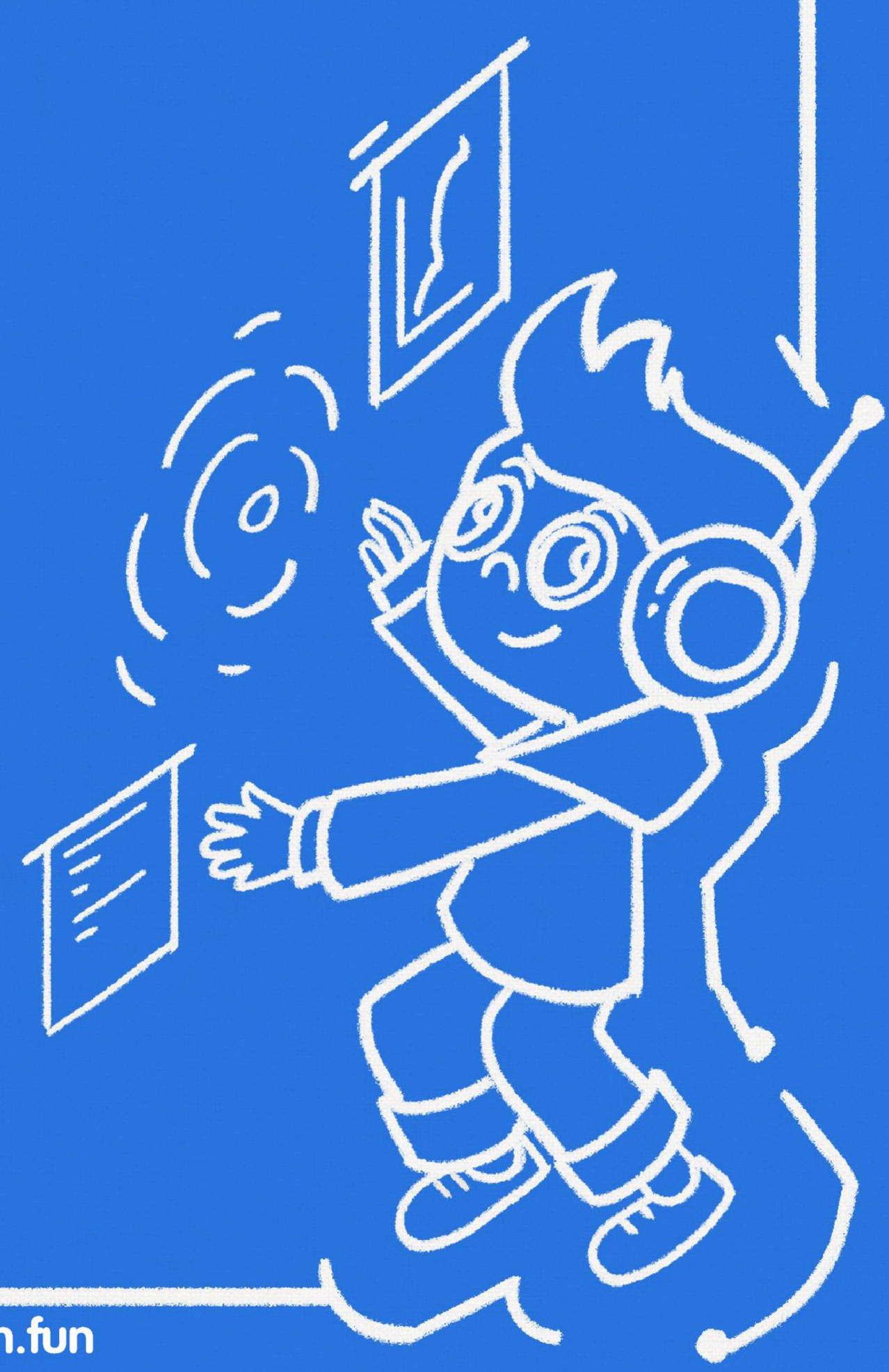
中国地图我来拼(一)



