

Pygame色彩与绘图机制



嵩 天
北京理工大学





单元开篇

Pygame色彩与绘图机制

Pygame



Pygame色彩机制



Pygame图形绘制机制



Pygame文字绘制机制



Pygame绘图机制原理精髓



壁球小游戏(文字型)



python

弹指之间·享受创新



Pygame色彩机制

色彩表达

pygame.Color

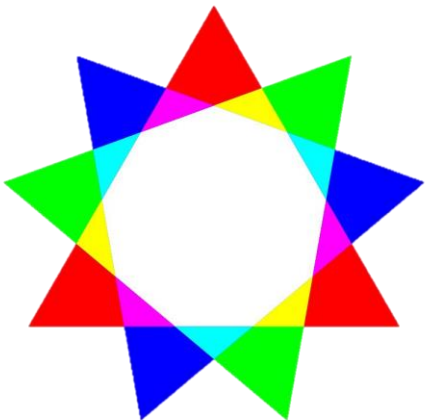
Color类用于表达色彩，使用RGB或RGBA色彩模式，A可选
Color类可以用色彩名字、RGBA值、HTML色彩格式等方式定义

Color(name) 例如：Color("grey")

Color(r,g,b,a) 例如：Color(190, 190, 190, 255)

Color(rgbavalue) 例如：Color("#BEBEBEFF")

RGB色彩模式



- 红绿蓝三个通道颜色组合
- 覆盖视力所能感知的所有颜色
- RGB取值范围0-255

英文名称	R.G.B.	中文名称
white	255 255 255	白色
black	0 0 0	黑色
grey	190 190 190	灰色
darkgreen	0 100 0	深绿色
gold	255 215 0	金色
violet	238 130 238	紫罗兰
purple	160 32 240	紫色

RGBA色彩模式

- RGB色彩模式之外增加了第四维度：alpha通道
- alpha通道表示不透明度，取值0-255，默认255
- alpha通道值越大，不透明度越高，255表示不透明

pygame.Color类

`pygame.Color.r`

获得Color类的红色值r

`pygame.Color.g`

获得Color类的绿色值g

`pygame.Color.b`

获得Color类的蓝色值b

`pygame.Color.a`

获得Color类的不透明度值a

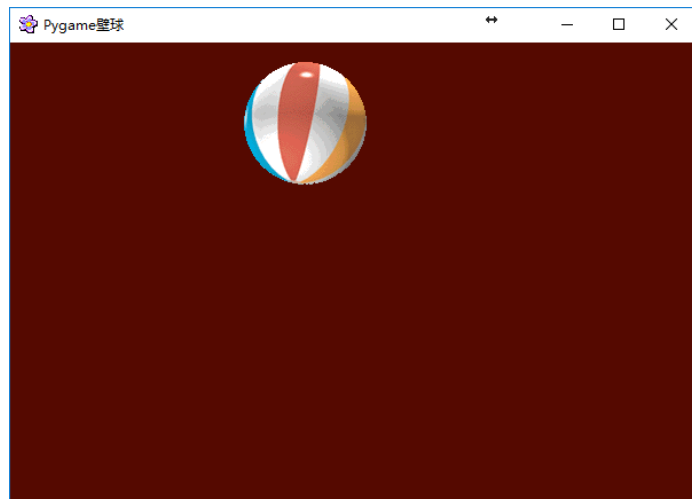
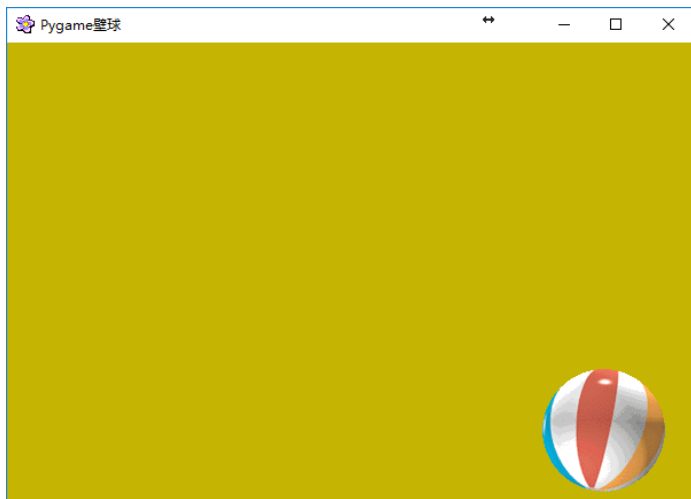
`pygame.Color.normalize`

