第11课：加速落到底部

步骤目标

本文要实现的目标是：

1. 一直按着方向键，达成等同于与反复敲击该方向键的效果。前面实现的程序版本无法做到这一点。
2. 按下 f 字母键，能加速落到底部。效果就是瞬间落到底部。

一直按着方向键的效果

只要添加一条语句，就能做到“一直按着方向键，达成等同于与反复敲击该方向键的效果”。这条语句加在 main.py 文件的 main 函数内，即下面代码段中的语句：`pygame.key.set_repeat(10, 100)`。

```
def main():
    #初始化pygame。启用Pygame必不可少的一步，在程序开始阶段执行。
    pygame.init()
    #创建屏幕对象
    screen = pygame.display.set_mode((1200, 900) )   #分辨率是1200*900
    pygame.display.set_caption("俄罗斯方块")   #窗口标题
    pygame.key.set_repeat(10, 100)   # 一直按下某个键，每过100毫秒就引发一个KEYDOWN事件
    ......   #与前一版本相同，故省略
```

修改代码完毕后，你有必要运行看看是否如你所愿。

加速落到底部

实现加速落到底部的功能，包含两项任务：

1. 检测玩家按下 f 字母键；
2. 方块瞬间落到底部。

GitChat

检测玩家按下 f 字母键的代码放在 main.py 文件的 `check_events` 函数内，即下面代码段中的最后两行语句。

```
def check_events(piece):
    '''捕捉和处理键盘按键事件'''
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sys.exit()
        elif event.type == pygame.KEYDOWN:
            if event.key == pygame.K_DOWN:
                # print("向下方向键被按下")
                piece.move_down()
              elif ......  #与前一版本相同，故省略
            elif event.key == pygame.K_f:
                piece.fall_down()
```

下面我们来实现 Piece 类的 `fall_down()` 方法，达成瞬间落下的效果。这要在 piece.py 文件内增加一个方法。我们把这个方法放在 `move_down()` 方法的定义体之下。这是因为两者有密切关联，放在一起比较好。代码1中第52~54行定义了 `fall_down` 方法。

```
# TetrisGame/piece.py
45  def move_down(self):
46      '''方块向下移动1格。如果到达了底部，设置is_on_bottom属性为True.'''
47      if self.can_move_down():
48          self.y += 1
49      else:
50          self.is_on_bottom = True
51
52  def fall_down(self):
53      while not self.is_on_bottom:
54          self.move_down()
```

代码1 加速落到底部的 `fall_down()` 方法（左侧数字是文件内代码行号）

`fall_down()` 方法实现瞬间到底的功能。这是通过循环执行 `move_down()` 方法做到的，循环测试条件是方块未到达底部。我们知道，当方块到达底部时，`is_on_bottom` 属性将设为 True，否则属性值为 False。

做了以上改动后，你可以自己运行程序测试一下。按 f 字母键后，你会发现游戏区域顶部的方

GitChat

块发生了变化，压根儿没见到“加速落到底部的效果”。

是不是哪里出错了呢？答案是，这正是“加速落到底部的效果”。由于我们没有实现落到底部后垒成一堵墙的功能，前一方块消失了，生成新方块出现在游戏区域顶部。这表明，下一步我们要实现必须是“方块到底后垒成一堵墙的功能”。

小结

本文实现了玩家控制方块加速落到底部的功能。玩家可以通过两种方式来加速：

- 第一种是一直按着向下方向键。在 main.py 内调用 pygame 模块的 `key.set_repeat()` 函数，就能达成目标。
- 第二种是按快捷键（我们选用的是 f 字母键，fall 的首字母）。我们修改 main.py 内的 `check_events` 函数，以响应按下 f 字母键事件。响应处理的逻辑是，只要方块没有触底，则反复调用方块对象的 `move_down` 方法。

> 提示：本文要实现的步骤是不是好简单？这是因为，我们重用了在前面步骤中实现的函数，也就是判断触底的 `can_move_down` 方法和向下移动方块的 `move_down` 方法。写程序多用函数是一个好习惯！

实现本文步骤全部功能的代码链接请见：Github。

下一篇中，我们要实现的是“垒墙功能”，也就是方块触底后垒成一堵墙的功能。

GitChat