

# Pygame最小开发框架

---



嵩 天  
北京理工大学





# Pygame最小开发框架





# Pygame简介与安装

pip install 

Projects ▾

News

About

Getting Started

Docs

Info ▾

Development ▾

## About — wiki

Edit

Source

History

Recent Changes

Table of Contents

[More silly nonsense about Pygame](#)

Hello internet traveller,

welcome to our humble (*and slightly strange*) little part of the **World Wide Web**. Let me give you a quick introduction about what you've stumbled upon here.

pygame (**the library**) is a Free and Open Source [python](#) programming language library for making multimedia applications like games built on top of the excellent [SDL library](#). Like SDL, pygame is highly portable and runs on nearly every platform and operating system. Millions of people have downloaded pygame itself, which is a whole lot of bits flying across the interwebs.

pygame.org (**the website**) welcomes all Python game, art, music, sound, video and multimedia projects. Once you have finished [getting started](#) you could [add a new project](#) or learn [about](#) pygame by reading the [docs](#). For more information on what is happening in the pygame world see the [community dashboard](#), which lists many things like our projects we are working on, news (our blog with [rss](#)), [twitter](#), [reddit](#) (forum), [stackoverflow](#) (Q&A), [Bitbucket](#) (development), [irc](#) (chat), [mailinglist](#) (we love writing electronic mail to each other) and other various bits and pieces about pygame from around the internets.

pygame (**the community**) is a small volunteer group of creative humans who ♥ making things (there may also be a few cats, several koalas, dozens of doggos, 3.14 gnomes, and 42 robots who also tinker amongst us). We respect each other, and follow the [Python community code of conduct](#), whilst we help each other make interesting things.

Enjoy yourself whilst looking around. We look forward to your creations.

Best humanly possible wishes and extra toasty warm regards,  
pygame

<http://www.pygame.org>



#### Main

- About
- Bugs
- Licensing
- Credits
- Feedback

#### Documentation

- Wiki
- Forums
- Mailing Lists

#### Download

- SDL 2.0
- SDL 1.2
- SDL Mercurial
- Bindings

## About SDL

Simple DirectMedia Layer is a cross-platform development library designed to provide low level access to audio, keyboard, mouse, joystick, and graphics hardware via OpenGL and Direct3D. It is used by video playback software, emulators, and popular games including Valve's award winning catalog and many Humble Bundle games.

SDL officially supports Windows, Mac OS X, Linux, iOS, and Android. Support for other platforms may be found in the source code.

SDL is written in C, works natively with C++, and there are bindings available for several other languages, including C# and Python.

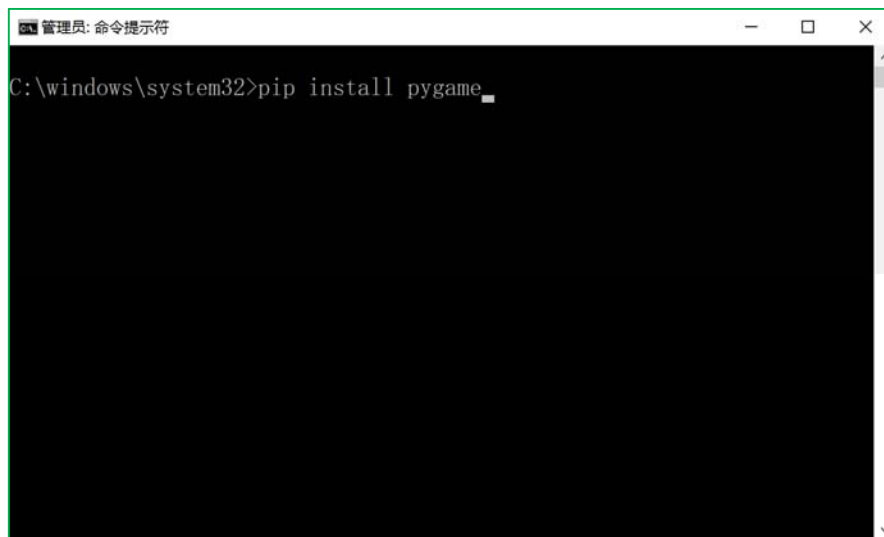
SDL 2.0 is distributed under the [zlib license](#). This license allows you to use SDL freely in any software.

<http://www.pygame.org>

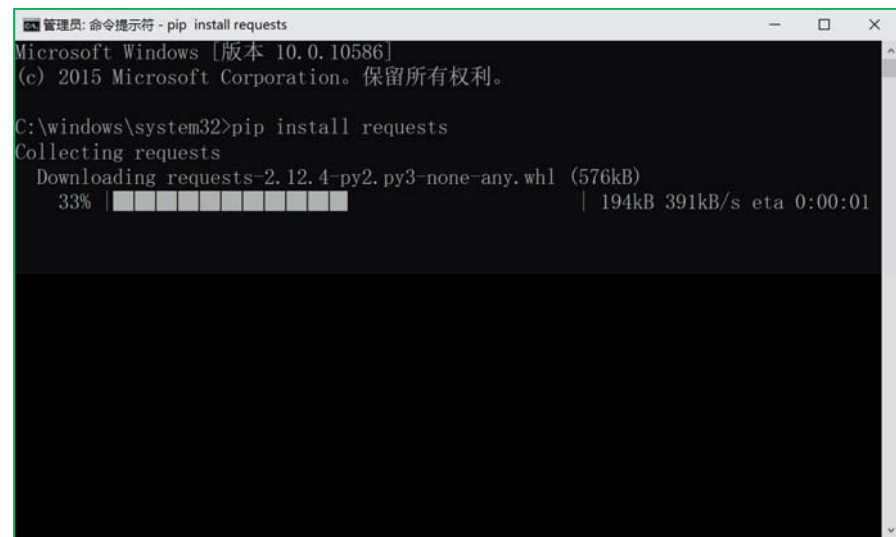


# Pygame的安装

Win平台： “以管理员身份运行” cmd , 执行 `pip install pygame`



```
管理员: 命令提示符
C:\windows\system32>pip install pygame
```

