第3章

Pygame游戏开发平台

3.1 Pygame 游戏开发平台简介

能吸引游戏玩家目光的，往往是动画类的游戏。如果需要设计配有动画和声音的游戏，可以通过专门为 Python 设计的 Pygame 游戏开发平台来编程实现。Pygame 游戏库的标志如图 3-1 所示。Pygame 的官方网站是 www.pygame.org。

图 3-1 Pygame 游戏库的标志

绝大多数的计算机游戏，其本质都是在窗口中绘制图形，检查游戏中的角色之间的碰撞情况，监测来自键盘或鼠标的事件，并根据各种预先设定的游戏规则作出响应，决定下一步的游戏动画的绘制方法和播放声音（例如爆炸声）等。

准确地说，Pygame 游戏库是一个功能强大的 Python 游戏开发平台，它提供了许多实用的游戏开发工具。借助 Pygame 游戏库，在屏幕上显示背景图片、显示游戏角色的动画，以及监听鼠标或键盘的事件、播放声音等功能，都可以轻松地实现。

Pygame 游戏库是一个跨平台的 Python 库，专为开发电子游戏而设计。它建立在 SDL（Simple DirectMedia Layer）的基础上，允许实时研发电子游戏而不被低级语言束缚，可以让开发者把精力集中在游戏架构的设计上。

Pygame 游戏库的主要模块如下：

（1）Pygame 初始化函数。Pygame 游戏库会自动导入其他的 Pygame 相关模块。Pygame 游戏库包括 surface 函数，可以返回一个新的 surface 对象。初始化函数 init()是 Pygame 游戏的核心，必须在进入游戏的主循环之前调用。初始化函数 init()会自动初始化其他相关模块。

（2）Pygame.locals 模块。Pygame.locals 模块包括在 Pygame 库作用域内使用的名称

(变量),包括事件类型、按键和视频模式等的名称。

(3) Pygame.display 模块。Pygame.display 模块包括处理 Pygame 显示方式的函数,可以选择普通窗口模式和全屏模式。Pygame.display 中一些常用的方法如下:

flip:更新显示。

update:更新屏幕上显示的图形时需要使用 update。

set_mode:设定显示的类型和尺寸。

set_caption:设定 Pygame 窗口的标题。

get_surface:调用 flip 和 blit 方法前返回一个可用于画图的表面(surface)对象。

(4) Pygame.font 模块。Pygame.font 模块包括 font 字体处理函数,用于设定文字的字体。

(5) Pygame.sprite 模块。Pygame.sprite 即游戏精灵,被 group 对象用作 sprite 对象的容器。调用 group 对象的 update 对象时,会自动调用所有 sprite 对象的 update 方法。

(6) Pygame.mouse 模块。Pygame.mouse 模块用于隐藏鼠标符号,或者获取鼠标位置。

(7) Pygame.event 模块。Pygame.event 模块用于响应事件,包括移动鼠标、单击鼠标按钮事件、按下某个键和释放某个键等事件。

(8) Pygame.image 模块。Pygame.image 模块用于处理保存在 GIF、PNG 或者 JPEG 文件内的图像。

3.2 安装 Pygame

在 Windows 系统中安装 Pygame 的方法很简单,只要在 DOS 命令行中输入以下命令即可:

```
pip install pygame
```

Pygame 的安装过程如图 3-2 所示。

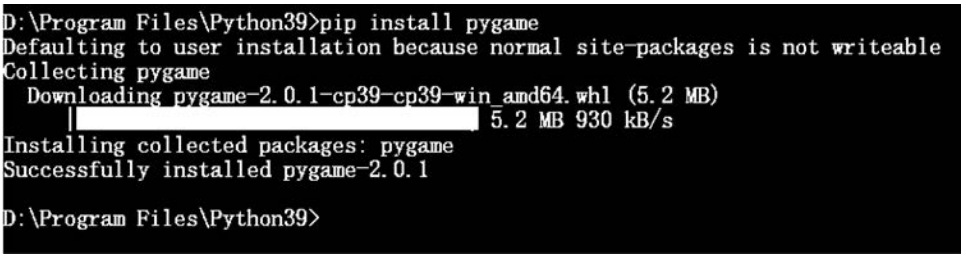

图 3-2 Pygame 的安装过程

3.3 用 Pygame 绘制几何图形

推出 Pygame 游戏开发平台的目的是让游戏角色的图形和动画的创建变得更容易。对于大多数游戏而言,设计者主要的精力往往花在响应玩家的输入以及对游戏角色的图案的刷新绘制上。游戏程序不断地执行循环,在每一个循环中都会在屏幕上重新绘制游戏角色

