目录

[2023 春编程水平测试 2 1](#_Toc137160421)

[作品概述 4](#_Toc137160422)

[结果截图 4](#_Toc137160423)

[成员分工 5](#_Toc137160424)

[文件说明 6](#_Toc137160425)

[类关系图 6](#_Toc137160426)

[代码说明 6](#_Toc137160427)

[图层类 6](#_Toc137160428)

[创建各类的对象来具体实现 6](#_Toc137160429)

[界面初始化 7](#_Toc137160430)

[静态部分绘制思路 7](#_Toc137160431)

[刷新界面 8](#_Toc137160432)

[分析鼠标位置 9](#_Toc137160433)

[主程序 9](#_Toc137160434)

[随机数种子 9](#_Toc137160435)

[播放音乐 9](#_Toc137160436)

[创建图层对象 9](#_Toc137160437)

[图形基类 10](#_Toc137160438)

[struct shapes 10](#_Toc137160439)

[Class shape 10](#_Toc137160440)

[星星 11](#_Toc137160441)

[派生类 11](#_Toc137160442)

[刷新 12](#_Toc137160443)

[烟花 12](#_Toc137160444)

[派生类 12](#_Toc137160445)

[烟花的展示程序 18](#_Toc137160446)

[动态小组成员名 20](#_Toc137160447)

[Logo 20](#_Toc137160448)

[动态小组成员名 22](#_Toc137160449)

[基类 22](#_Toc137160450)

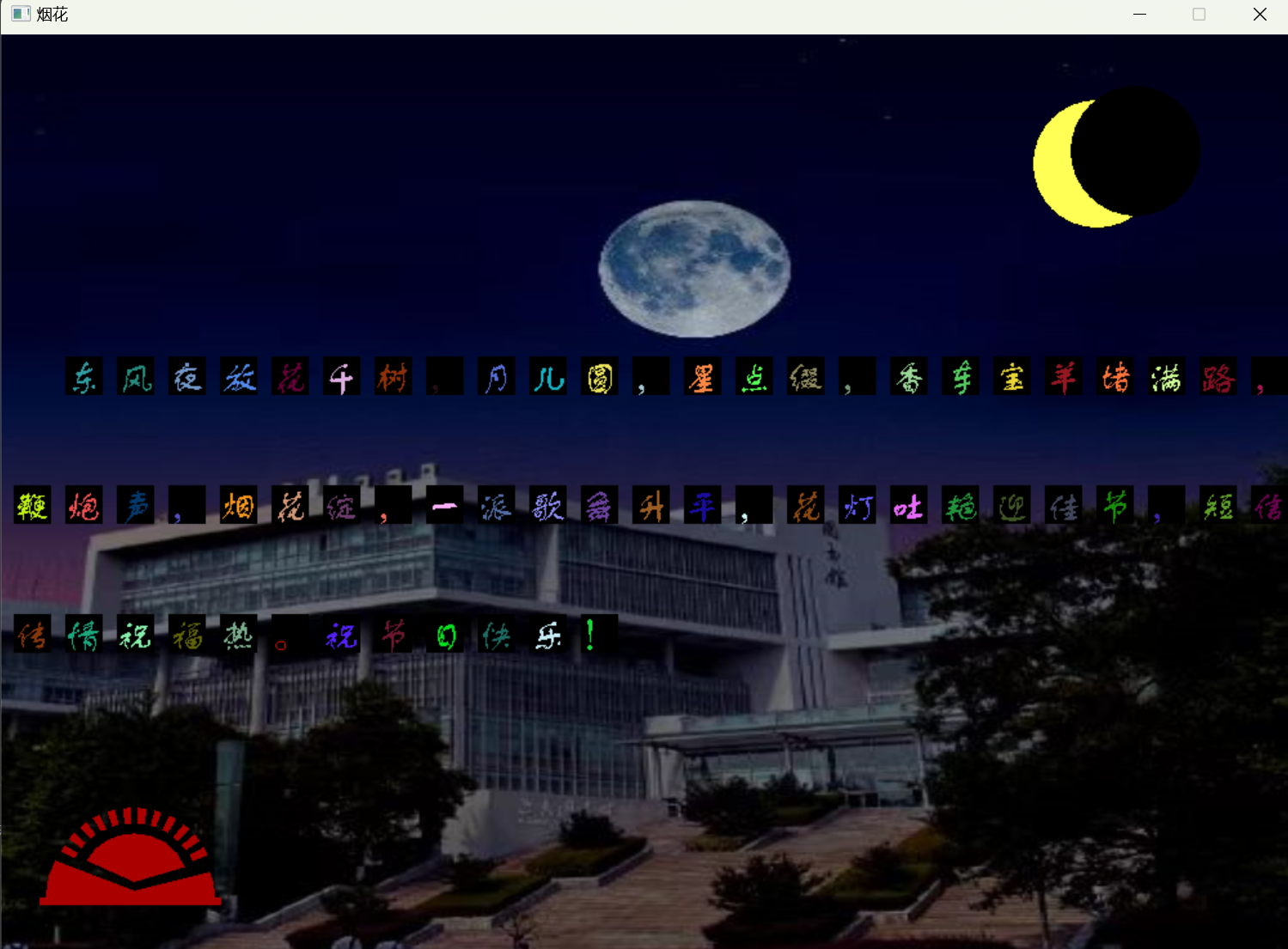
[类关系图 23](#_Toc137160451)

[祝福语 23](#_Toc137160452)

# 作品概述

## 结果截图

运行程序后出现音乐，祝福语慢慢出现。



然后文字消失，开始放烟花。其中背景、月亮、logo静态不变。星星每隔几秒中变换位置、角度、颜色、大小。鼠标指向不同的区域时，右下角会显示该部分由谁制作，烟花闪烁着不同的颜色，上升到随机高度后炸出爱心形状。

