

# 开发工具

**Python版本：** 3.6.4

**相关模块：**

pygame模块；

以及一些python自带的模块。

# 环境搭建

安装Python并添加到环境变量，pip安装需要的相关模块即可。

## 原理简介

### 游戏简介：

将图像分为 $m \times n$ 个矩形块，并将图像右下角的矩形块替换为空白块后，将这些矩形块随机摆放成原图像的形状。游戏目标为通过移动非空白块将随机摆放获得的图像恢复成原图像的模样，且规定移动操作仅存在于非空白块移动到空白块。

例如下图所示：



### 逐步实现：

#### Step1：游戏初始界面

既然是游戏，总得有个初始界面吧？

OK，我们先写一个游戏初始界面：

```

126 # 显示游戏开始界面
127 def Show_Start_Interface(Demo, width, height):
128     Demo.fill(Background_Color)
129     tfont = pygame.font.Font('./font/simkai.ttf', width//4)
130     cfont = pygame.font.Font('./font/simkai.ttf', width//20)
131     title = tfont.render('拼图游戏', True, Red)
132     content1 = cfont.render('按H、M、L键开始游戏', True, Blue)
133     content2 = cfont.render('H为高难度, M为中等难度, L为简单', True, Blue)
134     trect = title.get_rect()
135     trect.midtop = (width/2, height/10)
136     crect1 = content1.get_rect()
137     crect1.midtop = (width/2, height/2.2)
138     crect2 = content2.get_rect()
139     crect2.midtop = (width/2, height/1.8)
140     Demo.blit(title, trect)
141     Demo.blit(content1, crect1)
142     Demo.blit(content2, crect2)
143     pygame.display.update()
144     while True:
145         size = None
146         for event in pygame.event.get():
147             if event.type == QUIT:
148                 Stop()
149             if event.type == KEYDOWN:
150                 if event.key == K_ESCAPE:
151                     Stop()
152                 if event.key == ord('l'):
153                     size = 3
154                 elif event.key == ord('m'):
155                     size = 4
156                 elif event.key == ord('h'):
157                     size = 5
158             if size:
159                 break
160     return size

```

效果是这样子的：



根据玩家自身水平，可以选择不同难度的拼图游戏。

### **Step2: 定义移动操作**

定义移动操作的目的是为了移动拼图(好像是废话)，具体实现起来十分简单：

```

42 # 将空白Cell左边的Cell右移到空白Cell位置
43 def moveR(board, blankCell, columns):
44     if blankCell % columns == 0:
45         return blankCell
46     board[blankCell-1], board[blankCell] = board[blankCell], board[blankCell-1]
47     return blankCell-1
48
49
50 # 将空白Cell右边的Cell左移到空白Cell位置
51 def moveL(board, blankCell, columns):
52     if (blankCell+1) % columns == 0:
53         return blankCell
54     board[blankCell+1], board[blankCell] = board[blankCell], board[blankCell+1]
55     return blankCell+1
56
57
58 # 将空白Cell上边的Cell下移到空白Cell位置
59 def MoveD(board, blankCell, columns):
60     if blankCell < columns:
61         return blankCell
62     board[blankCell-columns], board[blankCell] = board[blankCell], board[blankCell-columns]
63     return blankCell-columns
64
65
66 # 将空白Cell下边的Cell上移到空白Cell位置
67 def MoveU(board, blankCell, row, columns):
68     if blankCell >= (row-1) * columns:
69         return blankCell
70     board[blankCell+columns], board[blankCell] = board[blankCell], board[blankCell+columns]
71     return blankCell+columns

```

### Step3: 游戏主界面

OK，有了前面的铺垫，我们可以开始实现我们的游戏主界面了。

首先，我们需要打乱拼图，但是随机打乱很可能导致拼图无解，因此我们通过随机移动拼图来实现打乱拼图的效果，这也是我们先定义拼图的移动操作的主要原因：

```

74 # 获得打乱的拼图
75 def CreateBoard(row, columns, Num_Cell):
76     board = []
77     for i in range(Num_Cell):
78         board.append(i)
79     # 去掉右下角那块
80     blankCell = Num_Cell - 1
81     board[blankCell] = -1
82     for i in range(Num_Random):
83         # 0: left
84         # 1: right
85         # 2: up
86         # 3: down
87         direction = random.randint(0, 3)
88         if direction == 0:
89             blankCell = moveL(board, blankCell, columns)
90         elif direction == 1:
91             blankCell = moveR(board, blankCell, columns)
92         elif direction == 2:
93             blankCell = MoveU(board, blankCell, row, columns)
94         elif direction == 3:
95             blankCell = MoveD(board, blankCell, columns)
96     return board, blankCell

