第 22 章贪吃蛇

贪吃蛇是一款经典的益智类游戏，玩家使用方向键控制蛇改变方向以吃掉食物，同时蛇身 会随着吃掉食物而不断变长。一旦蛇头撞到了窗口的边缘或者身体的其他部位，则判定游戏失 败。本章将带领大家使用pygame开发一款贪吃蛇游戏，让大家对面向对象开发有更好的体会 和理解。同时,再给大家介绍一下pygame的基本使用。

踏。1炒简命

22JJ欄場跟蠶腹盤

贪吃蛇小游戏是一款有趣的益智游戏，它不仅可以锻炼大脑的反应能力，而且可以在游戏 与挑战中娱乐放松自己，如图224所示。

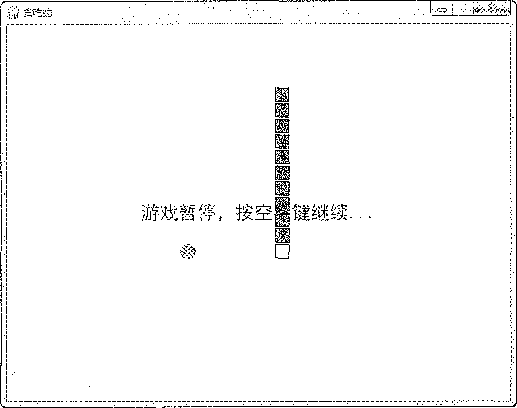
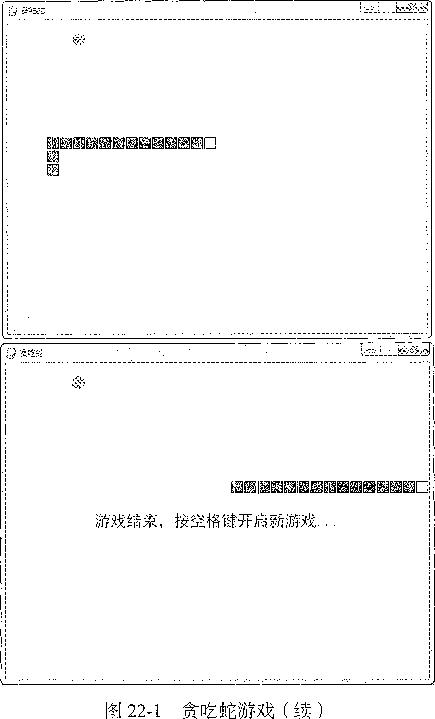


图224贪吃蛇游戏

482 w Pjfthoo篡臆疆鹳 風饕寥Pyth@o



从图22』中可以看出，游戏中设定了两个重要的角色：贪吃蛇和食物。其中，贪吃蛇是 由一组大小固定的方格组成的，蛇头使用边框不填充，其他身体部位使用深灰色填充表示。而 食物使用红色的圆形表示。

食物随机在某个位置出现，玩家通过方向键控制蛇不断地吃掉食物。每吃掉一颗食物，蛇 身就会变长，分数也会增加。此外，玩家在玩游戏的过程中，可以按空格键暂停或继续游戏。

一旦蛇头撞到了外侧的墙壁或者身体的其他部位，就判定游戏失败。要是想重新开始新一 轮游戏，同样按下空格键就行。

22J.2

在开发游戏之前•我们先来明确一下贪吃蛇游戏的游戏规则，具体分为如下3个方面：

（1 ）贪吃蛇初始岀现在游戏窗口的左上角位置，体长共有3节。

（2） 游戏过程中，一旦蛇头撞到了窗口的边缘或者身体的其他部位，游戏结束。

（3） 游戏过程中，点击游戏窗口的关闭按钮，或者按［Esc］键可以直接退出游戏。

（4） 一局游戏结束后，按下空格键可以重新开启一局新游戏。

1. 运动和控制

（1 ）贪吃蛇初始沿屏幕水平方向,向右侧运动，每隔0.5 s移动一节身体。

（2） 使用键盘的方向键（上f』、*in.* [<-i. [->1）可以改变蛇头的运动方向， 不过需要注意：

•当蛇头沿水平方向运动时，只能使用垂直方向的方向键将蛇头的运动方向修改为垂直 方向。

•当蛇头沿垂直方向运动时，只能使用水平方向的方向键将蛇头的运动方向修改为水平 方向。

（3） 游戏过程中按下空格键，可以暂停游戏；再次按下空格键，可以继续游戏。

1. 食物額得分

（1） 游戏开始后，会在游戏窗口的任一随机位置出现食物，食物不会与蛇身体重叠。

（2） 当蛇头与食物相遇时，代表蛇吃到了食物。此时：

•被吃掉的食物从屏幕上消失。

•蛇的身体会增加一节。

-游戏得分增加10分。

•在游戏窗口的其他任一随机位置再次出现新的食物，食物同样不会与蛇身体重叠。

（3） 如果食物出现的30 s内，贪吃蛇没有吃到食物，那么没有被吃到的食物会从屏幕上 消失，之后在游戏窗口的其他任一随机位置再次出现新的食物，食物同样不会与蛇身体重叠。

（4） 每吃掉1颗食物，贪吃蛇的运动速度会加快。初始蛇每隔05 s移动一节身体，它每 吃掉1颗食物，移动的时间间隔缩短50 ms,也就是045 s、0.40 s、035 s等依次类推。不过， 最短的移动时间间隔是0.1 so

22。2 .类和模块的设计

22.2J襲爾谩讲 . " .

根据游戏规则，要实现贪吃蛇游戏，一共需要4种对象，分别是游戏对象、蛇对象、食物 对象和文字标签对象。除游戏对象外歹其他3种对象在游戏窗口中的展现效果如图22以所示。

蛇（Snake）

按空格键继续（Label）

食物

(Food)

得分（Label）

' ""游戏（Game）

因此，我们可以考虑设计4个类，分别承担不同的职责：

（1）游戏类（Game）:负责提供整个游戏流程控制的相关操作，包括:

•创建游戏主窗口和游戏时钟。

®创建并显示游戏元素：蛇、食物、得分标签、提示标签。

•监听用户交互并做出相应处理。

（2）标签类（Label）：负责提供文本提示信息的相关操作。

（3）食物类（Food）：负责提供食物对象的相关操作

（4）蛇类（Snake）：负责提供蛇对象相关操作。

设计后的类结构如图22-3所示。

Game main\_window游戏主窗口 score\_label得分标签 tiplabel提示标忿 is\_game\_over结束标记 is\_pause暂停标记 food食物 snake 贪吃蛇 reset\_gameM®SS~"" 砒art开始游戏

Label

商绘制文本的季矿~  
is\_s（x）冀是否分数文本  
draw 在游戏窗口绘制文本

Food

color食物颜色

score 一颗食物的分数 &晚食物矩形区域 \_

random\_rect随机设置位置 draw在游戏窗口绘制食物

Snake dir运动方向 time\_interval运动时间间隔 score游戏得分 color身体颜色 body list 身体列養

重置蜿属性 add\_node添加一节身体 draw在游液窗口上绘制身体 update移动蛇的身体 change\_dir改变方向 is dead是否死亡 has eat是否吃到食物

图22-3类的设计图

到目前为止，由于还没有介绍如何使用pygamo来开发图形界面游戏，因此有关各个类的 详细设计，本小节暂不做介绍。类图中的属性和方法设计大家先有一个印象即可，在后续章节 中会详细展开。

22.2.2

根据类的设计，我们可以考虑在贪吃蛇游戏项目中设计2个模块，分别是：

° game.py :游戏主模块，封装Game类并负责启动游戏。

° gamejtems.py :游戏元素模块,封装Label. Food、Snake类,并定义全局变量。

2X3 '

22.3.1

使用PyCharm新建一个项目，项目取名为“贪吃蛇”*，*它所用到的解释器版本为Python 3.6。 在该项目中依次建立两个模块:game.py和game\_items.py,创建好的项目结构如图22-4所示。

game^items.py

223.2 pwam® B '

pygame是专门为开发电子游戏而设计的跨平台的Python包，程序员利用pygame提供的 APL可以方便地实现诸如创建图形用户界面、绘制图形或图像、监听用户键盘或鼠标操作、 播放音频等游戏中必需的功能。

要想使用pygame库，需要先将其导入项目中。在game.py文件中输入import pygame, 编辑器没有任何提示,表明此时还没有安装pygame库。PyCharm可以方便地安装和管理第三 方插件，它能够自动连接pypi,获取模块的最新版本并安装。接下来，给大家介绍一下如何在 PyCharm中安装pygame,具体步骤如下:

1.打葬谖置驟面

选择 “File" —> "settings...” —> 4<Project:PlaneWar （项目名称） —> uProject Interpreter", 打开设置界面，可以在右边区域看到已经存在的第三方包列表。如图22・5所示。

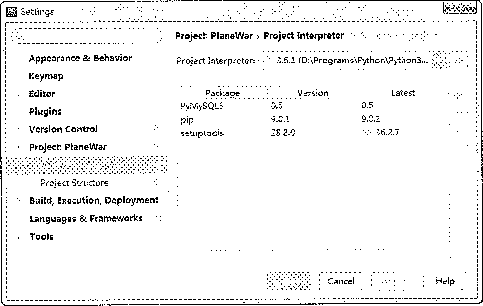
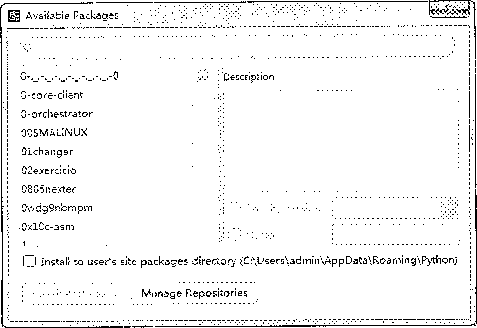