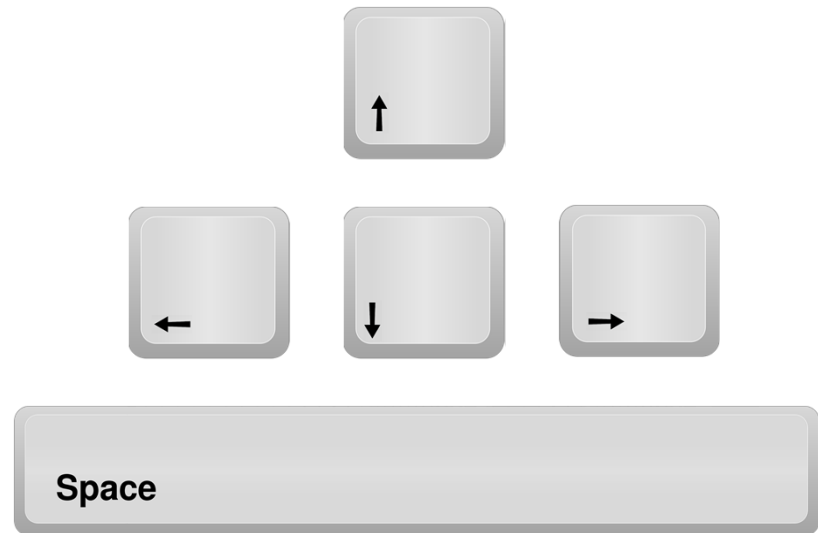
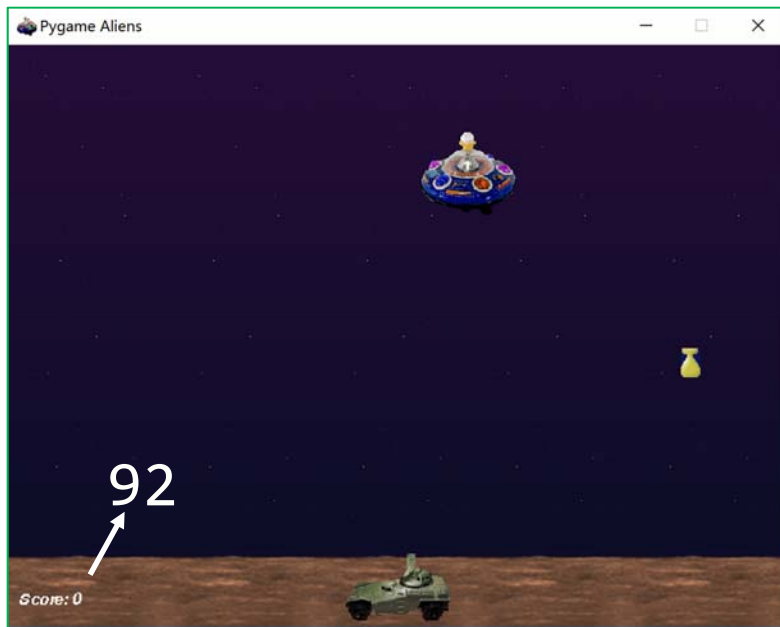


```
管理员: 命令提示符 - pip install requests
Microsoft Windows [版本 10.0.10586]
(c) 2015 Microsoft Corporation。保留所有权利。

C:\windows\system32>pip install requests
Collecting requests
  Downloading requests-2.12.4-py2.py3-none-any.whl (576kB)
    33% |#####| 194kB 391kB/s eta 0:00:01
```

# Pygame的安装小测

Win平台: cmd , 执行 `python -m pygame.examples.aliens`





# 游戏的开发环境：PyCharm



# 游戏的开发环境：PyCharm



PyCharm

Coming in 2017.1   What's New   2016.3   Features   Docs & Demos   Buy   **Download**



Version: 2016.3.2

Build: 163.10154.50

Released: December 30, 2016

[System requirements](#)

[Installation Instructions](#)

[Previous versions](#)

## Download PyCharm

[macOS](#)

**Windows**

[Linux](#)

### Professional

Full-featured IDE  
for Python & Web  
development

**DOWNLOAD**

232 MB

### Community

Lightweight IDE  
for Python & Scientific  
development

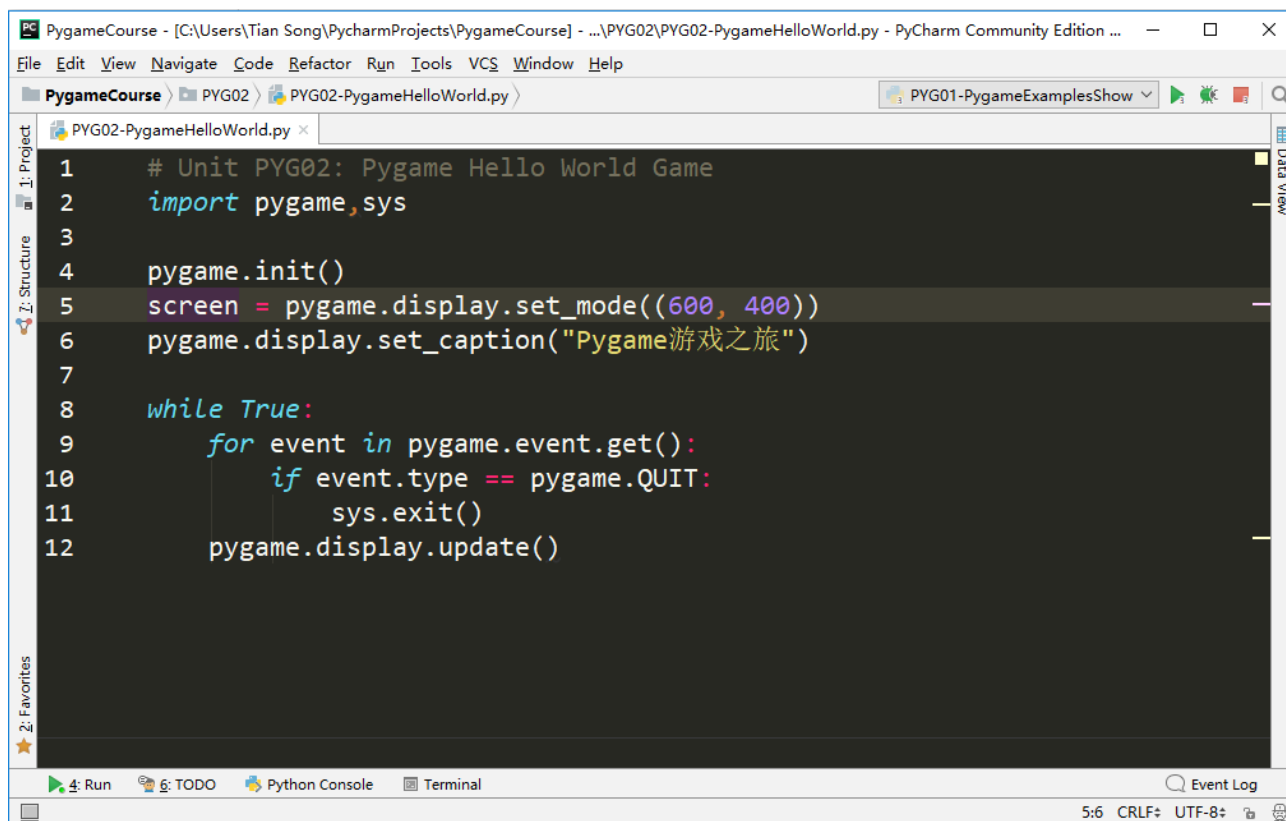
**DOWNLOAD**

179 MB



Get the **ToolBox App** to download PyCharm and its  
future updates with ease

# 游戏的开发环境：PyCharm



# 理解Pygame

- Python最经典的2D游戏开发第三方库，也支持3D游戏开发
- Pygame适合用于游戏逻辑验证、游戏入门及系统演示验证
- Pygame是一种游戏开发引擎，基本逻辑具有参考价值
- Pygame有些"过时"，但永远"不过时"
- 使用Pygame可以开发出优秀的游戏！