```



游戏主界面初始化：

```
165     # 初始化
166     pygame.init()
167     mainClock = pygame.time.Clock()
168     # 加载图片
169     gameImg = pygame.image.load(GetImagePath(filepath))
170     ImgRect = gameImg.get_rect()
171     # 设置窗口
172     Demo = pygame.display.set_mode((ImgRect.width, ImgRect.height))
173     pygame.display.set_caption('拼图游戏')
174     # 开始界面
175     size = Show_Start_Interface(Demo, ImgRect.width, ImgRect.height)
176     if isinstance(size, int):
177         row, columns = size, size
178         Num_Cell = size * size
179     elif len(size) == 2:
180         row, columns = size[0], size[1]
181         Num_Cell = size[0] * size[1]
182     else:
183         print('[Error]: Parameter Size error...')
184         Stop()
```

最后实现主界面的显示刷新以及事件响应等功能：

```
195     while True:
196         for event in pygame.event.get():
197             if event.type == QUIT:
198                 Stop()
199             if over:
200                 Show_End_Interface(Demo, ImgRect.width, ImgRect.height)
201             # 键盘操作
202             if event.type == KEYDOWN:
203                 if event.key == K_LEFT or event.key == ord('a'):
204                     blankCell = moveL(gameBoard, blankCell, columns)
205                 elif event.key == K_RIGHT or event.key == ord('d'):
206                     blankCell = moveR(gameBoard, blankCell, columns)
207                 elif event.key == K_UP or event.key == ord('w'):
208                     blankCell = MoveU(gameBoard, blankCell, row, columns)
209                 elif event.key == K_DOWN or event.key == ord('s'):
210                     blankCell = MoveD(gameBoard, blankCell, columns)
211             # 鼠标操作
212             if event.type == MOUSEBUTTONDOWN and event.button == 1:
213                 x, y = pygame.mouse.get_pos()
214                 x_pos = x // cellWidth
215                 y_pos = y // cellHeight
216                 idx = x_pos + y_pos * columns
217                 if idx==blankCell-1 or idx==blankCell+1 or idx==blankCell+columns or idx==blankCell-columns:
218                     gameBoard[blankCell], gameBoard[idx] = gameBoard[idx], gameBoard[blankCell]
219                     blankCell = idx
220             if isOver(gameBoard, blankCell, size):
221                 gameBoard[blankCell] = Num_Cell-1
222                 over = True
223             Demo.fill(Background_Color)
224             for i in range(Num_Cell):
225                 if gameBoard[i] == -1:
226                     continue
227                 x_pos = i // columns
228                 y_pos = i % columns
229                 rect = pygame.Rect(y_pos*cellWidth, x_pos*cellHeight, cellWidth, cellHeight)
230                 ImgArea = pygame.Rect((gameBoard[i]*columns)*cellWidth, (gameBoard[i]//columns)*cellHeight, cellWidth, cellHeight)
231                 Demo.blit(gameImg, rect, ImgArea)
232             for i in range(columns+1):
233                 pygame.draw.line(Demo, BLACK, (i*cellWidth, 0), (i*cellWidth, ImgRect.height))
234             for i in range(row+1):
235                 pygame.draw.line(Demo, BLACK, (0, i*cellHeight), (ImgRect.width, i*cellHeight))
236             pygame.display.update()
237             mainClock.tick(FPS)
```

## Step4：游戏结束界面

当玩家完成拼图后，需要显示游戏结束界面，和游戏初始界面类似，实现起来都比较简单：

```
107 # 显示游戏结束界面
108 def Show_End_Interface(Demo, width, height):
109     Demo.fill(Background_Color)
110     font = pygame.font.Font('./font/simkai.ttf', width//8)
111     title = font.render('Finished!', True, (233, 150, 122))
112     rect = title.get_rect()
113     rect.midtop = (width/2, height/2.5)
114     Demo.blit(title, rect)
115     pygame.display.update()
116     pygame.time.wait(500)
117     while True:
118         for event in pygame.event.get():
119             if event.type == QUIT:
120                 Stop()
121             elif event.type == KEYDOWN:
122                 if event.key == K_ESCAPE:
123                     Stop()
```

这样运行就可以了