# 第22课：难度级别和使用说明

## 步骤目标

本文有两个目标。一是实现难度级别功能。游戏一开始，难度级别为1。游戏得分每过5000分一次，难度级别升1级。这样，得分0~4999，难度级别为1；得分5000~9999，难度级别为2，依次类推。难度级别提高，方块自动下落速度加快——定时器闹铃的时间间隔越来越短。第二个目标是在游戏窗口左侧提供使用说明。

上述两个目标达成后，游戏窗口如图1所示。

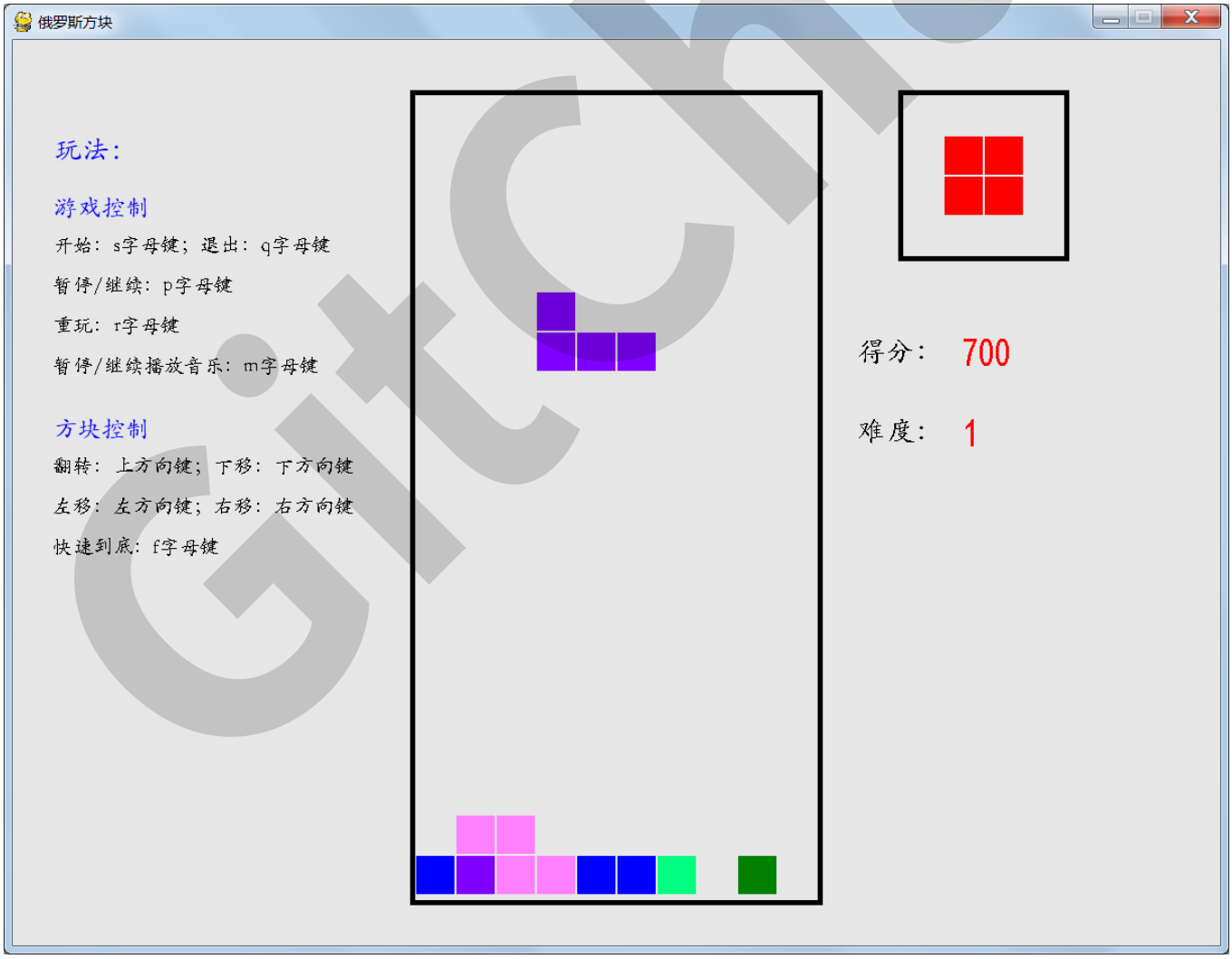


图1 难度级别和使用说明

GitChat

## 实现难度级别功能

实现难度级别功能的做法是：

1. 在 GameState 类内添加 difficulty 属性，用来记住当前的难度级别。游戏一开始，难度级别为1。
2. 玩家得分时，可能要把难度级别提高一级。
3. 如果提高一级，那么 difficulty 属性值增1，并缩短定时器闹铃时间间隔，因而方块自动下落速度加快了。

实现难度级别功能的代码如代码1所示。

```
# TetrisGame/gamestate.py
11  class GameState():
12      def __init__(self, screen):
13          self.screen = screen
14          self.wall = GameWall(screen)
15          self.piece = None
16          self.next_piece = None
17          self.timer_interval = TIMER_INTERVAL   #1000ms
18          self.game_score = 0
19          self.stopped = True
20          self.paused = False
21          self.session_count = 0
22          self.difficulty = 1
23
24      def set_timer(self, timer_interval):
25          pygame.time.set_timer(pygame.USEREVENT, timer_interval)
26
27      def stop_timer(self):
28          pygame.time.set_timer(pygame.USEREVENT, 0)   #传入0表示清除定时器
29
30      def add_score(self, score):
31          self.game_score += score
32          difficulty = self.game_score // DIFFICULTY_LEVEL_INTERVAL + 1
33          if difficulty > self.difficulty:
34              self.difficulty += 1
35              self.timer_interval -= TIMER_DECREASE_VALUE
36              pygame.time.set_timer(pygame.USEREVENT, self.timer_interval)