的图案。

在 Pygame 游戏库中，表面(surface)是屏幕上可以进行绘图的区域，Pygame 的图片导入就是通过调用 pygame.image.load()函数来实现的，这一操作将会返回一个可用的表面对象。尽管图片源文件的格式可能各不相同，但是表面对象能够把这些差异隐藏、封装起来。我们可以对表面对象进行绘制、填充、变形以及复制等多种操作。

在 Pygame 游戏库中包含了一系列用于处理基本图形的函数，使我们可以轻松地绘制圆形、长方形、多边形等几何图形。当我们绘制图形时，可以定义线条的粗细，还可以对图形指定填充的颜色。

在 Pygame 游戏库中绘制图形之前，必须使用 pygame.display.get_surface()函数创建游戏主窗口所对应的表面。接着，可以使用 surface.fill()函数向表面填充背景颜色。

在表面上绘制圆形使用 pygame.draw.circle()函数，这个函数包括五个参数：(1)绘制圆形对应的表面的名称。(2)圆形线条的颜色，如红色(255,0,0)。(3)圆心的横坐标和纵坐标。(4)圆的半径。(5)线条的宽度，如果取值为 0 表示填充。

绘制圆形的典型示例代码如下：

```
pygame.draw.circle(screen,(255,0,0),(100,100),30,0)
```

以上 Python 代码的功能是在名称为 screen 的表面对象上绘制一个圆形。线条的颜色为(255,0,0)，即红色；圆心的横坐标和纵坐标分别为 100 和 100，半径为 30，并用红色填充整个圆形。

一个用 Pygame 库绘制圆形的完整的 Python 程序如图 3-3 所示。

绘制圆形的程序.py

```
import pygame
import sys
from pygame.locals import *
from random import randint
pygame.init()

awindow=pygame.display.set_mode((400,300))
pygame.display.set_caption("Hello Pygame")
surface=pygame.display.get_surface()

clock=pygame.time.Clock()

while True:
    clock.tick(30)
    for event in pygame.event.get():
        if event.type==QUIT:
            pygame.quit()
            quit()
    surface.fill((255,255,255))
    r=randint(0,255)
    g=randint(0,255)
    b=randint(0,255)
    color=pygame.Color(r,g,b)
    pygame.draw.circle(surface,color,(200,160),80,0)
    pygame.display.update()
```

图 3-3 绘制圆形的 Python 程序

第 1 行，导入 Pygame 库；
第 2 行，导入 sys 库，这个库包含本例所需的 quit()方法；
第 3 行，导入 Pygame 库的相关模块；
第 4 行，导入 random 库，用于产生随机数；
第 5 行，对 Pygame 库进行初始化；
第 7 行，定义窗口的高度和宽度为 400×300；
第 8 行，设置窗口的标题为“Hello Pygame”；
第 9 行，定义表面对象，用于让后面的代码在表面上绘制图形；
第 11 行，对时钟进行初始化；
第 13 行，创建一个永远运行的循环；
第 14 行，设置时钟的触发间隔为每秒 30 次；
第 15 行，检测键盘和鼠标事件；
第 16～18 行，判断事件是否为关闭窗口事件，如果是，则退出程序；
第 19 行，填充表面为白色；
第 20～23 行，生成一个随机的颜色，并将颜色值保存到变量 color 中；
第 24 行，绘制一个圆心位于坐标(200,160)，半径为 80 的圆，并且填充颜色 color；
第 25 行，更新表面，即对表面重新进行绘制。
执行第 25 行后跳回到第 13 行，重复执行循环。
以上程序的运行结果如图 3-4 所示，将循环地绘制各种不同颜色的圆形。