图22-5第三方包列表

2.搜索模块

单击图22・5中包列表右边的“+”按钮，或者双击包列表中的任意一个包条目，进入管理 第三方包的窗口，如图22・6所示。



在图22-6的搜索框中*，*输入pygame搜索这个库*，*会出现该库当前版本和最新版本的信息, 如图22・7所示。

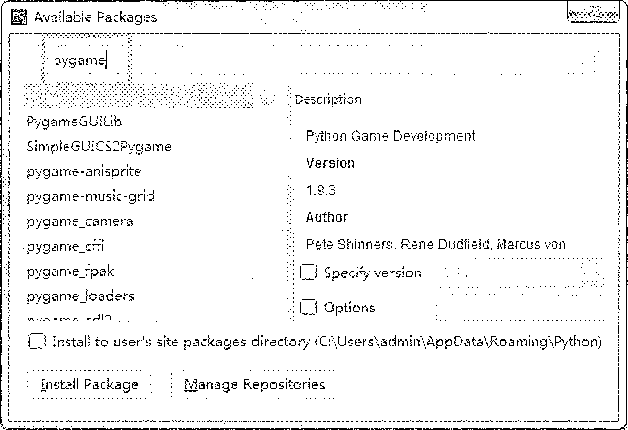


图22-7 搜索pygame包

1. 安装模块

图22-7显示了 pygame的最新版本。这时，单击Install Package按钮开始安装，按钮呈现 不可点击的状态。不到一分钟时间，界面出现包安装成功的信息，如图22・8所示。

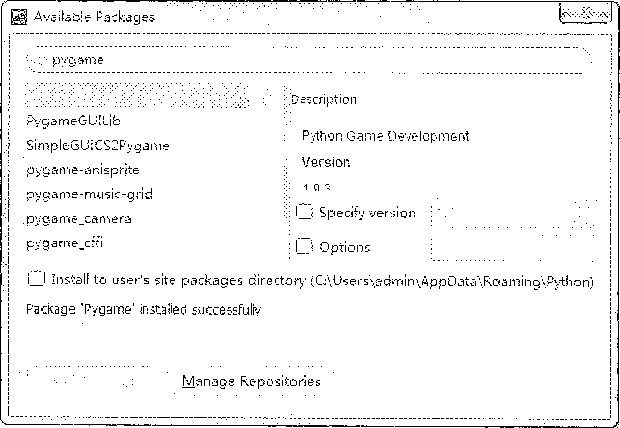


图22-8 pygame包安装成功

冬模煥匾原在列表申 '

关闭图22・8所示的窗口，在第三方包的条目区域显示了刚刚添加的pygame库，如图2A9 所示。

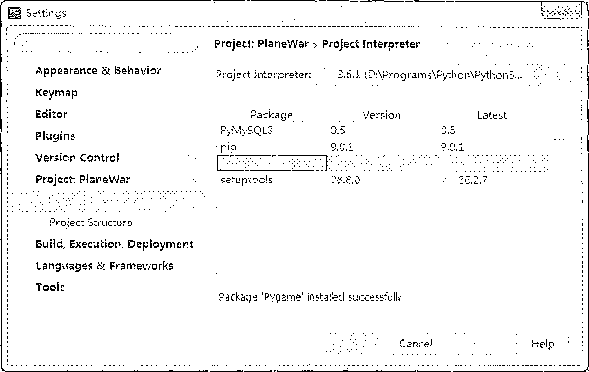


图22-9 pygam©显示在包列表中

单击0K按钮*，*PyCharm底部提示正在更新。更新完成后身在程序中导入pygame库进行测试, 只要编辑器编译通过，就证明成功地导入了库。

22。4 »»»

在这一小节，重点给大家介绍一下使用pygame开发图形界面游戏的几个要素，并且把贪 吃蛇游戏的整体框架搭建完成。

本节的知识重点包括：

（1） pygame的初始化和退岀。

（2） 游戏主窗口。

（3） 游戏循环和游戏时钟。

（4） 主窗口背景颜色。

（5 ）绘制文本。

（6 ） pygame的坐标系。

（7） 游戏事件监听。

（8） 绘制图形。

（9） 定时器事件。

通过对上述内容的学习，相信大家会对图形界面游戏的实现机制以及pygame的使用有个 基本的认识。

22O4J pygame

在之前的小节中已经介绍过，pygmn©是专为开发电子游戏而设计的跨平台的Pythoii包， 针对不同的开发需求它里面提供了不同的模块。例如，显示模块、字体模块、混音器模块等。 在这些模块中，有一部分模块是必须要初始化之后，才能够正常使用的，比如字体模块等。

pygame为了让程序员更加方便地使用包中的模块，提供了如下两个方法：

。init方法：可以一次性初始化pygame的所有模块。这样在后续开发中，程序员可以直接 使用这些模块，而不必再单独为某一个模块调用对应的初始化方法。

• quit方法：可以取消初始化之前已经初始化过的模块。由于Pythoii解释器在退出之前会 释放所有的模块，所以quit方法不是必须要调用的。

要实现pygame的初始化和退出,在game.py模块中实现以下代码:

import pygame # 导入 pygame

from game\_items import \* #导入游戏元素模块中的所有类和全局变量

if name == ? main ':

pygame . init () #初始化所有模块

#游戏代码

pygame . quit () #取消初始化所有模块

值得一提的是，虽然quit方法的调用不是必须的，但是很多程序员在开发程序时，还是 习惯按照谁申请、谁释放的原则来编写代码。因此，大家可以在很多pygame的开源代码中看 到init和quit成对调用的情况。

贪吃蛇游戏是一款图形界面游戏，而所谓的图形界面就是指程序启动时，首先应该呈现一 个图形化的窗口，所有游戏元素(如蛇、食物、得分等)都显示在这个窗口的内部。

pygame的display模块提供的一系列方法可以用于创建游戏窗口以及更新窗口显示内容等 操作，常用的方法如表224所示。

表22-1 display模块的常用方法

|  |  |
| --- | --- |
| 方 法 | 说 明 |
| pygame. display. set\_mode() | 初始化游戏窗口 |
| pygame.display.set\_\_caption() | 设置窗口标题 |
| pygame. display.updateQ | 更新屏幕显示内容 |

关于上述前两个方法的具体介绍如下:

t劎建游戏室窗口方懲 '

使用set\_mod©方法，可以非常方便地创建一个游戏主窗口，语法格式如下：

set\_mode(resolution-(0,0)7 flags=0z depth=0) -> Surface

参数含义如下：

° resolution参数：指定屏幕的宽和高的元组，默认创建的窗口大小和屏幕大小一致。

° fhgs参数 指定屏幕的附加选项，例如，是否全屏等。默认使用resolution指定的窗口大小。 ® depth参数：表示颜色的位数，默认自动匹配。

上述方法会返回Surface对象。可以把一个Suface对象看作是一个油画的画布，可以在这 个画布上作画。例如，绘制贪吃蛇、绘制食物、绘制分数文字等。所有游戏元素绘制在游戏的 主窗口之后,就是用户看到的游戏画面。

1. 设置窗口标题方法

使用set\_caption方法，可以设置窗口标题，语法格式如下:

set\_caption(title, icontitle=None) -> None

参数含义如下：

° title参数：指定显示在标题栏上的窗口标题。

• icontitle参数：在某些操作系统中，可以指定显示在任务栏上的程序标题，默认与fi庞 一致。

创建游戏主窗口并设置窗口的标题。在game.py中定义Game类，并且实现初始化方法， 代码如下。

class Game(object):

"n °游戏类° °

def init (self):

self.main\_window = pygame.display.set\_mode((640 f 480))

pygame . display. set\_caption ( ° 贪吃蛇")

if name == ' main f :

pygame . init () #初始化所有模块

#游戏代码

Game () #创建游戏对象

pygame . quit () #取消初始化所有模块

运行程序，在PyCharm中运行game.py文件进行测试，可以看到一个黑色的游戏窗口一闪 而过。为什么会一闪而过呢？主要是因为程序在启动之后立即结束。

22d4d3躊価漩戏循歸顧瘋赐爾鱒

要做到游戏程序启动执行之后，不会立即退岀，需要在游戏程序中增加一个游戏循环。所 谓游戏循环就是一个无限循环。在Gam©类中增加start方法，并且实现如下代码。

def start(self):

""°开始贪吃蛇游戏° ° "

while True:

pass

在gmue.py的末尾修改代码，调用stairt方法启动游戏，具体代码如下。

#游戏代码

Game () . start () #创建游戏对象并且启动游戏

此时运行程序，游戏窗口会一直显示在屏幕上。

在一款图形界面的游戏中，通常游戏的画面是不断变化的，也就是我们常说的动画。例如, 贪吃蛇的运动、食物的出现与消失以及分数值的变化等。那么，游戏中的动画效果是怎样实现 的呢？跟电影的原理类似，电影是将多张静止的电影胶片连续、快速播放，产生连贯的视觉效果。 游戏中的动画效果，本质上是快速地在屏幕上绘制图像。

一般情况下，在电脑上每秒绘制60次冬就能够达到非常连续、高品质的动画效果，每次 绘制的结果被称为帧。

因此，在我们刚刚完成的代码中，无限循环的执行频率(刷新帧率)只要能达到每秒60帧,

就能够达到我们预期的动画效果了。但是，实际运行中，这个无限循环的执行频率有多快呢？ 我们先调整一下start方法中的代码，运行观察一下循环体的执行频率，代码如下：

i = 0

while True:

print（i）

i += 1

运行程序会发现，循环体的执行频率非常高，远远超过了我们预期的60帧/s,而且CPU 的负荷也比较大。

为了解决上述问题，pygame的time模块中专门提供了一个Clock类，可以非常方便地设 置游戏循环的执行频率（刷新帧率）o以60帧/s举例，当循环体代码执行一遍之后，让程序 休息（1/60 ）s，然后再次执行循环体代码，这样不仅能够让循环体内部的代码每秒只被执行60次， 而且CPU的负荷也能够大大降低。

要设置刷新帧率需要如下两个步骤：

（1 ）在游戏循环的循环体上方创建一个时钟对象。

（2）在游戏循环的循环体的末尾让时钟对象调用tick （帧率）方法来设置刷新帧率。需要 注意的是，tick方法会根据上次被调用的时间，自动设置游戏循环中的延时。

要想设置刷新帧率，可以对start方法的代码调整如下：

def start（self）:

