****

**Southwest university of science and technology**

JAVA实验报告

**Applet程序设计**

|  |  |
| --- | --- |
| **学院名称** | 计算机科学与技术学院 |
| **专业班级** | 信安1802 |
| **学生姓名** | 侯天铭 |
| **学号** | 5120182377 |
| **指导教师** | 刘涌 |

2019年3月

**Applet程序设计**

1. **实验目的**

分析时钟的设计过程。

1. **实验内容**
2. **Applet程序的总体设计：**

设计时钟类：在抽象时钟类的时候，可以先考虑静态的东西，即时钟的**属性，**这些属性是每一个时钟都具有的，它们包括了整个时钟的一个框架，以及时针，分针，秒针，而这些属性在Applet程序的图形窗口中是可以通过某些方法画出来的。

作为一个Applet程序，其主要具有init(),start(),stop(),destroy()四种**方法**，init()方法是程序的入口，其余三个方法未在本次实验中使用。同时因为我们需要在图形界面中画出一个时钟，因此我们还需要使用paint()方法，通过图形对象g调用相关方法来绘制图形。

因此该类的**大致结构**也就是由init()和paint()方法所构成，init()方法用于做一些初始化的工作，在Applet程序加载后**首先执行且只执行一次**。paint()方法用于绘制时钟图形，是我们首先要考虑设计的方法，且其在程序的运行过程中**可以多次执行。**

Applet程序的执行过程：当Applet程序用于Web页面时首先执行init()方法，然后自动执行start()方法，离开页面时执行stop()方法，重回本页面则再次调用start()方法。就此次设计的程序而言，在程序加载完成后首先执行且只执行一次init()方法来完成初始化，然后再执行paint()方法绘制图形，并且paint()方法可以多次执行。

1. **时钟界面的静态设计：**
2. **如何获取Applet的图形对象：**首先我们创建了Clock这个类，而这个类继承自Applet类，Applet类又继承自Panel类，最终我们在Panel类的父类Container类中找到了paint()方法，这个方法带有一个图形对象参数g。因此Clock类继承了这一方法，我们可以在本类中重写paint()方法实现对父类中paint()方法的覆盖，同时通过传入的图形对象g调用相关方法来完成绘图。
3. **分析图形坐标系：**

****小应用程序几个字下方白色区域的最左端为坐标原点，向右为x轴，向下为y轴。

1. **如何通过图形对象绘制表盘，刻度等：**

在创建的Clock类里paint()方法的方法体里面输入g.可以查看图形对象g所具有的方法，其中发现了g.drawLine(),g.drawOval()两个方法，分别用来绘制直线和椭圆。首先通过g.drawOval()方法，传入所画椭圆的左顶角坐标，以及长和宽（长宽相等即是圆）可以绘制出整个时钟的圆形框架。然后在此基础上通过多次调用g.drawLine()方法，传入不同的参数（需要画出的直线的端点坐标x1,y1,x2,y2）可以分别绘制出表盘、刻度以及静态的时针、分针和秒针。

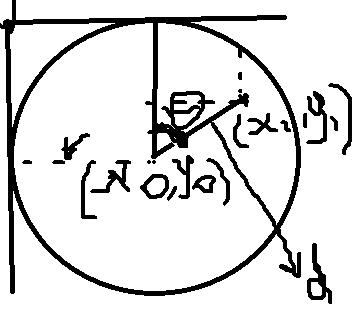
1. **当前系统时间的获取及绘制：**
2. **如何获取当前系统时间（时、分、秒）：**

获取系统时间方法挺多，上课讲了两种，我在网上也查询到了别的办法。本次实验我是利用课上讲的方法之一来完成的。首先使用util包内的日期Date类（下来想想，既然Date是日期，那这个类里是很有可能存在获取时间的方法的）创建一个对象now，通过now.可查看其该对象所具有的方法，其中找到了getSeconds(),getMinutes()以及getHours()三个方法，刚好可以用来获取系统的秒，分，时并且各返回一个int值，所以可分别定义三个int变量second,minute,hour用来储存当前秒，分，时。