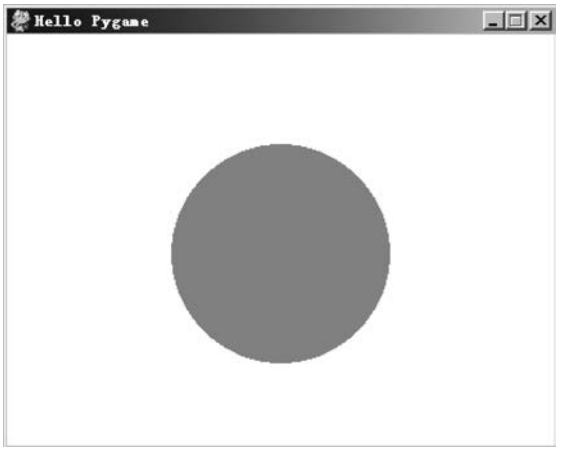

图 3-4 绘制圆形程序的运行结果

如果要绘制长方形，则需要使用 pygame.draw.rect()函数。pygame.draw.rect()函数包含四个参数：表面对象名称、线条的颜色、长方形左上角的横坐标和纵坐标、长方形的长和宽。绘制长方形的典型示例代码如下：

```
pygame.draw.rect(screen,(255,0,0),(250,150,300,200),0)
```

以上 Python 代码的功能是在名称为 screen 的表面对象上绘制一个长方形，线条的颜色为(255,0,0)，即红色；长方形左上角的横坐标和纵坐标分别为 250 和 150；长和宽分别为 300 和 200；并用红色填充整个长方形。

用 Pygame 库绘制长方形的完整的 Python 程序如图 3-5 所示。

绘制长方形的程序.py

```
1  import pygame
2  import sys
3  from pygame.locals import *
4  from random import randint
5  pygame.init()
6
7  screen = pygame.display.set_mode((600,600))
8  pygame.display.set_caption("Hello Pygame")
9  surface=pygame.display.get_surface()
10
11 clock=pygame.time.Clock()
12
13 while True:
14     clock.tick(30)
15     for event in pygame.event.get():
16         if event.type==QUIT:
17             pygame.quit()
18             quit()
19     surface.fill((255,255,255))
20     r=randint(0,255)
21     g=randint(0,255)
22     b=randint(0,255)
23     color=pygame.Color(r,g,b)
24     position_width_height = (randint(0,500),randint(0,500),randint(0,500),randint(0,500))
25     pygame.draw.rect(screen, color, position_width_height, 0)
26     pygame.display.update()
```