° ° "开始贪吃蛇游戏° ° "

clock = pygame . time . Clock （） # 游戏时钟

i = 0

while True:

print（i）

i += 1

clock. tick （60） # 刷新帧率

再次运行程序，对比一下增加游戏时钟前后的CPU占用情况，如图22J0所示。

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 京蝴舟*邮｝m* .渣.如.  浒押耳福*聞怂更涛海*闌产 |  |  |  |  | 3 X 吁符笋辽  文舛旧必如⑴吨即野  *也權*我能洵痢您肉诺温*榆*捋押 |  |  |  |  | a x |
|  | *2&%* | 58%  懲爵: | 0% | 0%  *做"* |  | 11% | 57%  -.■■■■ | 0% | 0% |  |
| *丄g：* |  |  |  | fl M邮任 | ■-密 ^Charm | *毎"* |  | 0 M盼瓣 | C Mbps |  |
| ； 也£迅加M WiTHioft H濒  \ 铲3鼻，如Ft |  | *"偷*  4.7  *VbE尽* | *播侧网* | 0 Mbps ；  CH負猝； | *\ W轡*卩渺'如⑵  *1* © G；顷任 枷心w村例? | （T电 仍，. | 23.2 羸  4.7 VS | 0雄姆  （?MW *而B淋* | *g"* 球雄/ C冊眇| |  |
| *州mgz驾m淨叫*  *.5F* |  |  | 河  0MW | 0 Mbps  0 Mbpfi | *\ 5如* |  | ‘ 72 MS  *咨術* | *宜*M W | 脇心  0興挡 |  |

图22-10增加游戏时钟前'后对比

pygame的颜色使用的是RGB色彩模式，即通过对红（R）、绿（G）、蓝（B）三个颜色 相互之间的叠加来得到各种各样的颜色。

在pygame中，使用一个（R, G, R）格式的元组来定义一个颜色，其中，红（R）、绿（G）、 蓝（B）的数值是（）〜255之间的一个整数。数值对应颜色的亮级，数值越大，亮级就越大。

常用的颜色定义如下？

(255, 255, 255)

色色色色色色色色 白黑红绿蓝黄橘灰

(0, 0, 0)

(255, 0, 0)

(0, 255, 0)

(0, 0, 255)

(255, 255, 0)

(255, 128, 0)

(128, 128, 128)

首先，在gamejtem.py模块中定义一个表示主窗口背景颜色的全局常量，代码如下:

import pygame

#全局变量定义

BACKGROUND\_COLOR = (232, 232, 232) # 主窗口背景颜色

然后，在game.py模块中导入gamejtem模块中所有的类和全局变量，代码如下: from game\_items import \* #导入游戏元素模块中的所有类和全局变量

修改Game类的start方法，在游戏循环中填充窗口颜色并更新显示，代码如下：

def start(self):

° " °开始贪吃蛇游戏° ° "

clock = pygame . time . Clock () # 游戏时钟

while True:

#依次绘制游戏元素

self .main\_windowofill (BACKGROUND\_COLOR)

#更新显示

pygame.display.update() clock. tick (60) # 刷新帧率

值得一提的是，在填充了主窗口的背景颜色之后，务必要使用pygameJisplay.updateQ方 法更新屏幕的内容，该方法的作用就是将在游戏主窗口中绘制的最终结果一次性显示在屏幕上。

运行程序,在PyCharm中运行game.py文件进行测试,可以看到一个背景色为浅灰色的游

戏窗口，如图22D1所示。

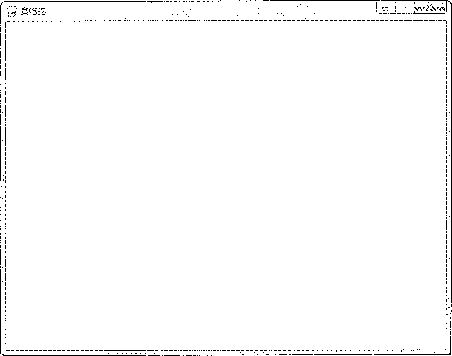


图2241填充窗口的背景色

；☆注意：

Pythoii中没有变量和常量的区别。所谓常量就是定义之后，只允许访问，但是不允许修改。 在Python中，如果要定义常量，可以使用全大写命名，单词之间使用\_分隔。

22a4n5餘割礦翁黨零

pygame的font模块中专门提供了一个SysFont类,可以创建系统字体对象,从而能够实现 在游戏窗口中绘制文字内容。要想在游戏窗口中绘制文本内容，需要执行以下3个步骤：

1. 创建字体对象。
2. 用字体渲染指定的文本内容，并生成一张图像(Surface对象)°
3. 将生成的图像绘制在游戏主窗口的指定位置。

以上三个步骤的示意图如图2242所示。

简体 出隔分：1000]——\*

游戏窗口

图2242绘制文本的步骤示意图

根据上述列岀的三个步骤进行逐一实现，具体如下：

t刨建寥体黝象

SysFont类的初始化方法语法如下:

SysFont(namesf sizer bold=Falsez italic=False) -> Font

参数含义如下;；

® name参数:系统字体的名称。可以设置的字体与操作系统有关,通过pygame.font.get\_ fonts()可以获取当前系统的所有可用字体列表。

® size参数:字体大小。

° bold参数？是否粗体9默认为否。

• italic参数：是否斜体，默认为否。

贪吃蛇游戏中的文字一共有两种颜色，分别是：

®分数文字颜色 浅灰色。

•游戏暂停和结束时的提示文字颜色——深灰色。

为了方便后续代码的编写，首先在gam©\_item<py模块的顶部定义两个字体颜色常量，代码 如下：

SCORE\_TEXT\_COLOR - (192, 192, 192) # 分数文字颜色

TIP\_TEXT\_COLOR = (64, 64, 64) # 提示文字颜色

然后在gamejtem.py中定义Label类，并且实现初始化方法，代码如下:

class Label (object):

""n文字标签类"""

def init (self, size=48is\_score^True):

° ° "初始化方法

:param size : 字体大小

:param is\_score : 是否分数

H H I?

self e font - pygame . font. SysFont ( "simhei" , size) # ，黑体字 self.is\_score = is\_score

2h渲染文本内容

使用创建的字体对象调用rende『方法，渲染生成一张图像(Surface对象)。gidei•方法的 语法如下:

render(text, antialias, color, background=None) "-> Surface

参数含义如下：

° text参数：文字内容。

* antialias 是否抗锯齿，抗锯齿效果会让绘制的文字看起来更加平滑。
* color参数：文字颜色。

° background参数：背景颜色，默认为None。

上述方法会返回一个Surface对象，我们可以将其理解为一张刚好包含文字内容的图像。 Surface对象通过调用get\_rect方法，可以获得图像大小。

1. 绘制渲染结果

pygame为Surface对象提供了一个blit方法，可以在一个Surface对象中绘制另外一个 Surface对象的内容。blit方法的语法如下:

blit (source^ destz area=Nonez special\_flags = 0) -> Rect

参数含义如下：

* source参数：要绘制的图像(Surface对象)。
* (test参数w目标位置。

上述方法会返回Rect对象，用于表示绘制结果对应的矩形区域，关于矩形区域的内容会 在后续小节进行介绍。

在Label类中定义draw*方法，*并接收游戏主窗口作为参数，代码如下:

def draw(selfz window^ text):

EE在窗口中绘制文本内容

:param window: 游戏主窗口 '

:param text:要显示的文本内容

n n n

* 1.使用字体渲染文本内容

color = SCORE\_TEXT\_COLOR if self.is\_score else TIP\_TEXT\_COLOR text\_surface = self.font render(textf True, color)

* 2 .在游戏窗口中绘制渲染结果 window.blit(text\_surfacef (0 7 0))

值得一提的是，在调用blit方法绘制渲染结果时暂时传入的目标位置为(0? 0)。

接下来，对Game类进行一个扩展，在屏幕的左上角位置显示游戏的分数。首先， 在Game类的初始化方法中,定义两个属性:score和scorejabel9代码(加粗部分)如下:

def init (self):

self main\_window = pygame . display. set \_mode ( (64 0f 480)) pygame . display. set\_caption ("贪吃蛇")

self. score = 0 # 游戏得分

self o score\_label = Label () # 得分文本标签

然后，在如rt方法的游戏循环内部，增加代码修改分数并绘制在游戏主窗口，代码(加粗 部分)如下:

while True:

#依次绘制游戏元素

self ..main\_window.fill (BACKGROUND\_COLOR)

#增加游處存分 ~

self o score += 1

self. score\_lab@l o draw (self «main\_windowf "得分：%d" % self» score)

#更新显示“ g

pygame.display。update()

clock. tick (60) # 刷新帧率

运行gamely文件，可以看到游戏窗口的左上角显示了分数标签，且分数的数值一直在不 断变化,如图22-13所示。

*q*畫臨密 i中"專T

图22-13游戏窗口显示分•数文本

如果大家运行修改后的代码，会发现5〜10 s之后，游戏分数就不再更新了。先不要着急， 这个问题在介绍完游戏坐标系之后就讲。

在上一小节中,我们把游戏得分的文字绘制在了屏幕的左上角。如果想要把文字绘制到屏 幕的其他位置，那么应该怎么做呢？要解决这个问题，首先需要给大家介绍一下pygame的坐 标系。

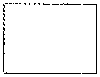
坐标系的用处就是以坐标原点为参照，准确地描述岀屏幕上的任意位置或者区域。 pygame的坐标系定义如下:

•坐标原点在游戏主窗口的左上角。

• X轴沿水平方向向右，逐渐增加。

® Y轴沿垂直方向向下，逐渐增加。

以某个矩形为例，介绍其在坐标系中的位置，如图2244所示。

(0, 0)

640 •

—T—► X

80

y

160

width = 16(3

height = 120

height = 48。

游戏窗口

480

width = 640

图22-14坐标系示意图

现在,我们回顾一下在上一小节Label的draw方法中的最后一句代码:

* 2见在游戏窗口中绘制渲染结果

window.blit(text\_surface7 (0f 0))

上述代码中，调用biit方法时传递的第二个参数为(0, 0)*，*用于指定在游戏窗口的什么位置 来绘制文字。

在游戏中，所有可见的元素都是以矩形区域来描述位置区域的，一个矩形区域包含四个要 素:(x, y) (width, height) o pygame专门提供T一个类Red用于描述矩形区域,Rect的初始化方 法的语法如下:

Rect(x, y, width, height) -> Rect

知道了矩形区域这个概念之后，接下来我们将得分文字绘制在游戏窗口的左下角。要实现 这一目标，一共需要如下3个步骤：

(1 )使用get\_rect方法获得渲染完成的文本图像(Surface对象)的矩形区域。

1. 通过计算设置文本图像的显示位置。
2. 利用Rect类提供的属性设置文本图像的显示位置。

根据上述列出的三个步骤进行逐一实现，具体如下：

牝藐得支本圏像的矩.

