第14课：方块自动下落

步骤目标

本文的目标是实现方块每隔一小段时间自动下落一行的功能。如图1所示，即使玩家没有按下向下方向键，J 型方块也会自动下落。

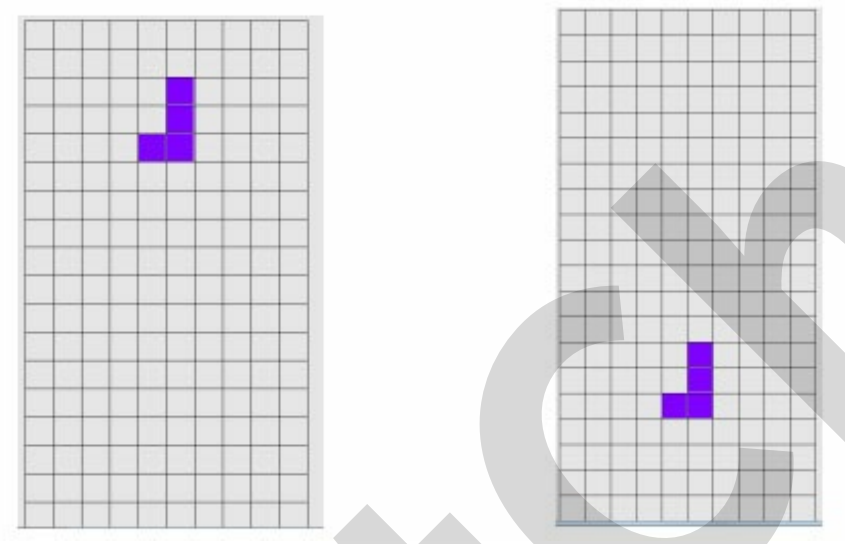

图1 方块自动下落

实现思路

本步骤的实现思路是：

1. 用定时器来触发方块自动下落。Pygame.time 模块的 `set_timer` 方法用来设置定时器。
2. 程序收到定时器闹铃事件后，调用方块的 `move_down()` 方法，使之下落一行。

玩过俄罗斯方块游戏的人都知道，随着游戏得分越来越高，游戏难度级别不断增大。这个难度主要体现在方块自动下落的速度越来越快，或者说定时器闹铃的时间间隔越来越小。因此，要用一个变量来保存定时器时间间隔。

我们决定用一个游戏状态对象来存储游戏进行过程中的各种状态。现阶段，有以下状态信息：

1. 定时器时间间隔
2. 当前方块
3. 墙体

你可能对第2、3项有些疑惑。为什么说它们属于游戏状态呢？原因在于游戏进行过程中它们是不断变化的。玩家能感知到的任何不断变化的元素都属于游戏状态的一部分。

下面，我们把游戏状态对象作为 GameState 类的实例。

游戏状态类

游戏状态类 GameState 在 gamestate.py 文件内定义，如代码1所示。

```
# TetrisGame/gamestate.py
5   import random
6   from settings import *
7   from piece import Piece
8   from gamewall import GameWall
9   import pygame
10  class GameState():
11      def __init__(self, screen):
12          self.screen = screen
13          self.wall = GameWall(screen)     #墙体对象
14          self.piece = Piece(random.choice(PIECE_TYPES), screen, self.wall)   #当前方块
15          self.timer_interval = TIMER_INTERVAL   #1000ms，定时器时间间隔初值
16          self.set_timer(self.timer_interval)         #启动定时器
17
18      def set_timer(self, timer_interval):
19          self.game_timer = pygame.time.set_timer(pygame.USEREVENT, timer_interval)
```

代码1 游戏状态类 GameState

GameState 类的构造函数中做了以下几件事：

1. 生成墙体对象。
2. 生成游戏的第一个方块。这会在下面作出更多说明。
3. 启动定时器。定时器时间间隔 `TIMER_INTERVAL` 设置为初始值 1000ms，即