# 哪些同学不适合学习Pygame？

- 不喜欢游戏或对游戏开发没兴趣的
- 希望使用Pygame开发直接商用游戏的
- 只希望学习Python语言或一般技术开发用于就业的



无论学什么专业，从事什么工作，在世界的哪里坚守着

即使我们不能仗剑天涯，即使要面对眼前的苟且

请学习Pygame，用技术让自己的心灵来趟"诗和远方"的旅行吧



# Pygame最小开发框架

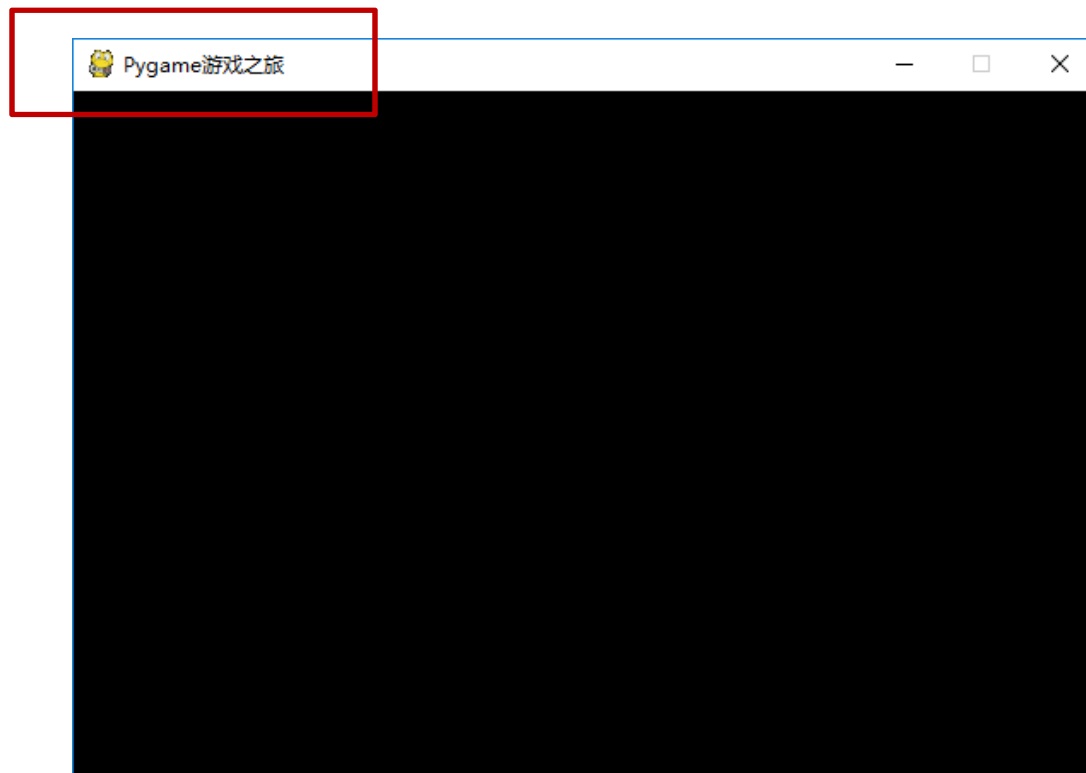
# Pygame的Hello World程序

```
1      # Unit PYG02: Pygame Hello World Game
2      import pygame, sys
3
4      pygame.init()
5      screen = pygame.display.set_mode((600, 400))
6      pygame.display.set_caption("Pygame游戏之旅")
7
8      while True:
9          for event in pygame.event.get():
10             if event.type == pygame.QUIT:
11                 sys.exit()
12             pygame.display.update()
```

9行有效代码



# Pygame的Hello World程序



# Pygame的最小开发框架

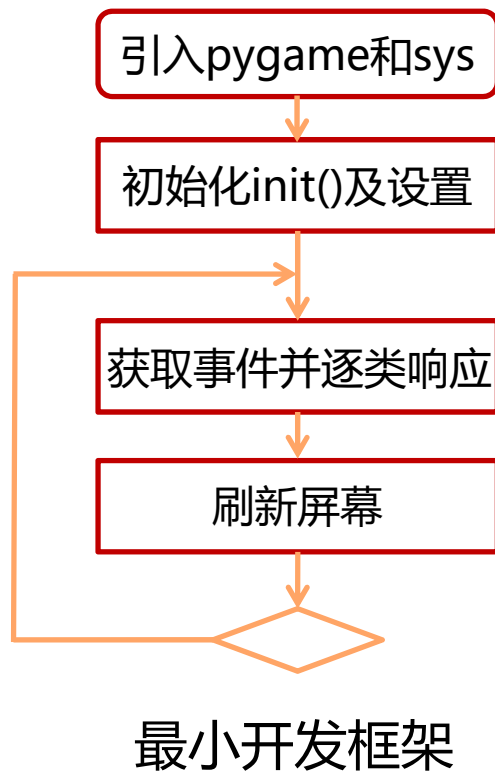
```
# Unit PYG02: Pygame Hello World Game
import pygame, sys

pygame.init()
screen = pygame.display.set_mode((600, 400))
pygame.display.set_caption("Pygame游戏之旅")

while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sys.exit()
    pygame.display.update()
```



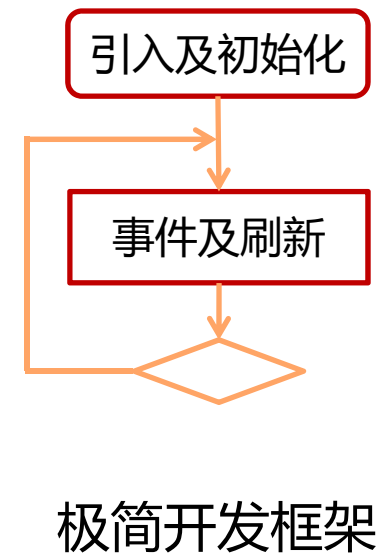
# Pygame的最小开发框架



```
# Unit PYG02: Pygame Hello World Game
import pygame, sys

pygame.init()
screen = pygame.display.set_mode((600, 400))
pygame.display.set_caption("Pygame游戏之旅")

while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sys.exit()
    pygame.display.update()
```



# Pygame的最小开发框架



# Pygame的最小开发框架



```
import pygame, sys  
sys.exit()
```

- sys是Python的标准库
- sys提供Python运行时环境变量的操控
- sys.exit()用于退出结束游戏并退出

# Pygame的最小开发框架



```
pygame.init()  
screen = pygame.display.set_mode((600, 400))  
pygame.display.set_caption("Pygame游戏之旅")
```

`pygame.init()`

对Pygame内部各功能模块进行初始化创建及变量设置，默认调用

# Pygame的最小开发框架



```
pygame.init()
screen = pygame.display.set_mode((600, 400))
pygame.display.set_caption("Pygame游戏之旅")
```

