用python开发一个2D角色游戏的地图Demo

如今很多大学生大学学习了编程语言，想做游戏却迟迟做不出一个游戏雏形来，接下来就和大家谈论下游戏中地图移动的简单原理并用python这一门非常火的语言进行详细的描述。

游戏使用python开发

为什么用python开发，因为python简单易学，对于初学者来说非常适合，而且python集成的库非常地多，我们这里采用python的安装库pygame进行开发。

首先你需要已经安装了python并且已经下载了pygame库

打开你所熟悉的Sublime Text文本编辑器，或者其他的文本编辑器【记事本也可以】创建好一个python开发文件，如下图：

游戏中pygame的框架雏形如下代码：

```python

import pygame,sys

pygame.init()

screen = pygame.display.set\_mode([840,550],0,32)

screen\_caption = pygame.display.set\_caption('game')

pygame.display.flip()

while True:

for event in pygame.event.get():

if event.type == pygame.QUIT:

sys.exit()

pygame.display.update()

1

2

3

4

5

6

7

8

9

10

11

12

代码运行如图：

一个白板啥也没有这就对了，说明游戏搭建成功！

代码解释

解释：

代码中第一行是导入pygame游戏库,其最初的雏形是SDL。

它包含图像、声音，pygame建立在SDL基础上，允许实时电子游戏研发而无需被低级语言（如机器语言和汇编语言）束缚。基于这样一个设想，所有需要的游戏功能和理念都（主要是图像方面）都完全简化为游戏逻辑本身，所有的资源结构都可以由高级语言提供，如Python。

Pygame 原为代替突然停止的 pySDL。

这就是为什么选择python这个库pygame来开发游戏，而不是c语言。

pygame.init()是导入所有pygame方法。

screen = pygame.display.set\_mode([840,550],0,32)

1

#代码是创建屏幕窗口宽840像素，高550像素，不全屏的窗口。

screen\_caption = pygame.display.set\_caption('game')

1

代码是写一个窗口的标题

pygame.display.flip()

1

代码是刷新屏幕，可以意味更新屏幕内容

while True:

for event in pygame.event.get():

if event.type == pygame.QUIT:

sys.exit()

pygame.display.update()

1

2

3

4

5

while True是死循环的意思，在游戏中程序就是一个死循环，当游戏结束时就让程序终结循环即可。这是游戏运行的原理。

for event in pygame.event.get():

if event.type == pygame.QUIT:

sys.exit()

这几行即便是获取游戏中用户输入的状态，if中捕获退出消息的状态，如果是退出窗口，那么就执行sys库中的exit()方法来结束游戏程序。

最后一行的pygame.display.update()也可意味刷新屏幕。

说到这里大家就会问了，pygame.display.flip()和 pygame.display.update()都是刷新，那二者有什么区别吗，其实也没太大的区别，update()是更新部分软件界面显示，而flip()更新整个待显示的Surface对象到屏幕上了，这里我们死循环中用到update()而不是flip()是为了更好的让游戏流畅，节省内存而已。

闲话不多说，我们步入正规

首先导入我的所用的所有库

# -\*- coding:utf-8 -\*-

import pygame,os,wx

from pygame.locals import \*

from sys import exit

from random import randint

pygame.init()

1

2

3

4

5

6

在这里建议下载一个叫做wx的库，这是用来写gui图形界面软件的库，我们用它来获得屏幕的大小好以后开发维护提供遍历条件。

接下来我的一切内容都会写入main方法中，整个main方法框架如下：

...省略了导入的库

def main():

pass

if \_\_name\_\_ == '\_\_main\_\_':

main()

1

2

3

4

5

pass就是空语句，接下来我会在main方法中填入所有的游戏内容。以上省略了导入的库

在此我们需要用到下面的地图资源文件

两个图都是我自己画的，虽然不是很好看但是能用就行

然后角色我是下载的网上的，包括了个方位行走的动作帧图如下：

这张图是不是比我画的不晓得好到哪儿去了，呃…。不管这些了，然后继续上我的代码：

别忘了代码都写在main函数中，下面基本都有注释记得稍微看看哦

app=wx.App()#创建wx.App

WHSIZE=wx.DisplaySize()#获取屏幕大小

WHSIZE=int(WHSIZE[0]\*0.75),int(WHSIZE[1]\*0.85)#变成屏幕宽度75%，变成屏幕高度的85%

Mapxy=950,950,-950,-950 地图宽度从950到-950高度也是一样

ax,ay=0,0#看作是角色的左上角位置

screen=pygame.display.set\_mode([WHSIZE[0],WHSIZE[1]],0,32)#载入屏幕上面讲到过

caption=pygame.display.set\_caption('project1')#屏幕标题