1.探索新知

1.1 基本游戏框架

我们编写程序框架模板，设定一个新窗口

```
# 导入pgzero库
import pgzrun
# 设置窗口
WIDTH = 1000  # 窗口的宽度
HEIGHT = 800  # 窗口的高度
# 设置窗口标题
TITLE = '中国地图我来拼'
# 启动游戏，必须放在最后一行
pgzrun.go()
```

- **pgzrun** 就是 pygame zero run
- 窗口大小设定为 **1000 * 800**

任务一新增代码：

```
# 导入pgzero库
import pgzrun
# 设置窗口
WIDTH = 1000  # 窗口的宽度
HEIGHT = 800  # 窗口的高度
# 设置窗口标题
TITLE = '中国地图我来拼'
# 启动游戏，必须放在最后一行
pgzrun.go()
```

1.2 加载地图碎片

在 pgzero 中，我们要加载地图碎片，也就是要创建出场的【地图碎片】角色

01新疆碎片.png

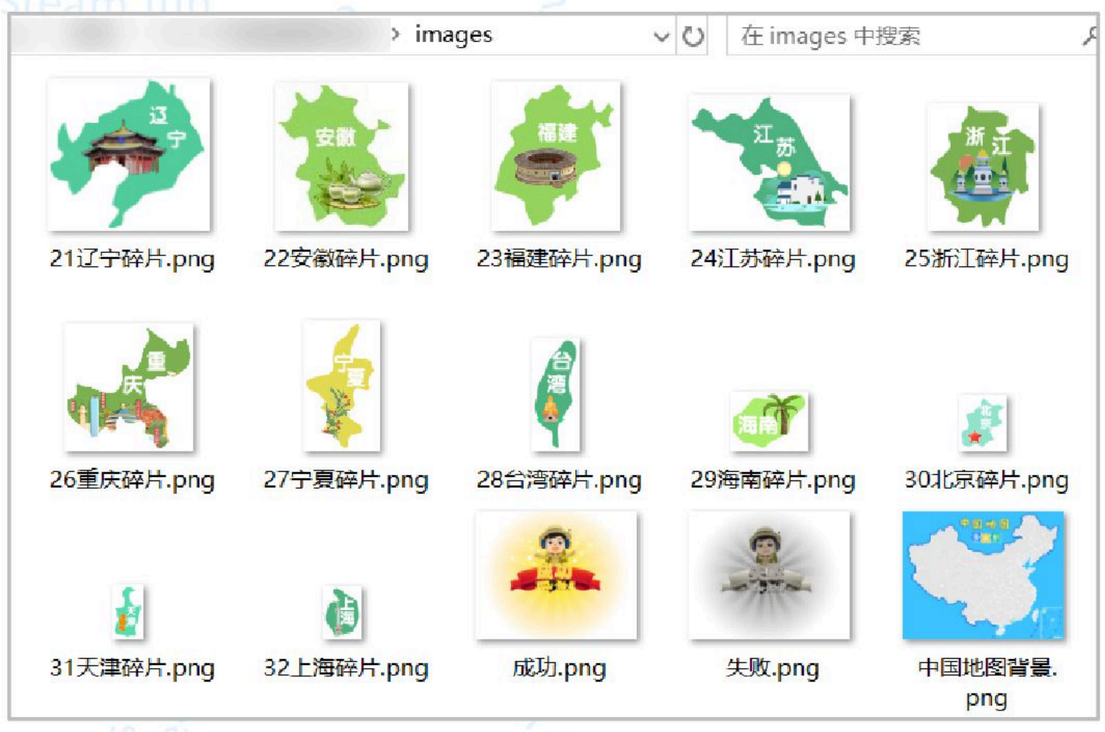
【创建碎片角色】

```
actor = Actor('图片名', (坐标))
```

- 创建角色所调用的图片必须放在项目的 images 文件夹中
- 通过 Actor() 创建的游戏角色，必须使用 draw() 绘制出来，否则不会显示
- 如果 Actor() 函数中不声明坐标，则默认将图片的左上角放在原点 (0,0) 位置

images文件夹里有各种各样的图片，我们遇到了几个困难：

1. 怎么从 images 里找图片？
2. 如何筛选出我们需要的碎片图片？
3. 如何把 .png 后缀前面的‘图片名’提取出来？



1.2 加载地图碎片

【如何从文件夹找图片】

os.listdir(path='.')

