潍坊科技学院
Weifang University of Science and Technology

第10章 Python计算生态与常用库

授课老师：刘国旭

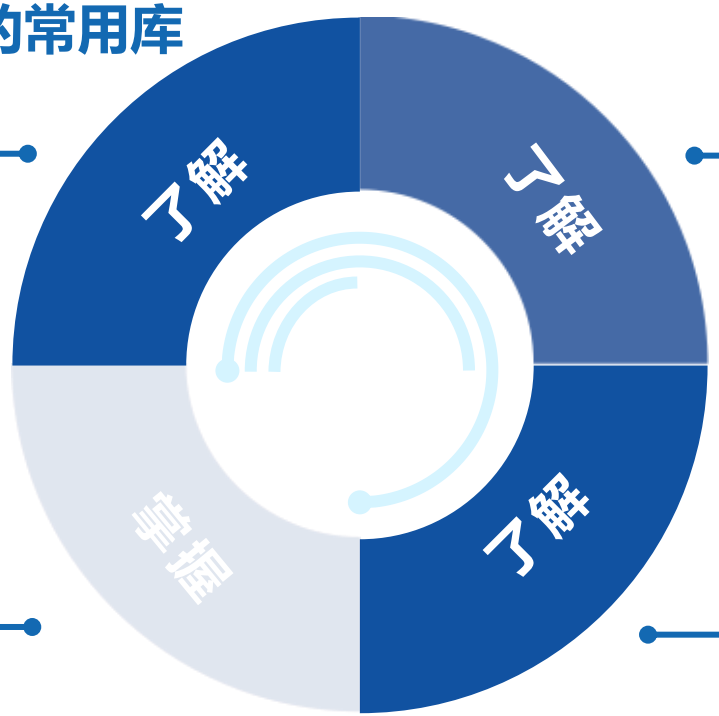
潍坊科技学院

- Python计算生态
- 各应用领域的常用库
- time库、random库
- turtle库
- jieba库、wordcloud库
- pygame库

学习目标

1 了解 Python计算生态及各应用领域的常用库

2 了解 Python生态库的构建与发布



3 了解 jieba、wordcloud、pygame库的基本用法

4 掌握 time、random、turtle库的基本用法

目录页

目录页

10.1 Python计算生态概述

Python计算生态涵盖网络爬虫、数据分析、文本处理、数据可视化、图形用户界面、机器学习、Web开发、网络应用开发、游戏开发、虚拟现实、图形艺术等多个领域，为各个领域的Python使用者提供了极大便利。

10.1 Python计算生态概述

网络爬虫是一种按照一定的规则，自动从网络上抓取信息的程序或者脚本。通过网络爬虫可以代替手工完成很多工作。

10.1 Python计算生态概述

网络爬虫程序涉及HTTP请求、Web信息提取、网页数据解析等操作，Python计算生态通过Requests、Python-Goose、Re、Beautiful Soup、Scrapy和PySpider等库为这些操作提供了强有力的支持，这些库各自的功能如表所示。

库名	功能说明
Requests	Requests提供了简单易用的类HTTP协议，支持连接池、SSL、Cookies，是Python最主要的、功能最丰富的网络爬虫功能库
Python-Goose	Python-Goose专用于从文章、视频类型的Web页面中提取数据
Re	Re提供了定义和解析正则表达式的一系列通用功能，除网络爬虫外，还适用于各类需要解析数据的场景
Beautiful Soup	Beautiful Soup用于从HTML、XML等Web页面中提取数据，它提供一些便捷的、Python式的函数，使用起来非常简单
Scrapy	Scrapy支持快速、高层次的屏幕抓取和批量、定时的Web抓取以及结构性数据的抓取，是一款优秀的网络爬虫框架
PySpider	PySpider也是一款爬虫框架，它支持数据库后端、消息队列、优先级、分布式架构等功能。与Scrapy相比，它灵活便捷，更适合小规模的爬取工作

10.1 Python计算生态概述

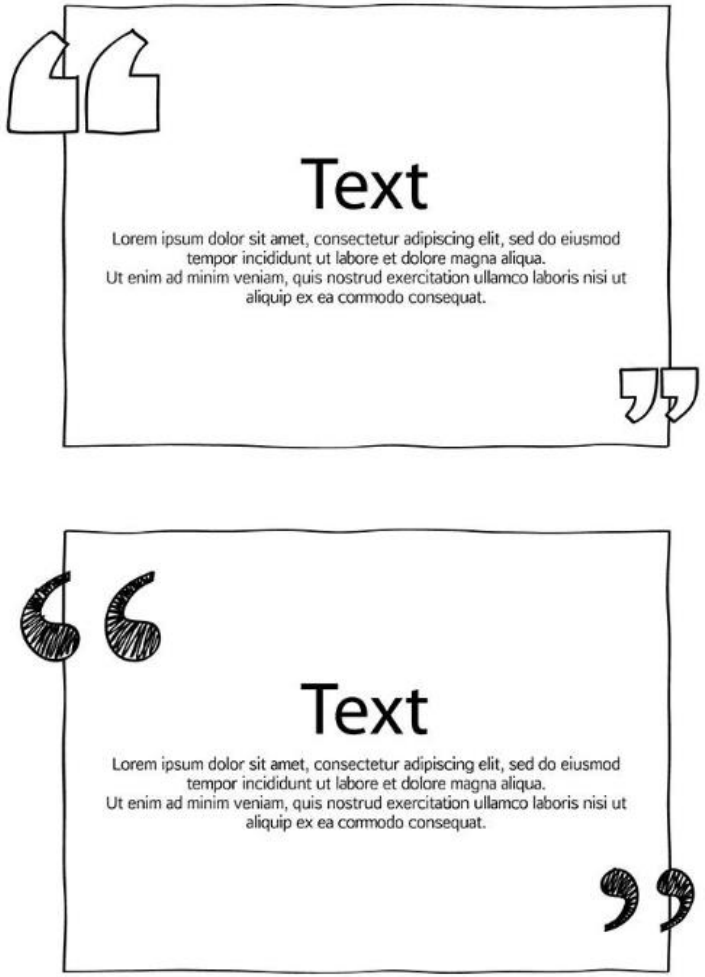
数据分析指用适当的统计分析方法对收集来的大量数据进行分析，将它们加以汇总、理解与消化，以求最大化地发挥数据的作用。