图 3-5　绘制长方形的程序

第 1 行，导入 Pygame 库；

第 2 行，导入 sys 库，这个库包含本例所需的 quit()方法；

第 3 行，导入 Pygame 库的相关模块；

第 4 行，导入 random 库，用于产生随机数；

第 5 行，对 Pygame 库进行初始化；

第 7 行，定义窗口的高度和宽度为 600×600；

第 8 行，设置窗口的标题为“Hello Pygame”；

第 9 行，定义表面对象，用于让后面的代码在表面上绘制图形；

第 11 行，对时钟进行初始化；

第 13 行，创建一个永远保持运行状态的循环；

第 14 行，设置时钟的触发间隔为每秒 30 次；

第 15 行，检测键盘和鼠标事件；

第 16～18 行，判断事件是否为关闭窗口事件。如果是，则退出程序；

第 19 行，填充表面为白色；

第 20～23 行，生成一个随机的颜色，并将颜色值保存到变量 color 中；

第 24 行，随机地生成长方形左上角的坐标、宽度和高度，并将各参数保存到变量 position_width_height 中；

第 25 行，绘制长方形，并填充颜色 color；

第 26 行，更新表面，即对表面进行重新绘制。

第 26 行语句执行后，跳回到第 13 行，重复执行循环。

以上程序的运行结果如图 3-6 所示，会不停地在窗口中的不同位置上绘制大小不等、颜色各异的长方形。

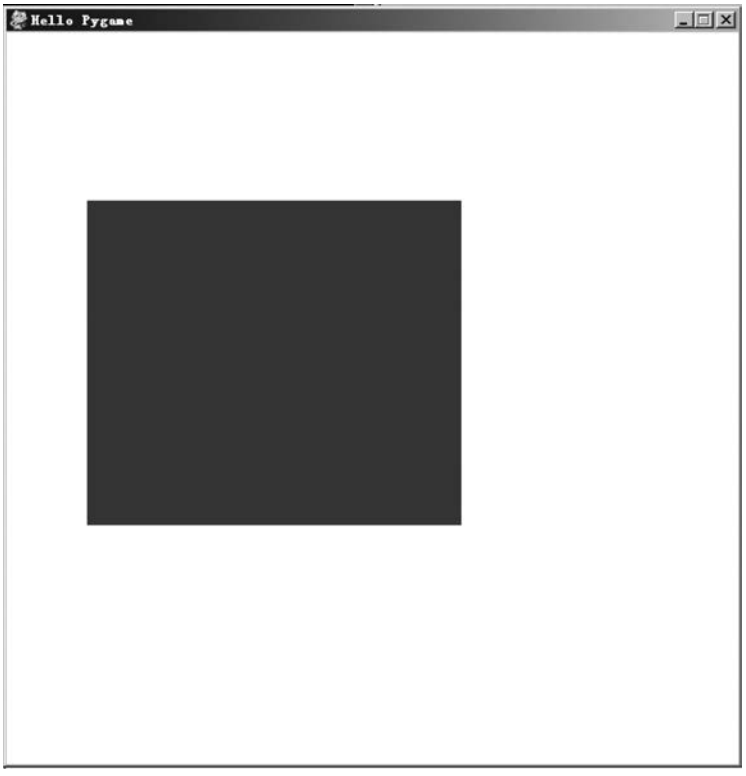

图 3-6 绘制长方形程序的运行结果

3.4 用 Pygame 显示文字

除了可以使用 Pygame 游戏库在屏幕上绘制图形以外，我们也可以使用 Pygame 游戏库在窗口中指定的位置上显示文字。

要将文字正确地放到表面上显示，需要使用 pygame.font.Font()函数创建一个字体对象，然后使用字体对象的 font.render()函数将文字渲染为图形，最后使用 blit()函数显示渲染生成的图形。

用 Pygame 游戏库显示文字的示例程序如图 3-7 所示。

第 1 行，导入 Pygame 库；

第 2 行，导入 Pygame 库的相关模块；

第 4 行，初始化 Pygame 库；

第 6 行，创建一个 400×300 的窗口；

第 7 行，设置窗口的标题为“Hello Pygame”；

第 8 行，创建一个表面对象；

第 9 行，将表面对象的背景填充为白色；

第 11 行，创建一个字体对象，字体为默认字体 None，并将字体大小设置为 36；

用pygame模块显示文字.py

```
1  import pygame
2  from pygame.locals import *
3
4  pygame.init()
5
6  screen = pygame.display.set_mode((400,300))
7  pygame.display.set_caption("Hello Pygame")
8  surface=pygame.display.get_surface()
9  surface.fill((255,255,255))
10
11 font=pygame.font.Font(None,36)
12 text1=font.render("Welcome to Pygame",1,(100,121,200))
13 screen.blit(text1,(100,100))
14 pygame.display.update()
15
```

图 3-7 用 Pygame 游戏库显示文字

第 12 行，将要显示的文字指定为“Welcome to Pygame”，将文字颜色的 RBG 代码指定为(100,121,200)，即浅蓝色，然后渲染为图形；

第 13 行，使用 blit()函数将图形绘制到窗口中坐标为(100,100)的位置上；

第 14 行，更新表面，即对表面进行重新绘制。

以上程序的运行结果如图 3-8 所示，在窗口中显示浅蓝色的文字“Welcome to Pygame”。

图 3-8 在窗口中显示文字的效果

3.5 用 Pygame 显示图片

在使用 Pygame 游戏库在窗口中显示图片之前，首先需要用 pygame.image.load()函数导入图片。这个函数可以支持 jpg、png、gif、bmp、pcx、tif、tga 等多种格式的图片。

例如，导入一个名称为 space.png 的图片，其 Python 代码如下：

```
photo = pygame.image.load("space.png").convert_alpha()
```

convert_alpha()方法使用透明的方法来绘制前景对象，当我们需要导入一个有透明(alpha)通道的图像文件时(如 PNG 或 TGA 格式的图片文件)，就需要使用 convert_alpha()方法。当然，普通格式的图片也可以使用这个方法，用了也不会产生什么副作用。

图片导入完成之后，可以使用 Surface 对象的 blit()函数显示图片。命令格式如下：

```
screen.blit(photo,(x,y))
```

第一个参数是导入完成的图片文件名，第二个参数是图片左上角的坐标。

用 Pygame 游戏库显示图片的示例程序如图 3-9 所示。

用pygame模块显示图片.py

```
1  import sys, random, math, pygame
2  from pygame.locals import *
3
4  pygame.init()
5  screen = pygame.display.set_mode((400,400))
6  pygame.display.set_caption("Star Space")
7
8  space = pygame.image.load("space.jpg").convert_alpha()
9
10 while True:
11     for event in pygame.event.get():
12         if event.type == QUIT:
13             pygame.quit()
14             quit()
15
16     screen.blit(space,(0,0))
17     pygame.display.update()
18
```