```

GitChat

```
28.     ...         ......
```

代码1 实现难度级别功能

下面简要说明代码1中的代码：

1. 第22行定义了 difficulty 属性，用来记住当前的难度级别。初值为1。
2. 第32行据目前的得分计算难度级别。`DIFFICULTY_LEVEL_INTERVAL` 常量在 settings.py 文件内定义，值为5000。这意味着， 5000分为一关，玩家每过一关，游戏难度级别增1。
3. 第33行判别玩家是不是又过了一关。如果是，执行第34~36行代码。
4. 第34行，difficulty 属性值增1。
5. 第35行，定时器闹铃间隔变短。`TIMER_DECREASE_VALUE` 是缩短的时间量，值为50ms。这意味着，难度级别增1，定时器闹铃间隔时间变短50ms，也就是方块前后两次自动下落的间隔缩短50ms。
6. 第36行，重新设定定时器，按变短后的间隔时间来触发闹铃事件。

## 绘制难度级别

图1中，在窗口右端，得分一栏下面，显示了难度级别。做法上，显示难度级别和显示得分是一样的。我们在 GameDisplay 类内用 `draw_difficulty_level` 方法来实现显示难度级别的功能，该方法的代码如下。

```
1.   @staticmethod
2.       def draw_difficulty_level(screen, level):
3.           '''绘制游戏难度级别'''
4.           level_label_font = pygame.font.SysFont('stkaiti', 28)  # 换
     成'arial', 无法显示中文。
5.
6.           level_label_surface = level_label_font.render(u'难度：', False,
     SCORE_LABEL_COLOR)
7.           level_label_position = (GAME_AREA_LEFT + COLUMN_NUM * CELL_WIDT
     H + MARGIN_WIDTH, GAME_AREA_TOP + 8 * CELL_WIDTH)
8.           screen.blit(level_label_surface, level_label_position)
9.
10.          level_font = pygame.font.SysFont('arial', 36)
11.          level_surface = level_font.render(str(level), False, SCORE_COLO
     R)
12.          level_label_width = level_label_surface.get_width()
```

```
13.             level_position = (level_label_position[0] + level_label_width +
    20, level_label_position[1])
14.             screen.blit(level_surface, level_position)
```