将RGBA各通道值归一到0-1之间

壁球小游戏(色彩型)

需求：

根据壁球移动状态修改游戏的背景色



壁球小游戏(色彩型)

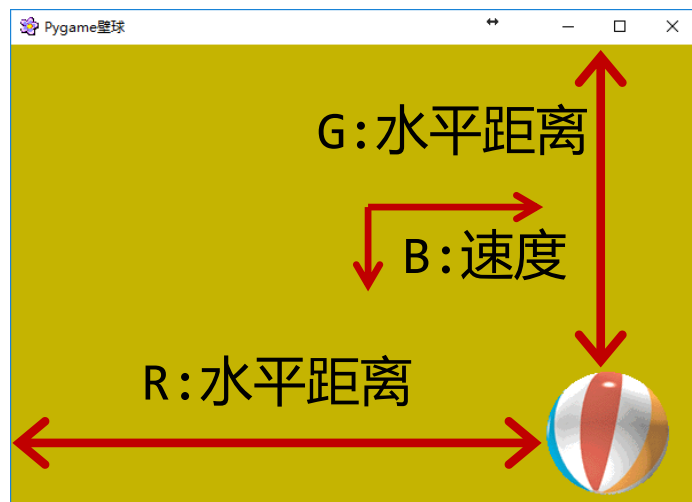
设计：

RGB颜色分别定义如下：

R：壁球水平距离

G：壁球垂直距离

B：壁球水平和垂直速度差别



壁球小游戏(色彩型)

具体实现：

R:水平距离/窗体宽度，取值0-255

G:垂直距离/窗体高度，取值0-255

B:最小速度/最大速度，取值0-255



壁球小游戏(色彩型)

```
# Unit PYG05: Pygame Wall Ball Game version 9 色彩型
import pygame, sys
```

```
def RGBChannel(a):
    return 0 if a<0 else (255 if a>255 else int(a))
```

```
pygame.init()
size = width, height = 600, 400
speed = [1,1]
bgcolor = pygame.Color("black")
screen = pygame.display.set_mode(size, pygame.RESIZABLE) #窗口大小可调
```

```
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sys.exit()
```

```
.....
.....
```

```
bgcolor.r = RGBChannel(ballrect.left*255/width)
bgcolor.g = RGBChannel(ballrect.top*255/height)
bgcolor.b = RGBChannel(min(speed[0], speed[1])*255/max(speed[0], speed[1], 1))
```

```
screen.fill(bgcolor)
screen.blit(ball, ballrect)
pygame.display.update()
fclock.tick(fps)
```

