



ANGRY PIGGY

汇报人:
陈逸茗
赵子璇
★王译哈

CONTENTS



01

开始的准备工作



02

更改图标和背景图的大小



03

定义我们的猪猪主角



04

定义猪猪前进路上的阻碍

CONTENTS



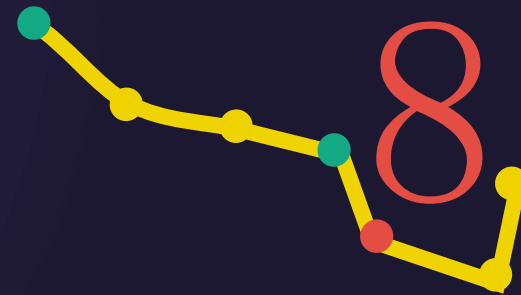
05
**为调用enemy
作准备**



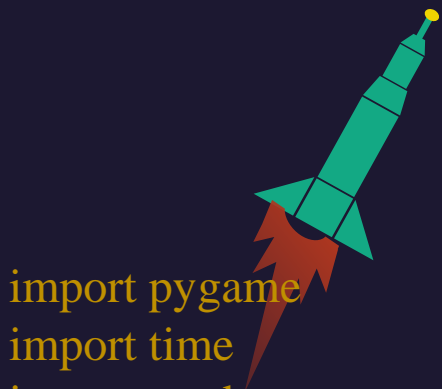
06
**让猪猪
移动起来**



07
**调用猪猪前进
路上的障碍**



08
**让游戏
运作起来**



```
import pygame
import time
import random
from pygame.locals import *
from sys import exit
```

```
pygame.init()
bgsz = width, height = 1000, 600
pygame.display.set_mode(bgsz)
pygame.display.set_caption('Angry Piggy')
bg =
pygame.image.load('C:\\Users\\19581\\Desktop\\Python\\angrypiggy\\
bgstar.jpg')
piggy_image =
pygame.image.load('C:\\Users\\19581\\Desktop\\Python\\angrypiggy\\
piggy.jpg')
enemy_image =
pygame.image.load('C:\\Users\\19581\\Desktop\\Python\\angrypiggy\\
enemy.png')
```

1

开始的准备工作

Pygame是一个设计用来开发游戏的模块。Time是用来计时的模块。第四行是将所有的Pygame常量导入，比如下面用到的第八行。再从sys模块中借一个exit函数来退出程序

初始化pygame，为使用硬件做准备
定义背景的长和宽
生成Windows窗口，返回的是一个Surface对象，
bgsz就是窗口的大小
获得窗口的标题
调用背景的图像
猪猪的图像
障碍物的图像

②

更改图标和背景图的大小

```
piggy_image =  
pygame.transform.scale  
(piggy_image,(30, 20))
```