bg=pygame.image.load('image/map/bg1.png').convert\_alpha()#地图壁纸

tree=pygame.image.load('image/map/tree.png').convert\_alpha()#地图中的树

man=pygame.image.load('image/pic/哥布林/145118l0哥布林\_01.gif').convert\_alpha()#载入哥布林角色图片

manxy=man.get\_width(),man.get\_height()#哥布林的尺寸

manxy=WHSIZE[0]/2-manxy[0]/2,WHSIZE[1]/2-manxy[1]/2#让哥布林的位置变成地图的正中央

man\_direction='down' #哥布林的方向默认为向下的方向

man\_walk\_time=0 #哥不林移动的时间

map\_tree\_list=[]#随机树列表

for i in range(500):#随机生成500个树的位置

map\_tree\_list.append([randint(50, 65),randint(50, 65)])

pygame.key.set\_repeat(20)#让按键一直响应【就只一直按着一个键不松开，就会一直响应】

screen.fill([0,0,0])#屏幕初始化填充为黑色

pygame.display.flip()#刷新屏幕

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

\*\*接下来是一个游戏的主体【也就是写在死循环里面】下面和一段是接受按键信息来控制地图拖动和哥布林移动的算法。

#外设事件判断

for event in pygame.event.get():

if event.type==QUIT:#按键判断是否退出游戏

exit()

if event.type==KEYDOWN:

#地图拖动和主角相反

if event.key==K\_UP:

ay+=100

man\_direction='up'

man\_walk\_time+=1

if event.key==K\_DOWN:

ay-=100

man\_direction='down'

man\_walk\_time+=1

if event.key==K\_LEFT:

ax+=100

man\_direction='left'

man\_walk\_time+=1

if event.key==K\_RIGHT:

ax-=100

man\_direction='right'

man\_walk\_time+=1

if event.type==KEYUP:

if event.key==K\_UP:

man\_walk\_time=0

man=pygame.image.load('image/pic/哥布林/145118l0哥布林\_13.gif').convert\_alpha()

if event.key==K\_DOWN:

man\_walk\_time=0

man=pygame.image.load('image/pic/哥布林/145118l0哥布林\_01.gif').convert\_alpha()

if event.key==K\_LEFT:

man\_walk\_time=0

man=pygame.image.load('image/pic/哥布林/145118l0哥布林\_05.gif').convert\_alpha()

if event.key==K\_RIGHT:

man\_walk\_time=0

man=pygame.image.load('image/pic/哥布林/145118l0哥布林\_09.gif').convert\_alpha()

#外设事件判断结束

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

在我设计的地图中宽度是宽高是1900,1900，也就是从坐标（-950，-950）到（950,950），屏幕正好是（0,0）的位置不会变，也就形成了哥布林在整个地图中央的逻辑

如下图：

接着下面的代码是用来绘制surface放到屏幕上：

screen.fill([0,0,0]) #绘制整个游戏背景为黑色

#把事先的绿色地块用来填满整个地图，绘制地图所有方块2000像素的地图

for i in range(int(2000/50)):

for j in range(int(2000/50)):

screen.blit(bg,( (-1000+WHSIZE[0]/2)+(50\*j)+ax,(-1000+WHSIZE[1]/2)+(50\*i)+ay ))

#绘制地图外圈的环境，也就是绘制之前随机的地图树，用四个for循环绘制

for i in range(52):

for j in range(7):

screen.blit( tree ,( -1100+(i\*map\_tree\_list[i\*j][0])+ax, -1100+(j\*map\_tree\_list[i\*j][1])+ay ))

screen.blit( tree ,( -1100+(i\*map\_tree\_list[i\*j][0])+ax, 1300+(j\*map\_tree\_list[i\*j][1])+ay ))

for i in range(10):

for j in range(50):

screen.blit( tree ,( -1100+(i\*map\_tree\_list[i\*j][0])+ax, -1100+(j\*50)+ay ))

screen.blit( tree ,( 1450+(i\*map\_tree\_list[i\*j][0])+ax, -1000+(j\*50)+ay ))

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