图 3-9 用 Pygame 游戏库显示图片

第 1 行，导入 Pygame、sys、math 和 random 等库；

第 2 行，导入 Pygame 库的相关库；

第 4 行，初始化 Pygame 库；

第 5 行，创建一个大小为 800×800 的窗口；

第 6 行，设置窗口的标题为“Star Space”；

第 8 行，导入文件名为“space.jpg”(夜空)的图片文件；

第 10 行，创建一个永远保持运行状态的循环；

第 11 行，检测键盘和鼠标事件；

第 12～14 行，判断事件是否为关闭窗口事件。如果是，则退出程序；

第 16 行，使用 blit()函数显示图形；

第 17 行，更新表面，即对表面进行重新绘制。

第 17 行语句执行后跳到第 10 行，重复执行循环。

以上显示图片程序的运行效果如图 3-10 所示。

图 3-10 显示图片程序的运行效果

如果要同时显示多幅图片，方法很简单。只要首先使用 pygame.image.load()函数分别导入这些图片，然后再分别使用 blit()函数逐一显示图片即可。

例如，我们需要同时在窗口中显示 spring.jpg（春天）和 bird.png（可爱的小鸟）这两幅图片，则 Python 程序如图 3-11 所示。

同时显示两幅图片.py

```
1  import pygame
2  from pygame.locals import *
3
4  pygame.init()
5  screen = pygame.display.set_mode((600,400))
6  background = pygame.image.load("spring.jpg").convert_alpha()
7  target = pygame.image.load("bird.png").convert_alpha()
8  screen.blit(background,(0,0))
9  screen.blit(target,(150,150))
10
11 while True:
12     for event in pygame.event.get():
13         if event.type == QUIT:
14             pygame.quit()
15             quit()
16
17     pygame.display.update()
18
```

图 3-11　同时显示两幅图片的程序

第 1 行，导入 Pygame 库；

第 2 行，导入 Pygame 的相关模块；

第 4 行，初始化 Pygame 库；

第 5 行，创建一个大小为 600×400 的窗口；

第 6 行，导入文件名为“spring.jpg”（春天）的图片文件用作背景；

第 7 行，导入文件名为“bird.png”（可爱的小鸟）的图片文件；

第 8 行，显示背景图片；

第 9 行，在坐标（150，150）处显示可爱的小鸟图片；

第 11 行，执行一个永不停止的循环，不停地更新画面；

第 12～15 行，检测鼠标事件，如果是点击右上角的关闭按钮，则退出程序；

第 17 行，刷新屏幕。

同时显示两幅图片程序的运行效果如图 3-12 所示。

如果需要旋转图形，可以使用旋转函数 pygame.transform.rotate（photo，angle）来实现。这个旋转函数需要两个参数，第 1 个参数 photo 是图片，第 2 个参数 angle 是旋转的角度。旋转的角度以角度制为单位，即每旋转一圈的角度为 360°。

我们仍以 spring.jpg（春天）作为背景图片，令 bird.png（可爱的小鸟）图形作顺时针旋转。实现这一旋转效果的 Python 程序如图 3-13 所示。

第 1 行，导入 Pygame 库和 time 计时库；

第 3 行，初始化 Pygame 库；

第 4 行，创建一个大小为 600×400 的窗口；

第 5 行，导入文件名为“spring.jpg”（春天）的图片文件用作背景；

第 6 行，导入文件名为“bird.png”（可爱的小鸟）的图片文件；

图 3-12 同时显示两幅图片程序的运行效果

旋转图形.py

```
1  import pygame,time
2
3  pygame.init()
4  screen = pygame.display.set_mode((600,400))
5  background = pygame.image.load("spring.jpg").convert_alpha()
6  bird = pygame.image.load("bird.png").convert_alpha()
7
8  angle = 0
9  while True:
10     angle = angle - 10
11     rotate_bird = pygame.transform.rotate(bird, angle)
12     screen.blit(background, (0, 0))
13     screen.blit(rotate_bird, (150, 150))
14     pygame.display.update()
15     time.sleep(0.1)
16
```