10.1 Python计算生态概述

Python计算生态通过Numpy、Pandas、SciPy库为数据分析领域提供支持，这些库各自的功能如表所示。

库名	功能说明
Numpy	数据分析离不开科学计算，Numpy定义了表示N维数组对象的类型ndarray，通过ndarray对象可以便捷地存储和处理大型矩阵；包含了成熟的用于实现线性代数、傅里叶变换和随机数生成的函数，能以优异的效率实现科学计算
Pandas	Pandas是一个基于Numpy开发的、用于分析结构化数据的工具集，它为解决数据分析任务而生，同时提供数据挖掘和数据清洗功能
SciPy	Scipy是Python科学计算程序中会使用的核心库，它用于有效地计算Numpy矩阵，可以处理插值、积分、优化等问题，也能处理图像和信号、求解常微分方程数值

10.1 Python计算生态概述



文本指书面语言的表现形式，从文学角度说，文本是具有完整、系统含义的一个句子或多个句子的组合。

文本处理即对文本内容的处理，包括文本内容的分类、文本特征的提取、文本内容的转换等等。

10.1 Python计算生态概述

Python计算生态通过Jieba、PyPDF2、Python-docx、NLTK等库为文本处理领域提供支持，这些库各自的功能如表所示。

库名	功能说明
Jieba	Jieba是一个优秀的Python中文分词库，它支持精确模式、全模式和搜索引擎模式这三种分词模式，支持繁体分词、自定义字典，可有效标注词性，从文本中提取关键词
NLTK	NLTK提供了用于访问超过50个语料库和语汇资源的接口，支持文本分类、标记、解析和语法、语义分析等功能，简单、易用且高效，是最优秀的Python自然语言处理库
PyPDF2	PyPDF2是一个专业且稳定的、用于处理PDF文档的Python库，它支持PDF文件信息的提取、文件内容的按页拆分与合并、页面裁剪、内容加密与解密等
Python-docx	Python-docx是一个用于处理Word文件的Python库，它支持Word文件中的标题、段落、分页符、图片、表格、文字等信息的管理，上手非常简单

10.1 Python计算生态概述

数据可视化是一门关于数据视觉表现形式的科学技术研究，它既要有效传达数据信息，也需兼顾信息传达的美学形式，二者缺一不可。

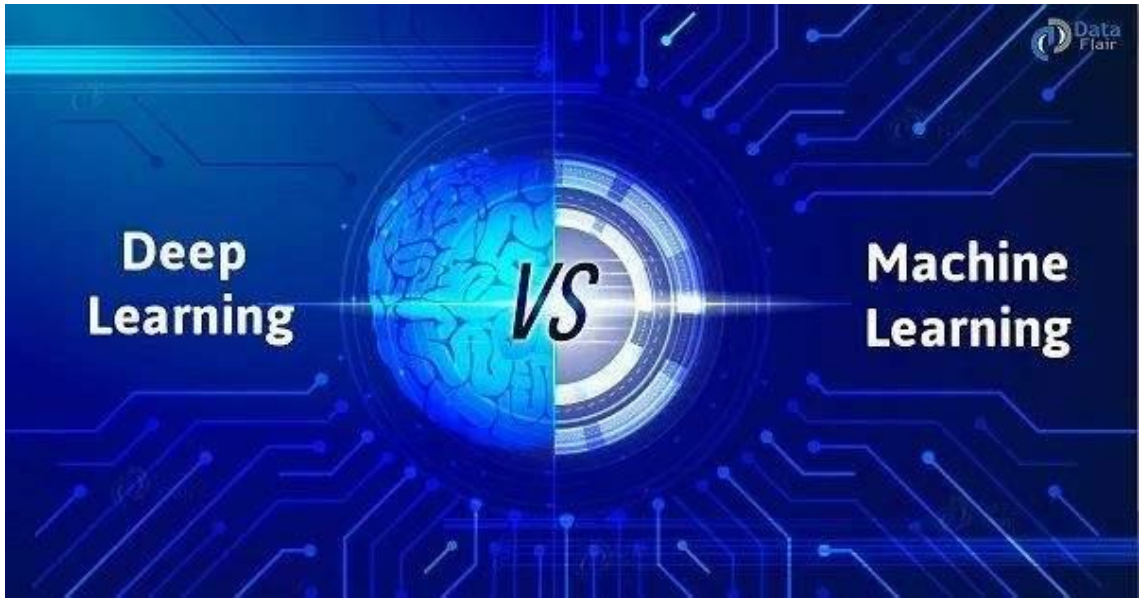
10.1 Python计算生态概述

Python计算生态主要通过Matplotlib、Seaborn、Mayavi等库为数据可视化领域提供支持，这些库各自的功能如表所示。

库名	功能说明
Matplotlib	Matplotlib是一个基于Numpy开发的2D Python绘图库，该库提供了上百种图形化的数据展示形式。Matplotlib库中pyplot包内包含一系列类似MATLAB中绘图功能的函数，利用Matplotlib.pyplot，开发者编写几行代码便可生成可视化图表
Seaborn	Seaborn在Matplotlib的基础上进行了更高级的封装，支持Numpy和Pandas，但它比Matplotlib调用更简单，效果更丰富，多数情况下可利用Seaborn绘制具有吸引力的图表
Mayavi	Mayavi是一个用于实现可视化功能的3D Python绘图库，它包含用于实现图形可视化和处理图形操作的mlab模块，支持Numpy库

10.1 Python计算生态概述

机器学习是一门涉及概率论、统计学、逼近论、凸分析、算法复杂度理论等多门学科的多领域交叉学科，该学科旨在研究计算机如何模拟或实现人类的学习行为，以获取新的知识或技能、重新组织已有知识结构并不断改善自身。机器学习是人工智能的核心，是使计算机具有智能的根本途径。



10.1 Python计算生态概述

Python计算生态主要通过Scikit-learn、TensorFlow、MXNet库为机器学习领域提供支持，这些库各自的功能如表所示。

库名	功能说明
Scikit-learn	Scikit-learn支持分类、回归、聚类、数据降维、模型选择、数据预处理，它提供了一批调用机器学习方法的接口，是Python机器学习领域中最优秀的免费库
TensorFlow	TensorFlow是一款以数据流图为基础，由谷歌人工智能团队开发和维护、免费且开源的机器学习计算框架，该框架支撑谷歌人工智能应用，提供了各类应用程序接口
MXNet	MXNet是一个轻量级分布式可移植深度学习库，它支持多机多节点多GPU计算，提供可扩展的神经网络以及深度学习计算功能，可用于自动驾驶、语音识别等领域

10.1 Python计算生态概述

图形用户界面（Graphical User Interface，简称GUI）指采用图形方式显示的计算机操作用户界面，该界面允许用户使用鼠标、键盘等输入设备操纵屏幕上的图标或菜单选项，以选择命令、调用文件、启动程序或执行一些其他的日常任务。