与老师一起热热身吧

PYG05-PygameWallBallv9.py

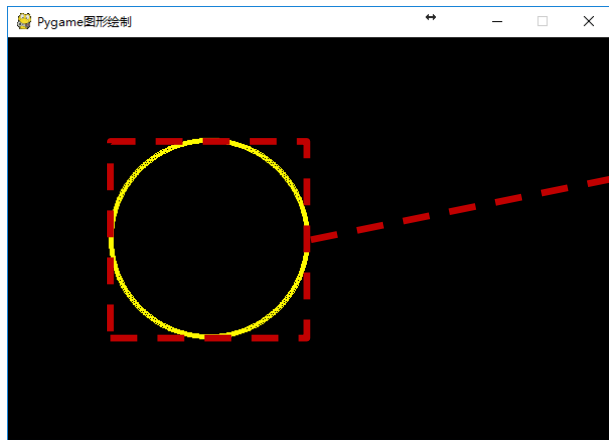


Pygame图形绘制机制

图形绘制

`pygame.draw`

向屏幕上绘制一些简单的图形，如直线、圆形、椭圆等
任何一个图形绘制后，会返回一个矩形Rect类表示该形状



→ 圆形绘制后，用Rect类表示

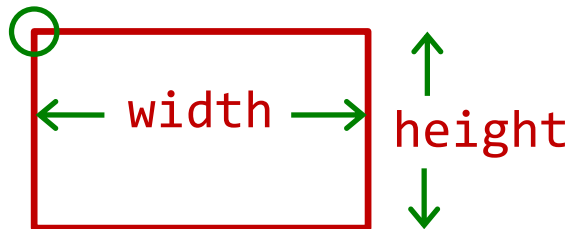
Rect类

`pygame.Rect`

表达一个矩形区域的类，用于存储坐标和长度信息

Pygame利用Rect类来操作图形/图像等元素

`(left, top)`



四个参数如下：

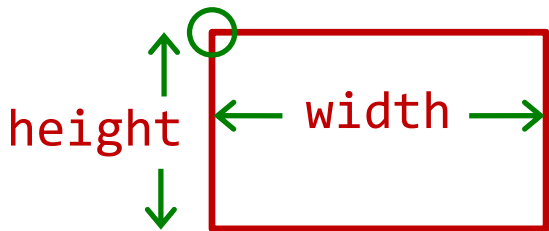
`left, top, width, height`

Rect类

`pygame.Rect`

Rect类提供了如下属性，返回一个数值或一个代表坐标的元组

(left, top)



`x, y, w, h, size, width, height`

`top, left, bottom, right`

`topleft, bottomleft, topright, bottomright`

`midtop, midleft, midbottom, midright`

`center, centerx, centery`

<http://www.pygame.org/docs/ref/rect.html>

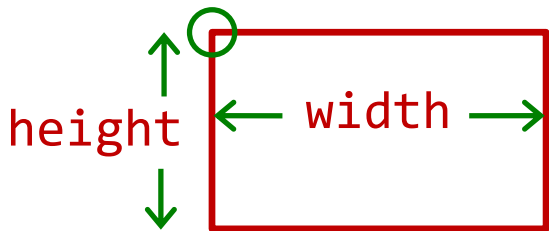
猜猜含义

Rect类

`pygame.Rect`

Rect类提供了如下方法，用来操作Rect类

(left, top)



`.copy()`, `.move()`, `.inflate()`, `.clamp()`, `.clip()`,
`.union()`, `.unionall()`, `.fit()`,
`.normalize()`, `.contains()`, `.collidepoint()`
`.colliderect()`, `.collidelist()`, `.collidelistall()`,
`.collidedict()`, `.collidedictall()`

<http://www.pygame.org/docs/ref/rect.html>

猜猜含义

图形绘制

`pygame.draw`

<code>.rect()</code>	矩形
<code>.polygon()</code>	多边形
<code>.circle()</code>	圆形
<code>.ellipse()</code>	椭圆形
<code>.arc()</code>	椭圆弧形

<code>.line()</code>	直线
<code>.lines()</code>	连续多线
<code>.aaline()</code>	无锯齿线
<code>.aalines()</code>	连续无锯齿线

矩形绘制

```
pygame.draw.rect(Surface, color, Rect, width=0)
```

- `Surface` 矩形的绘制屏幕
- `color` 矩形的绘制颜色
- `Rect` 矩形的绘制区域
- `width=0` 绘制边缘的宽度，默认为0，即填充图形

```
# Unit PYG05: Pygame Shape Draw Test
```

```
import pygame, sys
```

```
pygame.init()
```

```
screen = pygame.display.set_mode((600, 400))
```

```
pygame.display.set_caption("Pygame图形绘制")
```

```
GOLD = 255, 251, 0
```

```
RED = pygame.Color('red')
```

与老师一起热热身吧

```
r1rect = pygame.draw.rect(screen, GOLD, (100,100,200,100), 5)
```

```
r2rect = pygame.draw.rect(screen, RED, (210,210,200,100), 0)
```

```
while True:
```