`pygame.display.set_mode(size)`

初始化显示窗口，第一个参数size是一个二值元组，分别表示窗口的宽度和高度

该函数完整功能将后续内容中介绍

# Pygame的最小开发框架



```
pygame.init()
screen = pygame.display.set_mode((600, 400))
pygame.display.set_caption("Pygame游戏之旅")
```

`pygame.display.set_caption(title)`

设置显示窗口的标题内容，参数title是一个字符串类型



# Pygame的最小开发框架



```
while True:  
    for event in pygame.event.get():  
        if event.type == pygame.QUIT:  
            sys.exit()
```

**while True:**

无限循环，直到Python运行时退出结束

# Pygame的最小开发框架



```
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sys.exit()
```

`pygame.event.get()`

从Pygame的事件队列中取出事件，并从队列中删除该事件，例如：键盘按下是一个事件。

该函数完整功能将后续内容中介绍

# Pygame的最小开发框架



```
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sys.exit()
```

`event.type`

`pygame.QUIT`

获得事件类型，并逐类响应；`pygame.QUIT` 是Pygame中定义的退出事件常量

事件类型完整功能将后续内容中介绍

# Pygame的最小开发框架



```
pygame.display.update()
```

`pygame.display.update()`

对显示窗口进行更新，默认窗口全部重绘

该函数完整功能将后续内容中介绍

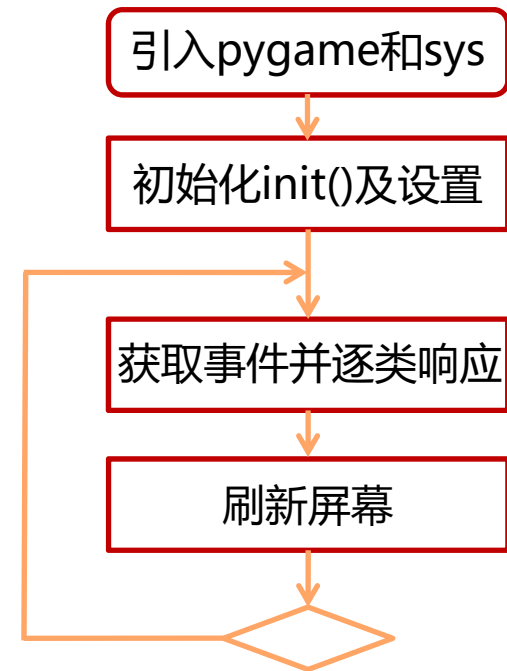
# Pygame的最小开发框架

```
# Unit PYG02: Pygame Hello World Game
import pygame, sys

pygame.init()
screen = pygame.display.set_mode((600, 400))
pygame.display.set_caption("pygame游戏之旅")

while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sys.exit()
    pygame.display.update()
```

与老师一起写一遍吧





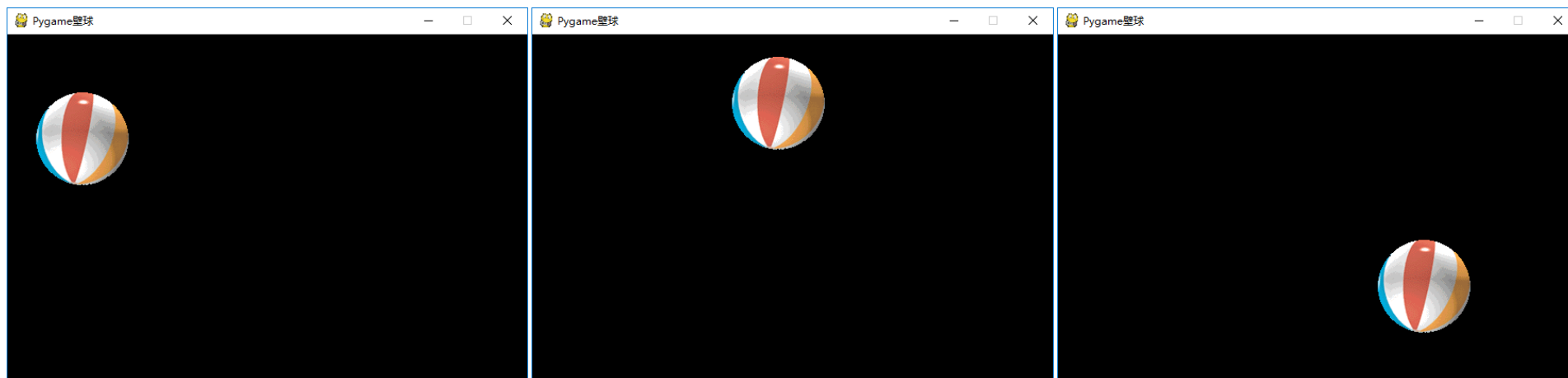
是不是觉得远方不那么遥远了？

接下来，与老师一同热热身，开发几款小游戏吧



# 壁球小游戏(展示型)与图像的基本使用

# 壁球小游戏(展示型)



四处碰壁且求索的小球



# 壁球小游戏(展示型)的关键要素

## 从需求到实现的三个关键要素：

- (1) **壁球**：游戏需要一个壁球，通过图片引入
- (2) **壁球运动**：壁球要能够上下左右运动
- (3) **壁球反弹**：壁球要能够在上下左右边缘反弹