10.1 Python计算生态概述

Python计算生态通过PyQt5、WxPython、PyGObject库为图形用户界面领域提供支持，这些库各自的功能如表所示。

库名	功能说明
PyQt5	PyQt5库是Python与强大的GUI库——Qt的融合，它提供了Qt开发框架的Python接口，拥有超过300个类、将近6000个函数和方法，可开发功能强大的图形用户界面
WxPython	WxPython是跨平台库WxWidgets的Python版本，该库开源、支持跨平台，允许Python开发人员创建完整的、功能健全的图形用户界面，是一个优秀的GUI库
PyGObject	PyGObject绑定了Linux下最著名的图形库GTK3+，该库简单易用、功能强大、设计灵活，具有良好的设计理念和可扩展性，是一个优秀的GUI库

10.1 Python计算生态概述

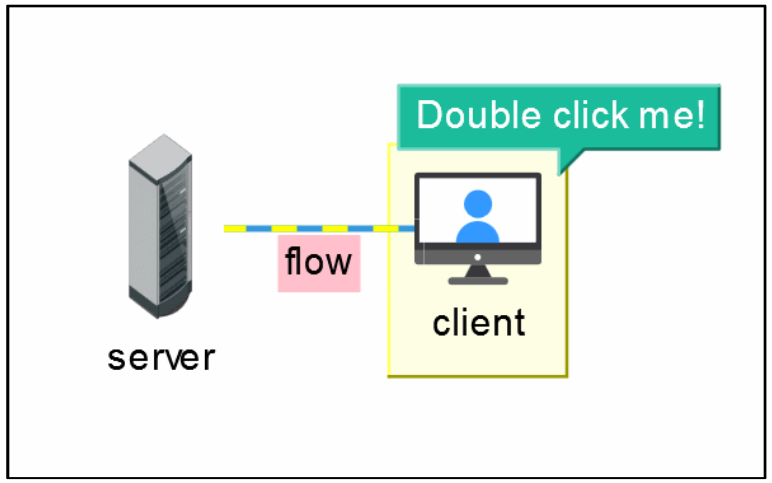
Web开发指基于浏览器而非桌面进行的程序开发。

10.1 Python计算生态概述

Python计算生态通过Django、Tornado、Flask、Twisted等库为Web开发领域提供了支持，这些库各自的功能如表所示。

库名	功能说明
Django	Django是一个免费开源且功能完善的Web框架，它采用MTV模式，提供URL路由映射、Request上下文和基于模板的页面渲染技术，内置一个功能强大的管理站点，适用于快速搭建企业级、高性能的内容类网站，是Python中最流行的Web开发框架
Tornado	Tornado是一个高并发处理框架，它常被用作大型站点的接口服务框架，而非如Django般建立完整网站的框架。Tornado同样提供URL路由映射、Request上下文和基于模板的页面渲染技术，此外它还支持异步I/O、提供超时事件处理，内置了可直接用于生产环境的HTTP服务器
Flask	Flask是Python Web领域一个新兴框架，它吸收了其他框架的优点，功能简单，但具有可扩展性，一般用于实现小型网站的开发
Twisted	Django、Tornado和Flask是基于应用层协议HTTP展开的框架，而Twisted是一个由事件驱动的网络框架。Twisted支持多种传输层和应用层协议，支持客户端和服务器双端开发，适用于开发追求服务器程序性能的应用

10.1 Python计算生态概述



网络应用开发指以网络为基础的应用程序的开发。

10.1 Python计算生态概述

Python计算生态通过WeRoBot、aip、MyQR等库为网络应用开发领域提供支持，这些库各自的功能如表所示。

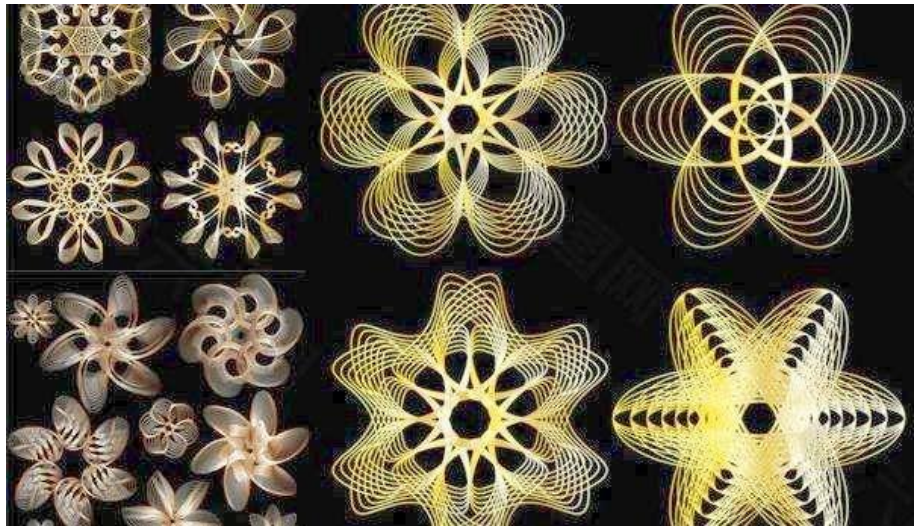
库名	功能说明
WeRoBot	WeRoBot库封装了很多微信公众号接口，提供了解析微信服务器消息及反馈消息的功能，该库简单易用，是建立微信机器人的重要技术手段
aip	aip封装了百度AI开放平台接口，利用该库中封装的接口可快速开发各类网络应用，如天气预报、在线翻译、快递查询等等
MyQR	MyQR是一个用于生成二维码的Python库

10.1 Python计算生态概述

Python计算生态通过PyGame、Panda3D库为游戏开发领域提供支持，这些库各自的说明如表所示。

库名	功能说明
PyGame	pygame是为开发2D游戏而设计的Python第三方跨平台库，开发人员利用pygame中定义的接口，可以方便快捷地实现诸如图形用户界面创建、图形和图像的绘制、用户键盘和鼠标操作的监听以及播放音频等游戏中常用的功能
Panda3D	panda3d是由迪士尼VR工作室和卡耐基梅隆娱乐技术中心开发的一个3D渲染和游戏开发库，该库强调能力、速度、完整性和容错能力，提供场景浏览器、性能监视器和动画优化工具，并通过完善代码来有效降低开发者跟踪和分析错误的难度