这个函数能帮助我们列出一个文件夹里所有的文件和子文件夹，返回一个列表。

```
import os
print(os.listdir('images'))
```

```
['01新疆碎片.png', '02西藏碎片.png', '03内蒙古碎片.png', '04青海碎片.png', '05四川碎片.png', '06黑龙江碎片.png', '07甘肃碎片.png', '08云南碎片.png', '09广西碎片.png', '10湖南碎片.png', '11陕西碎片.png', '12河北碎片.png', '13吉林碎片.png', '14湖北碎片.png', '15广东碎片.png', '16贵州碎片.png', '17河南碎片.png', '18江西碎片.png', '19山东碎片.png', '20山西碎片.png', '21辽宁碎片.png', '22安徽碎片.png', '23福建碎片.png', '24江苏碎片.png', '25浙江碎片.png', '26重庆碎片.png', '27宁夏碎片.png', '28台湾碎片.png', '29海南碎片.png', '30北京碎片.png', '31天津碎片.png', '32上海碎片.png', '中国地图背景.png', '失败.png', '成功.png']
```

拿到文件夹里的所有文件名称后，我们就可以对每一个图片进行操作了

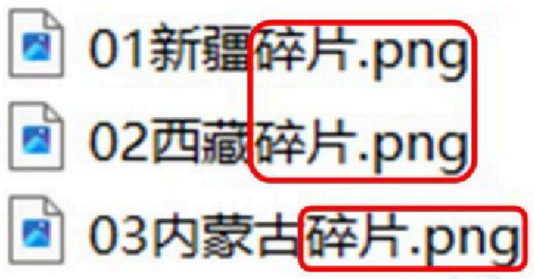
```
import os
for filename in os.listdir('images'):
    # 处理每一个图片
```

接下来解决问题2和问题3：

2. 如何筛选出我们需要的碎片图片？
3. 如何把 .png 后缀前面的‘图片名’提取出来？

【如何筛选】

我们发现地图碎片的文件末尾都为 碎片.png ，切片后判断即可



filename[-6:]

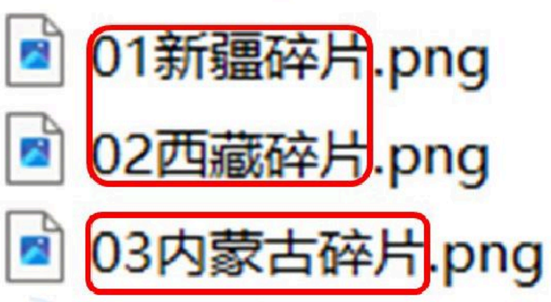
0	3	内	蒙	古	碎	片	.	p	n	g
…	…	…	…	…	-6	-5	-4	-3	-2	-1

1.2 加载地图碎片

```
for filename in os.listdir('images'):
    # 只处理以 碎片.png 结尾的文件
    if filename[-6:] == '碎片.png':
```

【如何提取‘图片名’】

获取地图文件名，只需要把 . 后面的后缀都去掉就可以了



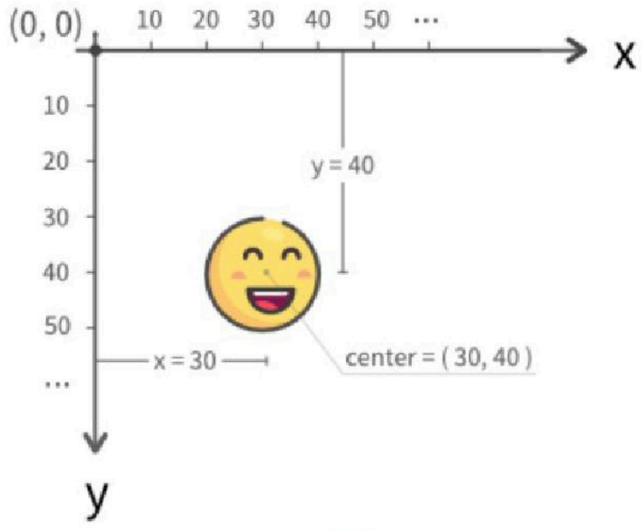
filename[:-4]