## (1) 壁球

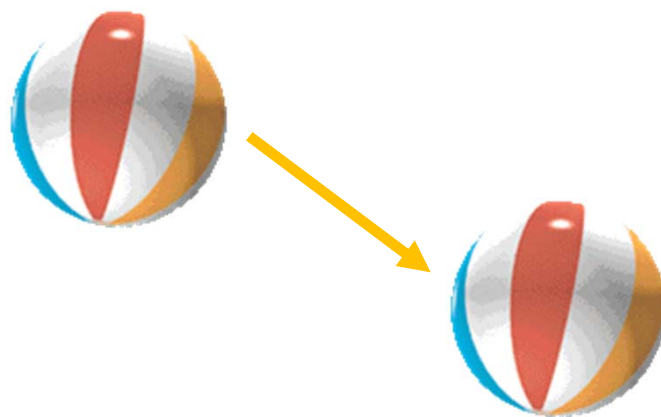


<https://python123.io/PY15/PYG02-ball.gif>

请通过上述链接下载壁球图片，请注意：

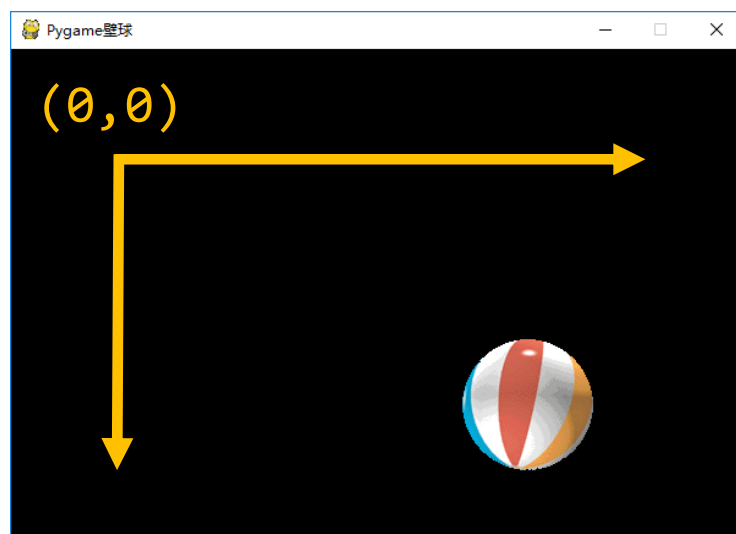
(1)注意大小写，PY和PYG大写；(2)下载图片保存为gif格式

## (2) 壁球运动



使图片每次向右及向下移动1个像素

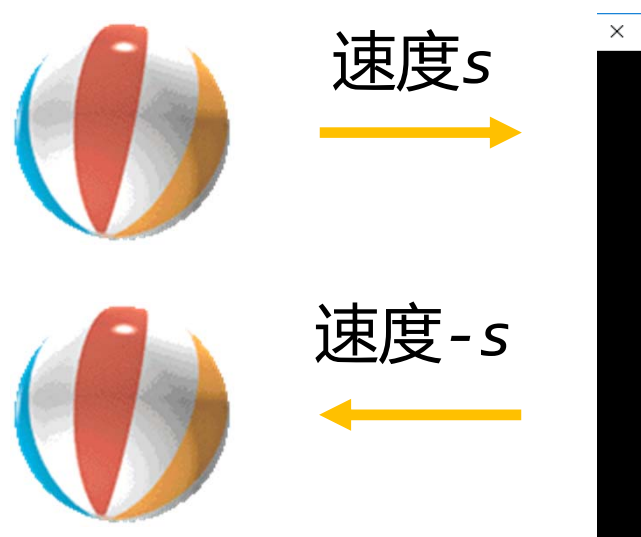
## (2) 壁球运动



与默认窗口处理方法一致，Pygame采用上图所示坐标体系：

窗口左上角坐标 $(0,0)$ ，横轴正向向右，纵轴正向向下

### (3) 壁球反弹



图片每次碰撞到边缘，速度取反

壁球小游戏(展示型)

**源代码来了！**

```
# Unit PYG02: Pygame Wall Ball Game version 1
import pygame, sys
```

```
pygame.init()
size = width, height = 600, 400
speed = [1,1]
BLACK = 0, 0, 0
screen = pygame.display.set_mode(size)
pygame.display.set_caption("Pygame壁球")
ball = pygame.image.load("PYG02-ball.gif")
ballrect = ball.get_rect()
```

```
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sys.exit()
    ballrect = ballrect.move(speed[0], speed[1])
    if ballrect.left < 0 or ballrect.right > width:
        speed[0] = - speed[0]
    if ballrect.top < 0 or ballrect.bottom > height:
        speed[1] = - speed[1]
```

```
screen.fill(BLACK)
screen.blit(ball, ballrect)
pygame.display.update()
```

引用

初始化

请浏览代码，找到开发框架

事件处理

窗口刷新

# 壁球小游戏(展示型)



```
pygame.init()
size = width, height = 600, 400
speed = [1,1]
BLACK = 0, 0, 0
screen = pygame.display.set_mode(size)
pygame.display.set_caption("Pygame壁球")
ball = pygame.image.load("PYG02-ball.gif")
ballrect = ball.get_rect()
```

`pygame.image.load(filename)`

将filename路径下的图像载入游戏，支持JPG、PNG、GIF(非动画)等13种常用图片格式



# 壁球小游戏(展示型)

```
ball = pygame.image.load("PYG02-ball.gif")  
ballrect = ball.get_rect()
```

Surface对象



Rect对象

Surface对象     `ball.get_rect()`

Pygame使用内部定义的Surface对象表示所有载入的图像，其中`.get_rect()`方法返回一个覆盖图像的矩形Rect对象

Surface对象和Rect对象的完整功能将后续内容中介绍

# 壁球小游戏(展示型)

(x,y)



(u,w)

```
ball = pygame.image.load("PYG02-ball.gif")
ballrect = ball.get_rect()
```

## Rect对象

Rect对象有一些重要属性，例如：

top, bottom, left, right 表示上下左右

width, height 表示宽度、高度

```
top = y
bottom = w
left = x
right = u
width = u-x
height = w-y
```

Rect对象的完整功能将后续内容中介绍

# 壁球小游戏(展示型)



```
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sys.exit()
    ballrect = ballrect.move(speed[0], speed[1])
    if ballrect.left < 0 or ballrect.right > width:
        speed[0] = - speed[0]
    if ballrect.top < 0 or ballrect.bottom > height:
        speed[1] = - speed[1]
```

**ballrect.move(x,y)**

矩形移动一个偏移量(x,y)，即在横轴方向移动x像素，纵轴方向移动y像素，xy为整数

# 壁球小游戏(展示型)



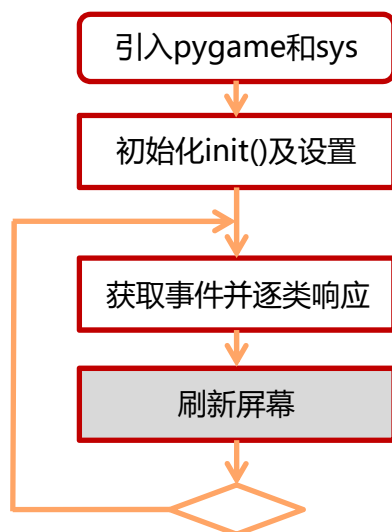
```
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sys.exit()
    ballrect = ballrect.move(speed[0], speed[1])
    if ballrect.left < 0 or ballrect.right > width:
        speed[0] = - speed[0]
    if ballrect.top < 0 or ballrect.bottom > height:
        speed[1] = - speed[1]
```

## 壁球的反弹运动

遇到左右两侧，横向速度取反；

遇到上下两侧，纵向速度取反。

# 壁球小游戏(展示型)



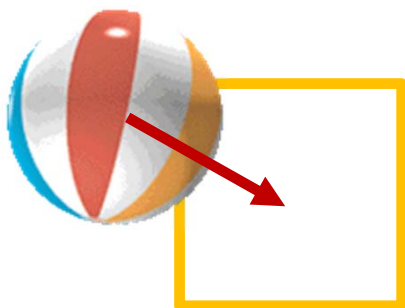
```
screen.fill(BLACK)
screen.blit(ball, ballrect)
pygame.display.update()
```

`screen.fill(color)`

显示窗口背景填充为color颜色，采用RGB色彩体系。由于壁球不断运动，运动后原有位置将默认填充白色，因此需要不断刷新背景色

# 壁球小游戏(展示型)

```
screen.fill(BLACK)
screen.blit(ball, ballrect)
pygame.display.update()
```



`screen.blit(src, dest)`

将一个图像绘制在另一个图像上，即将src绘制到dest位置上。通过Rect对象引导对壁球的绘制。

```
# Unit PYG02: Pygame Wall Ball Game version 1
import pygame,sys
```