10.1 Python计算生态概述



图形艺术是一种通过标志来表现意义的艺术。标志是一些单纯、显著、易识别的具有指代性或具有表达意义、情感和指令等作用的物象、图形或文字符号，也是图形艺术的表现手段。

10.1 Python计算生态概述

Python计算生态通过Quads、ascii_art和turtle库为图形艺术领域提供支持，这些库各自的说明如表所示。

库名	功能说明
Quads	Quads是一个基于四叉树和迭代操作的图形艺术库，其功能是以图像作为输入，将输入图像分为四个象限，根据输入图像中的颜色为每个象限分配平均颜色，误差最大的象限会被分成四个子象限以完善图像，以上过程重复N次
ascii_art	ascii_art是一种使用纯字符表示图像的技术，Python的ascii_art库提供了对该技术的支持，该库可对接收到的图片进行转换，以字符形式重构图片并输出
turtle	turtle提供了绘制线、圆以及其他形状的函数，使用该库可以创建图形窗口，在图形窗口中通过简单重复动作直观地绘制界面与图形

10.1 Python计算生态概述

图像处理一般指数字图像（数字图像是指用工业相机、摄像机和扫描仪等设备经过拍摄得到的一个大的二维数组，这个数组的元素称为像素，其值称为灰度值）处理，图像处理技术一般包括图像压缩、增强和复原、图像匹配、描述和识别。

10.1 Python计算生态概述

Python通过Numpy、Scipy、Pillow、OpenCV-Python等库为图像处理领域提供支持，这些库各自的说明如表所示。

库名	功能说明
Numpy	数字图像的本质是数组，Numpy定义的数组类型非常适用于存储图像；Numpy提供基于数组的计算功能，利用这些功能可以很方便地修改图像的像素值
Scipy	Scipy提供了对N维Numpy数组进行运算的函数，这些函数实现的功能，包括线性和非线性滤波、二值形态、B样条插值等都适用于图像处理
Pillow	Pillow库是PIL库的一个分支，也是支持Python3的图像处理库，该库提供了对不同格式图像文件的打开和保存操作，也提供了包括点运算、色彩空间转换等基本的图像处理功能
OpenCV-Python	OpenCV-Python是OpenCV的Python版API，OpenCV是基于BSD许可发型的跨平台计算机视觉库，该库内部代码由C/C++编写，实现了图像处理和计算机视觉方面的很多通用算法；OpenCV-Python以Python代码对OpenCV进行封装，因此该库即方便使用又非常高效

目录页

10.2Python生态库的构建与发布

库是Python中常常提及的概念，但事实上Python中的库只是一种对特定功能集合的统一说法而非严格定义。Python库的具体表现形式为模块（Module）和包（Package），下面分这两部分介绍Python库的构建与使用，并介绍如何发布第三方库。

10.2.1 模块的构建与使用

Python模块本质上是一个包含Python代码片段的.py文件，模块名就是文件名。那么创建一个.py文件，在其中编写功能代码并保存，便可构建一个模块。

10.2.1 模块的构建与使用

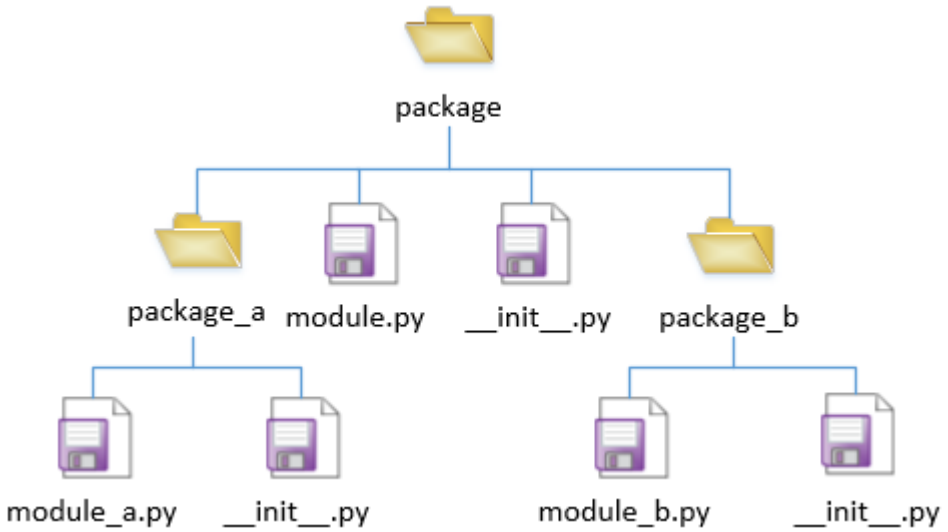
导入模块：

import 模块名

from … import …

10.2.2 包的构建与导入

- 将模块放入一个文件夹，并在该文件夹中创建__init__.py文件，就构建了一个Python包。
- 简单地说，Python中的包就是以目录形式组织起来的、具有层级关系的多个模块。
- Python包中可以包含子包，包结构示例如图所示。



此时若想在当前程序中导入以上包中的模块module_a，使用的导入语句如下：

- import package.package_a.module_a　　# 方式一
- from package.package_a import module_a　　# 方式二

10.2.3 库的发布

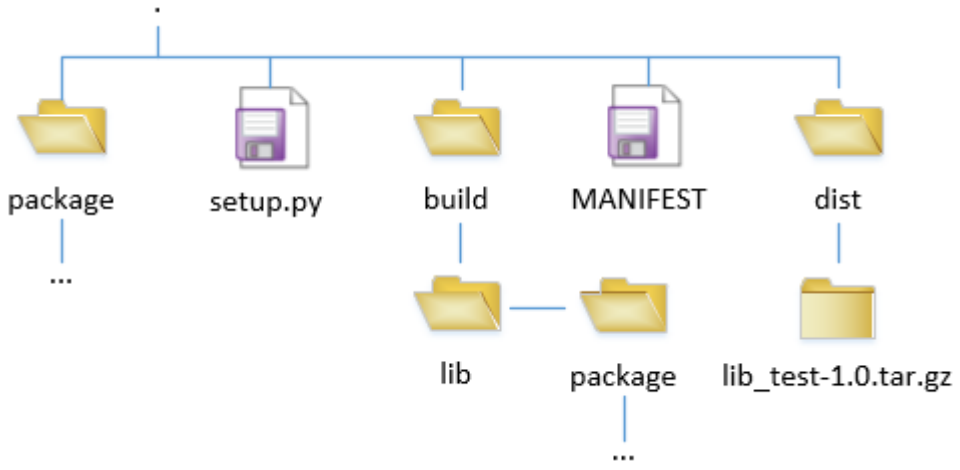
Python中的第三方库是由Python使用者自行编写与发布的模块或包，同样的，我们也可以将自己编写的模块与包作为库发布。具体步骤如下：