图 3-13 实现让可爱的小鸟图形顺时针旋转的程序

第 8 行，设置旋转角度变量 angle 的初值为 0；

第 9 行，定义一个永不停止的循环，循环体包括第 10～15 行的所有代码；

第 10 行，以 10 为单位递减角度变量 angle 的值，即每执行一次循环就令小鸟图形顺时针旋转 10°；

第 11 行，根据角度变量 angle 的当前值旋转小鸟图形；

第 12 行，显示背景图片；

第 13 行，显示旋转之后的小鸟图形；

第 14 行，更新画面；

第 15 行，延时 0.1s，然后跳回到第 9 行，不停地重复执行循环。

3.6 用 Pygame 检测键盘事件

我们可以使用 pygame.event.get()函数来检测键盘的当前状态是否改变，当用户按下每一个按键时，Python 都会产生一个类型为 KEYDOWN 的事件。此时，可以通过 event.key 来获取此按键的键盘编码，也可以用 event.unicode 来获取此按键对应的字符。

检测键盘事件的示例程序如图 3-14 所示。这个程序可以用方向键控制小鸟的移动，即通过监测键盘的事件，用四个方向键来改变小鸟在图中的位置。当按下退出键 Esc 时，结束程序。

检测键盘事件移动图片的程序.py

```
1  import pygame, sys
2  pygame.init()
3
4  x=300
5  y=300
6  d=20
7  surface = pygame.display.set_mode((600,600))
8  pygame.display.set_caption('Pygame Keyboard')
9  background = pygame.image.load("spring.jpg").convert_alpha()
10 bird = pygame.image.load("bird.png").convert_alpha()
11
12 while True:
13     for event in pygame.event.get():
14         if event.type == pygame.KEYDOWN:
15             print(event.key)
16             if event.key == pygame.K_LEFT:
17                 x = x-d
18             if event.key == pygame.K_RIGHT:
19                 x = x+d
20             if event.key == pygame.K_UP:
21                 y = y-d
22             if event.key == pygame.K_DOWN:
23                 y = y+d
24             if event.key == pygame.K_ESCAPE:
25                 pygame.quit()
26                 sys.exit()
27         position = (x,y)
28         surface.blit(background,(0,0))
29         surface.blit(bird,(position))
30     pygame.display.update()
31
```

图 3-14 检测键盘事件的示例程序

第 1 行，语句导入 Pygame 和 sys 库；

第 2 行，初始化 Pygame 库；

第 4 行，定义小鸟初始位置的横坐标 x；

第 5 行，定义小鸟初始位置的纵坐标 y；

第 6 行，定义小鸟移动一步的距离 d；

第 7 行，创建一个表面对象，窗口大小为 600×600；

第 8 行，设置窗口标题为“Pygame Keyboard”；

第 9 行，导入背景图片 spring.jpg；

第 10 行，导入小鸟图片 bird.png；

第 12 行，创建一个永远保持运行状态的循环；

第 13 行，检测 Pygame 键盘或鼠标事件；

第 14 行，判断事件的类型是否为按下了某个键的事件。如果是，则执行第 15～26 行的所有语句；

第 15 行，显示这个按键对应的编码；

第 16、17 行，判断用户是否按下了向左的方向键“←”(编码为 1073741904)。如果是，则将小鸟左移 1 步；

第 18、19 行，判断用户是否按下了向右的方向键“→”(编码为 1073741903)。如果是，则将小鸟右移 1 步；

第 20、21 行，判断用户是否按下了向上的方向键“↑”(编码为 1073741906)。如果是，则将小鸟上移 1 步；

第 22、23 行，判断用户是否按下了向下的方向键“↓”(编码为 1073741905)。如果是，则将小鸟下移 1 步；

第 24、26 行，判断用户是否按下了退出键“Esc”(编码为 27)。如果是，则结束程序；

第 27 行，更新小鸟的位置坐标；

第 28 行，显示背景图片；

第 29 行，显示小鸟图片；

第 30 行，刷新表面对象，即重新显示表面上的所有控件。

第 30 行的语句执行后，将跳回第 12 行语句，重复执行循环。

以上示例程序运行的效果如图 3-15 所示。运行这个检测键盘事件程序以后，每当用户按下某个按键时，在图 3-15 所示 IDLE 主窗口中就会显示这个按键对应的键盘编码。例如：按“←”键会显示“1073741904”，按“→”键会显示“1073741903”，按“↑”键会显示“1073741906”，按“↓”键会显示“1073741905”。