来到game items.py模块中,修改Label类的draw方法,代码如下:

* 1.使用字体渲染文本内容

color = SCORE\_TEXT\_COLOR if self.is\_score else TIP\_TEXT\_COLOR

text\_surface = self.font.render(text, True, color)

#获存文本图像的矩形区域

text\_rect = text\_surface.get\_rect() print(text\_rect)

在上述最后一行设置断点，利用断点调试，可以看到文本图像text\_surface的原点为（0,0）, 宽高为（168, 49）,如图22』5所示。

"竺鹭:鑒愛 *笠..*

*We虹蕙莽鑼継囁■* ,1 ■>,奏：篆题电:聲

*W三京辭忑爲;茶辭;;專孫源忑云気墓;忑京;；：;；；忑忑忑忑盘；云；指；提忑=謨技慕;;握京云;;:壬;；遠；忑;；对总;；専萇;云牍搖日忑損：爲三爲云;;上芯憲:感uww京点忐:靛;:云W* 中顷崩 z y明tte

color 二*整\*:\*終* <cls££ 'tuple'>r （192. 192, 192} *"KtSf!"蠢—…賈* j麟*損*希也卵mspwl *盗伊冬AUS,空—*

；'•憊 <module, gamwpy:圭 ■: -：«■ : *；' %燮件竺*=即疽； <佶ct遗芒 16& 塑》

窿灣..如“.即晶.;"；."由广？

*楚dM艺：:普也d* L .：?■..*二*…'

分"辭皂::鶯蠱陽籍賞寳館巒3歩号#©◎；；"\*项:疇说}迁位如丈尊日鼓屁；二""3鼬領话枫n試\*.

图22』5 Debug模式下查看的矩形区域

2.醴过谤算设置支暴圖像位置

要想把文本图像移动到左下角的位置，需要让其X坐标保持不变，Y坐标变大。那么，如 何确定y值的大小呢？我们可以使用如下公式进行计算：

文本矩形的Y值=窗口的高度-文本矩形的高度

修改Label类的draw方法代码如下:

* 1.使用字体渲染文本内容

color = SCORE\_TEXT\_\_COLOR if self»is\_score else TIP\_TEXT\_COLOR text\_surface = self.font.render(text7 Truez color)

#获房文本图像的矩形区域

text\_rect = text\_surface »get\_rect()

#获存主窗口的矩形if域 」

window\_rect = window.get\_rect()

#计算旻本图像的y值 -

text\_rect. y = window\_rect. height - text\_r€$ct. height

* 2 .在游戏窗口中绘制渲染结果

window.blit(text\_surface7 text\_rect)

运行gamely文件，可以发现文本图像已经被绘制在游戏窗口的左下角了，如图22J6所示。

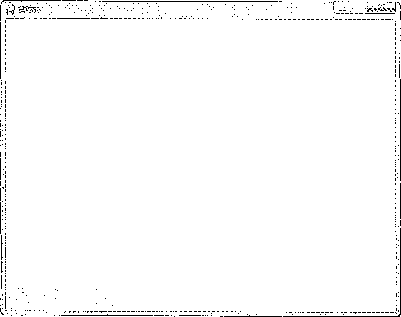


图22』6游戏窗口显示得分

但是，有没有更简单的方法歹不需要计算就能够方便地设置文本图像的位置呢？

3a *m* Rect的属燧谩置显示位置 「 ：

为了方便程序员对矩形区域的操作，pygame的Rect类中提供了一系列方便的属性。 表22-2以矩形区域red = Rect(10, 80, 168, 50)举例说明这些属性。

表22-2矩形区域的常用属性

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | | 属 性 | 说 | 明 |  | 示 例 | |
| 1 | | x、left | 水平方向和Y轴的距离 | | | rect.x =10, rect.left =10 | |
| 2 | | y、top | 垂直方向和X轴的距离 | | | rect.y = 80、rect.top = 80 | |
| 3 | | width、w | 宽度 | | | rect.width = 168、rect.w = 168 | |
| 4 | | height s h | 高度 | | | rect.height = 50、rect.h = 50 | |
| 5 | | right | 右侧=X + W | | | rect.right = 178 | |
| 6 | | bottom | 底部=y + h | | | rect.bottom = 130 | |
| 7 | | size | 尺寸(w, h) | | | rect.size = (168, 50) | |
| 8 | | topleft | (E | | | rect.topleft = (10, 80) | |
| 9 | | bottomleft | (x, bottom) | | | rect.bottomleft = (10, 130) | |
| 10 | | topright | (right, y) | | | rect.topright *= (178, 80)* | |
| 11 | | bottomright | (right, bottom) | | | rect.bottomright = (178, 130) | |
| 12 | | centerx | 中心点x = x + 0.5\* | w |  | rect. centerx = 94 | |
| 13 | | centery | 中心点y = y + 0.5\* | h |  | rect. centery = 105 | |
| 14 | | center | (centerx, centery) | | | rect.center = (94, 105) | |
| 15 | | midtop | (centerx, y) | | | rect.midtop = (94, 80) | |
| 16 | | midleft | (x, centery) | | | rect.midleft = (10, 105) | |
| 17 | | midbottom | (centerx, bottom) | | | rect.midbottom = (94, 130) | |
| 18 | | midright | (right, centery) | | | rect.midright = (178, 105) | |
| 属性示意图如图2247所示。 | | | |  |  |  |  |
| (S,*研* |  |  | 飼必 侬，0? | |  |  | |
|  |  | y |  |  |  | midtop |  |
|  |  | top |  |  | topleft : topriyht | |  |
|  |  |  | battara = *y* h |  |  |  |  |
|  | -4 - | 扁•…4一~一"-一-1 | ■6. : |  |  |  |  |
|  |  |  |  |  | i5i Id I eft … | 0 I ■ - ■ - r ight |  |
|  |  | 1 5.U-4? |  |  |  | center 1 |  |
|  |  |  |  |  |  | 1 centery) j |  |
|  |  |  |  |  |  | *.：；； t：* ‘a. |  |
|  |  | W : |  |  | bottowleft bottGRiright | |  |
|  | ■■■■< - | --•…right = x 申 v/ - - |  |  |  |  |  |
|  |  | 游戏德口 | |  |  | 游戒滋口 |  |
| i |  |  |  |  |  |  |  |

图22J7 属性示意图

介绍了 Rect的属性之后,现在我们使用属性来修改一下Label类的draw方法,大家可以

对比一下修改前后的代码，修改后的代码如下:

# 1.使用字体渲染文本内容 .

color = SCORE\_TEXT\_COLOR if self.is\_score else TIP\_TEXT\_COLOR text\_surface = self.font.render(textz True7 color)

#获存文本图像的矩形区域

[text\_rect = text\_surface.get\_rect() #获存主窗口的矩形if域 —](#bookmark2695" \o "Current Document)

window\_rect = window.get\_rect()

#设置企置 —

text\_rect »bottomleft = window\_rect obottomleft

# 2.在游戏窗口中绘制渲染结果

window.blit(text\_surfacef text\_rect)

再次运行game.py文件，可以发现效果跟图22D6所示一样。

宏2 損 3

通过之前小节的学习，我们已经知道，游戏循环的主要目的是保证游戏不会立即退出。在 这一小节，我们要重点学习一下，在游戏循环中需要做的最后一件事情，那就是事件监听。首先, 我们来明确两个概念M

-事件：游戏启动后，用户针对游戏所做的操作。例如罗单击“关闭”按钮、单击鼠标、 按下键盘等。

«监听辭在游戏循环中，判断用户在当前这一时刻所做的具体操作。只有监听到用户具体 的操作，才能有针对性的做出响应。

明确了事件监听的概念后，我们来看一下游戏循环工作示意图，如图22』8所示。

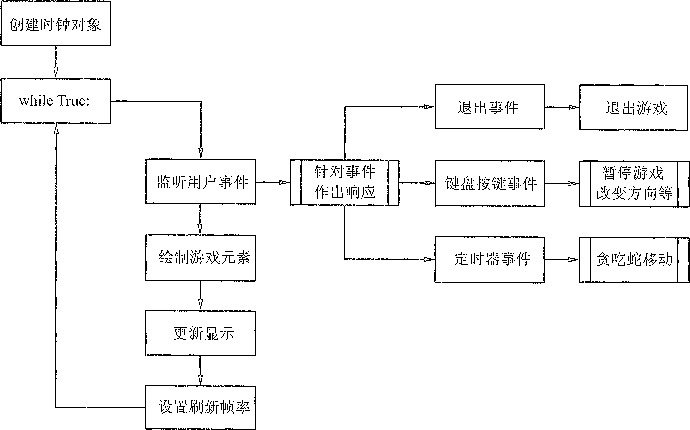


图2248游戏循环工作示意图

通过图2248可以很直观地看到：

•只有监听到退出事件，才能实现用户单击“关闭”按钮时终止游戏。

•只有监听到键盘按键事件，才能在用户按下方向键后，改变贪吃蛇的运动方向，才能在 用户按下空格键后，暂停、继续或者开启新一轮游戏；

•只有监听到定时器事件，才能每隔05 s移动一节身体。

pygame专门提供了一个evenf模块用于处理游戏事件。同一时刻可能会发生很多事件，例 如，在用户按下方向键修改贪吃蛇方向的同时，定时器事件被触发了，此时还应该让贪吃蛇向 前移动。因此，可以通过调用pygame.event.getO方法获得当前这一时刻发生的所有事件列表。

首先，我们实现对退出事件的监听和处理。找到Game类的如rt方法，对代码(加粗部分) 调整如下：

def start(self):

"5开始贪吃蛇游戏n h »

clock = pygame o time » Clock()

#游戏时钟

while True:

#事件监听

for event in pygame„©vent.get() : # 遍历同一时刻发生的事件列表

if event. type == pygame QUIT: # 判断退出事件

return

#依次绘制游戏元素

self 8main\_window.fill (BACKGROUND\_COLOR)

省略其苑代码—

紧接着，我们再来监听一下用户按键事件，按下［Esc］键时同样直接退出游戏。在遍历 事件的for循环中，增加对按键事件的判断，代码(加粗部分)如下：

#事件监听 for event in pygame.event„get(): if event. type == pygame .. QUIT :