括号里是缩放的
Surface和长宽

```
screen =  
pygame.display.get_sur  
face()
```

获取当前的
Surface对象



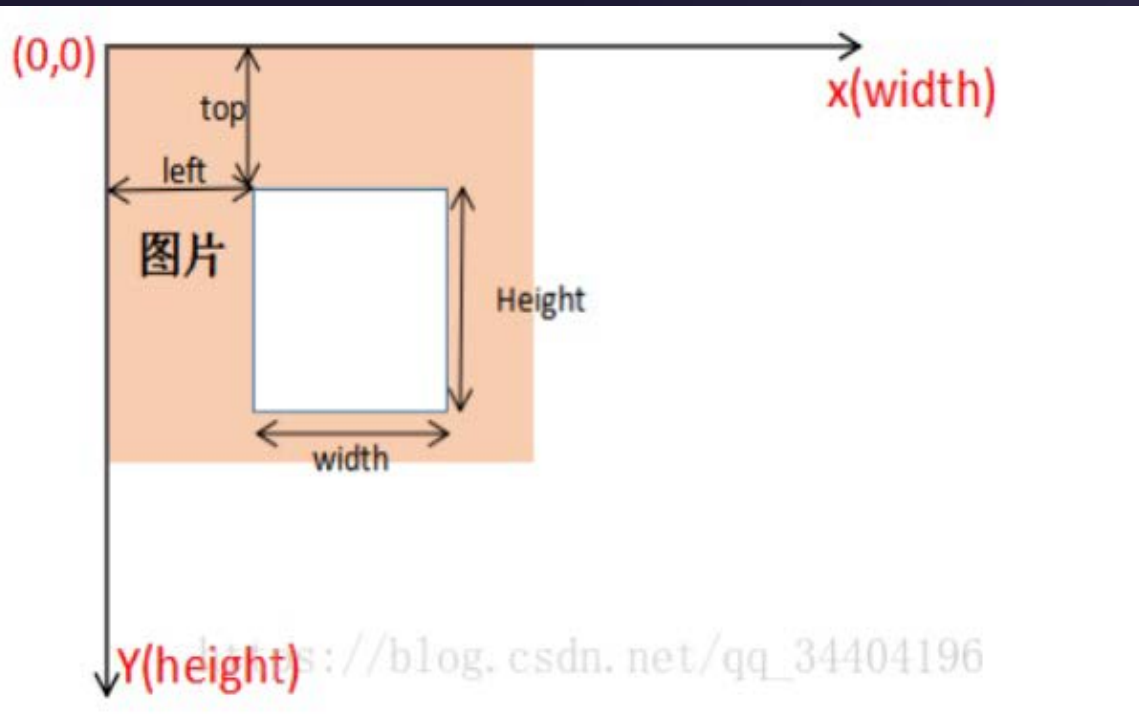
```
bg =  
pygame.transform.scale  
(bg,(1000, 600))
```

```
enemy_image =  
pygame.transform.scale  
(enemy_image,(20, 15))
```

3

定义我们的猪猪主角 ✨

我们即将用到用到课堂上学过的
“Class”。但首先要引入rect的概念



Pygame 通过 Rect 对象存储和操作矩形区域。一个 Rect 对象可以由 left, top, width, height 几个值创建。Rect 也可以是由 Pygame 的对象所创建，它们拥有一个属性叫“rect”。任何需要一个 Rect 对象作为参数的 Pygame 函数都可以使用以上值构造一个 Rect。这样使得作为参数传递的同时创建 Rect 成为可能。

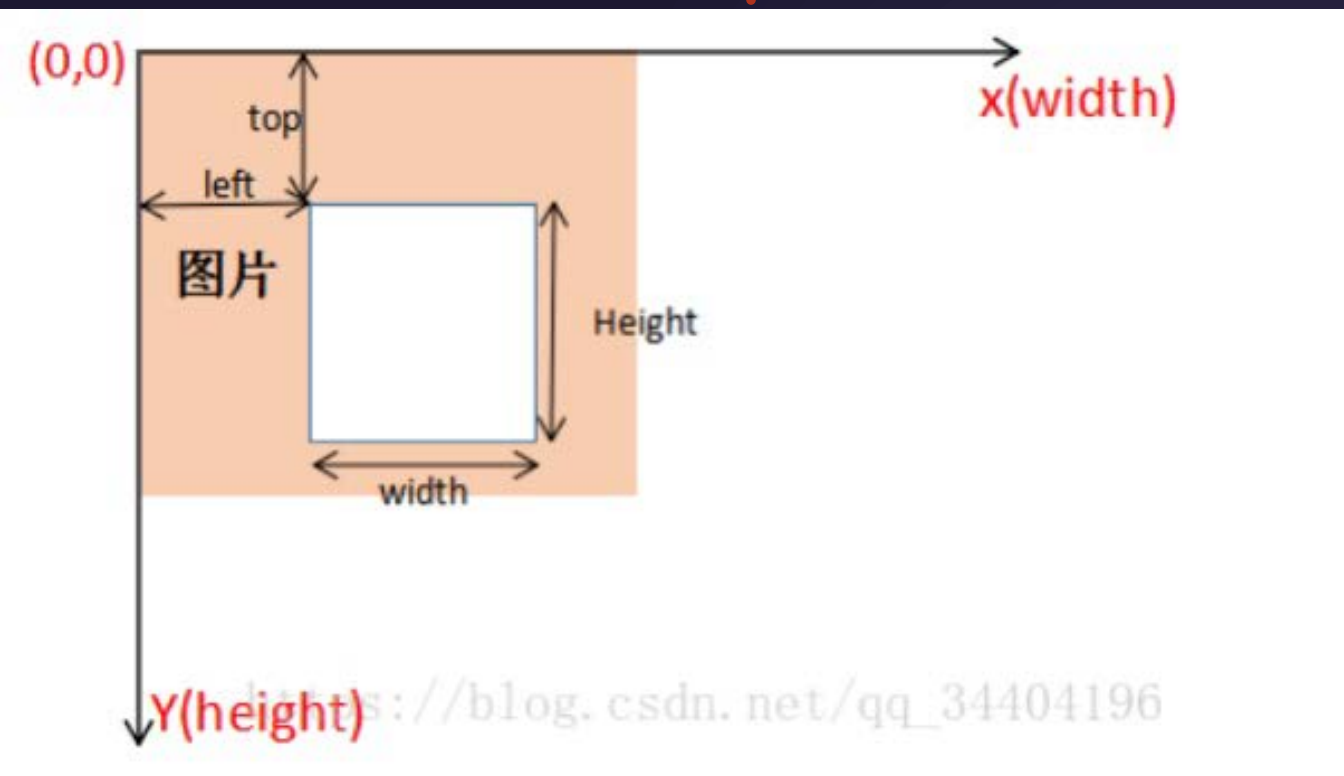
Rect 对象中的大部分方法在修改矩形的位置、尺寸后会返回一个新的 Rect 拷贝，原始的 Rect 对象不会有任何改变。但有些方法比较特殊，它们会“原地”修改 Rect 对象（也就是说它们会改动原始的 Rect 对象）。