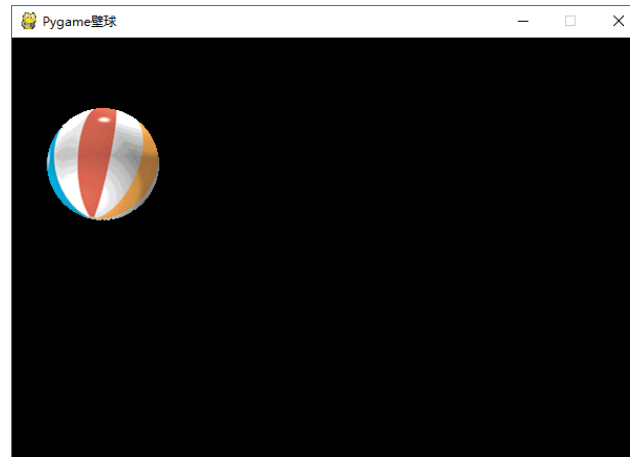
```
pygame.init()
size = width, height = 600, 400
speed = [1,1]
BLACK = 0, 0, 0
screen = pygame.display.set_mode(size)
pygame.display.set_caption("Pygame壁球")
ball = pygame.image.load("PYG02-ball.gif")
ballrect = ball.get_rect()
```

```
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sys.exit()
        ballrect = ballrect.move(speed[0], speed[1])
        if ballrect.left < 0 or ballrect.right > width:
            speed[0] = - speed[0]
        if ballrect.top < 0 or ballrect.bottom > height:
            speed[1] = - speed[1]

    screen.fill(BLACK)
    screen.blit(ball, ballrect)
    pygame.display.update()
```

与老师一起写一遍吧





A：老师老师，这个壁球太快了，怎么看都看不清？

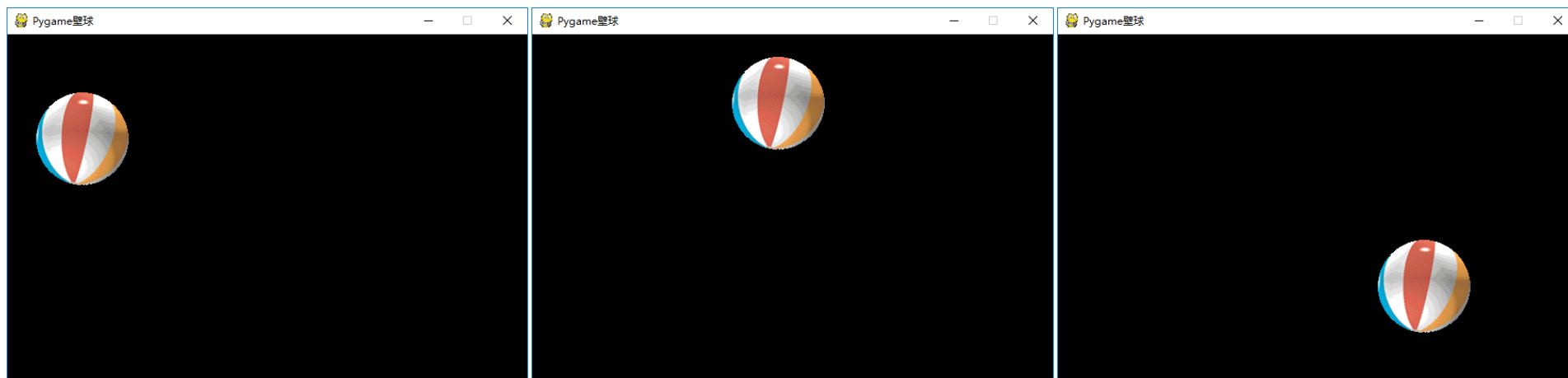
B：想控制壁球的节奏，且听老师继续讲解...





# 壁球小游戏(节奏型)与屏幕的帧率设置

# 壁球小游戏(节奏型)



四处有节奏碰壁的小球

# 壁球小游戏(节奏型)的关键要素

**需求：**

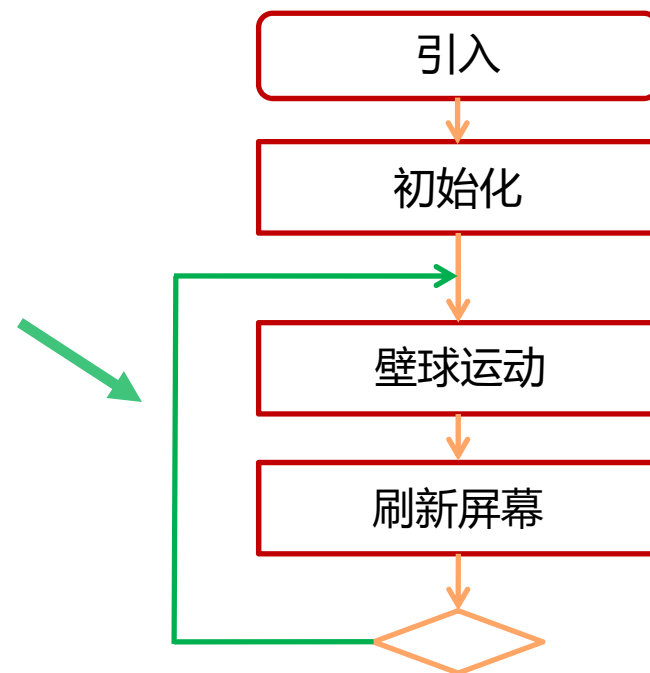
壁球可以按照一定速度运动

**从需求到实现的关键要素：**

- **壁球速度**：如何控制壁球的运动速度呢？

# 壁球速度

- 每次循环壁球运动一步
- 控制循环间隔即可控制速度
- (展示型)在尽最大能力运动



壁球小游戏(展示型)

**源代码来了！**

```
# Unit PYG02: Pygame Wall Ball Game version 2
import pygame, sys
```

```
pygame.init()
size = width, height = 600, 400
speed = [1,1]
BLACK = 0, 0, 0
screen = pygame.display.set_mode(size)
pygame.display.set_caption("Pygame壁球")
ball = pygame.image.load("PYG02-ball.gif")
ballrect = ball.get_rect()
fps = 300
fclock = pygame.time.Clock()
```

```
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sys.exit()
    ballrect = ballrect.move(speed[0], speed[1])
    if ballrect.left < 0 or ballrect.right > width:
        speed[0] = - speed[0]
    if ballrect.top < 0 or ballrect.bottom > height:
        speed[1] = - speed[1]
```

```
screen.fill(BLACK)
screen.blit(ball, ballrect)
pygame.display.update()
fclock.tick(fps)
```