0	3	内	蒙	古	碎	片	.	p	n	g
…	…	…	…	…	…	…	-4	-3	-2	-1

```
for filename in os.listdir('images'):
    if filename[-6:] == '碎片.png':
        # 提取文件名
        actor = Actor('filename[:-4]', (坐标))
```

【创建碎片角色】 actor = Actor('filename[:-4]', (坐标))

‘图片名’已经有了，接下来我们分析角色坐标。
游戏开始后，碎片会散落在窗口的随机位置



x坐标：random.randint(0, WIDTH)
y坐标：random.randint(0, HEIGHT)

别忘了导入
random库

1.2 加载地图碎片

【创建碎片角色】

能筛选出图片，有“图片名”，有坐标，现在可以开始创建所有的角色了

```
for filename in os.listdir('images'):
    if filename[-6:] == '碎片.png':
        img = Actor(filename[:-4],(random.randint(0, WIDTH), random.randint(0, HEIGHT)))
```

为了方便后面使用，我们在加载地图碎片时存储一些信息。

碎片角色 —— 后面绘制角色的时候要用到
碎片名称 —— 鼠标释放后，我们要看碎片是否放到了正确位置，就要根据碎片名找到对应的正确坐标
碎片状态 —— 碎片拼好就不能再拖动，拖动碎片前要判断碎片的状态，所以碎片状态也要用到，初始为False，拼好后状态变更为True

准备一个空列表 pieces 用来存储所有的地图碎片和碎片信息

```
pieces = []       # 加载碎片
```

碎片的信息包括:
img - 创建的游戏角色
filename[:-4] - 指文件名
False - 碎片状态，未正确放置为False、正确放置后要更新为True

```
pieces.append([img, filename[:-4], False])
```

1.2 加载地图碎片

任务二新增代码：

```
# 导入pgzero库
import pgzrun
# （新增）导入操作系统和随机库
import os
import random

# 设置窗口
WIDTH = 1000  # 窗口的宽度
HEIGHT = 800  # 窗口的高度

# 设置窗口标题
TITLE = '中国地图我来拼'

# （新增）加载碎片
pieces = []
# （新增）遍历指定文件夹中的所有文件
for filename in os.listdir('images'):
    # 只处理以 碎片.png 结尾的文件
    if filename[-6:] == '碎片.png':
        img = Actor(filename[:-4],(random.randint(0,WIDTH), random.randint(0,HEIGHT)))
        pieces.append([img,filename[:-4],False]) # 添加碎片信息到列表

# 启动游戏
pgzrun.go()
```

1.3 绘制背景、碎片

在 pgzero 中，创建完角色以后，并不能直接显示在屏幕上。我们需要**draw()函数**来刷新屏幕，确保我们把这些角色 Actor 绘制在屏幕上

- 我们不需要将 draw() 放在循环中，pgzero会不断的调用（使用）它
- 每秒 draw 的次数是 60 次

```
def draw():
```

【绘制背景】

screen.blit('image', (left,top))

- 参数 image 是图片的名字
- (left,top)是图片左上角坐标

中国地图背景.png

```
screen.blit('中国地图背景', (0,0))     # 绘制背景
```

【绘制碎片】

首先用 piece 遍历存储所有信息的二维列表 pieces ，每个碎片的信息依次存在列表 piece 中，里面存有三个信息：

- **创建的碎片角色** -
- **文件名**（无后缀）-
- **碎片是否正确放置** -

要绘制碎片角色，首先通过索引，取到【-创建的碎片角色-】，piece[0]

```
# 绘制每个碎片
for piece in pieces:
    piece[0].draw()
```

1.3 绘制背景、碎片

任务三新增代码：

```
    # 遍历指定文件夹中的所有文件
    ...
    #  （新增）刷新屏幕
    def draw():
        screen.blit('中国地图背景',(0,0))      # 绘制背景
        # 绘制每个碎片
        for piece in pieces:
            piece[0].draw()
    # 启动游戏
    pgzrun.go()
```

