人工智能程序设计


python
import turtle
turtle.setup(650,350,200,200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for 1 in range(4):
turtle.circle(40, 80)
turtle.circle(-40, 80)
turtle.fd(40)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)

人工智能程序设计

9.1 PYGAME游戏开发入门

北京石油化工学院 人工智能研究院

刘 强

Pygame 简介

Pygame 是 Python 中最流行的游戏开发库

- 基于 SDL（Simple DirectMedia Layer）构建
- 提供创建 2D 游戏所需的所有基本功能
- 简单易学，非常适合初学者入门游戏开发

9.1.1 Pygame 环境搭建

Pygame 具有以下优势：

- **简单易学**：API 设计简洁，适合初学者
- **功能完整**：支持图形绘制、声音播放、事件处理等
- **跨平台**：支持 Windows、macOS、Linux 等操作系统
- **活跃社区**：拥有丰富的教程和示例代码

安装 Pygame

使用 **pip** 命令安装 **Pygame**：

安装成功后会显示 Pygame 的版本号

```
## 安装Pygame
pip install pygame

## 验证安装
python -c "import pygame; print(pygame.version.ver)"
```

9.1.2 基本游戏框架

游戏循环 是游戏程序的心脏，它不断重复执行以下步骤：

1. **处理输入**：检测键盘、鼠标等输入事件
2. **更新游戏状态**：根据输入和游戏逻辑更新对象位置、状态等
3. **渲染画面**：将游戏对象绘制到屏幕上
4. **帧率控制**：控制游戏运行速度，保持稳定的帧率

这个模式是所有游戏的基础，无论是简单的贪吃蛇还是复杂的3D游戏

示例 9.1.1：基础游戏框架程序

创建一个简单的 `Pygame` 程序，分为5个步骤：

1. **步骤1**：导入必要的库
2. **步骤2**：初始化 Pygame 和创建窗口
3. **步骤3**：定义颜色、变量和时钟
4. **步骤4**：游戏主循环
5. **步骤5**：程序退出

步骤1：导入必要的库

这一步导入游戏开发所需的核心模块。

- **pygame**：Python游戏开发库，提供窗口创建、图形绘制、事件处理等功能
- **sys**：Python系统库，提供程序退出等系统级功能

```
import pygame
import sys
```

步骤2：初始化Pygame和创建窗口

这一步建立游戏运行环境和显示界面。

- **pygame.init()**：初始化所有 Pygame 模块（音频、视频、字体等）
- **pygame.display.set_mode()**：创建游戏窗口，参数为**（宽度，高度）**元组

```
## 初始化Pygame
pygame.init()

## 设置窗口大小
screen = pygame.display.set_mode((800, 600))
pygame.display.set_caption("我的第一个Pygame窗口")
```

步骤3：定义颜色、变量和时钟

这一步设置游戏运行所需的基础数据和控制参数。

```
## 创建时钟对象控制帧率
clock = pygame.time.Clock()

## 定义颜色（RGB值）
WHITE = (255, 255, 255)
RED = (255, 0, 0)
LIGHT_GREEN = (144, 238, 144)

## 游戏运行状态
running = True
```

步骤3：变量说明

- `pygame.time.Clock()`：创建时钟对象，用于控制游戏帧率
- `WHITE = (255, 255, 255)`：定义白色，RGB 三个分量都是 255
- `RED = (255, 0, 0)`：定义红色，只有红色分量为 255
- `LIGHT_GREEN = (144, 238, 144)`：定义浅绿色
- `running`：布尔变量，控制游戏循环的启动和停止

步骤4：游戏主循环

这一步实现游戏的核心运行逻辑，持续处理用户交互和画面更新。

```
while running:
    # 处理事件
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False

    # 填充背景色
    screen.fill(WHITE)

    # 在屏幕中央绘制一个浅绿色矩形
    pygame.draw.rect(screen, LIGHT_GREEN, (300, 200, 200, 150))

    # 更新显示
    pygame.display.flip()

    # 控制帧率为60FPS
    clock.tick(60)
```

步骤4：函数说明

- `pygame.event.get()`：获取事件队列中的所有事件，返回事件列表
- `pygame.QUIT`：窗口关闭事件类型常量
- `screen.fill()`：用指定颜色填充整个屏幕表面
- `pygame.draw.rect()`：绘制矩形，参数为`(表面，颜色，(x，y，宽，高))`
- `pygame.display.flip()`：更新整个屏幕显示，显示所有绘制内容
- `clock.tick(60)`：控制游戏帧率为`60FPS`