上述代码首先绘制了“难度：”这一字眼，然后绘制难度值这个整数。绘制的做法是：

1. 生成字体对象。调用 pygame.font.SysFont() 来生成字体对象。它的第一个参数是字体，比如 `stkaiti` 指的是楷体。第二个参数是字体大小。
2. 生成图元对象。调用字体对象的 render 方法来生成图元对象。第一个参数是要绘制的文字，字符串前面的修饰符 u 代表这是 nicode字符串。第二个参数讲要不要反显，False 表示不要。第三个参数是颜色。`SCORE_COLOR`和 `SCORE_LABEL_COLOR` 是 settings.py 文件内定义的颜色常量。
3. 计算出图元显示的位置。
4. 调用 screen.blit 方法来显示图元。

> 注意：如果你是在 Linux 系统中编写程序，那么有可能因为系统没有安装 stkaiti（即楷体）这种字体而导致程序运行错误。错误发生在调用 pygame.font.SysFont() 的地方。纠正该错误需要安装字体文件。做法参阅以下文档，你也可以自行搜索“如何在 Linux 中安装字体”的答案。在 Mac 系统中遇到字体问题，请自行搜索答案。
>
> 1.《Linux CentOS 7 安装字体库 & 中文字体》：该文适用于 Redhat 或 CentOS 系统。文内详细讲解了：（1）安装字体库的方法；（2）在 Windows 系统内找到字体文件和复制到 Linux 系统的字体文件目录的做法；（3）配置字体的做法。
>
> 2.《Linux 命令之 FC，手动安装字体》：该文适用于 Redhat 或 CentOS 系统，比上一文档更简练，信息更少，不过步骤是完整的。可与上一文档配合使用。
>
> 3.《Linux 中安装字体》]：该文适用于 Ubuntu 系统。它扼要地讲解了操作步骤。第4步加载字体中的第三条命令 `fc -cache -fv`，应改为 `fc-cache -fv`，`fc-cache` 是命令。建议与第一份文档配合使用。

## 绘制使用说明

绘制使用说明的做法与绘制难度级别是类似的。我们在 GameDisplay 类内定义 `draw_mannual` 方法来实现绘制使用说明功能。该方法的代码如下所示。这里，中文字体都用了楷体。我们把显示“得分”的字体也改成了楷体，不改动的话是黑体。你不改动也没有关

系。

```
    @staticmethod
    def draw_mannual(screen):
        base_position_x = 40
        base_position_y = GAME_AREA_TOP + 40
        title_font = pygame.font.SysFont('stkaiti', 28)
        title_surface = title_font.render(u'玩法：', True, TITLE_COLOR)
        title_position = (base_position_x, base_position_y)
        screen.blit(title_surface, title_position)

        base_position_y += 60
        gamectrl_label_font = pygame.font.SysFont('stkaiti', 24)
        gamectrl_label_surface = gamectrl_label_font.render(u'游戏控制',
True, TITLE_COLOR)
        gamectrl_label_position = (base_position_x, base_position_y)
        screen.blit(gamectrl_label_surface, gamectrl_label_position)

        base_position_y += 40
        man_font = pygame.font.SysFont('stkaiti', 20)
        man_down_surface = man_font.render(u'开始：s字母键；退出：q字母键',
False, HANZI_COLOR)
        man_down_position = (base_position_x, base_position_y)
        screen.blit(man_down_surface, man_down_position)

        base_position_y += 40
        # man_font = pygame.font.SysFont('stkaiti', 20)
        man_pause_surface = man_font.render(u'暂停/继续：p字母键', False,
HANZI_COLOR)
        man_pause_position = (base_position_x, base_position_y)
        screen.blit(man_pause_surface, man_pause_position)

        base_position_y += 40
        # man_font = pygame.font.SysFont('stkaiti', 20)
        man_restart_surface = man_font.render(u'重玩：r字母键', False, HA
NZI_COLOR)
        man_restart_position = (base_position_x, base_position_y)
        screen.blit(man_restart_surface, man_restart_position)

        base_position_y += 40
        # man_font = pygame.font.SysFont('stkaiti', 20)
        man_music_surface = man_font.render(u'暂停/继续播放音乐：m字母键',
False, HANZI_COLOR)
        man_music_position = (base_position_x, base_position_y)
```