## 成员分工

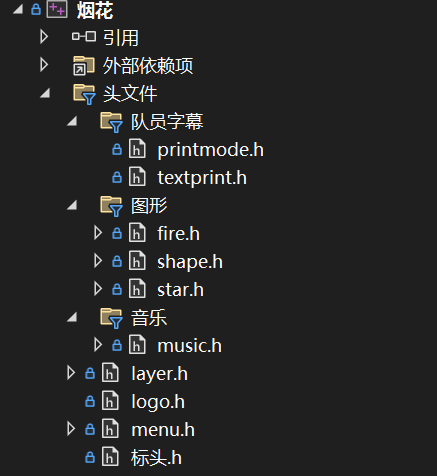
沈辰：技术文档、整合和优化、星空、鼠标交互

蒋子韬：烟花、技术文档

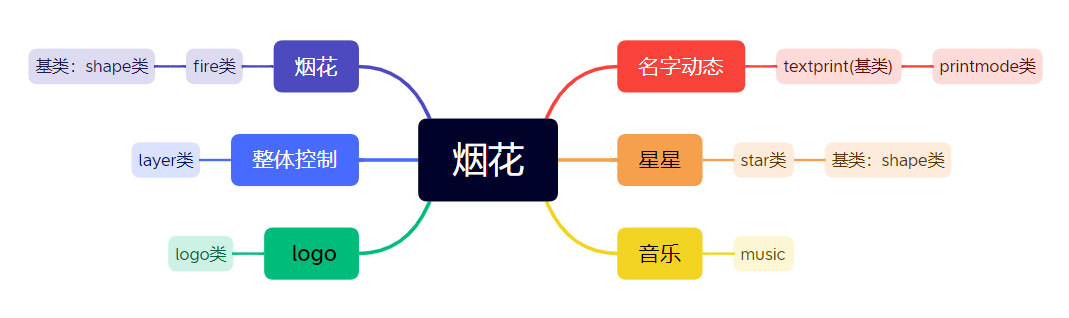
张梓瑶：音乐、logo

韩建军：动态小组成员名

## 文件说明



## 类关系图

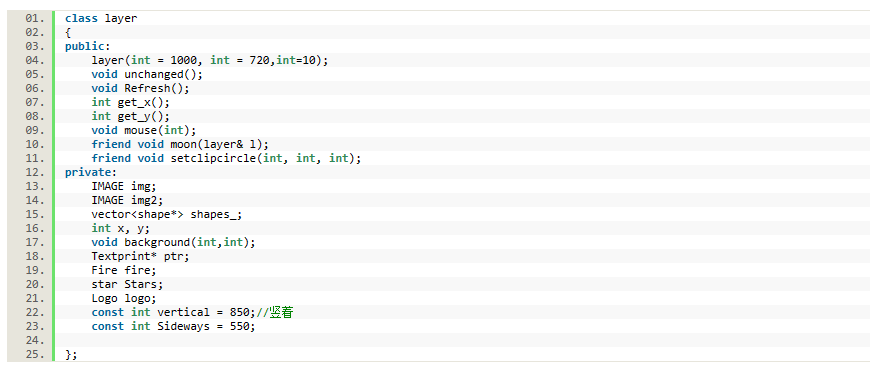


# 代码说明

## 图层类

### 创建各类的对象来具体实现

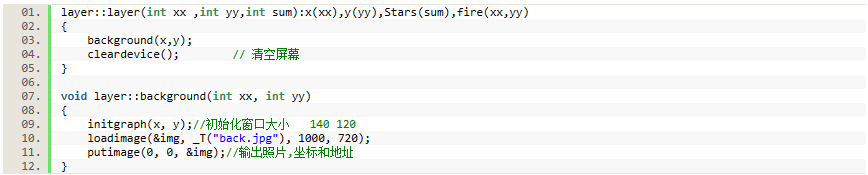
**layer** 类是一个图层管理类，其设置了展示界面的各种属性，18行-21行。

其中用了2种方式来实例化类

**18行，Textprint\* ptr;**：声明了一个指针变量 **ptr**，类型为 **Textprint\***，表示它可以指向 **Textprint** 类的对象。

20-21行 直接创建烟花类、星星类和logo类的对象。

### 界面初始化

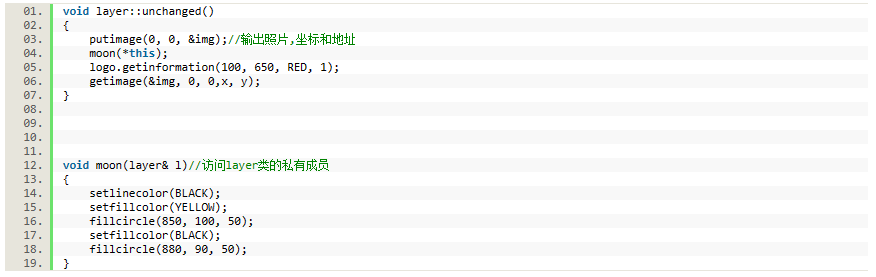


**layer** 类的构造函数接受三个参数 **xx**、**yy** 和 **sum**，并使用成员初始化列表初始化了类的成员变量 **x**、**y**、**Stars** 和 **fire**。

**void layer::background(int xx, int yy)**：它接受两个参数 **xx** 和 **yy**，用于指定图层的宽度和高度。函数首先调用 **initgraph** 函数初始化图形窗口的大小，然后使用 **loadimage** 函数加载名为 "back.jpg" 的图片，图片大小为 1000x720。最后，使用 **putimage** 函数将加载的图片输出到图形窗口的左上角位置。