1.在与待发布的包同级的目录中创建setup.py文件。

2.编辑setup.py文件，在该文件中设置包中包含的模块。

3.在setup.py文件所在目录下打开命令行，使用python setup.py build命令构建Python库。

4.在setup.py文件所在目录下打开命令行，使用python setup.py sdist命令创建库的安装包。



目录页

10.3.1 time库

time库

time是最基础的时间处理库，该库本质上是一个模块，它包含的所有内容都定义在time.py文件中。该库中定义了time()、strftime()、localtime()、sleep()和一些用于实现时间格式转换的函数。

time()函数

time()函数返回以浮点数表示的从世界标准时间的1970年1月1日00:00:00开始到现在的总秒数，也就是时间戳。

10.3.1 time库

time库

time是最基础的时间处理库，该库本质上是一个模块，它包含的所有内容都定义在time.py文件中。该库中定义了time()、strftime()、localtime()、sleep()和一些用于实现时间格式转换的函数。

localtime()与gmtime()函数

localtime()函数和gmtime()函数都可将时间戳转换为以元组表示的时间对象（struct_time），localtime()得到的是当地时间，gmtime()得到的是世界统一时间（Coordinated Universal Time，简称UTC），它们的语法格式如下：

localtime([secs])

gmtime([secs])

参数secs是一个表示时间戳的浮点数，若不提供该参数，默认以time()函数获取的时间戳作为参数。

10.3.1 time库

struct_time元组元素的含义与取值

元素	含义	取值
tm_year	年	4位数字
tm_mon	月	1~12
tm_mday	日	1~31
tm_hour	时	0~23
tm_min	分	0~59
tm_sec	秒	0~61（60或61是闰秒）
tm_wday	一周的第几日	0~6（0为周一，依此类推）
tm_yday	一年的第几日	1~366
tm_isdst	夏令时	1：是夏令时 0：非夏令时 -1：不确定

10.3.1 time库

time库

time是最基础的时间处理库，该库本质上是一个模块，它包含的所有内容都定义在time.py文件中。该库中定义了time()、strftime()、localtime()、sleep()和一些用于实现时间格式转换的函数。

strftime()和asctime()函数

strftime()函数借助时间格式控制符来输出格式化的时间字符串，该函数的语法格式如下：

strftime(format[, t])

- 参数format是指代时间格式的字符串。
- 参数t为struct_time对象，默认为当前时间，即localtime()函数返回的时间，该参数可以省略。

10.3.1 time库

时间格式控制符

时间格式控制符	说明
%Y	四位数的年份，取值范围为0001～9999
%m	月份（01～12）
%d	月中的一天
%B	本地完整的月份名称，比如January
%b	本地简化的月份名称，比如Jan
%a	本地简化的周日期
%A	本地完整周日期
%H	24小时制小时数（0～23）
%I	12小时制小时数（01～12）
%p	上下午，AP或PM
%M	分钟数（00～59）
%S	秒（00～59）

10.3.1 time库

time库

time是最基础的时间处理库，该库本质上是一个模块，它包含的所有内容都定义在time.py文件中。该库中定义了time()、strftime()、localtime()、sleep()和一些用于实现时间格式转换的函数。

strftime()和asctime()函数

asctime()函数同样用于输出格式化的时间字符串，但它只将struct_time对象转化为Sat Jan 13 21:56:34 2018'这种形式。asctime()函数的语法格式如下：

asctime([t])

以上格式中的参数t与和strftime()函数的参数t意义相同。

10.3.1 time库

time库

time是最基础的时间处理库，该库本质上是一个模块，它包含的所有内容都定义在time.py文件中。该库中定义了time()、strftime()、localtime()、sleep()和一些用于实现时间格式转换的函数。

ctime()函数

ctime()函数用于将一个时间戳（以秒为单位的浮点数）转换为'Sat Jan 13 21:56:34 2018'这种形式（结果同time.asctime()），若该函数未接收到参数，则默认以time.time()作为参数。

10.3.1 time库

time库

time是最基础的时间处理库，该库本质上是一个模块，它包含的所有内容都定义在time.py文件中。该库中定义了time()、strftime()、localtime()、sleep()和一些用于实现时间格式转换的函数。

strptime()函数

strptime()函数用于将格式化的时间字符串转化为struct_time，该函数是strftime()函数的反向操作。

strptime()函数的语法格式如下：

strptime(string, format)

以上格式中的参数string表示格式化的时间字符串，format表示时间字符串的格式，string与format必须统一。

10.3.1 time库

time库

time是最基础的时间处理库，该库本质上是一个模块，它包含的所有内容都定义在time.py文件中。该库中定义了time()、strftime()、localtime()、sleep()和一些用于实现时间格式转换的函数。

sleep()函数

sleep()函数可让调用该函数的程序进入睡眠态，即让其暂时挂起，等待一定时间后再继续执行。sleep()函数接收一个以秒为单位的浮点数作为参数，使用该参数控制进程或线程挂起的时长。

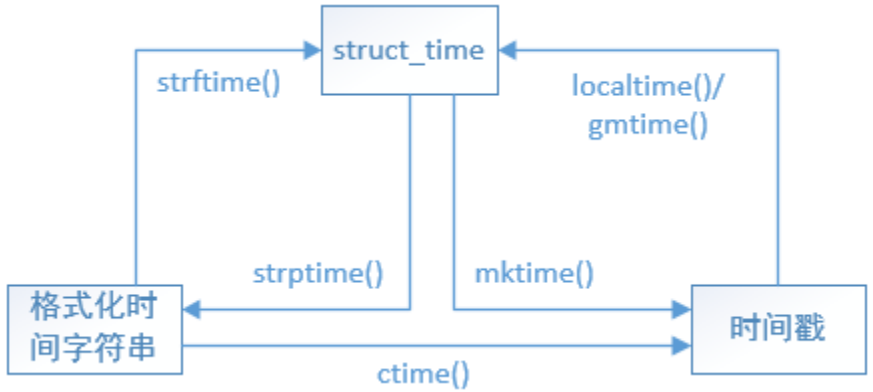
10.3.1 time库

time库

time是最基础的时间处理库，该库本质上是一个模块，它包含的所有内容都定义在time.py文件中。该库中定义了time()、strftime()、localtime()、sleep()和一些用于实现时间格式转换的函数。

时间计算

时间计算通常指时间的加减，时间可以时间戳形式进行加减运算。若要对非时间戳形式表示的时间进行计算，在计算之前可以先将其转换为时间戳形式。各形式之间的转换方式如图所示。



10.3.2 random库

random是Python内置的标准库，在程序中导入该库，可利用库中的函数**生成随机数据**。

random库中常用的函数如表所示。

函数	功能说明
random.random()	用于生成一个随机浮点数n，0<=n<1.0
random.uniform(a,b)	用于生成一个指定范围内的随机浮点数n，若a<b，则a<=n<=b；若a>b，则b<=n<=a
random.randint(a,b)	用于生成一个指定范围内的整数n，a<=n<=b
random.randrange([start,]stop[,step])	生成一个按指定基数递增的序列，再从该序列中获取一个随机数
random.choice(sequence)	从序列中获取一个随机元素，参数sequence表示一个有序类型
random.shuffle(x[,random])	将序列x中的元素随机排列
random.sample(sequence,k)	从指定序列中获取长度为k的片段，随机排列后返回新的序列。该函数可以基于不可变序列进行操作