图 3-15 显示按键所对应的编码

如果用户按下的是方向键，窗口中的小鸟就会随着方向键所指的方向移动，如图 3-16 所示；如果按下其他按键则仅仅显示键盘编码，而小鸟不会移动，如图 3-16 所示；如果按下 Esc 键则结束程序。

图 3-16 用方向键控制小鸟移动的结果

3.7 用 Pygame 检测鼠标事件

我们也可以使用 pygame.event.get()函数来检测鼠标的当前状态是否改变。每当用户移动鼠标或者单击鼠标时，都会触发一个 MOUSEMOTION 或 MOUSEDOWN 事件。此时，我们可以通过 pygame.mouse.get_pos()函数来获得当前鼠标指针的坐标，同时也可以用 pygame.mouse.get_press()函数来获得当前单击了鼠标的哪一个按键。

检测鼠标事件的示例程序如图 3-17 所示。这个程序显示一个会跟随鼠标移动的小鸟图形，即在主循环中不断检测鼠标指针的位置坐标，并且用这个坐标确定小鸟的位置。

跟随鼠标移动的图形程序.py

```
import pygame
import sys
from pygame.locals import *

pygame.init()

screen = pygame.display.set_mode((600,600))
pygame.display.set_caption("Hello Pygame")
surface = pygame.display.get_surface()
background = pygame.image.load("spring.jpg").convert_alpha()
bird = pygame.image.load("bird.png").convert_alpha()

while True:
    for event in pygame.event.get():
        position_mouse_x,position_mouse_y = pygame.mouse.get_pos()
        position = (position_mouse_x,position_mouse_y)
        screen.blit(background,(0,0))
        screen.blit(bird,(position))
        left_button,mid_button,right_button = pygame.mouse.get_pressed()
        if right_button:
            pygame.quit()
            sys.exit()
    pygame.display.update()
```

图 3-17 跟随鼠标移动的图形程序

第 1 行，导入 Pygame 库；

第 2 行，导入 sys 库；

第 3 行，导入 Pygame 库的相关模块；

第 5 行，初始化 Pygame 库；

第 7 行，定义一个名字为 screen，大小为 600×600 的窗口；

第 8 行，设置窗口的标题为“Hello Pygame”；

第 9 行，创建一个表面对象 surface；

第 10 行，导入背景图片 spring.jpg；

第 11 行，导入小鸟图片 bird.png；

第 13 行，创建一个永不停止的循环；

第 14 行，检测是否有键盘或鼠标事件。如果有，则执行第 15～22 行的语句，否则不执行这几行语句；

第 15 行，获取当前鼠标指针位置的坐标；

第 16 行，设置小鸟的位置坐标为当前鼠标指针位置的坐标；

第 17 行，显示背景图片；

第 18 行，显示小鸟图片；

第 19 行，检测鼠标的按钮事件；

第 20～22 行，判断是否单击了鼠标右键。如果是，则退出程序；

第 23 行，刷新屏幕，重新显示所有控件。

第 23 行语句执行后，将跳回第 13 行语句，重新执行循环。

以上检测鼠标事件的示例程序运行时，用户移动鼠标指针，小鸟就会跟随鼠标移动；如果单击了鼠标右键，就会退出程序。

3.8 用 Pygame 播放音乐和声音

在计算机系统中，常用的音乐和声音文件有 WAV 格式和 MP3 格式。

WAV 文件是在 PC 机平台上很常见的、最经典的多媒体音频文件，最早于 1991 年 8 月出现在 Windows 3.1 操作系统上，文件扩展名为 WAV，是 WaveForm 的简写，也称为波形文件，可直接存储声音波形，还原的波形曲线十分逼真。WAV 文件格式简称 WAV 格式，是一种存储声音波形的数字音频格式，是由微软公司和 IBM 联合设计的，经过了多次修订，可用于 Windows、Macintosh、Linux 等多种操作系统。WAV 支持多种音频数字、采样频率和声道。标准格式化的 WAV 文件和 CD 格式一样，也是 44.1kHz 的取样频率，使用 16 位量化数字，因此声音文件质量和 CD 相差无几。WAV 的特点是真实记录自然声波形，基本无数据压缩，但是数据量大。