### 静态部分绘制思路

**月亮和背景同为不变的部分，但是我在写的时候，动态的部分每刷新一次，它就需要重新绘制一次。虽然这是一个小项目，但为了明确可复用的理念和可读性我设置了一个叫unchanged的成员函数在每次刷新界面之前将图层的内容设置为不变的部分。然后刷新时复制到现有图层上。**

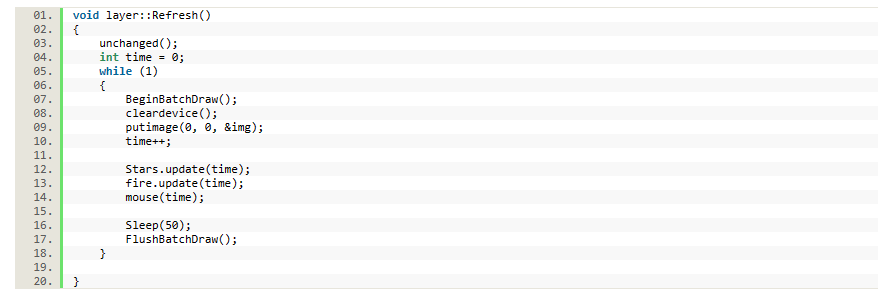


### 刷新界面

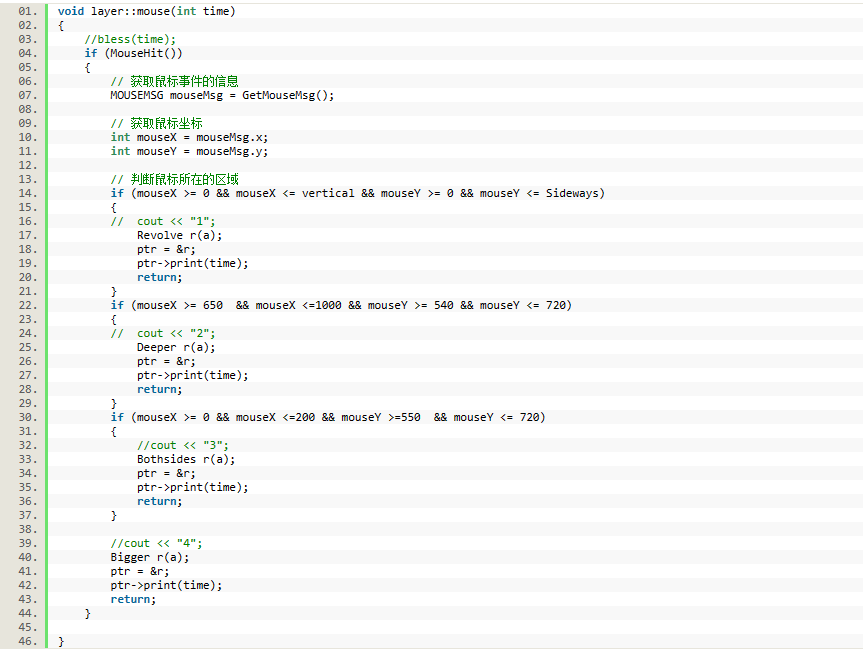
先将图层的不变部分加载并绘制到屏幕上。这包括背景图片、固定形状（月亮）和标志。然后进入无限循环，用于实时刷新界面。

在每次循环开始时，调用 **BeginBatchDraw()** 函数，**BeginBatchDraw()** 函数标志着开始进入批量绘制模式。在该模式下，绘制的结果不会立即显示在屏幕上，而是被缓存起来，直到调用 **FlushBatchDraw()** 函数时才一次性将所有绘制的内容刷新到屏幕上。这样做的好处是可以减少屏幕刷新的次数，提高绘图效率。

每刷新一次，次数加一，将次数传入各个部分中（12-14行）

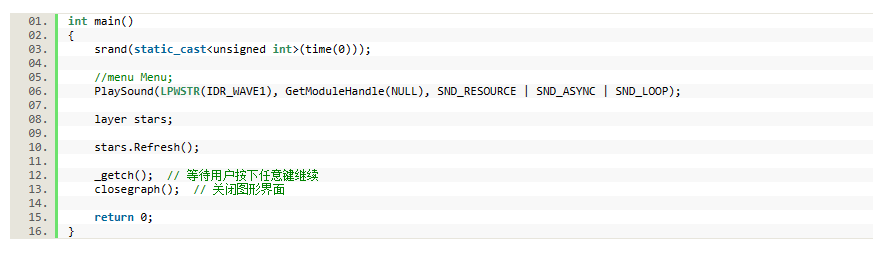


### 分析鼠标位置



GetMouseMsg()获取鼠标实时信息，

## 主程序



### 随机数种子

srand(static\_cast<unsigned int>(time(0)))**用于初始化随机数种子，因为各个功能都需要随机，所以我们选择在主程序初始化随机数种子。**

### 播放音乐

### 创建图层对象

**这行代码创建了一个名为 stars 的 layer 对象，layer类是以图层的形式管理所有部分的正常运行，并进行了初始化。**

**stars.Refresh()**用于刷新图层，所有图像的变换也只需要用到这个函数。

## 图形基类

文件shape.h 和shape.cpp





### struct shapes

**struct shapes** 是一个基类结构体，表示图形的基本属性。它包含了图形的位置（x 和 y 坐标）、颜色（color）、半径（r）和角度（angle），以及一个 **POINT** 类型的数组 **place**，用于存储图形的多点的信息（比如星星的5个角，每个角的横纵位置）。