遍历同一时刻发生的事件列表 判断退出事件

return

elif event. type == pygame. KEYDOW: if event. key == *pjgaiae*.. K\_ESCAPE :

判断按键事件

return

根据贪吃蛇游戏规则的描述，空格键可以改变游戏的状态，可分为如下两种情况：

1. 一局游戏结束后，按下空格键可以重新开启一局新游戏。
2. 游戏过程中按下空格键，可以暂停游戏；再次按下空格键？可以继续游戏。

要实现游戏状态的切换，我们分两步来完成:

1. 创建提示标签及游戏状态标记，游戏状态不同显示不同的提示信息。
2. 监听空格键按键，并根据当前游戏状态标记，修改游戏状态。

根据上述列出的步骤进行逐一实现，具体如下：

1。削建提示标签和状态标记

修改Game类的初始化方法，创建提示标签和游戏状态标记，代码(加粗部分)如下言

def init (self):

self.main\_window = pygame odisplay o set\_mode((64 0f 480))

pygame . display. set\_caption (" 贪吃蛇 ")

self . score = 0 # 游戏得分

self . score label = Label () # 得分文本标签

, 'self. tip\_label = Label (24, False) # 提示标签

se 1 f«is\_game\_over = False # 游成1结束标记

self. isjause = False # 游戏暂停标记

修改游戏循环中的代码，在游戏暂停和游戏结束时，分别显示不同的提示信息。同时，只

有在游戏处在既不暂停又不结束的状态时，才修改游戏得分，代码(加粗部分)如下：

•…省略N行

#依次绘制游戏元素

sel f <, main\_\_window. fil 1 (BACKGR0UND\_\_C0L0R)

#判断游殲柬态 一

if self. is\_game\_over:

self. tip label. draw (self»main\_windowf研游戏结束，按空格锭开房鼎j•游成“。"八\* ?里) elif self o is\_pause:

self. tip^label edraw(self omain\_window,"游戒暂停，’按空格键继续

else:

#增加游双得分 self\score += 1

self. score\_label. draw (self .main\_window, "得分:%d" % self . score)

#更新显示

pygame.display.update()

。…省略N行.…

修改Label的draw方法，将提示标签的文字显示在游戏窗口的中间位置，代码(加粗部分) 如下：

def draw(selfwindow^ text):

.…省略N行。…

#获得主窗口的矩形区域

window\_rect = window. get\_\_rect ()

* 置 —

if self .is\_scoxe: #分数文字显示在左下角

text\_rect o bottomleft = window\_rect 0 bottomleft else:繼3算他文字显示在中间 5

text\_rect。center = window\_rect.center

* 2 •在游戒窗口中绘制渲染结果

window.blit(text\_surface, text\_rect)

如果游戏已经结束，按下空格键需要先对游戏复位，包括重设游戏状态、重设游戏得分、 重设贪吃蛇、重设食，物等;如果游戏没有结束，按下空格键，只需要切换selfisjause的状态即可。

为了方便后续的游戏复位,我们先在Game类的初始化方法的下方,定义一个resct\_ game方法9由这个方法负责后续的游戏复位的工作，代码如下：

def reset\_ga.me (self):

° ° °游戏复位"° °

self. is\_game\_over = False # 游戏结束标记

self. is\_pause = False # 游戏暂停标记

self. score = 0 # 游戏得分

然后在游戏循环的事件监听循环中，增加对空格键的监听及相应处理，代码(加粗部分) 如下：

while True:

#事件监听

for event in pygame . event. get () : # 遍历同一时刻发生的事件列表

if event. type == pygame . QUIT : # 判断退出事件

return

elif event. type == pygame . KEYDOWN : # 判断按键事件

if event o key == pygame.K\_ESCAPE:

return

elif event o key == pygame o K^SPACE:

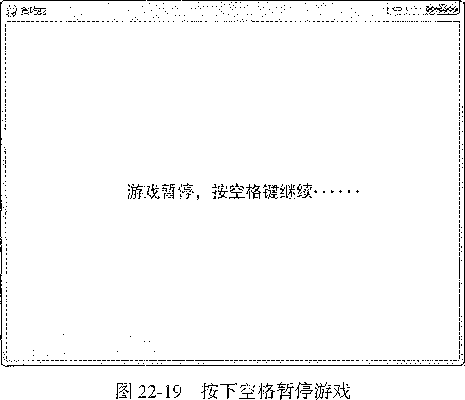
if se 1 f o is\_game\_over:

self»reset\_game()

else:

self.is\_pause = not self.is\_pause

运行gamely文件，弹出游戏的窗口，分数的值一直在不断增加。这时，按下空格键，可 以在窗口中央看到图2249所示的提示文本，且分数停止变化。再次按下空格键，分数继续增加。



22d4d8饑剳嵐物圖膨

pygmne的dmw模块中专门提供了一系列方法，可以方便地绘制图形。常用的绘制图形方 法如表22-3所示。

表22-3 draw模块的常用方法

|  |  |  |
| --- | --- | --- |
| 序号 | 语 法 | . . 说明 |
| 1 | rect(Surface, color, Rect, width=0) -> Rect | 绘制矩形，width>0则绘制边框 |
| 2 | ellipse(Surface, color, Rect, width=0) -> Rect | 绘制与Rect内切的椭圆，width同上 |
| 3 | line(Surface, color, start\_pos, end\_pos, width=l) -> Rect | 绘制从start\_pos至!j endjpos的线条 |
| 4 | lines(Surface, color, closed, pointlist, width=l) -> Rect | 绘制多个线条，closed设置首尾相连 |

游戏中的食物使用圆形表示，为此，我们就利用&aw模块提供的ellipse方法实现食物的

绘制工作。要实现这一目标，一共需要如下3个步骤：

1. 在屏幕的左上角绘制一个圆形表示食物。
2. 让食物随机出现。
3. 增加食物出场动画。

根据上述列出的步骤进行逐一实现，具体如下。

t简擘绘制彦物

首先，在game.\_item.py模块中定义屏幕矩形区域和小格子大小的全局常量，之所以要定义 全局常量有三个原因：

1. 定义屏幕矩形区域是为了方便后续计算食物的位置以及蛇身体的位置。
2. 在贪吃蛇游戏中，我们可以把屏幕矩形区域划分成若干个小格子。
3. 每一个小格子中，可以显示一个食物或者一节蛇的身体。

定义全局常量的代码如下：

#全局变量定义

SCREEN\_RECT = pygame Rect (0, 0\ 640, 480) # 游戏窗口矩形区域

CELL\_SIZE = 2 0 # 小格子大小

在game\_item.py模块的末尾,定义Food类,并实现如下代码:

class Food(object):

H H H食类2 H队n

def init (self):

self . color = (255〃 0z 0) # 食物颜色-红色

self score = 10 #每颗食物得分10分

self . rect = pygame . Rect (0, 0, CELL\_SIZEz CELL\_\_SIZE) # 食物位置 def draw(selff window):

pygame.draw.ellipse(window^ self.colorr self o rect)

注意，这里暂时设定食物的初始位置为(0,0)。

在Game类初始化方法的末尾，创建食物对象，代码如下：

self . food = Food () # 食物

修改start方法，在游戏循环中让食物对象调用draw方法，代码(加粗部分)如下：

。…省略N行“…

self <, score\_label „ draw (self »main\_window7 "彳异分：% self. score)

self o food»draw (self offlain^window)

#更新显示

pygame.display.update()

运行game.py文件，可以看到，游戏窗口的左上角显示了一个红色圆点。

1. 隨桃食物位置.

在game\_item.py模块的顶部，导入random模块，以方便使用随机数，代码如下：

import random

在Food类中定义random\_rect方法，随机确定游戏窗口的任一小格子设置食物出现的位置， 代码如下：

def random\_rect(self):

col = SCREEN\_RECTeW / CELL\_\_SIZE - 1 # 屏幕上小格子的歹4数

row = SCREEN\_RECT.h / CELL\_SIZE - 1 # 屏幕上小格子的行数

x = random. randint (0 f col) \* CELL\_\_SIZE

y = random o randint(0 z row) 大 CELL\_SIZE self.rect = pygame.Rect(x^ yz CELL\_SIZEf CELL\_SIZE)

在Food类初始化方法的末尾，设置食物随机出现的位置，代码如下？

self.random\_rect()

#设置食物随机位置

修改Game类的reset\_game方法，在游戏复位时，重新设置食物的位置，改后的代码如下:

def reset\_game(self):

° ° "游戏复位"° °

self.is\_game\_over = False

游戏结束标记

游戏暂停标记 游戏得分

重新设置食物位置

self.is\_pause = False

self » score = 0

self.food.random\_rect()

此时，每运行一次程序，窗口中食物显示的位置都是不同的。

1. 食物岀现动画

在实际开发中，我们只需要让连续帧绘制的结果相差不大，这样就能够在视觉上产生连续 的动画效果了，如图22・20所示。



图22-20食物出现动画示意图

pygame在Rect类中提供了两个方法inflate和inflate\_ip,其可以非常方便地实现食物由小 变大的动画效果，语法如下：

inflate (xz y) -> Rect

inflate\_ip (x7 y) ™> None

inflMe和inflate\_ip方法都可以以矩形区域的中心点为中心，向四周扩大或缩小。其中， x表示水平方向缩放的像素，正数放大，负数缩小；y表示垂直方向缩放的像素，正数放大， 负数缩小。切记，x和y必须要是偶数。例如，传入x的值为-2,表示左边减少一个像素，右 边同样减少1个像素。

上述两个方法的区别在于，inflate方法会返回一个新的Rect对象，不会修改调用Rect对 象的属性；而inflate\_ip方法不返回新的Rect对象，会修改调用Rect对象的属性。

现在，我们就利用inflate方法来实现食物在出现时，由小变大的动画效果。首先，在Food 类的random\_rect方法末尾，设置食物大小的初始矩形区域，增加的代码如下：

#食物初始不可见

self.rect。inflate\_ip(-CELL\_SIZE, -CELL\_SIZE)

然后，修改Food类的draw方法。在绘制食物前，判断一下食物的矩形区域宽度，如果还 没有和小格子宽度一致，则向四周放大，代码如下：

def draw(self, window):

#判断宽度是否达到小格子宽度

if self.rect.w < CELL\_SIZE:

self . rect. inflate\_ip *{2 r* 2) #向四周各自放大1个像素

pygame.draw.ellipse(window, self» colorf self.rect)