1.4 拖动地图碎片

拖动地图碎片的操作，其实触发了鼠标的三种状态：

鼠标按下时响应

`on_mouse_down(pos, button)`

- pos 将鼠标的坐标存放在元组中 (x,y)，函数中如果不用pos，可省略

- button 表示鼠标按键，分为鼠标左、中、右键 ，函数没有指定button，默认都可响应

1.4 拖动地图碎片

```
# 鼠标按下事件
def on_mouse_down(pos):
    遍历 pieces 中每个碎片的每个信息 piece, name 和 placed
        如果 鼠标点击未正确放置的碎片：
            将 s_piece 设置为被选中的碎片
            将 s_name 设置为被选中的碎片名称
            跳出循环
```

for piece, name, placed in pieces:

s 是 selected（被选中的） 缩写

按下鼠标后，要确认是否点击到了碎片，而且是没有放到正确位置的碎片

如何确认是否点击了碎片呢？

actor.collidepoint((x, y))

- 检测角色是否与指定点产生碰撞

- 如果碰撞，返回 True ，否则返回 False

碎片 **piece** 是否碰到**鼠标指针所在坐标**，即 **piece**.collidepoint(**pos**)

如何确保被点击的碎片还没有被正确放置？

pieces 列表中存有所有的碎片信息，包括：

- **创建的碎片角色** -
- **文件名（无后缀）** -
- **碎片是否正确放置** -

正确放置值为True，否则为False

1.4 拖动地图碎片

```
# 鼠标按下事件
def on_mouse_down(pos):
    for piece, name, placed in pieces:
        if piece.collidepoint(pos) and not placed:
            将 s_piece 设置为被选中的碎片
            将 s_name 设置为被选中的碎片名称
            跳出循环
```

我们创建全局变量 s_piece、s_name，初始值均设置为 None

```
s_piece = None # 被选中的碎片角色
s_name = None  # 被选中的碎片名字


def on_mouse_down(pos):
    global s_piece, s_name
    for piece, name, placed in pie
        if piece.collidepoint(pos)
            s_piece = piece  # 记录选择的碎片
            s_name = name  # 记录碎片名称
            跳出循环
```

想在函数内部修改这两个变量的值，所以需要使用 global 关键字来声明它们为全局变量

在函数内部修改 s_piece、s_name 的值

鼠标按下，只记录一个被选择的碎片信息，所以记录一个信息后，需要跳出循环。最后使用 break 跳出循环。

1.4 拖动地图碎片

新增代码如下：

```
# （新增）初始值设置
s_piece = None   # 当前被选择的碎片
s_name = None    # 当前被选择的碎片名字

# 加载所有碎片，信息存入 pieces
# 刷新屏幕

# （新增）鼠标按下事件
def on_mouse_down(pos):
    global s_piece, s_name
    for piece, name, placed in pieces:
        # 检查是否点击到未正确放置的碎片
        if piece.collidepoint(pos) and not placed:
            s_piece = piece  # 记录选择的碎片
            s_name = name  # 记录碎片名称
            break

# 启动游戏
pgzrun.go()
```

鼠标移动时响应

on_mouse_move(pos, rel, buttons)

- pos 将鼠标的坐标存放在元组中 (x,y)
- rel 将鼠标的坐标改变的值存放在元组中 (delta_x,delta_y)
- button 表示鼠标按键，分为鼠标左键、中键、右键