常用的 Rect 参数有这个形式：`pygame.Rect(left, top, width, height)`

我们来用课堂知识定义一只猪猪

```
class Piggy(pygame.sprite.Sprite):  
    def __init__(self, up_speed=5, down_speed=5, left_speed=5, right_speed=5):  
        pygame.sprite.Sprite.__init__(self)  
        self.up_speed = up_speed  
        self.down_speed = down_speed  
        self.left_speed = left_speed  
        self.right_speed = right_speed  
        self.image = piggy_image  
        self.rect = self.image.get_rect()  
        self.rect.top = 300  
        self.rect.left = (width - self.image.get_width()) // 2
```

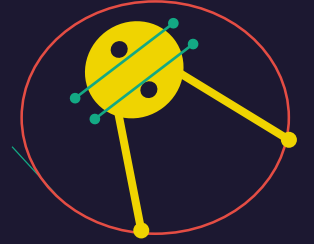
sprite是pygame附属的一个叫精灵的模块，Sprite是sprite模块下的一个类
我们定义猪猪可以上下左右移动的变量
把四个数值赋值给四个变量
image是一个模块，用于图像的传输，这里是把猪猪的图像传进去



```
def moveup(self):  
    self.rect.top -= self.up_speed  
  
def movedown(self):  
    self.rect.top += self.down_speed  
  
def moveleft(self):  
    self.rect.left -= self.left_speed  
  
def moveright(self):  
    self.rect.left += self.right_speed
```