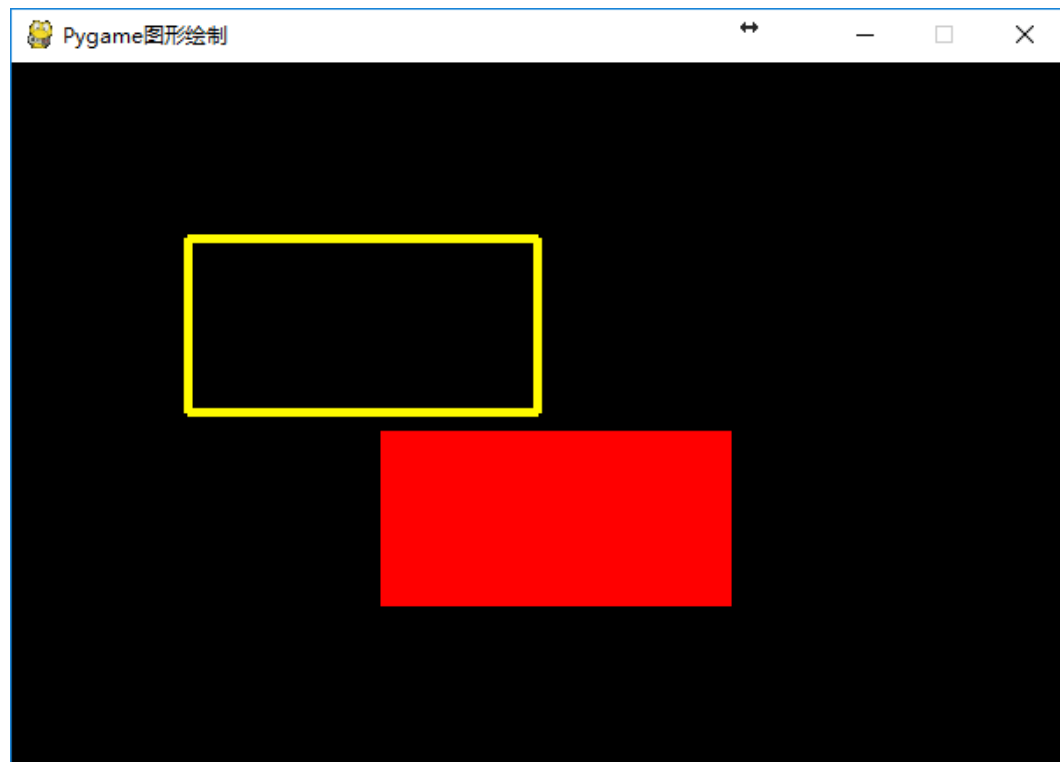
```
    for event in pygame.event.get():
```

```
        if event.type == pygame.QUIT:
```

```
            sys.exit()
```

```
    pygame.display.update()
```

PYG05-PygameShapeDraw.py



多边形绘制

```
pygame.draw.polygon(Surface, color, pointlist, width=0)
```

- `Surface` 多边形的绘制屏幕
- `color` 多边形的绘制颜色
- `pointlist` 多边形顶点坐标列表
- `width=0` 绘制边缘的宽度，默认为0，即填充图形

圆形绘制

```
pygame.draw.circle(Surface, color, pos, radius, width=0)
```

- `Surface` 圆形的绘制屏幕
- `color` 圆形的绘制颜色
- `pos` 圆形的圆心坐标
- `radius` 圆形的半径
- `width=0` 绘制边缘的宽度，默认为0，即填充图形


椭圆形绘制

```
pygame.draw.ellipse(Surface, color, Rect, width=0)
```

- `Surface` 椭圆形的绘制屏幕
- `Color` 椭圆形的绘制颜色
- `Rect` 椭圆形的绘制区域
- `width=0` 绘制边缘的宽度，默认为0，即填充图形

椭圆弧形绘制

```
pygame.draw.arc(Surface, color, Rect, start_angle,  
                stop_angle, width=0)
```