### Class shape

**shape** 被定位有关于图形的基类，并在烟花类和星星类中运用。

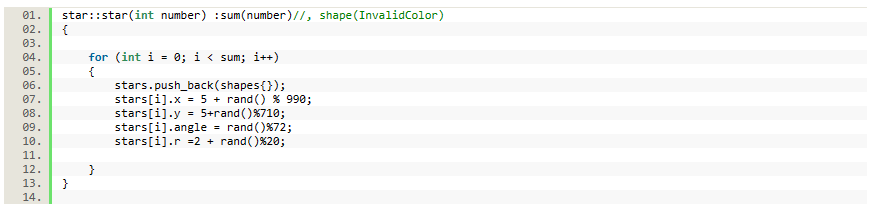
它具有 **struct shapes** 的所有成员变量，并提供了一些可能会普遍用到的行为如绘制和实时更新位置，因为其他类的情况各不相同所以我多用虚函数，期望能提高代码的可扩展性。

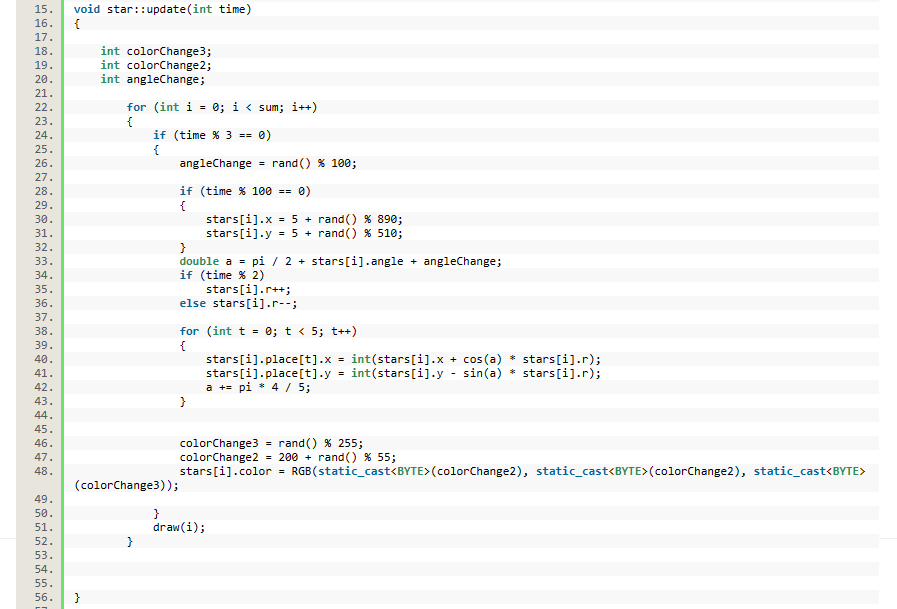
## 星星

### 派生类

**star 类是派生自基类 shape 的。在 star 类的构造函数中，它接收一个参数 number，该参数表示要创建的星星数量。在构造函数中，它使用一个循环将指定数量的 shapes 结构体对象添加到 stars 向量中。每个 shapes 对象都具有随机生成的位置、角度、半径等属性。star 类还重写了基类的 update 和 draw 。**







### 刷新

在 **update** 方法中，它对每个星星进行更新。根据给定的时间 **time**，它会更改星星的角度、半径、位置和颜色等属性。然后调用 **draw** 方法将更新后的星星绘制出来。Update是在整合个代码的图层类中唯一被循环调用的成员函数，由图层的类统一传入时间来确保各部分同时、正常的运行。

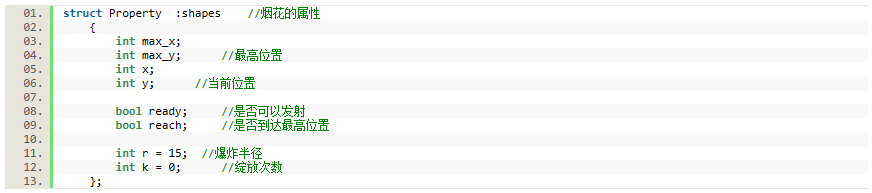
在 **draw** 方法中，通过调用 **solidpolygon** 方法，使用星星的位置信息 **stars[i].place** 绘制出五角星的形状。

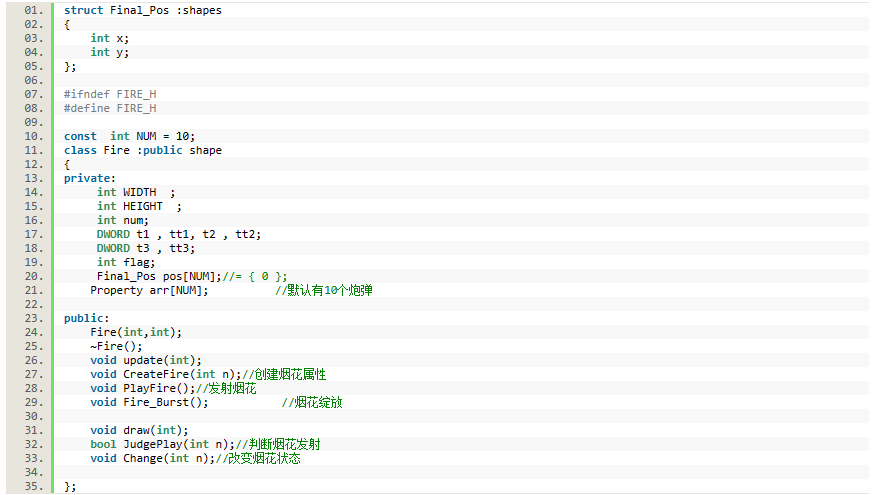
通过将 **star** 类派生自基类 **shape**，它继承了基类的公共方法和变量。这样，在 **star** 类中就可以使用基类的 **fillcolor** 方法来设置绘制颜色，并可以通过基类的 **getcolor** 方法来获取颜色。同时，通过重写基类的 **update** 和 **draw** 方法，**star** 类可以实现自己特定的更新和绘制逻辑。