再次运行gamely文件，可以看到食物以由小变大的动画显示到窗口的某个位置，如 图22-21所示。



按照贪吃蛇的游戏规则描述，如果食物出现的30 s内，贪吃蛇没有吃到食物，那么没有被 吃到的食物从屏幕上消失，并且在游戏窗口的其他任一随机位置出现新的食物。现在，如果暂 时不考虑贪吃蛇吃食物的情况，要实现游戏规则的需求，我们只需要每隔30 s让食物对象调用 一下random\_rect方法即可。

像这种每隔一段时间重复执行一个固定动作的场景,我们可以使用定时器事件来实现。 pygame的time模块中提供的set\_timer方法，就是专门用来定义定时器事件的。

要想在游戏中添加定时器事件，一共需要如下3个步骤：

1. 定义定时器事件代号常量。

(2 )使用set\_timer方法设置定时器事件。

1. 在游戏循环中监听定时器事件。

首先，在gamejtem.py模块中定义定时器事件常量，代码如下：

FOOD\_UPDATE\_EVENT = pygame s USEREVENT # 更新食物事件

上述事件代号是一个整数，初始值可以使用pygameJJSEOVENTo如果要定义多个事件 代号，可以从pygarne.USEREVENT开始顺序递增。

然后，在Food类random\_rect方法的末尾设置定时器事件，代码如下：

pygame . time . set\_timer (FOOD\_UPDATE\_EVENTA 30000) # 设置更新食物事件

setjimer^的时间间隔参数以毫秒(ms)为单位。

将设置定时器事件的代码放在raiidom\_rect方法中，可以做到只要设置了食物的位置，都 会重新开始计时。这样在后续开发中，无论是重新开始游戏还是食物被吃掉，都可以不必再考 虑定时设置食物位置这个问题了。

在游戏循环中，增加更新食物事件监听，代码(加粗部分)如下？

while True: #事件监听 for event in pygame . event. get () : #遍历同一时刻发生的事件列表

if event. type == pygame . QUIT : # 判断退出事件

return 省略N行

#仅在游礙状态才会更新食物的位置 if not self。is\_\_pause and not self。is\_\_game\_over: if event, type == FOOD\_UPDATE\_EVENT: # 判斷更新食物爭件

self. food” xand.om\_\_zect ()

#依次绘制游戏元素 ~

self .main\_windowofill (BACKGROUND\_COLOR) *…*省略N行…。

运行game.py文件，游戏窗口的某个位置显示了食物。等待30 s后，可以看到，窗口中显 示的食物消失，之后在新的位置出现一颗食物。

1 22。5蠢唸蟾勒離实瑰

在这一小节，我们以已经完成的游戏框架为基础，加入贪吃蛇的功能，完成整个游戏的开发。 本节重点要实现的功能如下所示。

1. 贪吃蛇类的设计。
2. 定义贪吃蛇类。
3. 添加蛇身体。
4. 绘制和移动身体。
5. 改变贪吃蛇的方向。

(6 )贪吃蛇吃食物。

(7)贪吃蛇的死亡。

本节所使用的知识点，基本都是之前已经学习过的。通过上述内容的学习，相信大家 对pygmie的使用会有一个更好的理解和体会，尤其对游戏的事件监听机制会有一个更好的 认识。

22.5J

我们首先来明确一下在贪吃蛇类中应该封装哪些属性和方法，这些内容的确定，会为后续 的开发做好准备和铺垫。按照贪吃蛇的游戏规则描述须贪吃蛇类的设计具体如下：

1L属燧设it

贪吃蛇类包含的属性如下：

（1） 运动方向属性

要描述贪吃蛇的运动方向，我们可以使用pygame预置的方向键常量，分别如下：

° pygame.K\_UP :上。

° pygame .K\_DWON :下。

® pygame.K\_LEFT : 左。

° pygame K\_RIGHT :右。

在后续监听键盘按键事件以及修改蛇的运动方向时，还会使用到这些常量。

（2） 时间间隔属性

在初始状态时，贪吃蛇是每隔0.5 s移动一节身体的，而随着吃到的食物增加，这个时间 间隔会不断缩小。因此，我们还需要考虑使用一个属性来记录这个时间间隔。

（3） 身体列表属性

如果把贪吃蛇的每节身体看作屏幕上的一个小格子，则可以使用一个Rect对象来表示贪 吃蛇的一节身体。这样，要记录贪吃蛇的整.个身体，只需要定义一个列表属性body\_list9用来 记录从蛇头到蛇尾的每节身体的R©ct对象即可。

（4） 游戏得分属性

每一轮新游戏开始之后，游戏中只有一条贪吃蛇，并且只有在这条蛇吃到食物之后，游戏 分数才会增加。一旦贪吃蛇在运动的过程中，蛇头撞到了窗口的边缘或者身体的其他部位，则 判定为死亡。而再次开启新一轮游戏后，游戏分数会归零重新计算。

因此，我们可以定义一个游戏分数的属性，用来记录当前这条贪吃蛇对象的累计得分。

2。方法Wt

贪吃蛇类包含的方法如下：

（1）重置贪食蛇属性

按照游戏规则描述，在每一轮游戏结束后，用户按空格键会开启新一轮游戏。这就意味着, 开启新一轮游戏需要把贪吃蛇对象的属性设置为初始值。

因此，可以定义一个reset\_\_sanke方法，专门负责来重设贪吃蛇的属性值。

（2 ）添加身体

除了在初始化贪吃蛇时，要在水平向右的方向（蛇的初始运动方向）上增加三节身体之外, 在游戏过程中,如果贪吃蛇再吃到食物，身体长度同样也要增加。

因此，可以定义一个add\_node方法，专门负责向贪吃蛇的身体列表中增加一节身体。

（3）绘制身体和更新位置

为了方便游戏循环中的代码调用，可以定义两个方法来负责绘制身体和更新位置，分别如下：

* draw方法：负责在游戏窗口中绘制贪吃蛇的身体。
* update方法:负责移动蛇的身体。

|  |  |
| --- | --- |
| 1. 改变蛇的方向   为了方便游戏循环中的代码调用，可以定义一个change\_dir 方法，专门负责根据用户按键改变蛇的运动方向。   1. 判断死亡和吃到食物   为了方便游戏循环中的代码调用，可以定义两个方法来负责 判断蛇的死亡和蛇是否吃到食物，分别如下：  • is\_dead方法言返回是否死亡。  ° has\_\_eat方法：返回是否吃到指定的食物。  综上所述，我们可以得到贪吃蛇的类图，如图22・22所示。 | Snake |
| dir运动方向 time\_interval 运动时间间隔 score 游戏得分 color 身体颜色 bodylist 身体列表 |
| reset\_snake 重置蛇属性  add\_node 添加一节身体  draw在游戏窗口上绘制身体 update 移动蛇的身体 change\_dir 改变方向  isdead 是否死亡  has eat 是否吃到食物 |
| 22.5.2竈莫赞IS盤鑿 | 图22-22贪吃蛇类图 |

在这一小节,我们先做一些准备工作,具体包括:

1. 定义贪吃蛇类，并准备reset\_\_snake方法。
2. 使用贪吃蛇的分数属性调整游戏中的得分显示。

首先，在game item.py中的最下方定义Sanke*类,*并且实现初始化方法禾口 reset snake方法, 代码如下：

class Snake(object): n ° n贪吃蛇类" n def init (self):

| self.dir = pygame «K\_RIGHT | #初始向右运动 |
| --- | --- |
| self o score = 0 | #初始得分 |
| self.time\_interval = 500 | #运动间隔时间 |
| self.color = (64’ 64 f 64) | #身体颜色-深灰色 |
| self.body\_list =[] | #身体列表 |
| self.reset\_snake() |  |
| def reset\_\_snake (self): |  |
| n "重置蛇属性° H |  |
| self o dir = pygame . ,K\_\_RIGHT |  |
| self.score = 0 |  |
| self.time\_interval = *500* |  |
| self.body\_list.clear() | #清空身体列表 |
| *for* i in range(3): | #添加三节身体 |
| self.add\_node() |  |

def add\_node(self):

n"°在蛇的运动方向上,增加一节身体"°

pass

需要注意的是，类的所有实例属性都应该定义在\_init\_方法内部。

按照之前小节的分析，由每条蛇负责来记录自己的游戏得分。因此，我们不需要再 在Game类中记录得分了。接下来，我们就调整一下Game类的代码，包括：

•在初始化方法中删除游戏得分属性，并且新建蛇对象。

®在游戏复位方法中，让贪吃蛇对象调用reset\_snake方法重置蛇属性。

调整后的初始化方法和reset\_game方法如下:

def init (self):

self.main\_\_window = pygame.display.set\_mode((640r 480)) pygame . display. set\_caption ("贪吃虫它 °)

|  |  |
| --- | --- |
| self o score\_label = Label() | #得分文本标签 |
| self.tip\_label = Label*(24f* | False） #提示标签 |
| self.is\_game\_over = False | #游戏结束标记 |
| self. is\_\_pause = False | #游戏暂停标记 |
| self o food = Food() | #食物 |
| self» snake = Snake() | #贪吃蛇 |
| def reset\_game(self): |  |
| "n w游戏复位"n ° |  |
| self.is\_game\_over = False | #游戏结束标记 |
| self.is\_pause = False | #游戏暂停标记 |
| self.food.random\_rect() | #重新设置食物位置 |
| self.snake.reset\_snake() | #设置蛇属性 |

然后，调整Game类中游戏循环部分的代码，把蛇对象的游戏得分绘制在游戏窗口中，代 码（加粗部分）如下:

def start(self):

开始贪吃蛇游戏» n »»

clock = pygame . time . Clock () # 游戏时钟

while True:

•…省略N行

#依次绘制游戏元素

self. main\_window. fill (BACKGR0UND\_C0L0R)

#判断游戏萩态 —

if self » is\_game\_\_over:

self o tip\_label。draw(self omain\_window^

°游戏结束，按空格键开启新游戏...°)

elif self.is\_pause:

self. tip\_label. draw (self .main\_windowz "游戏暂停，按空格键继续.…”) else :

#增加游戏得分

self.snake•score += 1

self。score\_label。draw(self omain windowz

° 得分:%d.n % self .snake, score)

self.food»draw(self.main\_window)