- `Surface` 椭圆弧形的绘制屏幕
 - `Color` 椭圆弧形的绘制颜色
 - `Rect` 椭圆弧形的绘制区域
 - `start_angle, stop_angle` 弧形绘制起始和结束弧度值
 - `width=0` 绘制边缘的宽度，默认为0，即填充图形
- 横向右侧为0度
- 

直线绘制

```
pygame.draw.line(Surface, color, start_pos, end_pos, width=1)
```

- `Surface` 直线的绘制屏幕
- `Color` 直线的绘制颜色
- `start_pos, end_pos` 直线的起始和结束坐标
- `width=1` 直线的宽度，默认值为1

连续多线绘制

```
pygame.draw.lines(Surface, color, closed, pointlist, width=1)
```

- `Surface` 连续多线的绘制屏幕
- `Color` 连续多线的绘制颜色
- `closed` 如果为`True`,起止节点间自动增加封闭直线
- `pointlist` 连续多线的顶点坐标列表
- `width=1` 连续多线的宽度, 默认值为1

无锯齿线绘制

```
pygame.draw.aaline(Surface, color, start_pos, end_pos, blend=1)
```

- `Surface` 无锯齿线的绘制屏幕
- `Color` 无锯齿线的绘制颜色
- `start_pos, end_pos` 无锯齿线的起始和结束坐标
- `blend=1` 不为0时，与线条所在背景颜色进行混合

连续无锯齿线绘制

```
pygame.draw.aalines(Surface, color, closed, pointlist, blend=1)
```

- `Surface` 连续无锯齿线的绘制屏幕
- `Color` 连续无锯齿线的绘制颜色
- `closed` 如果为`True`,起止节点间自动增加封闭直线
- `pointlist` 连续无锯齿线的顶点坐标列表
- `blend=1` 不为0时,与线条所在背景颜色进行混合

```
# Unit PYG05: Pygame Shape Draw Test
import pygame, sys
from math import pi

pygame.init()
screen = pygame.display.set_mode((600, 400))
pygame.display.set_caption("Pygame图形绘制")
GOLD = 255, 251, 0
RED = pygame.Color('red')
WHITE = 255, 255, 255
GREEN = pygame.Color('green')

#r1rect = pygame.draw.rect(screen, GOLD, (100,100,200,100), 5)
#r2rect = pygame.draw.rect(screen, RED, (210,210,200,100), 0)

e1rect = pygame.draw.ellipse(screen, GREEN, (50,50,500,300), 3)
c1rect = pygame.draw.circle(screen, GOLD, (200,180), 30, 5)
c2rect = pygame.draw.circle(screen, GOLD, (400,180), 30)
r1rect = pygame.draw.rect(screen, RED, (170,130, 60, 10), 3)
r2rect = pygame.draw.rect(screen, RED, (370,130, 60, 10))
plist = [(295,170), (285,250), (260,280), (340,280), (315,250), (305,170)]
#l1rect = pygame.draw.lines(screen, GOLD, True, plist, 2)
allrect = pygame.draw.aalines(screen, GOLD, True, plist, 2)
a1rect = pygame.draw.arc(screen, RED, (200,220,200,100), 1.4*pi, 1.9*pi, 3)

while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sys.exit()
    pygame.display.update()
```

与老师一起热热身吧

猜猜会输出什么？

PYG05-PygameShapeDraw.py



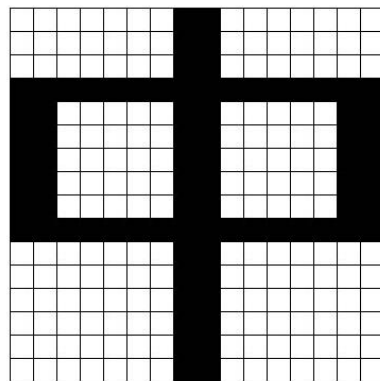
Pygame文字绘制机制

文字绘制

`pygame.freetype`

向屏幕上绘制特定字体的文字

文字不能直接`print()`，而是用像素根据字体点阵图绘制



文字绘制

`pygame.freetype`

`pygame.freetype`是绘制文字的增强方法，建议使用
必须额外增加import引用，如下：

```
import pygame, sys
import pygame.freetype
```