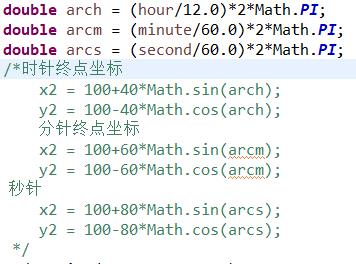


1. **如何通过图形对象绘制时针、分针、秒针：**

在上面的一步中，我们已经获取了当前系统时间的时，分，秒，并分别储存在hour,minute,second三个变量中。现在的问题是，我们想要使这个钟表的指针能够随着时间的变化指向对应的方位。首先，我们是利用了draw.Line()方法，传入四个参数绘制出了静态的指针，那么显然只需通过改变相应的传入参数，且只需改变三个指针（实际就是三条直线）的终点坐标即可（起点均在表盘中心即圆心处），那么需要做的便是在三条直线的终点坐标与second,minute,hour三个变量之间建立一个关系。如下图：



明显可以看到，我们只需表示出x1,y1即可。以平行y轴过圆心的直线为标准，通过分析知，x1=r+d\*sinθ，y1=r-d\*cosθ。将r,d分别换成对应的半径和针长即可。接着就是θ的问题了。就时针来说，转一圈为2Π，12个小时，那么经过hour个小时转过的角度可表示为：(hour/12)\*2Π，分针秒针也是同理。这样就把当前时间与指针的指向方位联系了起来。

在传入参数时将坐标强转为int型即可。。如此便完成了时针分针秒针的设计。

1. **时钟的动态显示：**
2. **Timer对象的设计：**

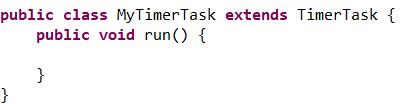
Timer是计时器的意思，因此可以想到使用Timer类创建对象来调用相关方法实现计时的作用。由于Timer类本身不是一个抽象类，所以可以直接new一个对象。如：

Timer timer = new Timer();随后输入timer.可查看该对象所具有的方法，发现其存在多个shcedule()方法，即时间表。其中有如下方法：

三个传入参数分别为：TimerTask类的任务，程序执行延迟时间，每两次执行间隔时间（时间单位均为ms）。由于需要传入一个TimerTask类的对象，所以自然引出下面的内容。

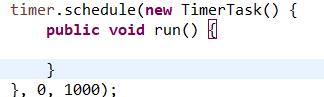
1. **TimerTask对象的设计：**

最初的设想是，在传入参数时直接new一个对象，即传入new TimerTask()，但随后发现不可行，因为TimerTask类为一个抽象类，不能直接使用new来创建对象，因此需要用别的方法。**一**是先创建一个类使其继承TimerTask类，即创建一个TimerTask类的子类，并实现TimerTask类的抽象方法，随后使用这个新建的类来创建对象作为参数传入即可。

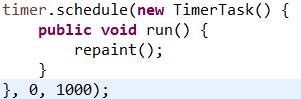




**二**是在传入参数时创建对象，但是需要随后在new TimerTask()后用一对大括弧表示出该类，并在类体中实现TimerTask()类中的抽象方法。



1. **如何实现窗体定时刷新：**

在上面的步骤中，我们已经实现了定时器对象timer的创建，并使用timer对象调用了schedule()方法，传入了三个参数，其中第一个参数为TimerTask类的对象。该方法的含义为：每隔period执行一次task，delay为延迟时间，可设为0。之前在paint()方法中使用g.drawLine(),g.drawOval()只能画出静态的固定的时钟，因为paint()方法此时只执行了一次，那么若要实现窗体的动态刷新，想来是要不停地反复地调用paint()方法，从而每隔一段时间（当然是1s）画一个时钟出来，这样就形成了一个动态的效果，给人的感觉就是时钟走了起来。所以在使用timer.schedule()时应做如下操作：即延迟为0，每隔1s执行一次repaint()重绘方法，即每秒钟执行一次paint()方法，画出一个当前时间下的时钟。由此实现了窗体的定时刷新。

1. **实验分析**

**测试过程，结果与结论：**

实验课上我基本完成了该钟表的设计，使其能够每秒刷新一次，即每一秒钟绘出一个当前时间下的钟表，从而实现了一个动态的变化效果。在随后的测试中，我先是修改了时钟的大小以及三个指针的长度，照样能正常运行。之后我对timer.schedule()方法进行了修改，一是延迟时间设为5s，刷新间隔设为5s，结果为程序启动5s后钟表开始转动，秒针每隔5s转动一次，一次转动5小格。

重新将其设为延迟0s,每秒钟刷新一次，再次实现正常的钟表功能。测试证明，设计结果正确。