…“省略N行*… ~*

22.5.3

按照之前的分析，我们知道在游戏中添加一节身体共分为两种情况：

（1） 初始化贪吃蛇时，在屏幕的左上角沿水平向右方向，依次添加3节身体。

（2） 在游戏过程中，贪吃蛇吃到食物后同样需要增加1节身体。

接下来，通过一张示意图来分析一下贪吃蛇在吃到食物后，增加身体的处理方式，如 图22-23所示。

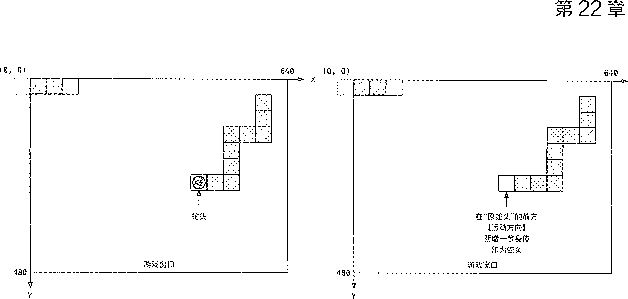


图22.23添加蛇头示意图

观察图22以3不难发现，当贪吃蛇吃到食物时，我们只需要在蛇头的前方位置，也就是沿 着蛇当前的运动方向偏移一个小格子，再增加一节身体作为蛇头就可以了。

这种处理方式最大的好处就是，增加身体和向前移动可以一步到位，而不需要先在蛇尾添 加一节身体，再向前依次移动每一节蛇的身体。接下来，我们再来看一下添加蛇身体的流程图， 从而进一步确认一下添加蛇身体的处理方式，如图22罗4所示。

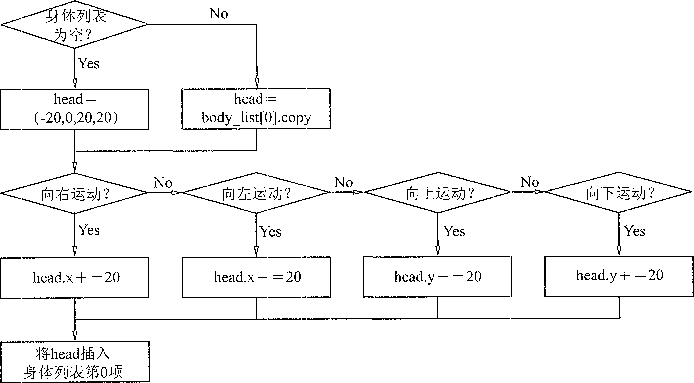


图22-24添加蛇身体流程图

接下来，我们就按照图22・24所描述的流程来实现一下Snake类中的add\_node方法，代码 如下：

def add\_\_node (self):

° H °在蛇的运动方向上,增加一节身体° ° °

* 1.判断是否有身体 if self »body\_list:

head = self.body\_list[0].copy()

else:

head = pygame.Rect(-CELL\_\_SIZE, 0, CELL\_SIZE, CELL\_SIZE)

* 2 .才艮据运动方向，调整head的位置

if self.dir == pygame.K\_RIGHT:

head.x += CELL\_SIZE

|  |  |
| --- | --- |
| elif self.dir  head.x 一= elif self.dir  head.y -= elif self o dir | == pygame . K\_\_LEFT : CELL\_SIZE  ==pygame.K\_UP: CELL\_SIZE  == pygame.K\_DOWN: |

head.y += CELL\_SIZE

# 3。将蛇头插入身体列裒第0项

self.body\_list.insert(0z head)

修改G為me类的初始化方法，在方法的末尾测试输出贪吃蛇的身体列表，代码如下： print(self.snake.body\_list)

运行程序，可以看到如下输出结果：

[<rect(40, 0r 20z 20)>z <rect (20, 0f 2020)>z <rect (0f 0, 20z 20)>] 这个结果与我们预期的屏幕左上角连续的三个矩形区域完全一致。

22.5.4

在这一小节,我们要完成如下两个任务:

1. 利用draw模块提供的方法绘制蛇的身体。
2. 利用定时器事件实现蛇的移动。

在Snake类中定义draw方法,并实现如下代码:

def draw(selfr window):

#遍历绘制每一节身体

for idxz rect in enumerate(self.body\_list): pygaime . draw. rect (window^

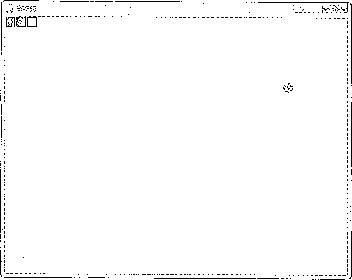
self.colory rect. inflate (-2 z -2) , # 缩小矩形区域 idx == 0) #蛇头绘制边框不填充

在Game类的游戏循环中，让蛇对象调用draw方法，代码(加粗部分)如下：

self score\_l.abel» draw (self .main\_window7 "得分：%d" % self . snake 8 score) self » food.draw(self »main window)

self.snake.draw(self " main^window)

此时，运行gamely文件，可以看到窗口的左上角显示了贪吃蛇，如图22・25所示。



经过之前的学习，我们已经知道，要想在pygame中每隔一段时间重复执行一个固定动作， 例如，蛇每一节身体沿当前运动方向偏移一个小格子，则可以使用定时器事件来实现。

首先，我们先来准备一下定时器事件发生后，需要让蛇对象执行的方法。在Sn&k©类中定 X update方法,并实现以下代码:

def update(self):

self .add\_node () #沿着运动方向在蛇头位置增加一节身体

self .body\_list.pop () # 删除蛇尾

值得一提的是，蛇向前移动本质上就是蛇头沿当前运动方向增加一节，之后删除蛇尾，而 身体的其他位置保持不变。

然后，在gamejtem.py模块顶部定义定时器事件常量，代码如下:

SNAKE：\_UPDATE\_EVENT = pygame . USEREVENT + 1 # 贪吃蛇移动事件

事件代码彼此是不能重复的。

然后，在Snake类中add\_node方法的末尾，设置定时器事件，代码如下《

# 4.设置贪吃蛇移动定时器

pygame.time.set\_timer(SNAKE\_UPDATE\_EVENT, self.time\_interval)

意： 将设置定时器事件的代码放在add\_node方法中，可以做到只要添加了一节身体， 都会重新开始计时。这样做的好处是，能够非常方便地实现一个目的，那就是吃的食物越多， 蛇的速度越快。

最后，在游戏循环中，增加贪吃蛇移动事件的监听，代码(加粗部分)如下：

#事件监听

for event in pygame . event. get () : #遍历同一时刻发生的事件列表

if event. type == pygame . QUIT : # 判断退出事件

return

elif event. type == pygame . KEYDOWN: # 判断按键事件

if event o key == pygame e K\_ESCAPE:

return

elif event» key == pygame.K\_SPACE:

if self.is\_game\_over:

self。reset\_game()

else :

self.is\_pause = not self.is\_pause

#仅在游戏状态才会更新食物和移动蛇的位置

if not self o is\_pause and not self.is\_game\_over:

if event .type == FOOD\_UPDATE\_EVENT : # 判断更新食物事件

self.food.random\_rect()

if event. type == SNAKE\_UPDATE\_EVENT: # 判断贪吃蛇移动事件

self e snake.update() ； . : 「 .、

…省略N行代码•…

运行game.py文件，可以看到，贪吃蛇一直以一定的速率，沿着窗口向右运动，如 图22-26所示。



图22-26贪吃蛇移动

22.5.5

在这一小节中，我们要完成如下两个任务：

(1 )在Snake类中定义change\_dir方法,修改贪吃蛇的运动方向。

(2)在游戏循环的事件监听方法中，监听用户按键并且修改贪吃蛇的运动方向。

按照贪吃蛇的游戏规则描述，使用键盘的方向键(/?』、tn. [<-i > [-1)可 以改变蛇头的运动方向，不过需要注意：

•当蛇头沿水平方向运动时，只能使用垂直方向的方向键将蛇头的运动方向修改为垂直 方向°

•当蛇头沿垂直方向运动时，只能使用水平方向的方向键将蛇头的运动方向修改为水平 方向。

在Snake类中定义change\_dir方法，代码如下:

def change\_dir(selfF to\_dir):

E5改变贪吃蛇的运动方髙

:param to\_dir :要变化的方向

H H H

hor\_dirs = (pygame . K\_RIGHT, pygame . K\_LEFT) # 水平方向 ver\_dirs = (pygame . K\_UP, pygame . K\_DOWN) # 垂直方向

#判断当前运动方向及要木參改的方向

if ((self.dir in hor\_dirs and to\_dir not in hor\_dirs) *or* (self.dir in ver\_dirs and to\_dir not in ver\_dirs)): self o dir = to\_dir

在Game类的游戏循环的事件监听循环中，判断用户是否按下方向键，如果是，则让贪吃 蛇对象调用change\_dir方法修改运动方向,代码(加粗部分)如下:

#仅在游戏状态才会更新食物、移动蛇的位置以及改变蛇的运动方向

if not self o is\_pause and not self . is\_game\_\_over :

if event, type == FOOD\_UPDATE\_EVENT : # 判断更新食物事件

self。food.random\_rect()

if event .type == SNAKE\_UPDATE\_\_EVENT : # 判断贪吃蛇移动事件

self.snake.update()

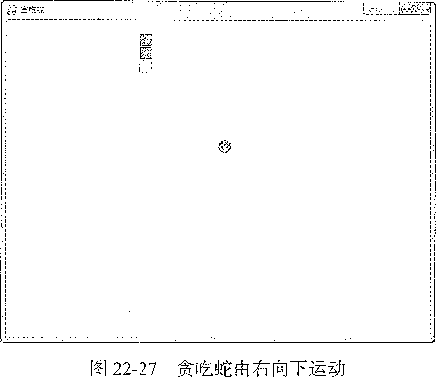
if event o type == pygame . CTYDOWN: # 判断接链事件

#判断是否方向键

if event„key in (pygame„K\_LEFT pygame.K\_RIGHT z pygame. K\_UP pygame»K\_DOW):

self.snake.change\_dir(event。key)

运行game.py文件,可以看到,贪吃蛇一直在向右移动。按［| ］键,贪吃蛇的运动方向 发生偏转，开始向下移动，如图22・27所示。再按［f ］键9贪吃蛇的运动方向不会发生任何 改变。