## 烟花

### 派生类

描述了烟花类 Fire 的成员函数及其功能。Fire 类是shapes基类的公有派生类，用于创建和管理一个烟花动画。





**成员数据**

● WIDTH、HEIGHT: 定义了烟花展示的范围

● num: 烟花绽放粒子的数量

● DWORD t1 = timeGetTime()：获取程序运行开始时的时间，将结果存储在t1变量中。

DWORD t2 = timeGetTime()：获取程序运行结束时的时间，将结果存储在t2变量中。

DWORD t3 = timeGetTime()：获取程序当前时间，将结果存储在t3变量中。

DWORD tt1,tt2,tt3：均用于存储时间间隔。

● flag:随机烟花编号

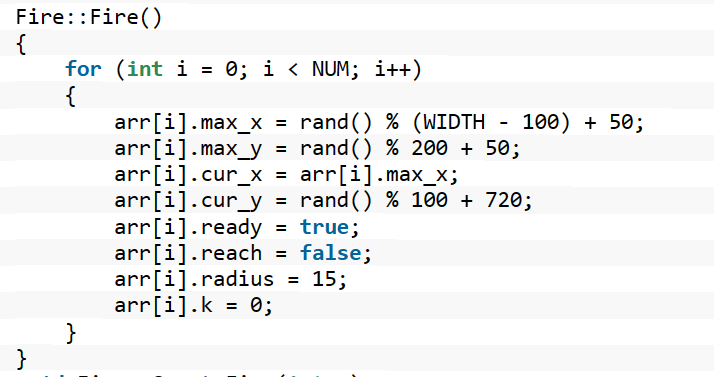
● arr[NUM]: 存储所有烟花属性的结构体数组

● pos[NUM]: 存储所有烟花终点位置的结构体数组

**成员函数**

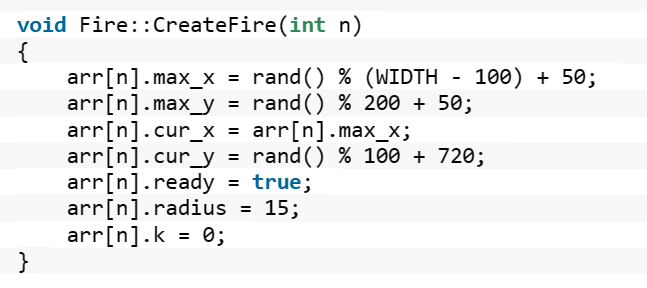
● 构造函数Fire()

初始化烟花对象数组的各个属性。

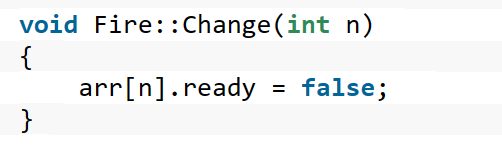


● Void CreateFire(int n)

此成员函数用于创建一个新的烟花。它接受一个整数参数 n，表示要创建的烟花在数组中的索引。该函数将为新烟花生成一个随机的初始位置和最大高度，并设置其初始状态为“准备发射”。

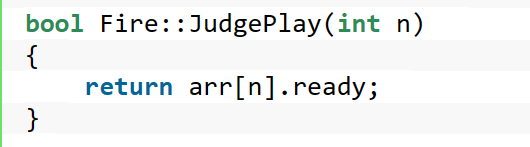


● Void Change(int n)

此成员函数用于将指定烟花的状态更改为“不可发射”。

● Bool JudgePlay(int n)