1s。`TIMER_INTERVAL` 是在 settings.py 内定义的常量。

有了 GameState 类，我们无须在 main 函数内生成 piece 变量引用的方块对象，和 `game_wall` 变量引用的墙体对象。生成 `game_state` 对象之际，构造方法内将生成方块对象和墙体对象，如代码2所示。

```
# TetrisGame/main.py
…   ……  #与前一版本相同，故省略
13  from gamestate import GameState
14
15  def main():
…       …… #与前一版本相同，故省略
27      # game_wall = GameWall(screen)
28      # piece = Piece(random.choice(PIECE_TYPES), screen, game_wall)
29      game_state = GameState(screen)
30      #游戏主循环
31      while True:
32          #方块触底的话
33          if game_state.piece.is_on_bottom:
34              game_state.wall.add_to_wall(game_state.piece)
35              game_state.piece = Piece(random.choice(PIECE_TYPES), sc
reen, game_state.wall)
36
37          #监视键盘和鼠标事件
38          check_events(game_state.piece)
39
40          #设定屏幕背景色
41          screen.fill(bg_color)
42          #绘制游戏区域网格线和墙体
43          GameDisplay.draw_game_area(screen, game_state.wall)
44          #绘制方块
45          game_state.piece.paint()
46          #让最近绘制的屏幕可见
47          pygame.display.flip()
```

代码2 主函数内使用游戏状态对象

代码2的代码说明如下：

1. 第27、28行代码是原先使用 piece 变量和 `game_wall` 变量的代码，现在不用了，所以把这两行注释掉了。第29行代码完成了这两行代码的功能。它调用了 GameState 类的构造

函数，这一构造函数内生成了游戏启动后第一个方块和墙体对象。

2. 要记得导入 GameState 类，见13行。
   3.从第30行开始，用 `game_state.piece` 引用当前方块对象，用 `game_state.wall` 引用墙体对象。

方块自动下落

生成游戏状态对象的构造函数内，启动了定时器，见代码1中第16行。程序一开始，执行主函数 main()，main 函数内调用 GameState 类的构造函数。执行构造函数中间，执行代码1第16行，导致代码1中第19行被执行。第19行调用了 pygame.time 模块的 `set_timer` 函数，如下所示。

```
1.    self.game_timer = pygame.time.set_timer(pygame.USEREVENT,
      timer_interval)
```

这个函数的第一个参数是事件类型编号，第二个参数是定时器闹铃的时间间隔，初始值为1000ms。这里，传入的第一个参数是 `pygame.USEREVENT`。它是一个整数值，是一个常量，用作定时器闹铃这一事件的类型编号。上一语句执行后，经过 `timer_interval` 时间后“闹铃”，这里讲的闹铃的实际动作是向程序的事件队列尾部添加编号为 `pygame.USEREVENT` 的事件。

程序该怎么接收到定时器闹铃事件呢？你可以在下面代码3中第57行看到答案。可见，定时器闹铃事件跟键盘按键事件是类似的。只不过，定时器闹铃事件的类型编号是启动定时器时设置的第一个参数的值。

如果，你用下面的语句来启动定时器：

```
1.    self.game_timer = pygame.time.set_timer(pygame.USEREVENT + 1,
      timer_interval)
```

那么，你要把代码3第57行写作：

```
1.      elif event.type == pygame.USEREVENT + 1:
```

```
# TetrisGame/main.py
50  def check_events(piece):
51      '''捕捉和处理键盘按键事件'''
52      for event in pygame.event.get():
53          if event.type == pygame.QUIT:
54              sys.exit()
55          elif event.type == pygame.KEYDOWN:
56              on_key_down(event, piece)
57          elif event.type == pygame.USEREVENT:
58              piece.move_down()
59
60
61  def on_key_down(event, piece):
62      if event.key == pygame.K_DOWN:
63          # print("向下方向键被按下")
64          piece.move_down()
65      elif event.key == pygame.K_UP:
66          # print("向上方向键被按下")
67          piece.turn()
68      elif event.key == pygame.K_RIGHT:
69          # print("向右方向键被按下")
70          piece.move_right()
71      elif event.key == pygame.K_LEFT:
72          # print("向左方向键被按下")
73          piece.move_left()
74      elif event.key == pygame.K_f:
75          piece.fall_down()
```

代码3 响应定时器闹铃事件

由于 `check_events` 函数的代码越来越长，我们决定把处理键盘按键事件的功能代码分离出来。代码3中的 `on_key_down` 函数完成了处理键盘按键事件的功能。以后，你会看到这一分离带来的好处。

小结

至此，方块自动下落的功能实现完毕。我们的做法是，利用定时器产生闹铃事件，主循环内响应闹铃事件，然后调用向下移动方块的方法。你可以自己运行程序测试一下，看看方块是否每隔 1s 就会自动下落一行。以后，我们会随着难度增大加快下落速度。

实现上述功能的全部代码可从以下链接阅览或下载：

- Github

俄罗斯方块游戏程序逐渐成形，希望你从中学到了一些东西。只要你多尝试，多思考，你就会有大进步。