系统中的字体



Windows系统

C:\Windows\Fonts

字体文件的扩展名

*.ttf

*.ttc

文字绘制机制简介

`pygame.freetype`

`pygame.freetype.Font`



`Font.render_to()`

`Font.render()`

根据字体和字号生成
一个Font对象

用Font对象的render*方
法绘制具体文字

Font类

```
pygame.freetype.Font(file, size=0)
```

- `file` 字体类型名称或路径
- `size` 字体的大小

Font类的绘制方法(1)

```
Font.render_to(surf, dest, text, fgcolor=None,  
                bgcolor=None, rotation=0, size=0) -> Rect
```

- surf 绘制字体的平面，Surface对象
- dest 在平面中的具体位置，(x,y)
- text 绘制的文字内容
- fgcolor 文字颜色

Font类的绘制方法(1)

```
Font.render_to(surf, dest, text, fgcolor=None,  
                bgcolor=None, rotation=0, size=0) -> Rect
```

- `bgcolor` 背景颜色
- `rotation` 逆时针的旋转角度，取值0-359，部分字体可旋转
- `size` 文字大小，赋值该参数将覆盖Font中的设定值

Font类的绘制方法(1)

```
Font.render_to(surf, dest, text, fgcolor=None,  
                bgcolor=None, rotation=0, size=0) -> Rect
```

Rect 返回一个Rect对象

```
# Unit PYG05: Pygame Font Draw 1
```

```
import pygame, sys
```

```
import pygame.freetype
```

```
pygame.init()
```

```
screen = pygame.display.set_mode((600, 400))
```

```
pygame.display.set_caption("Pygame文字绘制")
```

```
GOLD = 255, 251, 0
```

```
f1 = pygame.freetype.Font('C://Windows//Fonts//msyh.ttc', 36) #微软雅黑
```

```
f1rect = f1.render_to(screen, (200, 160), "世界和平", fgcolor=GOLD, size=50)
```

```
while True:
```

```
    for event in pygame.event.get():
```

```
        if event.type == pygame.QUIT:
```

```
            sys.exit()
```

```
    pygame.display.update()
```

与老师一起热热身吧

PYG05-PygameFontDraw1.py



Font类的绘制方法(2)

`Font.render(text, fgcolor=None, bgcolor=None, rotation=0, size=0) -> (Surface, Rect)`

- `text` 绘制的文字内容
- `fgcolor, bgcolor` 字体颜色、背景颜色
- `rotation` 逆时针的旋转角度，取值0-359，部分字体可旋转
- `size` 文字大小，赋值该参数将覆盖Font中的设定值

Font类的绘制方法(2)

```
Font.render(text, fgcolor=None, bgcolor=None,  
rotation=0, size=0) -> (Surface, Rect)
```

返回一个元组，包含Surface对象和Rect对象

```
# Unit PYG05: Pygame Font Draw 2
```

```
import pygame, sys
```

```
import pygame.freetype
```

```
pygame.init()
```

```
screen = pygame.display.set_mode((600, 400))
```

```
pygame.display.set_caption("Pygame文字绘制")
```

```
GOLD = 255, 251, 0
```

与老师一起热热身吧

```
f1 = pygame.freetype.Font('C://Windows//Fonts//msyh.ttc', 36) #微软雅黑
```

```
f1surf, f1rect = f1.render("世界和平", fgcolor=GOLD, size=50)
```

```
while True:
```

```
    for event in pygame.event.get():
```

```
        if event.type == pygame.QUIT:
```

```
            sys.exit()
```

```
    screen.blit(f1surf, (200, 160))
```

```
    pygame.display.update()
```