此成员函数用于判断指定烟花是否处于可发射状态。如果烟花可以发射，该函数返回 true；否则返回 false。

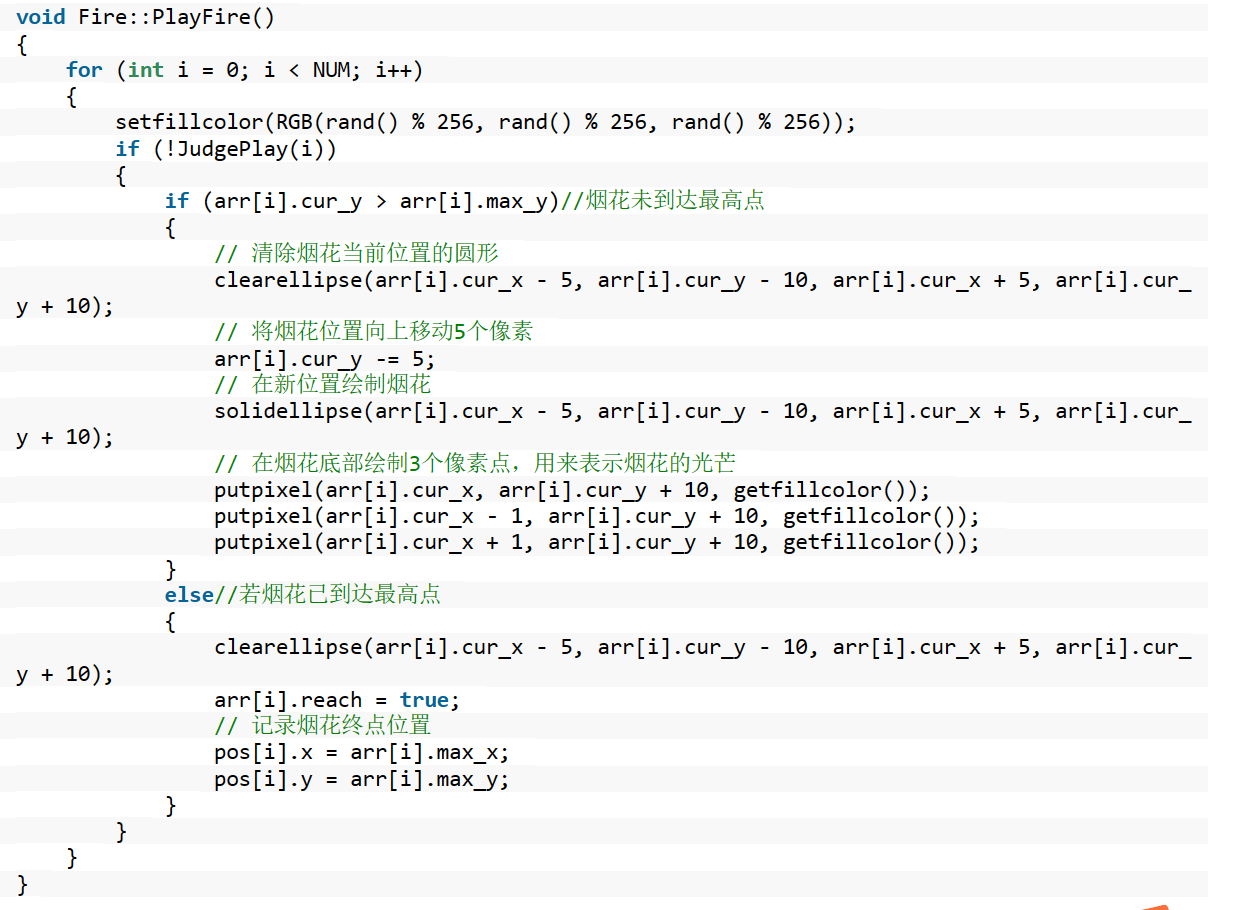


● Void PlayFire()

此成员函数用于演示烟花的发射过程。它遍历所有烟花，并为处于“可发射”状态的烟花执行以下操作：

如果烟花尚未到达最大高度，则将烟花向上移动5个像素，在新位置绘制烟花，并在烟花底部绘制3个像素点，用来表示烟花的光芒。

如果烟花已经到达最大高度，记录烟花的终点位置，并将其状态更改为“已到达最高点”。



● Void Fire\_Burst()

此成员函数用于模拟烟花爆炸的过程。它遍历所有烟花，并为处于“已到达最高点”状态的烟花执行以下操作：

首先确定各个粒子的位置：

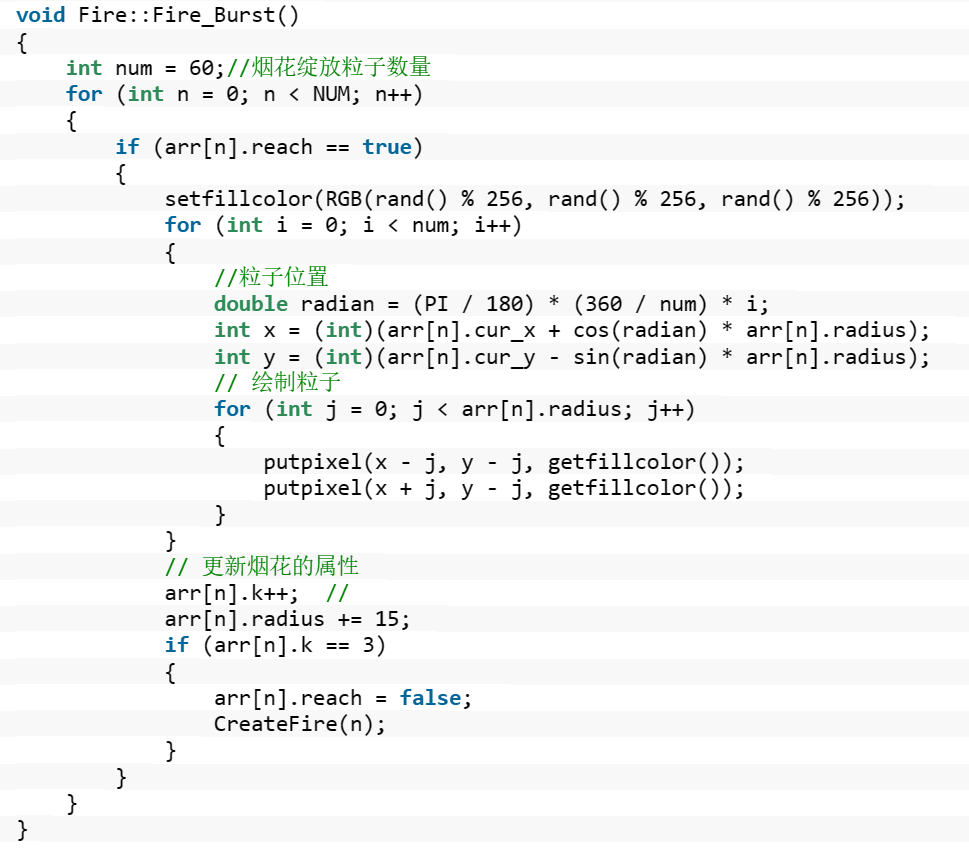
radian变量计算了每个粒子的角度。

x = (int)(arr[n].cur\_x + cos(radian) \* arr[n].radius)表示粒子的x坐标，其中arr[n].cur\_x表示烟花当前的x坐标，arr[n].radius表示粒子距离烟花中心的距离，cos(radian)表示粒子在圆形轨迹上的x轴偏移量。

y = (int)(arr[n].cur\_y - sin(radian) \* arr[n].radius)表示粒子的y坐标，其中arr[n].cur\_y表示烟花当前的y坐标，arr[n].radius表示粒子距离烟花中心的距离，sin(radian)表示粒子在圆形轨迹上的y轴偏移量。