22ra5D6盖腹盤腹鶯魏 •

按照贪吃蛇的游戏规则描述，当蛇头与食物相遇时，代表蛇吃到了食物。此时，蛇的身 体会增加1节，游戏得分增加10分"贪吃蛇的运动速度会缩短50 ms (最短间隔为(Ms), 被吃掉的食物从屏幕上消失，在游戏窗口的其他任一随机位置(不会与蛇身体重叠)出现新的 食物。

在这一节中，我们需要完成如下两个任务:

(1 )在Snake类中定义has\_eat方法。如果蛇吃到食物,则在方法内部修改游戏得分和运 动时间间隔，然后增加一节身体，最后返回Tow；如果没有吃到，返回False。

(2)在游戏循环中，增加贪吃蛇吃食物的方法调用。如果蛇吃到食物，则让食物对象调 用random\_rect方法，将食物设置到一个新的随机位置。

pygame的Rect类中提供了一个contains方法，可以判断一个矩形区域是否包含另一个矩 形区域,语法如下:

contains(Rect) -> bool

我们知道，蛇的每一节身体和食物的宽高都是(20,20),而要判断蛇头与食物是否相遇， 只需要让蛇头调用contains方法即可。

在Snake类中定义has eat方法,代码如下:

def has\_eat(selffood):

Ef判断蛇头是否与食物相遇-吃到食物

:param food:食物对象

:return :: 是否吃至U食物

H H队

if self.,body\_list [ 0 ]。contains (food. rect): self o score += food. score # 增加分数 #修改运动时间间隔

if self o time\_interval > 100: self »time\_interval -= 50 self . add\_node () # 增加一节身体

return True

return False

在Gam©类的游戏循环中，在游戏状态下判断贪吃蛇是否吃到食物。如果是，则让食物对 象调用random\_rect方法，将食物设置到一个新的随机位置，代码(加粗部分)如下？

#依次绘制游戏元素

self.main\_window. fill(BACKGROUND\_COLOR)

#判断游戒萩态 ~~

if self o is\_\_game\_over : self . tip\_\_label. draw (self .main\_windowz ° 游戏结束,按空格键开启新游戏...") elif self.is\_pause:

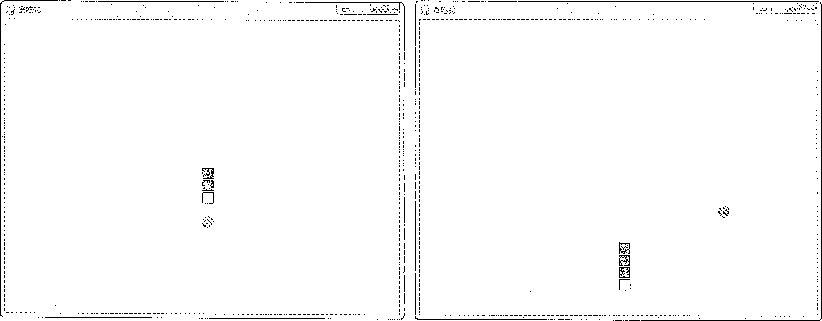
self . tip\_label. draw (self .main\_windowz ° 游戏暂停,按空格键继续 …。”) else :

#判断是否吃到食物

if self•snake.has^eat(self。food):

self.food.random\_rect()

self . score\_]\_abel. draw (self .main\_windowz ° 得分:%d" % self . snake . score) 运行game.py文件，可以看到，窗口中显示的游戏得分为饥 通过使用键盘的方向键让贪 吃蛇吃掉这颗食物后，贪吃蛇的身体增加一节，游戏得分增加1。分，其他位置又出现了新食物, 如图22-28所示。



22.5.7 寰應鹽酶遞直

按照贪吃蛇的游戏规则描述，一旦蛇头撞到了窗口的边缘或者身体的其他部位，则判定为

死亡。同时，为了给用户一个更好的体验,

在贪吃蛇死亡之后，游戏主窗口中应该显示贪吃蛇

临死前的最后一个画面，如图22・29所示。

抡头移密了游戏観口

QgSp蛇头物権疆

*g埃魂，*按空*W我榆".*

游競紡靈，按空櫓纜并庖鏡游戏…，

图 22-29

贪吃蛇死亡示意图

从图22.29中可以看到，贪吃蛇死亡之后，要么蛇头移出了屏幕，要么蛇头与身体重叠， 给用户的视觉体验非常不好。因此，我们要保证蛇头包含在窗口范围内，或不与身体其他位置 重合。

在这一小节，既要判断贪吃蛇的死亡，又要记录临终前的画面，我们要完成如下3个任务: (1 )在Snake类中定义is\_(fead方法。在该方法中进行判断，如果蛇头撞到了窗口的边缘 或者身体的其他部位金则返回True,否则返回Falseo

1. 修改已经完成的update方法。.一旦发现移动身体之后，如果贪吃蛇死亡，则恢复之 前的身体数据，并且返回False,表示无法移动蛇的身体。
2. 在游戏循环中，增加判断蛇死亡的代码处理。

要判断蛇是否死亡，仍然可以使用pygame中Rect类提供的contains方法，可分为如下两 种情况：

•判断窗口的矩形区域是否包含蛇头的矩形区域，如果不包含，说明蛇头已经移出屏幕。

•判断蛇头的矩形区域是否包含身体其他部分的矩形区域，如果包含，说明蛇头与其他部 分发生重叠，也就是说蛇头撞上自己的身体。

在Snake类中定义is\_dead方法，代码如下:

def is\_dead(self):

° ""判断贪吃蛇是否死亡

:return :死亡返回True,否则返回False

* 1.记录蛇头的矩形区域

head = self.body\_list[0]

* 2 .判断蛇头是否移出屏幕

if not SCREEN\_RECT.contains(head):

return True

# 3。判断是否与身体其他部分重叠 for body in self <» body\_list [ 1:]: if headocontains(body):

return True

return False

按照之前的分析，在移动贪吃蛇的身体时，一旦发现贪吃蛇死亡,需要恢复之前的身体数据, 从而可以保证在绘制游戏窗口时，用户能够看到贪吃蛇临终前的画面。针对这一需求，我们对 已经完成的update方法进行调整，如下所示。

def update(self):

° ° °移动贪吃蛇的整个身体

一旦发现贪吃蛇移动后会死亡，则恢复整个身体数据

:return :移动成功返回True,贪吃蛇死亡表示不能移动,返回False

# 1.备份身体列表

body\_li s t\_copy = self.body\_list.copy()

* 2.移动身体 self.add\_node() self.body\_list.pop()

#沿着运动方向在蛇头位置增加一节身体 #删除蛇尾

恢复备份的身体

* 3。判断是否死亡，如果是， if self.is\_dead():

self.body\_list = body\_list\_copy return False

return True

在Game类游戏循环的事件监听循环中，修改一下贪吃蛇移动事件发生后处理的代码，使 用is\_game\_over属性来记录一下本轮游戏是否已经结束，代码(加粗部分)如下：

省略N行•…

#仅在游戏状态才会更新食物、移动蛇的位置以及改变蛇的运动方向

if not self o is\_pause and not self o is\_game\_\_over :

if event „ type '= FOOD\_UPDATE\_EVENT : # 判断更新食物事件

self o food.random\_rect()

if event. type == SNAKE\_UPDATE\_EVENT : # 判断贪吃蛇移动事件

self o is\_game\_over = not selfosnakeoupdate()

if event „ type == pygame . KEYDOWN: # 判断按键事件

#判断是否方向键

if event.key in (pygame.K\_LEFTA pygame.K\_RIGHTZ

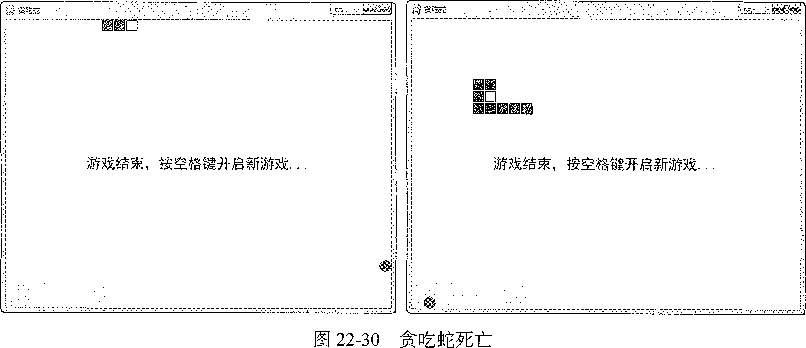
pygame.K\_UP, pygame.K\_DOWN):

self.snake.change\_dir(event•key)

….省略N行

运行gamely文件，可以看到弹出了游戏的窗口。按[f ] 窗口的中央显示了游戏结

束的提示信息；按下空格键，游戏恢复到初始状态；使用方向键控制贪吃蛇身体变长，蛇头碰 到身体其他位置后同样显示了游戏结束时显示的信息。具体如图22・30所示。



:縁书项K俸会

贪吃蛇项目至此已经全部完成。首先，我们根据游戏的规则进行了类的设计和模块准备， 然后结合pyganw库的使用逐步实现了游戏的功能。通过贪吃蛇项目的开发，达到了如下两个 目的w

1. 让大家会使用Python和pygame实现图形界面的开发。pygame是Pythoii下的一个开 源游戏库，可免费使用，它不仅继承了 Python特有的语言优势，诸如语法简单、API操作简单、 面向对象编程等，而且增加了更多游戏开发的API,使得2D游戏开发的进度大大加快。针对 2D游戏开发来讲，pygMiw在这个方面表现得比其他语言更加优秀，而这正是选择pygame进 行2D游戏开发最有利的技术支持。随着越来越多优秀的程序员加入pygame的开发，将会使 pygame的完善性越来越好。
2. 让大家学会对游戏项目占用的CPU进行合理优化。界面是一款游戏吸引玩家的第一 个门户，美观的界面渲染是吸引用户的制胜法宝。因此，一款游戏需要设计游戏元素用到的图 片素材。大家知道，游戏启动后，会将其所用到的素材全部加载到内存中，要是不对内存进行 管理，势必会造成游戏有卡顿或闪退的现象，用户的体验非常不好。在开发本项目的过程中9 虽然没有用到图片素材，但是使用时钟来降低了游戏的刷新频率，从而减少了 CPU的负荷， 大家一定要格外注意。