上面用四个for循环绘制树的地图，绘制之后会有他的魔力的，我们待会儿见证

我们都知道在2D游戏中，地图中的角色是不是不管怎么走都不会越过地图之内，永远都不会超出地图之外呢？

那么接下来我们需要判断一个碰壁弹回的方法，如下代码

#出界判断

#左，上

if ax>=Mapxy[0]:

ax=Mapxy[0]

if ay>=Mapxy[1]:

ay=Mapxy[1]

#右，下

if ax<=Mapxy[2]:

ax=Mapxy[2]

if ay<=Mapxy[3]:

ay=Mapxy[3]

1

2

3

4

5

6

7

8

9

10

11

ax,ay就是哥布林角色的左上坐标，Mayxy就是地图的四个坐标【左上角，和右下角】（x1,y1,x2,y2）

接着，我们需要绘制哥布林随着方向移动而改变它的帧图

原理如下：

最开始角色的方向是向下的

如上图方向的变量是“down”，如果方向是向上边"up",左边“left”,右边“right”

每一个方向有几个四个动作，整合起来如下

每个动作占时100毫秒，接下来我们会在程序的最后面写入延迟delay(100),当每个动作都设置好，每个动作的每一帧都播放一段时间，然后循环播放就形成了我们人眼中所看到的动画片，这里插播一下： 游戏就是动画片的一种多元化陈述。

如上图就是图片轮询播放从而事先了动画片的效果，是不是很有趣呢，不过在好看的前提下我们的设计师是有很大的作用哦！！！

如下程序就判断绘制了所有方向所有的动作帧

#人物绘制和行走

if man\_direction=='up':

if man\_walk\_time==1:

man=pygame.image.load('image/pic/哥布林/145118l0哥布林\_14.gif').convert\_alpha()

if man\_walk\_time==2:

man=pygame.image.load('image/pic/哥布林/145118l0哥布林\_15.gif').convert\_alpha()

if man\_walk\_time==3:

man=pygame.image.load('image/pic/哥布林/145118l0哥布林\_16.gif').convert\_alpha()

man\_walk\_time=0

if man\_direction=='down':

if man\_walk\_time==1:

man=pygame.image.load('image/pic/哥布林/145118l0哥布林\_02.gif').convert\_alpha()

if man\_walk\_time==2:

man=pygame.image.load('image/pic/哥布林/145118l0哥布林\_03.gif').convert\_alpha()

if man\_walk\_time==3:

man=pygame.image.load('image/pic/哥布林/145118l0哥布林\_04.gif').convert\_alpha()

man\_walk\_time=0

if man\_direction=='left':

if man\_walk\_time==1:

man=pygame.image.load('image/pic/哥布林/145118l0哥布林\_06.gif').convert\_alpha()

if man\_walk\_time==2:

man=pygame.image.load('image/pic/哥布林/145118l0哥布林\_07.gif').convert\_alpha()

if man\_walk\_time==3:

man=pygame.image.load('image/pic/哥布林/145118l0哥布林\_08.gif').convert\_alpha()

man\_walk\_time=0

if man\_direction=='right':

if man\_walk\_time==1:

man=pygame.image.load('image/pic/哥布林/145118l0哥布林\_10.gif').convert\_alpha()

if man\_walk\_time==2:

man=pygame.image.load'(image/pic/哥布林/145118l0哥布林\_11.gif').convert\_alpha()

if man\_walk\_time==3:

man=pygame.image.load('image/pic/哥布林/145118l0哥布林\_12.gif').convert\_alpha()

man\_walk\_time=0

screen.blit(man,(manxy))

#人物绘制和行走结束

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

看如上代码是不是很简单。

最后是程序的结尾刷新和延迟

pygame.display.update() #更新屏幕

pygame.time.delay(100) #100表示延迟100毫秒

1

2

到此整个游戏demo就完成，我么来操作控制台运行游戏如下

在控制台输入：

敲击回车运行游戏如下：

右边的移动和行走如下：

看是不是很有成就感，不过这个游戏有很多地方可以优化的，看看广大朋友们有什么好的算法和方法优化一定要联系我哦，如需要源代码和素材的可以联系，免费提供，开源交流，一步一步看我变强变凸。

————————————————

版权声明：本文为CSDN博主「码来的小朋友」的原创文章，遵循CC 4.0 BY-SA版权协议，转载请附上原文出处链接及本声明。

原文链接：https://blog.csdn.net/qq\_25755645/java/article/details/104355271