引用

初始化

请找到与(展示型)的不同

事件处理

窗口刷新

# 壁球小游戏(节奏型)



```
fps = 300 # Frames per Second 每秒帧率参数  
fclock = pygame.time.Clock()
```

`pygame.time.Clock()`

创建一个Clock对象，用于操作时间

Clock对象的完整功能将后续内容中介绍

# 壁球小游戏(节奏型)



```
pygame.time.Clock().tick(fps)
```

`clock.tick(framerate)`

控制帧速度，即窗口刷新速度，例如：

`clock.tick(100)`表示每秒钟100次帧刷新

视频中每次展示的静态图像称为帧



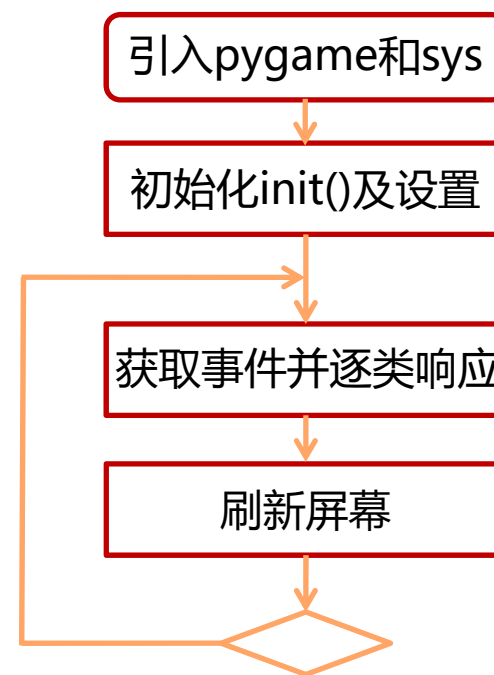
```
# Unit PYG02: Pygame Wall Ball Game version 2
import pygame, sys
```

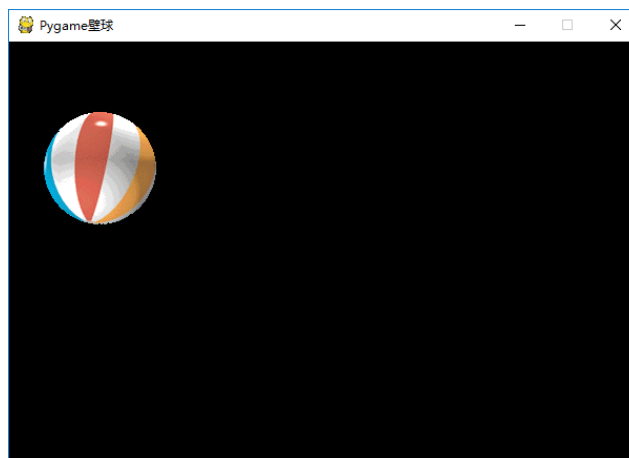
```
pygame.init()
size = width, height = 600, 400
speed = [1,1]
BLACK = 0, 0, 0
screen = pygame.display.set_mode(size)
pygame.display.set_caption("Pygame壁球")
ball = pygame.image.load("PYG02-ball.gif")
ballrect = ball.get_rect()
fps = 300
fclock = pygame.time.Clock()
```

```
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sys.exit()
    ballrect = ballrect.move(speed[0], speed[1])
    if ballrect.left < 0 or ballrect.right > width:
        speed[0] = - speed[0]
    if ballrect.top < 0 or ballrect.bottom > height:
        speed[1] = - speed[1]

    screen.fill(BLACK)
    screen.blit(ball, ballrect)
    pygame.display.update()
    fclock.tick(fps)
```

与老师一起写一遍吧





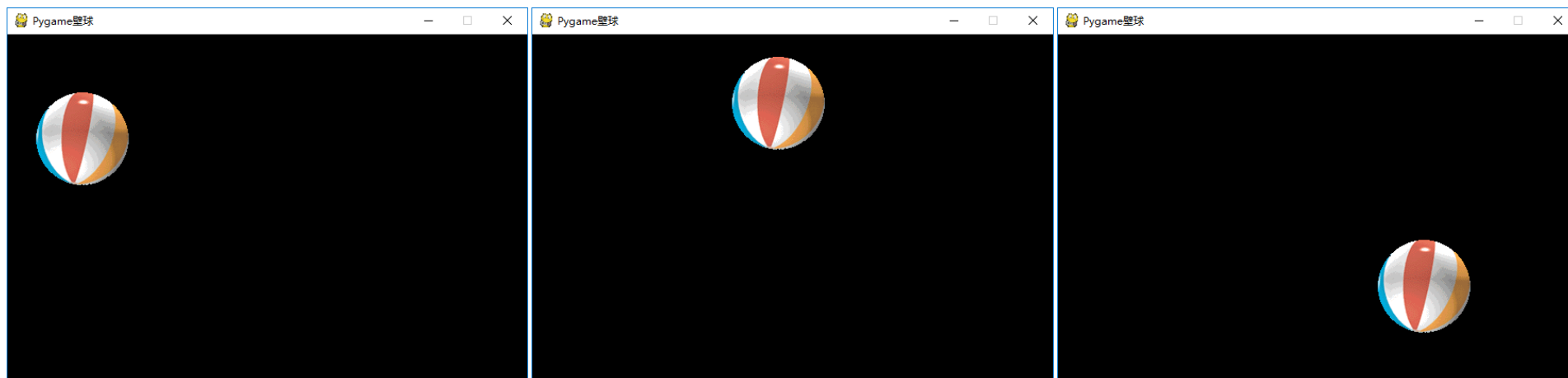
A：老师老师，这个程序能叫游戏吗？怎么都操作不了。

B：操控壁球，做个真正的游戏，且听老师继续讲解...



# 壁球小游戏(操控型)与键盘的基本使用

# 壁球小游戏(操控型)



被用户操控着且四处碰壁的小球

# 壁球小游戏(操控型)的关键要素

**需求：** 通过键盘的上下左右控制壁球运动速度，规则如下：

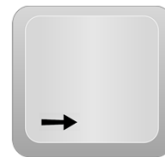
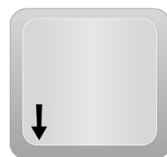
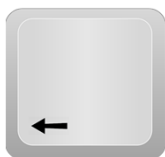
纵向绝对速度增加1个像素(纵向加速)



横向绝对速度减少1个像素

横向绝对速度增加1个像素

(横向减速)



(横向加速)

纵向绝对速度减少1个像素(纵向减速)

# 壁球小游戏(操控型)的关键要素

**从需求到实现的关键要素：**

- **键盘使用**：如何获取键盘的操作事件
- **速度调节**：根据对应按键调节壁球运动速度

# 键盘使用



- Pygame采用事件来对应键盘操作
- 获取事件将得到键盘输入
- 不同按键编写操作函数即可

壁球小游戏(操控型)