本课只用到 pos 参数，其他不用的参数简单了解即可

1.4 拖动地图碎片

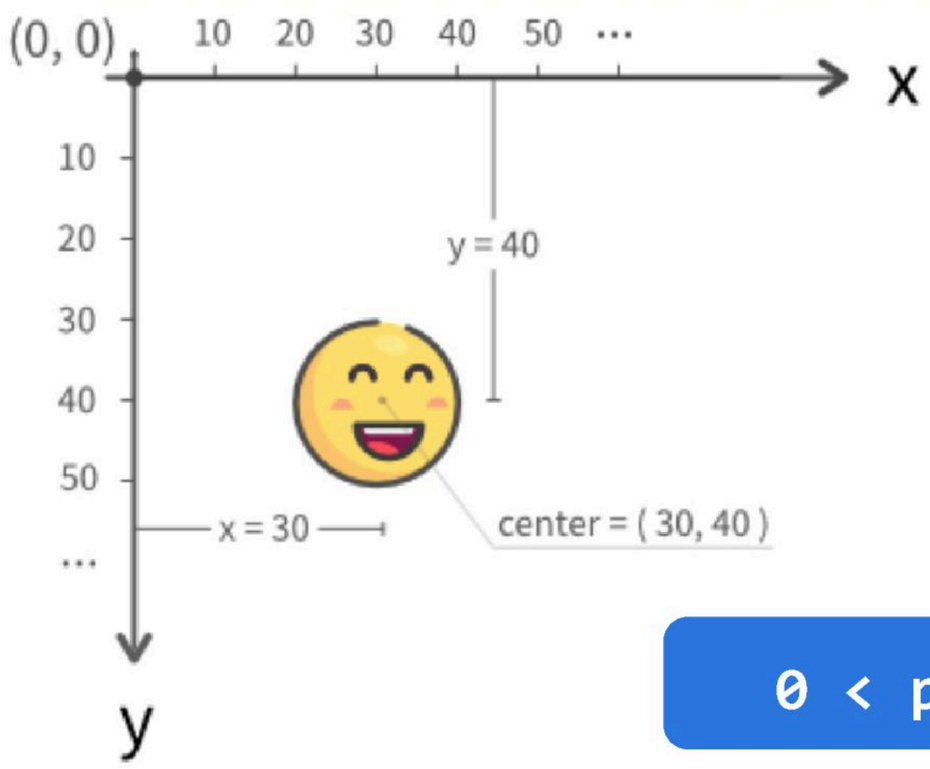
按下鼠标要带着选中的碎片移动，所以我们判断是否选中了碎片（即s_piece != None），则将碎片的pos设置为鼠标指针的pos

```
# 鼠标移动事件
def on_mouse_move(pos):
    if s_piece != None:
        s_piece.pos = pos
```

现在碎片可以跟随鼠标移动，但又发现新的问题：按下鼠标拖动的过程中，碎片会被拖动到窗口之外的区域

这样的话，就会出现找不到碎片的情况。看来我们**需要对鼠标移动的范围加以限制**

pgzero中，以左上角为原点（0，0），可以参考下图：



游戏窗口 WIDTH 宽度为1000，HEIGHT 高度为800。

所以我们可以限制鼠标：

0 < x坐标 < 1000，

0 < y坐标 < 800

```
0 < pos[0] < 1000 and 0 < pos[1] < 800
```

1.4 拖动地图碎片

新增代码如下：

```
# 鼠标按下事件

# （新增）鼠标移动事件
def on_mouse_move(pos):
    # 如果有选中碎片且鼠标未移出窗口，就随着鼠标移动
    if s_piece != None and 0<pos[0]<1000 and 0<pos[1]<800:
        s_piece.pos = pos  # 更新选中碎片的位置

# 启动游戏
pgzrun.go()
```

鼠标按键抬起时响应

```
on_mouse_up(pos, button)
```

- pos 将鼠标的坐标存放在元组中 (x,y)

- button 表示鼠标按键，分为鼠标左键、中键、右键

```
# 鼠标释放事件
def on_mouse_up(pos):
    # <检测碎片是否正确放置>
    pass
```