一般来说，由 WAV 文件还原而成的声音的音质取决于声卡的采样频率。采样频率越高，音质就越好，但开销就越大，WAV 文件也就越大。但 WAV 文件有一个致命的缺点，就是它所占用的磁盘空间太大（每分钟的音乐大约需要 12MB 的磁盘空间）。

MP3是一种音频压缩技术，其全称是动态影像专家压缩标准音频层面3（Moving Picture Experts Group Audio Layer 3），简称为MP3。它用来大幅降低音频数据量。1991年，位于德国埃尔朗根的研究组织Fraunhofer-Gesellschaft的一组工程师发明了MP3技术并进行了标准化。

MP3利用人耳对高频声音信号不敏感的特性，将时域波形信号转换成频域信号，并划分成多个频段，对不同的频段使用不同的压缩率，对高频使用大压缩比（甚至忽略信号），对低频信号使用小压缩比，保证信号不失真。这样一来就相当于抛弃人耳基本听不到的高频声音，只保留能听到的低频部分，从而将声音用1∶10甚至1∶12的压缩率压缩。对于大多数用户而言，压缩后的音频与原始音频相比，音质并没有明显的下降。

在Pygame中，我们可以使用混音器库的pygame.mixer.Sound()函数导入WAV格式的声音文件，然后在混音器的某个通道pygame.mixer.Channel(n)中播放声音。也就是说，可以同一时间在多个不同的通道上播放声音。

同时播放两个WAV格式的声音文件的Python程序如图3-19所示。在这里，我们需要事先准备好music1.wav和music2.wav两个声音文件。

播放WAV声音文件.py

```
1  import pygame.mixer
2  from time import sleep
3
4  pygame.mixer.init(48000,-16,1,1024)
5
6  sound1 = pygame.mixer.Sound("music1.wav")
7  channelA=pygame.mixer.Channel(1)
8  channelA.play(sound1)
9
10 sound2 = pygame.mixer.Sound("music2.wav")
11 channelB=pygame.mixer.Channel(2)
12 channelB.play(sound2)
13
14 sleep(30)
15
```

图3-19 同时播放两个WAV格式的声音文件

第1行，导入pygame.mixer混音器库；

第2行，导入time计时库；

第4行，初始化混音器，设置采样频率为48kHz，16位精度；

第6行，导入名称为music1.wav的声音文件；

第7行，创建第1个声音通道；

第8行，在声音通道1上播放music1.wav声音文件；

第10行，导入名称为music2.wav的声音文件；

第11行，创建第2个声音通道；

第12行，在声音通道2上播放music2.wav声音文件；

第14行，延时30s，等待声音播放完毕。

如果要播放MP3格式的音乐文件，则首先要用pygame.mixer.music.load()函数导入MP3格式文件，然后就可以使用pygame.mixer.music.play()函数播放这个文件了。

播放 MP3 格式的音乐文件的示例程序如图 3-20 所示。在这里，我们也需要事先准备好一个名称为 music.mp3 的音乐文件。

播放MP3声音文件.py

```
1  import time
2  import pygame
3  pygame.mixer.init()
4  filename='music.mp3'
5  track = pygame.mixer.music.load(filename)
6  pygame.mixer.music.play()
7  time.sleep(60)
8  pygame.mixer.music.stop()
9
```

图 3-20 播放 MP3 格式的声音文件

第 1 行，导入 time 计时库；

第 2 行，导入 Pygame 库；

第 3 行，初始化混音器；

第 4 行，指定 mp3 音乐文件名为 music.mp3；

第 5 行，导入 mp3 音乐文件；

第 6 行，播放 mp3 音乐文件；

第 7 行，延时 60s；

第 8 行，停止播放音乐。

3.9 小结与练习

本章探讨了专门针对 Python 的 Pygame 游戏开发平台，包括使用 Pygame 显示文字、作图、检测键盘事件和鼠标事件以及播放声音等编程知识和程序实例。

(1) 请用 Pygame 在窗体中显示蓝色的文字“人生苦短，我用 Python!”

(2) 请用 Pygame 在窗体中画一个红色的五角星。

(3) 请用 Pygame 实现一只飞鸟在天空背景前面从左向右飞翔的动画。

(4) 请用 Pygame 播放一首你喜欢的歌曲(注：mp3 格式文件)。