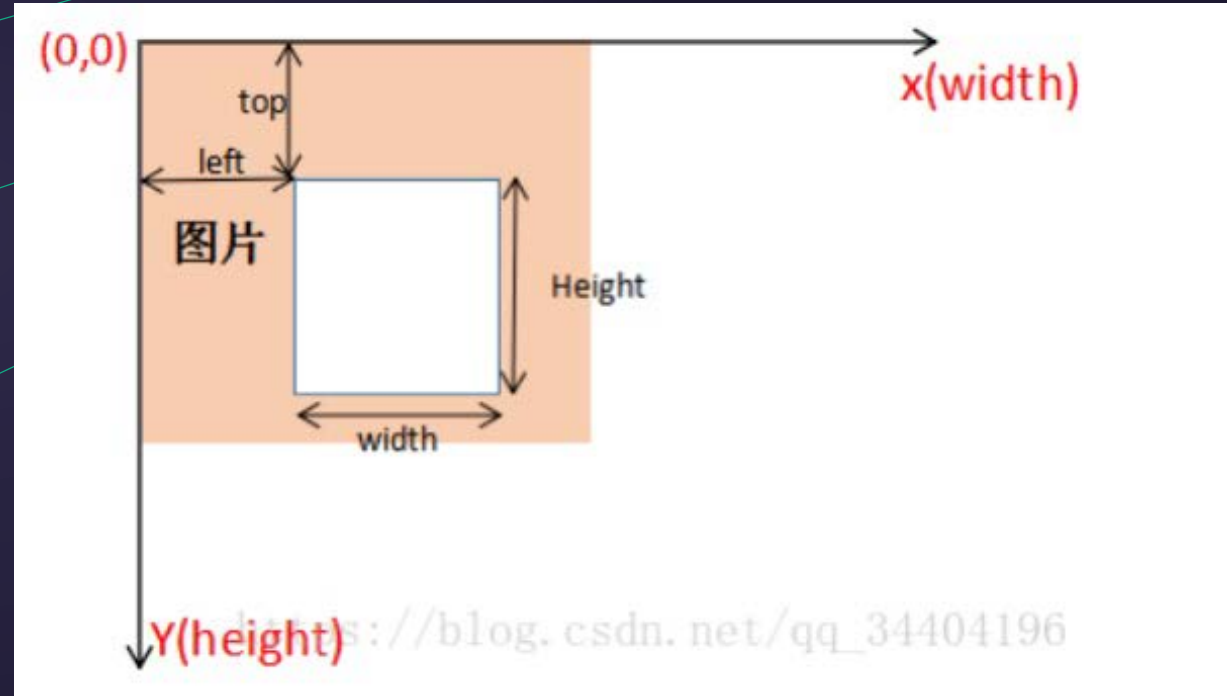



接下来我们来定义猪猪前进路上的阻碍



```
class Enemy(pygame.sprite.Sprite):
    def __init__(self, speed, top):
        pygame.sprite.Sprite.__init__(self)
        self.speed = speed
        self.image = enemy_image
        self.rect = self.image.get_rect()
        self.rect.top = top
        self.rect.left = width - self.image.get_width()

    def moveleft(self):
        self.rect.left -= self.speed
```



```
piggy = Piggy(6, 6, 6, 6)
```

```
clock=pygame.time.Clock()start = time.time()
```

```
enemy_total = []
```

```
time_for_enemy = 2 enemy_number_low = 5
```

```
enemy_number_high = 10
```

```
enemy_speed_low = 5
```

```
enemy_speed_high = 10
```

为调用
enemy
作准备

调用piggy函数

计时

start与后面的end连用计算

当前的系统时长

创建enemy的一个空列表

敌人出现的间隔为2

敌人的数量为5~10

敌人的速度为5~10



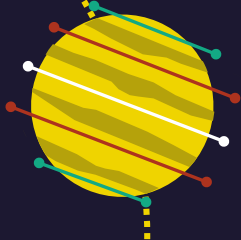
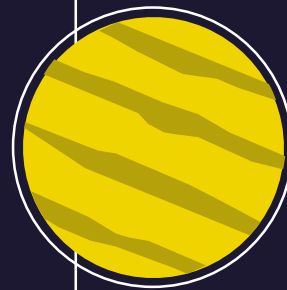
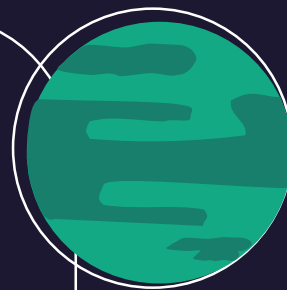
让猪猪移动起来

```
while 1:
```

**不断循环刷新
屏幕**

```
for event in  
pygame.event.g  
et():
```

**获取事件的返
回值**





让猪猪移动起来

```
key_press = pygame.key.get_pressed()
```

获取按键行为

```
elif key_press[K_UP] and piggy.rect.top > 0:  
    piggy.moveup()
```