10.3.3 turtle库

turle绘图模块

turtle（海龟）是Python内置的一个标准模块，它提供了绘制线、圆以及其它形状的函数，使用该模块可以创建图形窗口，在图形窗口中通过简单重复动作直观地绘制界面与图形。

turtle的使用主要分为以下三个方面：

- 创建窗口
- 设置画布
- 绘制图形

10.3.3 turtle库

创建窗口

- 图形窗口也称为画布（canas）。
- 控制台无法绘制图形，使用turtle模块绘制图形化界面，需要先使用setup()函数创建图形窗口。

```
turtle.setup(width, height, startx=None, starty=None)
```

参数含义：

- width： 窗口宽度
- height： 窗口高度

值为整数时，表示以像素为单位的尺寸；
值为小数时，表示图形窗口的宽或高与屏幕的比例

- startx：窗口在计算机屏幕上的横坐标
- starty：窗口在计算机屏幕上的纵坐标

startx、starty的取值可以为整数或None；
当取值为整数时，分别表示图形窗口左侧、顶部与屏幕左侧、顶部的距离（单位为像素）；
当取值为None时，窗口位于屏幕中心。

10.3.3 turtle库

设置画笔

画笔（pen）的设置包括画笔属性，如尺寸、颜色的设置，和画笔状态的设置。

（1）画笔属性函数

```
turtle.pensize(<width>)      # 设置画笔尺寸
turtle.speed(speed)          # 设置画笔移动速度
turtle.color(color)          # 设置画笔颜色
```

参数含义：

- pensize()函数的参数width可以设置画笔绘制出的线条的宽度。
- speed()函数的参数speed用于设置画笔移动的速度。
- color()函数的参数color用于设置画笔的颜色。

10.3.3 turtle库

设置画笔

（2）画笔状态函数

```
turtle.penup()      # 提起画笔
turtle.pendown()    # 放下画笔
```

说明：

- turtle模块中为penup()和pendown()函数定义了别名;
- penup()函数的别名为pu();
- pendown()函数的别名为pd()。

10.3.3 turtle库

绘制图形

在画笔状态为DOWN时，通过移动画笔可以在画布上绘制图形，可以将画笔想象成一只海龟（这也是turtle模块名字的由来）：海龟落在画布上，它可以向前、向后、向左、向右移动，海龟爬动时在画布上留下痕迹，路径即为所绘图形。

（1）移动控制函数

```
turtle.forward(distance)          # 向前移动
turtle.backward(distance)         # 向后移动
turtle.goto(x,y=None)             # 移动到指定位置
```

参数含义：

- 函数forward()和backward()的参数distance用于指定画笔移动的距离，单位为像素；
- 函数goto()用于将画笔移动到画布上指定的位置，该函数可以使用x、y分别接收表示目标位置的横坐标和纵坐标，也可以仅接收一个表示坐标向量的参数。

10.3.3 turtle库

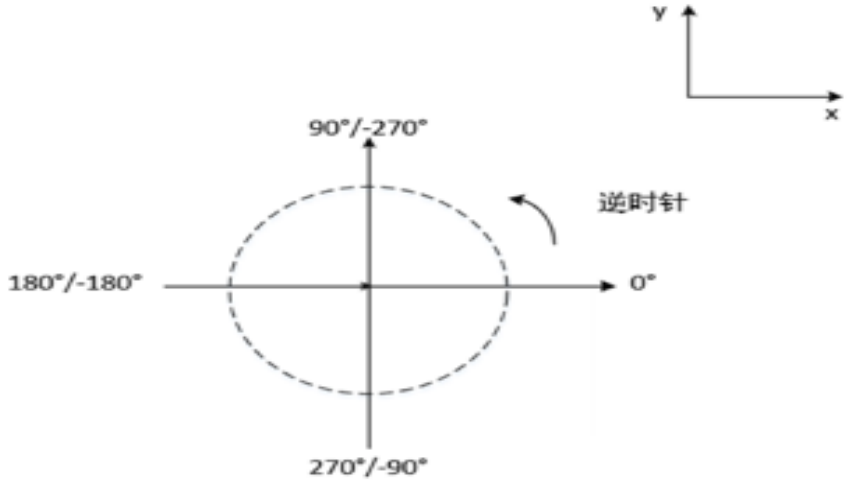
绘制图形

(2) 角度控制函数

```
turtle.right(degree)        # 向右转动
turtle.left(degree)         # 向左转动
turtle.seth(angle)          # 转动到某个方向
```

参数含义:

- 函数right()和left()的参数degree用于指定画笔向右与向左的角度;
- 函数seth()的参数angle用于设置画笔在坐标系中的角度。



10.3.3 turtle库

绘制图形

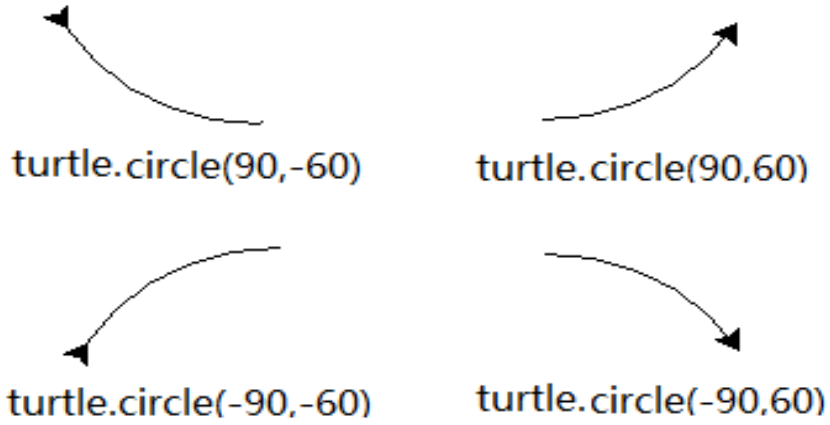
(3) 绘制函数

```
turtle.circle(radius, extent=None, steps=None)
```

参数含义:

- 参数radius用于设置半径;
- 参数extent用于设置弧的角度。

- 当radius为正时，画笔以原点为起点向上绘制弧线；radius为负时，画笔以原点为起点向下绘制弧线。
- 当extent为正时，画笔以原点为起点向右绘制弧线；extent为负时，画笔以原点为起点向左绘制弧线。