PYG05-PygameFontDraw2.py



Pygame绘图机制原理精髓

理解Pygame的两个重要类型

pygame.Surface

绘图层，或绘图平面，或图层

- 用于表示图形、文字或图像的绘制效果
- 与当前屏幕主图层可以并列存在
- 如果不绘制在主图层上，则不会被显示

pygame.Rect

矩形区域

- 对应于当前主图层的某个具体区域
- 相当于某个矩形区域的指针或标识信息
- 可以指定图层绘制在某个矩形区域中

主图层

由pygame.display.set_mode()生成的Surface对象

```
size = width, height = 600, 400  
screen = pygame.display.set_mode(size)
```

在主图层上绘制其他图层使用.blit()方法

```
screen.blit(ball, ballrect)
```

pygame.Surface

pygame.Rect



理解绘制过程

pygame.Surface

包含某个图
形的图层



pygame.Rect

定位某个区域

将图层绘制在区域内



壁球小游戏(文字型)



壁球小游戏(文字型)

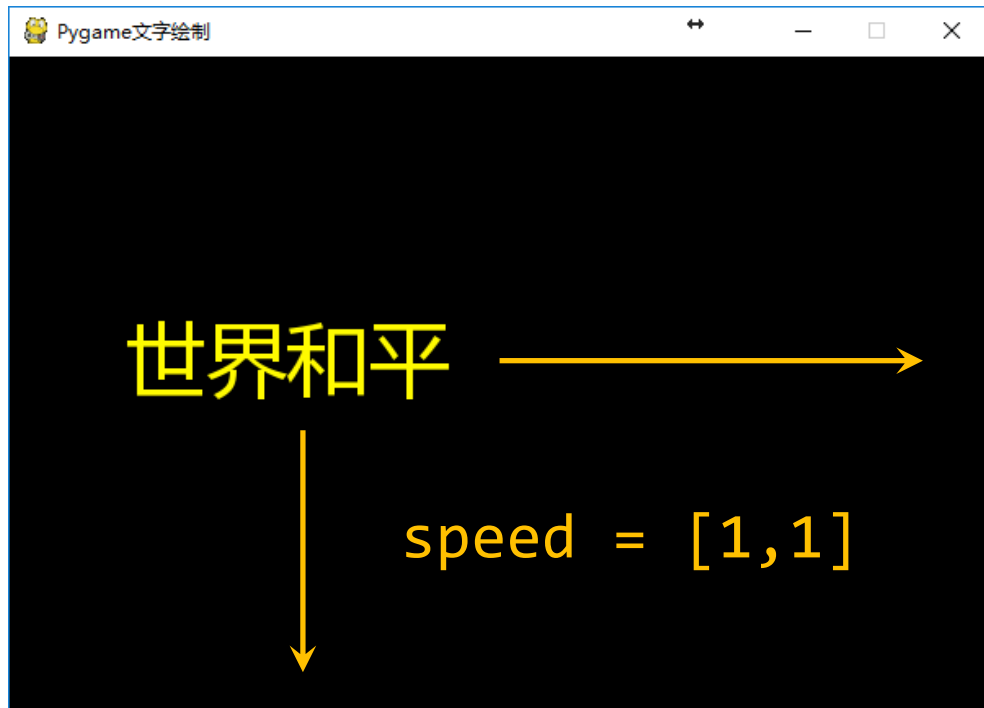
需求：

把壁球改为一段文字，可以进行移动

从需求到实现的关键要素：

- **文字移动**：文字的移动绘制及刷新

壁球小游戏(文字型)



壁球小游戏(文字型)