如果按了向上的按键并且猪猪顶部没有超出屏幕:
那么猪猪向上移动

```
clock.tick(25)
```

控制游戏绘制的最大帧率为25 每秒不超过25帧

```
pygame.display.flip()
```

不断刷新屏幕 让动画尽可能流畅

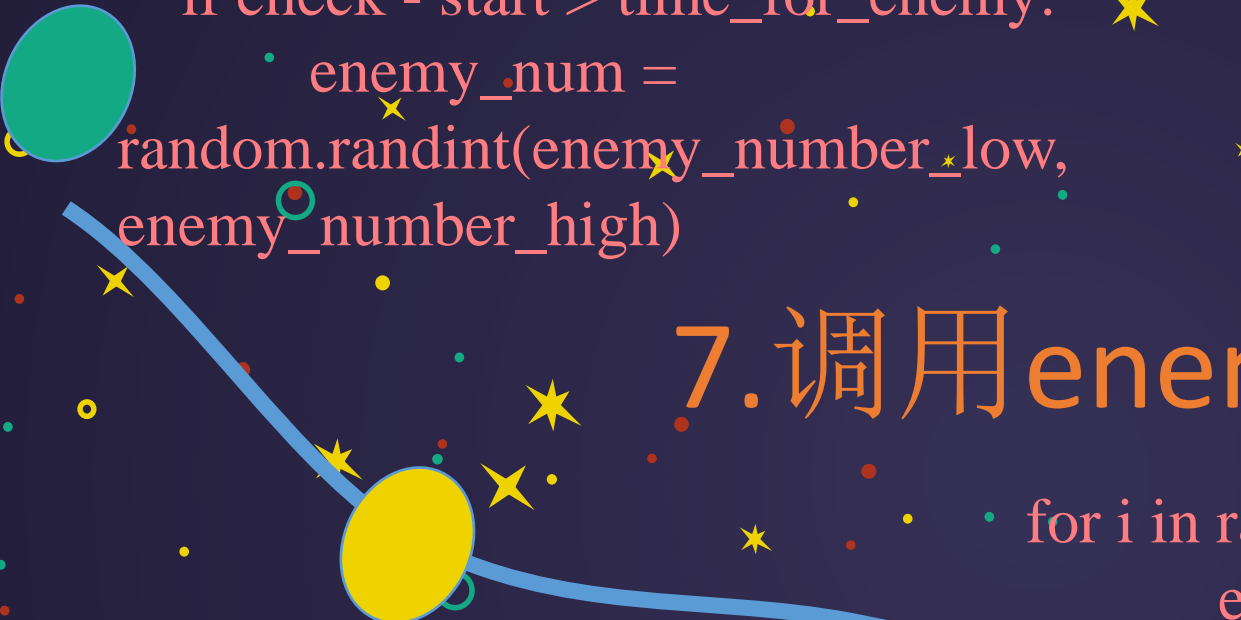
```
if event.type==QUIT:  
    exit()
```

判断是否按下关闭按钮:
是的话 就关闭

```
if key_press[K_DOWN] and piggy.rect.bottom < height:  
    piggy.movedown()
```

如果满足按下向下的按键并且猪猪底部没有超出屏幕
那么猪猪向下移动

```
screen.fill((0,0,0)) #填充背景  
screen.blit(bg, (0,0)) #显示背景图
```



```
check = time.time()
if check - start > time_for_enemy:
    enemy_num =
    random.randint(enemy_number_low,
    enemy_number_high)
```

end与前面的start一起计算时长
如果时长大于敌人的时长:
那么敌人的数量为5~10之间的随机数

7.调用enemy对象

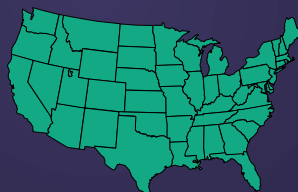
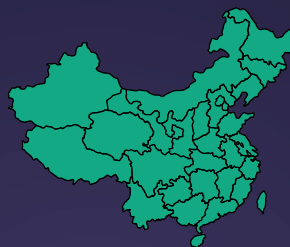
生成了多少敌人 就进行几次循环
每次循环中都调用敌人
敌人的速度参数为5~10之间的随机数
并且在enemy_total列表中增加enemy

```
for i in range(enemy_num):
    enemy = Enemy(speed =
    random.randint(enemy_speed_low,
    enemy_speed_high),
    top = height*(i+1)/enemy_num +
    random.randint(-20,20))
    enemy_total.append(enemy)
```