**源代码来了！**



```
# Unit PYG02: Pygame Wall Ball Game version 3
```

```
import pygame, sys
```

```
pygame.init()
```

```
size = width, height = 600, 400
```

```
speed = [1,1]
```

```
BLACK = 0, 0, 0
```

```
screen = pygame.display.set_mode(size)
```

```
pygame.display.set_caption("Pygame壁球")
```

```
ball = pygame.image.load("PYG02-ball.gif")
```

```
ballrect = ball.get_rect()
```

```
fps = 300
```

```
fclock = pygame.time.Clock()
```

```
while True:
```

```
    for event in pygame.event.get():
```

```
        if event.type == pygame.QUIT:
```

```
            sys.exit()
```

```
        elif event.type == pygame.KEYDOWN:
```

```
            if event.key == pygame.K_LEFT:
```

```
                speed[0] = speed[0] if speed[0] == 0 else (abs(speed[0]) - 1)*int(speed[0]/abs(speed[0]))
```

```
            elif event.key == pygame.K_RIGHT:
```

```
                speed[0] = speed[0] + 1 if speed[0] > 0 else speed[0] - 1
```

```
            elif event.key == pygame.K_UP:
```

```
                speed[1] = speed[1] + 1 if speed[1] > 0 else speed[1] - 1
```

```
            elif event.key == pygame.K_DOWN:
```

```
                speed[1] = speed[1] if speed[1] == 0 else (abs(speed[1]) - 1)*int(speed[1]/abs(speed[1]))
```

```
    ballrect = ballrect.move(speed)
```

```
    if ballrect.left < 0 or ballrect.right > width:
```

```
        speed[0] = - speed[0]
```

```
    if ballrect.top < 0 or ballrect.bottom > height:
```

```
        speed[1] = - speed[1]
```

```
    screen.fill(BLACK)
```

```
    screen.blit(ball, ballrect)
```

```
    pygame.display.update()
```

```
    fclock.tick(fps)
```

引用

初始化

请找到与(节奏型)的不同

事件处理

窗口刷新

# 壁球小游戏(节奏型)

```
elif event.type == pygame.KEYDOWN:
    if event.key == pygame.K_LEFT:
        speed[0] = speed[0] if speed[0] == 0 else (abs(speed[0]) - 1)*int(speed[0]/abs(speed[0]))
    elif event.key == pygame.K_RIGHT:
        speed[0] = speed[0] + 1 if speed[0] > 0 else speed[0] - 1
    elif event.key == pygame.K_UP:
        speed[1] = speed[1] + 1 if speed[1] > 0 else speed[1] - 1
    elif event.key == pygame.K_DOWN:
        speed[1] = speed[1] if speed[1] == 0 else (abs(speed[1]) - 1)*int(speed[1]/abs(speed[1]))
```

**pygame.KEYDOWN**

Pygame对键盘敲击的事件定义，键盘每个键对应一个具体定义



**pygame.K\_UP**



**pygame.K\_LEFT**



**pygame.K\_DOWN**



**pygame.K\_RIGHT**

键盘事件的完整功能将后续内容中介绍

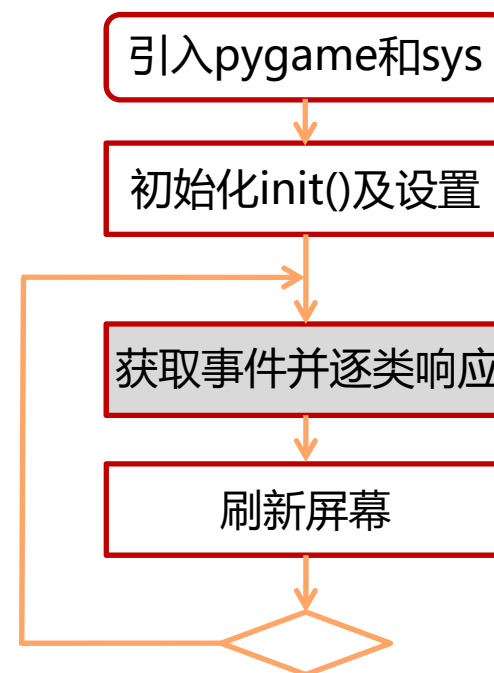
```
# Unit PYG02: Pygame Wall Ball Game version 2
import pygame,sys
```

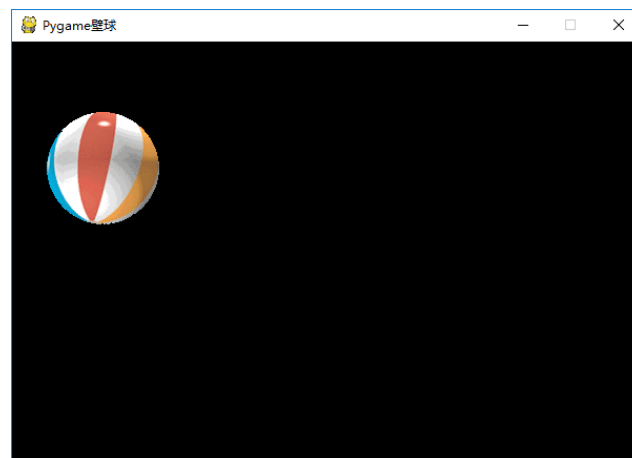
```
pygame.init()
size = width, height = 600, 400
speed = [1,1]
BLACK = 0, 0, 0
screen = pygame.display.set_mode(size)
pygame.display.set_caption("Pygame壁球")
ball = pygame.image.load("PYG02-ball.gif")
ballrect = ball.get_rect()
fps = 300
fclock = pygame.time.Clock()
```

```
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sys.exit()
    ballrect = ballrect.move(speed[0], speed[1])
    if ballrect.left < 0 or ballrect.right > width:
        speed[0] = - speed[0]
    if ballrect.top < 0 or ballrect.bottom > height:
        speed[1] = - speed[1]

    screen.fill(BLACK)
    screen.blit(ball, ballrect)
    pygame.display.update()
    fclock.tick(fps)
```

与老师一起写一遍吧



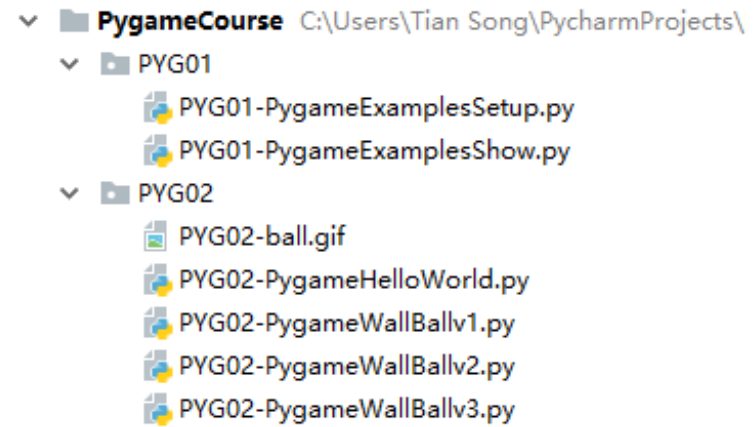


A：老师老师，这个壁球像个游戏了，要快快学习更多内容

B：且听老师继续讲解，但你想好心中的那个游戏了吗？



# 单元小结



- Pygame最小开发框架
- 图像的基本使用
- 屏幕的帧率设置
- 键盘的基本使用

Pygame是不是很有趣呢？