10.3.3 turtle库

绘制图形

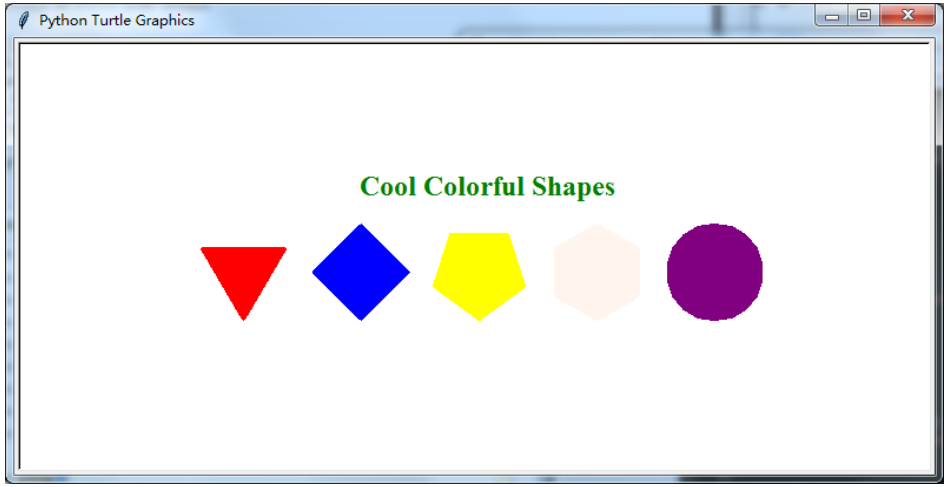
(4) 图形填充

```
turtle.begin_fill()      # 开始填充
turtle.end_fill()        # 结束填充
```

目录页

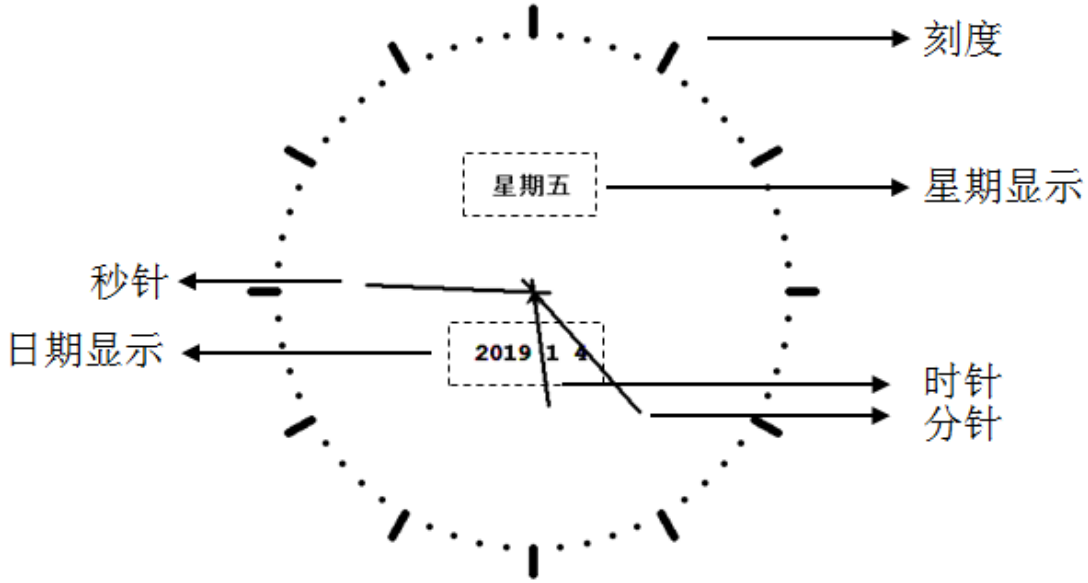
10.4.1 图形绘制

本实例要求编写程序，在程序中利用turtle模块绘制几何图形，绘制效果如图所示。



10.4.2 模拟时钟

本实例要求利用turtle和time绘制下图所示的钟表，并使钟表的日期、周日期、时间跟随本地时间实时变化。



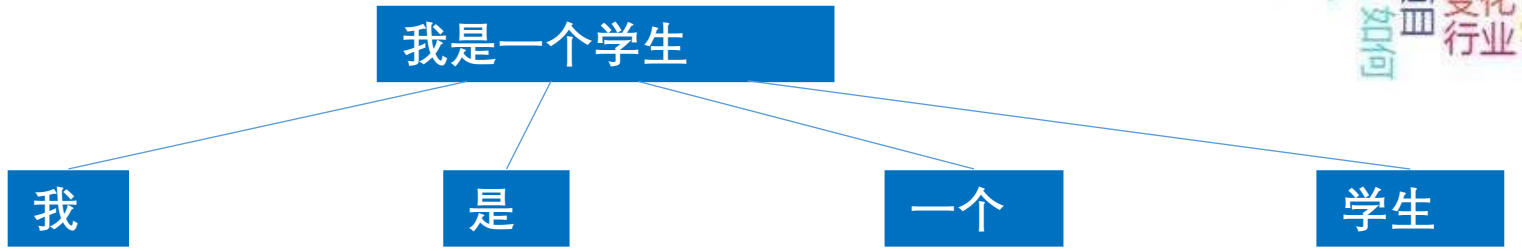
目录页

10.5.1 jieba库

中文分词

中文分词是指将一个汉字序列切分成一个一个单独的词，也就是说将连续的字序列按照一定的规范重新组合成词序列的过程，其作用就是将用户输入的中文语句或语段拆成若干汉语词汇。

示例：



10.5.1 jieba库

中文分词模块——jieba

安装jieba

pip install jieba/pip3 install jieba

导入jieba

import jieba

jieba模块的分词模式

精确模式 + 全模式 + 搜索引擎模式

10.5.1 jieba库

常用分词函数

函数	功能说明
jieba.cut(s)	采用精准模式对文本s进行分词，返回一个可迭代对象
jieba.cut(s, cut_all=True)	默认采用全模式对文本s进行分词，输出文本s中出现的所有词。
jieba.cut_for_search(s)	采用搜索引擎模式对文本s进行分词
jieba.lcut(s)	采用精准模式对文本s进行分词，分词结果以列表形式返回
jieba.lcut(s,cut_all=True)	采用全模式对文本s进行分词，分词结果以列表形式返回
jieba.lcut_for_search(s)	采用搜索引擎模式对文本s进行分词，分词结果以列表形式返回