然后，根据与烟花的距离在烟花周围绘制一定数量的粒子，以模拟爆炸效果。

一个烟花包含三次绽放，每次绽放后更新烟花的爆炸半径，当绽放次数达到三次后，重建烟花。



### 烟花的展示程序

1.创建一个Fire对象，并将其存储在指针变量fire中。

2.调用BeginBatchDraw()函数开始绘制，将绘图操作缓存起来，以提高绘图效率。

3.进入主循环，不断执行以下操作：

● 使用timeGetTime()函数获取当前系统时间，计算时间间隔，用于控制烟花的播放和绽放速度。

if (tt1 - t1 > 500)：如果自上一次执行到现在的时间间隔超过了500毫秒，就随机选择一个烟花，并判断该烟花是否可以发射。如果可以发射，则调用Change()函数切换烟花效果。

if (tt2 - t2 > 20)：如果自上一次执行到现在的时间间隔超过了20毫秒，就调用PlayFire()函数播放当前烟花效果。

if (tt3 - t3 > 50)：如果自上一次执行到现在的时间间隔超过了50毫秒，就调用Fire\_Burst()函数绘制烟花的绽放效果。

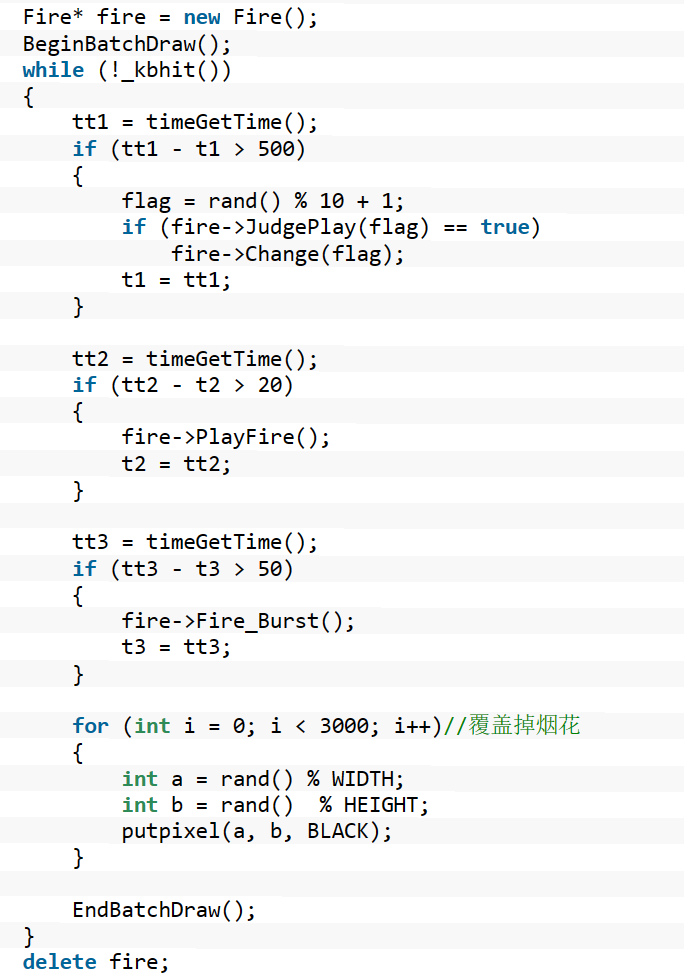
变量t1、t2和t3分别记录了上一次执行随机选择烟花效果、播放烟花效果和绘制烟花绽放效果的时间。当程序再次执行与这些操作相关的代码时，会通过计算当前时间与上一次执行的时间差，来控制烟花的播放和绽放速度。