鼠标释放后，我们需要检测碎片是否放在了正确的位置，这部分具体内容我们下节课继续完成~

完整代码

```
# 预置内容，请勿改动
a = {"01新疆碎片": (225, 231), "02西藏碎片": (244, 457), "03内蒙古碎片": (623, 203),
     "04青海碎片": (383, 396), "05四川碎片": (491, 521), "06黑龙江碎片": (861, 122),
     "07甘肃碎片": (472, 358), "08云南碎片": (457, 621), "09广西碎片": (580, 656),
     "10湖南碎片": (640, 584), "11陕西碎片": (586, 415), "12河北碎片": (718, 330),
     "13吉林碎片": (859, 228), "14湖北碎片": (654, 506), "15广东碎片": (678, 677),
     "16贵州碎片": (549, 593), "17河南碎片": (672, 446), "18江西碎片": (724, 584),
     "19山东碎片": (758, 392), "20山西碎片": (650, 369), "21辽宁碎片": (805, 284),
     "22安徽碎片": (740, 487), "23福建碎片": (766, 609), "24江苏碎片": (770, 462),
     "25浙江碎片": (793, 538), "26重庆碎片": (573, 525), "27宁夏碎片": (548, 375),
     "28台湾碎片": (824, 652), "29海南碎片": (610, 753), "30北京碎片": (715, 310),
     "31天津碎片": (730, 326), "32上海碎片": (818, 495)}    # 省份文件名称和对应的正确位置

# 导入库
import pgzrun
import os
import random
# 设置窗口
WIDTH = 1000  # 窗口的宽度
HEIGHT = 800  # 窗口的高度
# 设置窗口标题
TITLE = '中国地图我来拼'

# 初始值设置
s_piece = None  # 当前被选择的碎片
s_name = None   # 当前被选择的碎片名字

# 加载碎片
pieces = []
# 遍历指定文件夹中的所有文件
for filename in os.listdir('images'):
    if filename[-6:] == '碎片.png':  # 只处理以 碎片.png 结尾的文件
        img = Actor(filename[:-4],(random.randint(0, WIDTH),random.randint(0, HEIGHT)))
        pieces.append([img, filename[:-4], False]) #添加碎片信息到列表

# 刷新屏幕
def draw():
    screen.blit('中国地图背景', (0,0))    # 绘制背景
    for piece in pieces:    # 绘制每个碎片
        piece[0].draw()
```

完整代码

```
# 鼠标按下事件
def on_mouse_down(pos):
    global s_piece, s_name
    for piece, name, placed in pieces:
        # 检查是否点击到未正确放置碎片
        if piece.collidepoint(pos) and not placed:
            s_piece = piece  # 记录选择的碎片
            s_name = name  # 记录碎片名称
            break

# 鼠标移动事件
def on_mouse_move(pos):
    # 如果有选中碎片且鼠标未移出窗口，就随着鼠标移动
    if s_piece != None and 0<pos[0]<1000 and 0<pos[1]<800:
        s_piece.pos = pos  # 更新选中碎片的位置

# 鼠标释放事件
def on_mouse_up(pos):
    # <检测碎片是否正确放置>
    pass

# 启动游戏
pgzrun.go()
```

2.强化练习

1．以下哪个选项是当鼠标按下时触发的事件？（　　）

A. on_mouse_down()

B. on_mouse_up()

C. on_mouse_move()

D. on_mouse()

2．使用下面哪行代码可以导入pgzero库？（　　）

A. import pgzero

B. import pygame

C. import pygame zero

D. import pgzrun

3．想要绘制碎片角色，能使用下面哪种方法？（　　）

A. draw(碎片名)

B. 碎片名.draw()

C. screen.blit(碎片名, (0,0))

D. Actor(碎片名)

2.强化练习

4．字符串变量s中存有小李身份证号码(s＝"331004200608160037"，其中7至14位"20060816"代表出生日期)。若想提取小李身份证中的出生日期，下列Python表达式有错误的是？（　　）

A．s[－12:－4]

B．s[6:14]

C．s[7:15]

D．s[6:－4]

5．在开发一个拼图游戏时，你需要检测玩家是否将一个碎片角色放在了正确位置。对于此检测，你可以使用 碎片角色.collidepoint((x, y)) 方法。请问下面哪个说法正确？（　　）

A．使用碎片的中心点作为（x，y）

B．使用屏幕的中心点作为（x，y）

C．使用正确位置的坐标作为（x，y）

D．使用鼠标的坐标作为（x，y）

3.术语箱

OS	操作系统	mouse	鼠标
actor	角色	piece	碎片
collidepoint	碰撞点	selected	被选中的

4.课后挑战

连连看

鲁　粤　蜀　冀　黔