步骤5：程序退出

这一步安全清理游戏资源并正常退出程序。

- `pygame.quit()`：清理并释放所有 Pygame 资源（窗口、音频、内存等）
- `sys.exit()`：终止 Python 程序进程，确保程序完全退出

```
## 退出游戏
pygame.quit()
sys.exit()
```

完整代码

```
import pygame
import sys

## 初始化Pygame
pygame.init()

## 设置窗口大小
screen = pygame.display.set_mode((800, 600))
pygame.display.set_caption("我的第一个Pygame窗口")

## 创建时钟对象控制帧率
clock = pygame.time.Clock()

## 定义颜色（RGB值）
WHITE = (255, 255, 255)
RED = (255, 0, 0)
LIGHT_GREEN = (144, 238, 144)

## 游戏运行状态
running = True
```

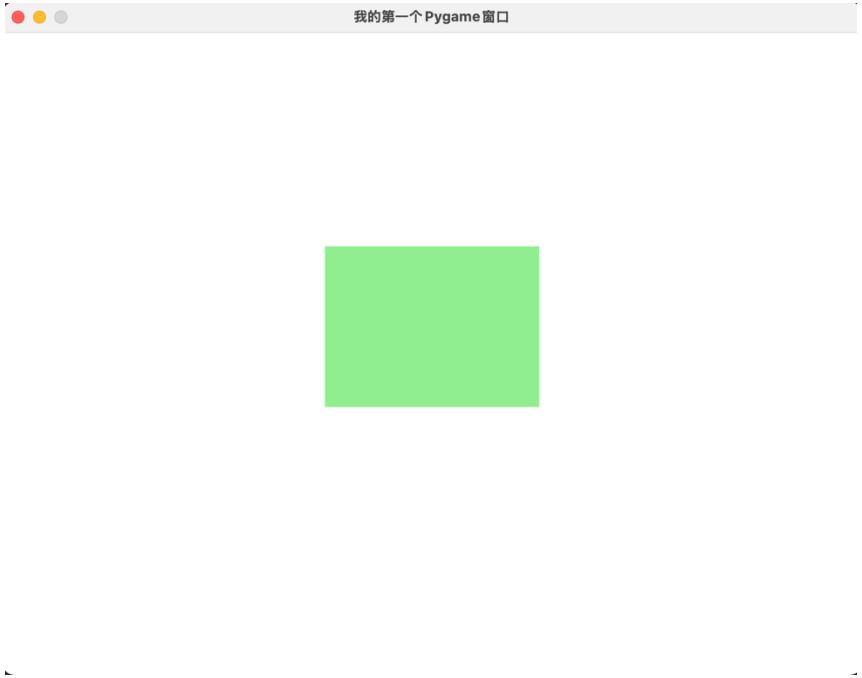
完整代码（续）

```
while running:
    # 处理事件
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False

    # 填充背景色
    screen.fill(WHITE)

    # 在屏幕中央绘制一个浅绿色矩形
    pygame.draw.rect(screen, LIGHT_GREEN, (300, 200, 200, 150))

    # 更新显示
    pygame.display.flip()

    # 控制帧率为60FPS
    clock.tick(60)

## 退出游戏
pygame.quit()
sys.exit()
```

程序运行效果

程序功能：

- 创建 800×600 像素窗口
- 白色背景
- 屏幕中央显示浅绿色矩形
- 点击关闭按钮退出程序



实践练习

练习 9.1.1：环境搭建与基础程序

1. 安装Pygame并验证安装成功
2. 运行第一个Pygame程序，观察窗口和浅绿色矩形的显示
3. 尝试修改窗口大小、背景颜色和矩形参数

实践练习

练习 9.1.2：游戏循环理解

1. 尝试修改帧率（将`60`改为`30`或`120`），观察程序运行的变化
2. 添加更多图形元素，如圆形、直线或多个矩形
3. 理解游戏循环中每个步骤的作用和顺序

本节小结

- Pygame 是基于 SDL 的 Python 游戏开发库
- 游戏循环是游戏程序的核心架构
- 掌握了基本游戏框架的5个步骤
- 了解了事件处理、图形绘制、帧率控制