GitChat

```
            screen.blit(man_music_surface, man_music_position)

            base_position_y += 60
            # gamectrl_label_font = pygame.font.SysFont('stkaiti', 24)
            gamectrl_label_surface = gamectrl_label_font.render(u'方块控制', True, TITLE_COLOR)
            gamectrl_label_position = (base_position_x, base_position_y)
            screen.blit(gamectrl_label_surface, gamectrl_label_position)

            base_position_y += 40
            # man_font = pygame.font.SysFont('stkaiti', 20)
            man_down_surface = man_font.render(u'翻转：上方向键；下移：下方向键', False, HANZI_COLOR)
            man_down_position = (base_position_x, base_position_y)
            screen.blit(man_down_surface, man_down_position)

            base_position_y += 40
            man_move_surface = man_font.render(u'左移：左方向键；右移：右方向键', False, HANZI_COLOR)
            man_move_position = (base_position_x, base_position_y)
            screen.blit(man_move_surface, man_move_position)

            base_position_y += 40
            man_speed_surface = man_font.render(u'快速到底：f字母键', False, HANZI_COLOR)
            man_speed_position = (base_position_x, base_position_y)
            screen.blit(man_speed_surface, man_speed_position)
```

## 调用绘制难度级别和绘制使用说明

我们在 `GameDisplay` 类的 `draw_game_area` 方法内调用 `draw_difficulty_level` 方法和 `draw_mannual` 方法。由于 `draw_game_area` 方法绘制的整个窗口的内容，因此我们把方法名改为 `draw_game_window`。修改后的代码如下所示：

```
    @staticmethod
    def draw_game_window(screen, game_state, game_resource):  #此处有修改
        '''绘制游戏窗口'''
        GameDisplay.draw_border(screen, GAME_AREA_LEFT - EDGE_WIDTH, GAME_AREA_TOP, LINE_NUM, COLUMN_NUM)

        GameDisplay.draw_wall(game_state.wall)
```

GitChat

```
            GameDisplay.draw_score(screen, game_state.game_score)
            if game_state.stopped:
                if game_state.session_count > 0:
                    GameDisplay.draw_game_over(screen, game_resource)
                GameDisplay.draw_start_prompt(screen, game_resource)
            if game_state.paused:
                GameDisplay.draw_pause_prompt(screen, game_resource)
            GameDisplay.draw_next_piece(screen, game_state.next_piece)
            GameDisplay.draw_mannual(screen)      #此处有修改
            GameDisplay.draw_difficulty_level(screen, game_state.difficulty
    )                       #此处有修改
```

## 一处修订

在试玩游戏过程中，我发现会发生以下异常现象：按 f 字母键一次，连续有两个方块落到底部。我判断，原因是我们通过以下语句设定了“一直按着键盘，就产生连续的按键事件”的效果。

```
pygame.key.set_repeat(10, 100)
```

这一语句位于 main.py 文件的 main 函数内。它的第一个参数是指过 10ms 后就产生按键事件。第二个参数是指以后每过 100ms 就产生按键事件。我觉得，第一个参数 10ms 太小了，玩家（比如我这个反应迟钝的玩家）稍微手慢点，就产生了第二个按键事件。因此，我把语句改为：

```
pygame.key.set_repeat(100, 100)
```

你也可以自己试验调整，找到最合适的参数值。你可以到网上搜索“`pygame.key.set_repeat`”，查找更多的资料。

## 小结

本步骤实现了以下功能：

1. 难度级别。玩家每得到5000分，就把难度级别增1。难度级别越大，方块自动下落速度越快。我们在 GameState 类添加 difficulty 属性来记录难度级别，初值为1。玩家得分时，

程序要检查是不是又过了一关，是的话把 difficulty 属性增1，并调整定时器。

2. 绘制难度级别和游戏使用指南。做法上，这与绘制游戏得分没什么两样。
3. 把绘制游戏界面的 `draw_game_area()` 改为 `draw_game_window()`，这样更加贴切。

完成全部功能的代码可从以下链接浏览或下载。

- Github

下一篇，我们将实现背景音乐功能。

GitChat