1”后“%8”的操作，得到0~7的循环数组，在拼接图片路径的时候将变量 num+1 即可进行拼接，得到01~08的图片。但是后来发现这样图片闪动太快了，于是我将变换速度放慢了十倍，即采用“%80”的操作，在拼接路径之前对num进行除10取商。  使用变量x控制图片人物字在坐标轴上的位置，每帧x+1。  在处理图片人物转向的时候起初遇到了一些困难，没有找到直接针对图片做翻转的函数，于是我使用了对GraphicsContext进行翻转，但是翻转之后图片会跑出画布，于是又添加了GraphicsContext的平移操作。之后又遇到了翻转之后，负x轴的图片不能被清除，所以在平移时加上了一个图片的宽度。  代码：  package Exp1;  import javafx.animation.AnimationTimer; import javafx.application.Application; import javafx.scene.Scene; import javafx.scene.canvas.Canvas; import javafx.scene.canvas.GraphicsContext; import javafx.scene.image.Image; import javafx.scene.layout.AnchorPane; import javafx.scene.paint.Color; import javafx.stage.Stage;  public class MyApp extends Application {  @Override  public void start(Stage primaryStage) throws Exception {  AnchorPane p = new AnchorPane();  *// 创建一个画布用于绘画* Canvas c = new Canvas();  p.getChildren().add(c);  c.heightProperty().bind(p.heightProperty());  c.widthProperty().bind(p.widthProperty());  *// 获取用于调用绘画功能的图形上下文对象* GraphicsContext gc = c.getGraphicsContext2D();  *// 创建计时器匿名类对象，在handle方法中定义每帧的行为* AnimationTimer at = new AnimationTimer() {  final int velocity = 1; *// 每一帧前进的距离* final int startPoint = 0;  final int endPoint = 500;  double x = startPoint;  int num = 0; *// 动画编号* @Override  public void handle(long now) {  String Path = "file:E:\\设计模式\\0\_实验报告\\实验1\\JavaFX\_Exp\\saber\\0" + Integer.*toString*(num / 10 + 1) + ".png";  num = (num + 1) % 80;   System.*out*.println(x);   *// 用白色清除上一帧的画布* gc.setFill(Color.*WHITE*);  gc.fillRect(0, 0, c.getWidth(), c.getHeight());  gc.setFill(Color.*BLACK*);  Image image = new Image(Path);  gc.drawImage(image, x, 10);   *// 更新位置信息* x += velocity;  if (x == endPoint) {  *// 图像翻转* x = startPoint; *// 回归起点  // 将gc的坐标原点平移到画布的(x,0)  // 这里加上一个图片宽度(image.getWidth())是为了让人物可以原地旋转* gc.translate(startPoint + endPoint + image.getWidth(), 0);  gc.scale(-1, 1); *// 水平翻转* System.*out*.println("======");  }  }  };  at.start();  primaryStage.setScene(new Scene(p));  primaryStage.show();  }   public static void main(String[] args) {  *launch*(args);  } }  总结：  本次实验复习了JavaFX的基本结构，包括窗格（Pane）、画布（Canvas）、绘图类（GraphicsContext），并认识了定时器（AnimationTimer）。  在实验中使用取模的方式了范围内连续的整数序列，并通过拼接字符串获得图片。另外，在实验中针对图片人物的切换速度过快的问题，我使用除法取商的方法进行调优。针对人物转向的问题，我通过翻转坐标轴和平移坐标原点的方法进行解决。虽然我知道这种方法效率极低，但是目前对Java语法不够熟悉，在今后的学习中，希望可以直接对图片进行翻转的操作。  代码运行结果如下： | | | | | |
| 教师评阅 | 实验内容（60%）： | | | |  |
| 实验思路（20%）： | | | |  |
| 实验总结（20%）： | | | |  |
| 实验成绩： | | | | |