PygameWallBallv10.py采用.render_to()方法

PygameWallBallv11.py采用.render()方法

```
# Unit PYG05: Pygame Wall Ball Game version 10, 文字型A
```

```
import pygame, sys
```

```
import pygame.freetype
```

```
pygame.init()
```

```
size = width, height = 600, 400
```

```
screen = pygame.display.set_mode(size)
```

```
speed = [1,1]
```

```
GOLD = 255, 251, 0
```

```
BLACK = 0, 0, 0
```

```
pos = [230, 160]
```

```
pygame.display.set_caption("Pygame文字绘制")
```

```
f1 = pygame.freetype.Font('C://Windows//Fonts//msyh.ttc', 36)
```

```
f1rect = f1.render_to(screen, pos, "世界和平", fgcolor=GOLD, size=50)
```

```
fps = 300
```

```
fclock = pygame.time.Clock()
```

```
while True:
```

```
    for event in pygame.event.get():
```

```
        if event.type == pygame.QUIT:
```

```
            sys.exit()
```

```
    if pos[0] < 0 or pos[0] + f1rect.width > width:
```

```
        speed[0] = -speed[0]
```

```
    if pos[1] < 0 or pos[1] + f1rect.height > height:
```

```
        speed[1] = -speed[1]
```

```
    pos[0] = pos[0] + speed[0]
```

```
    pos[1] = pos[1] + speed[1]
```

```
    screen.fill(BLACK)
```

```
    f1rect = f1.render_to(screen, pos, "世界和平", fgcolor=GOLD, size=50)
```

```
    pygame.display.update()
```

```
    fclock.tick(fps)
```

与老师一起重写壁球小游戏(文字型)吧
使用.render_to()方法

PYG05-PygameWallBallv10.py

```
# Unit PYG05: Pygame Wall Ball Game version 11, 文字型B
```

```
import pygame, sys
```

```
import pygame.freetype
```

```
pygame.init()
```

```
size = width, height = 600, 400
```

```
screen = pygame.display.set_mode(size)
```

```
speed = [1,1]
```

```
GOLD = 255, 251, 0
```

```
BLACK = 0, 0, 0
```

```
pos = [230, 160]
```

```
pygame.display.set_caption("Pygame文字绘制")
```

```
f1 = pygame.freetype.Font('C://Windows//Fonts//msyh.ttc', 36)
```

```
f1surf, f1rect = f1.render("世界和平", fgcolor=GOLD, size=50)
```

```
fps = 300
```

```
fclock = pygame.time.Clock()
```

```
while True:
```

```
    for event in pygame.event.get():
```

```
        if event.type == pygame.QUIT:
```

```
            sys.exit()
```

```
    if pos[0] < 0 or pos[0] + f1rect.width > width:
```

```
        speed[0] = -speed[0]
```

```
    if pos[1] < 0 or pos[1] + f1rect.height > height:
```

```
        speed[1] = -speed[1]
```

```
    pos[0] = pos[0] + speed[0]
```

```
    pos[1] = pos[1] + speed[1]
```

```
    screen.fill(BLACK)
```

```
    f1surf, f1rect = f1.render("世界和平", fgcolor=GOLD, size=50)
```

```
    screen.blit(f1surf, (pos[0], pos[1]))
```

```
    pygame.display.update()
```

```
    fclock.tick(fps)
```

与老师一起重写壁球小游戏(文字型)吧
使用.render()方法

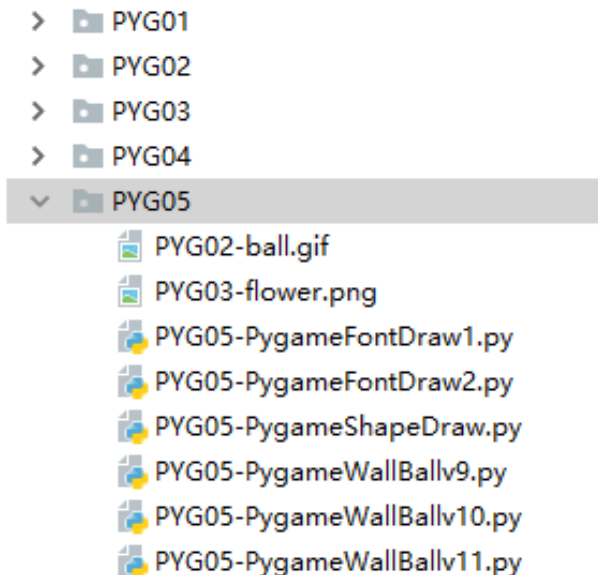
PYG05-PygameWallBallv11.py



单元小结



- Pygame色彩机制
- Pygame图形绘制机制
- Pygame文字绘制机制
- Pygame绘制机制原理精髓
- 壁球小游戏(文字型)



Pygame色彩与绘制机制