7. 调用enemy对象

```
for enemy in enemy_total:  
    enemy.moveleft()  
    screen.blit(enemy_image, enemy.rect)  
    if enemy.rect.left <= piggy.rect.left  
and enemy.rect.left >= piggy.rect.left-40  
and enemy.rect.top >= piggy.rect.top and  
enemy.rect.top <= piggy.rect.top+30:  
    print('You Lose!')  
    exit()
```

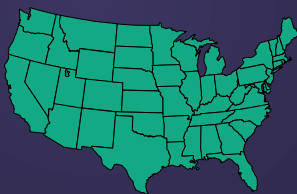


对于列表中的每一个随机生成的enemy
enemy向左移动
显示enemy图像
如果同时猪猪碰到了障碍物
就打印 'You lose'
退出循环



7. 调用enemy对象

```
for enemy in enemy_total:  
    enemy.moveleft()  
    screen.blit(enemy_image, enemy.rect)  
    if enemy.rect.left <= piggy.rect.left  
and enemy.rect.left >= piggy.rect.left-40  
and enemy.rect.top >= piggy.rect.top and  
enemy.rect.top <= piggy.rect.top+30:  
    print('You Lose!')  
    exit()
```



对于列表中的每一个随机生成的enemy
enemy向左移动
显示enemy图像
如果同时猪猪碰到了障碍物
就打印 'You lose'
退出循环



让游戏运作起来

```
screen.blit(piggy_image,piggy.rect)
```

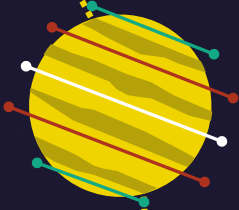
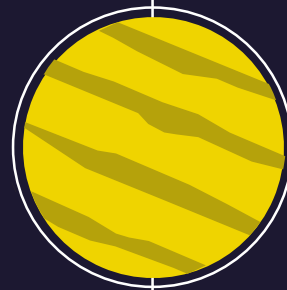
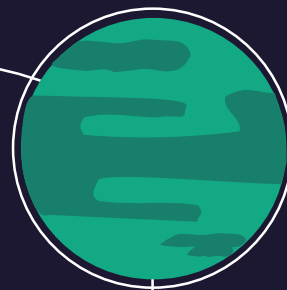
显示小猪图像

```
clock.tick(25)
```

控制游戏绘制的最大帧率为25 每秒不超过25帧

```
pygame.display.flip()
```

不断刷新屏幕 让动画尽可能流畅



```
intro_sound = .
```

```
pygame.mixer.Sound('C:\\Users\\19581\\Desktop\\Python\\angrypiggy
```

```
\\bgmusic.wav')
```

```
intro_sound.play()
```

9

一点小优化

让我们重温
这段熟悉的
Background
Music~

```
def gameover (playSurface):  
    redColour = pygame.Color(255,0,0)  
    gameoverFont = pygame.font.SysFont  
('arial.ttf',80)  
    gameoverSurf=  
gameoverFont.render('YOU LOSE!', True,  
redColour)  
    gameoverRect = gameoverSurf.get_rect()  
    gameoverRect.midtop = (500, 250)  
    playSurface.blit(gameoverSurf,  
gameoverRect)  
    pygame.display.flip()  
    time.sleep(5)  
    pygame.quit()
```

9

一点小优化

我们来定义 `Gameover` 函数

定义字体的颜色

定义字体的格式和大小

定义字体出现的位置坐标

`PlaySurface.blit` 是把字体

传入游戏界面。括号中第一个

参数是 `Surface` 对象，第二个

是它的坐标

然后用 `flip` 让字体在游戏结

束的界面上出现

持续时间为五秒

退出游戏