● 使用putpixel()函数在屏幕上随机覆盖掉一些像素点，以清除掉之前绘制的烟花。

● 调用EndBatchDraw()函数结束绘制，将绘图操作刷新到屏幕上。

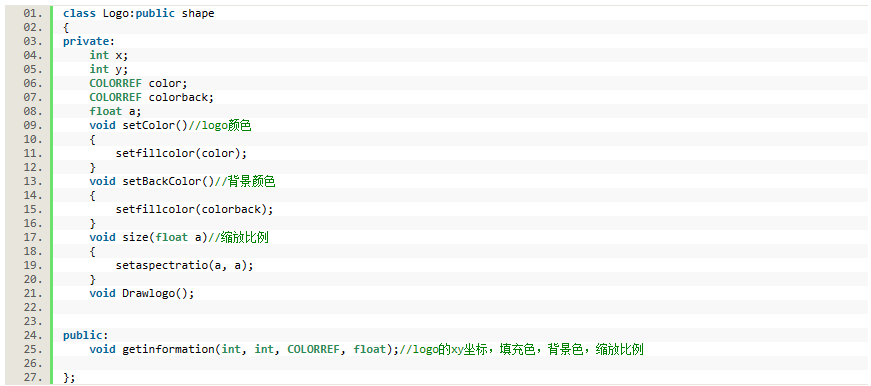
● 当用户按下键盘上的任意键时，退出程序。

4.释放Fire对象的内存空间，结束程序。



## 动态小组成员名

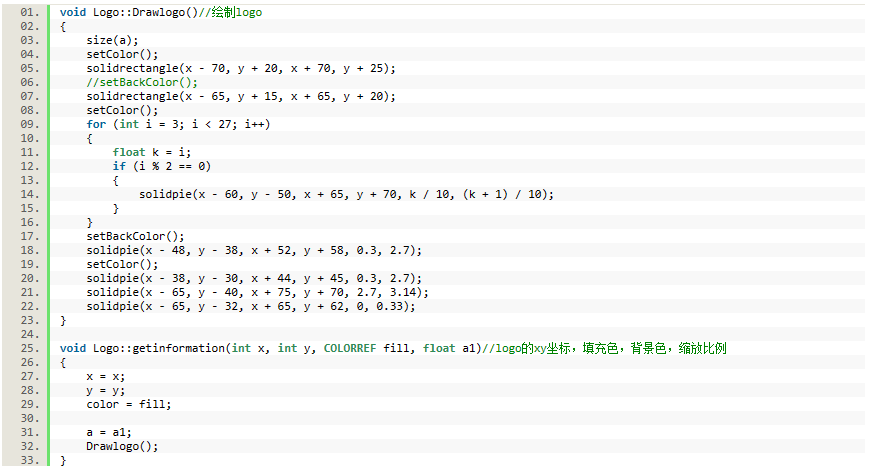
## Logo

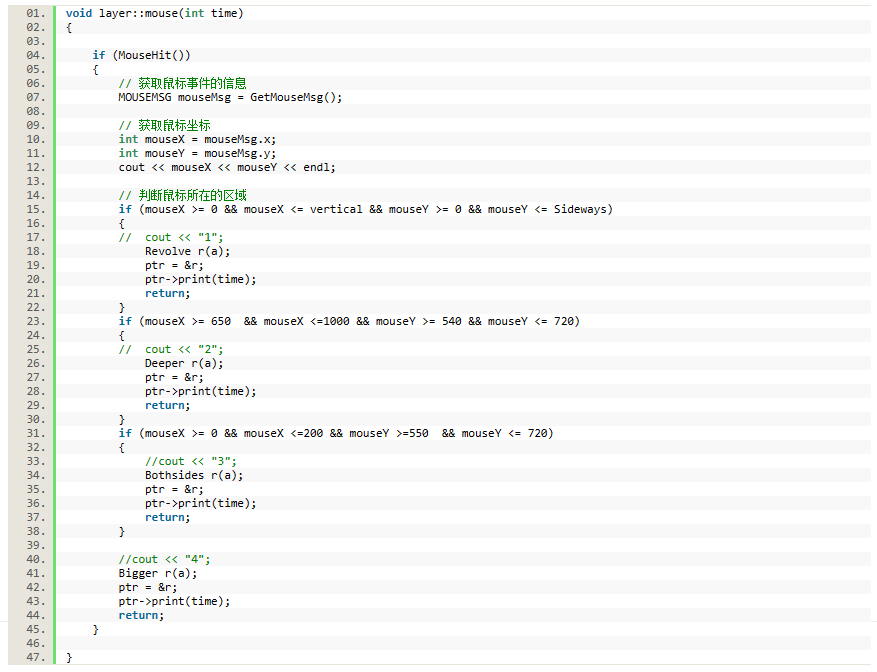
该类继承自基类 **shape** ，有较高的灵活性，可以自由设置大小、颜色、位置。

**getinformation(int, int, COLORREF, float)** 函数：

，用于接收 logo 的位置、填充颜色、背景色和缩放比例等参数，并调用 **Drawlogo()** 函数来绘制 logo。

**Drawlogo()** 函数实现了绘制 logo 的具体逻辑，根据一系列绘制操作绘制出具有特定形状的 logo。





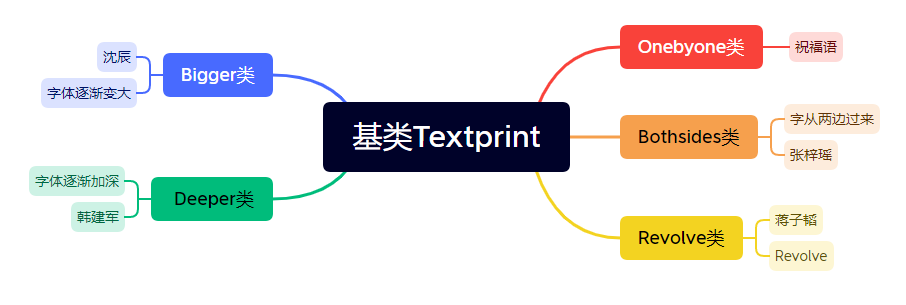
## 动态小组成员名

### 基类

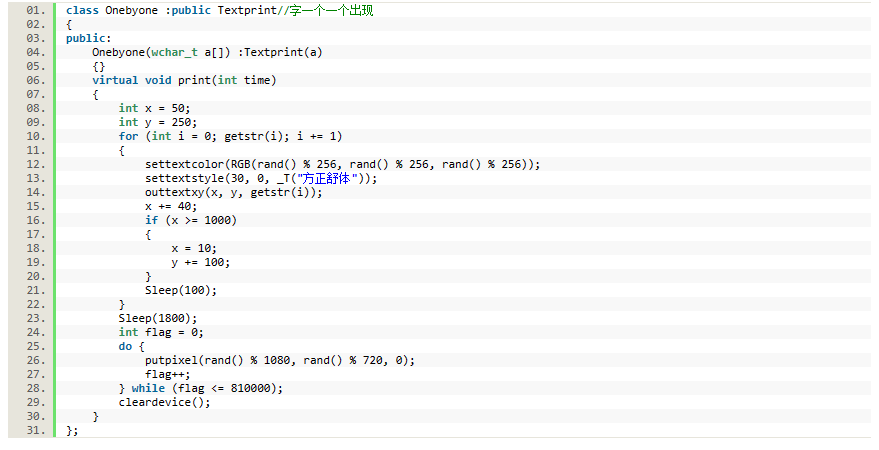


Textprint为衍生的其他文字类的基类。这个抽象基类，提供了设置文本内容、获取文本内容和打印文本的功能。它的派生类需要实现纯虚函数**print**来具体实现文本的打印功能。

### 类关系图



### 祝福语

成员函数**print**：接受一个整数参数**time**，重写了基类的纯虚函数**print**。在该函数中，按照一个字一个字的方式逐个输出文本内容。首先设置文本颜色为随机生成的RGB颜色，然后设置文本样式。接着使用**outtextxy**函数在指定位置绘制当前字的内容。每绘制一个字后，将坐标**x**向右移动40个单位，如果**x**超过了1000，则将**x**重置为10，**y**向下移动100个单位。通过循环逐个输出所有字，并在每个字之间暂停100毫秒。完成字的逐个输出后，程序暂停1800毫秒，并在画布上随机绘制大量的像素点，然后清除画布上的内容。