10.5.1 jieba库

示例：分别采用三种模式对中文进行分词操作

```
import jieba
seg_list = jieba.cut("我打算到中国科学研究院图书馆学习", cut_all=True)
print("【全模式】: " + "/ ".join(seg_list))                    # 全模式
seg_list = jieba.lcut("我打算到中国科学研究院图书馆学习")
print("【精确模式】: " + "/ ".join(seg_list))                  # 精确模式
# 搜索引擎模式
seg_list = jieba.cut_for_search("我打算到中国科学研究院图书馆学习")
print("【搜索引擎模式】: " + ", ".join(seg_list))
```

10.5.1 jieba库

增加新词——add_word()

示例

```
jieba.add_word("好天气")
jieba.lcut("今天真是个好天气")
```

10.5.2 wordcloud库

wordcloud库

Python的第三方库wordcloud是专用于实现词云功能的库，该库能将文本中词语出现的频率作为参数来绘制词云，并支持对词云的形状、颜色和大小等属性进行设置。

生成词云的主要步骤

1.利用WordCloud类的构造方法WordCloud()创建词云对象。

2.利用WordCloud对象的generate()方法加载词云文本。

3.利用WordCloud对象的to_file()方法生成词云。

10.5.2 wordcloud库

WordCloud()函数参数

参数	说明
width	指定词云对象生成图片的宽度，默认为400像素
height	指定词云对象生成图片的高度，默认为200像素
min_font_size	指定词云中字体的最小字号，默认为4号
max_font_size	指定词云中字体的最大字号，默认根据高度自动调节
font_step	指定词云中字体字号的步进间隔，默认为1
font_path	指定字体文件的路径，默认为当前路径
max_words	指定词云显示的最大单词数量，默认为200
stop_words	指定词云的排除词列表，即不显示的单词列表
background_color	指定词云图片的背景颜色，默认为黑色
mask	指定词云形状，默认为长方形，需要引用imread()函数

10.5.2 wordcloud库

generate()方法

generate()方法需要接收一个字符串作为参数，需要注意的是，若generate()方法中的字符串为中文，在创建WordCloud对象时必须指定字体路径。

to_file()方法

to_file()方法用于以图片形式输出词云，该方法接收一个表示图片文件名的字符串作为参数，图片可以为.png或.jpg格式。

imread()方法

matplotlib.image中定义的imread()函数用于加载图片文件，其语法格式如下：

imread(filename, flags=1)

利用imread()函数读取.png格式的图片，wordcloud会根据图片的可见区域生成相应形状的词云。

10.5.3 pygame库

pygame简介

pygame是为开发2D游戏而设计的Python跨平台模块。

开发人员利用pygame模块中定义的接口，可以方便快捷地实现游戏中的一些功能，如：

- 图形用户界面创建
- 图形和图像的绘制
- 用户键盘和鼠标操作的监听
- 播放音频

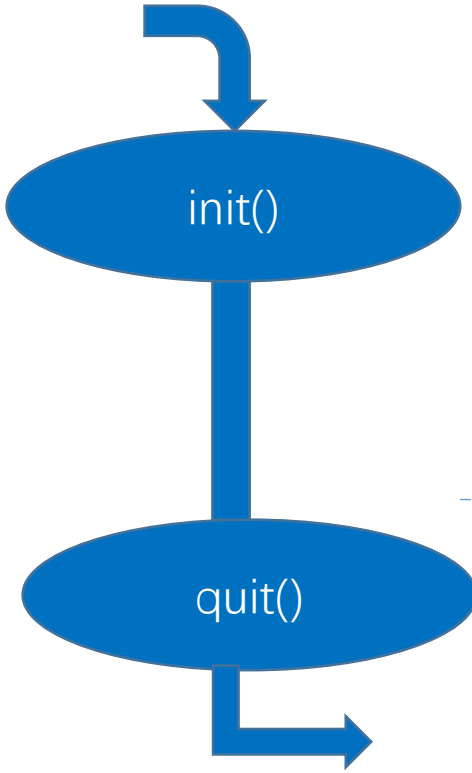
安装pygame

pygame的安装命令：pip install pygame

10.5.3 pygame库

pygame的初始化和退出

- init()：一次性初始化pygame的所有模块
- quit()：可以卸载所有之前被初始化的pygame模块



一次性初始化pygame的所有模块，如：

- 显示模块
- 字体模块
- 混音器模块

卸载所有之前被初始化的pygame模块。

pygame的初始化和退出

示例：

导入pygame模块，并在主函数中实现pygame的初始化和退出。

```
import pygame                          # 导入pygame
def main():
  pygame.init()                        # 初始化所有模块
  pygame.quit()                        # 卸载所有模块
if __name__ == '__main__':
    main()
```

创建游戏窗口

pygame通过display子模块创建图形界面窗口，该子模块中与窗口相关的常用函数如下：

函数	说明
set_mode()	初始化游戏窗口
set_caption()	设置窗口标题
update()	更新屏幕显示内容

10.5.3 pygame库

初始化游戏窗口

set_mode()函数声明如下：

```
set_mode(resolution=(0,0), flags=0, depth=0) -> Surface
```

参数含义：

- resolution：图形窗口的分辨率。本质上是一个元组(宽,高)，单位为像素。默认与屏幕大小一致。
- flags：标志位。用于设置窗口特性，默认为0。
- depth：色深。该参数只取整数，范围为[8,32]。

返回值含义：

- 返回值为Surface对象。
- 可以将Surface对象看作画布，必须先有画布，绘制的图形才能够被呈现。

10.5.3 pygame库

set_mode()函数创建的窗口默认为黑色背景，使用Surface对象的fill()方法可以填充画布，修改窗口颜色。

示例：创建一个窗体，并修改其背景颜色。

```
import pygame                                                    # 导入pygame
WINWIDTH = 640                                                   # 窗口宽度
WINHEIGHT = 80                                                   # 窗口高度
BGCOLOR = (  125,  125,  0)                                      # 预设颜色
def main():
    pygame.init()                                                # 初始化所有模块
    # 创建窗体，即创建Surface对象
    WINSET = pygame.display.set_mode((WINWIDTH, WINHEIGHT))
    WINSET.fill(BGCOLOR)                                         # 填充背景颜色
    pygame.quit()                                                # 卸载所有模块
if __name__ == '__main__':
    main()
```

10.5.3 pygame库

设置窗口标题

set_caption()函数声明如下：

```
set_caption(title, icontitle=None)      -> None
```

参数含义：

- title：用于设置显示在窗口标题栏上的标题。
- icontitle：用于设置显示在任务栏上的程序标题，默认与title一致。

示例：修改程序代码，在其中调用set_caption()函数设置窗口标题。

```
pygame.display.set_caption('小游戏')
```

刷新窗口

实际上前面代码中使用fill()方法填充背景后背景颜色却未改变，正是因为程序中未调用该函数对窗口进行刷新。

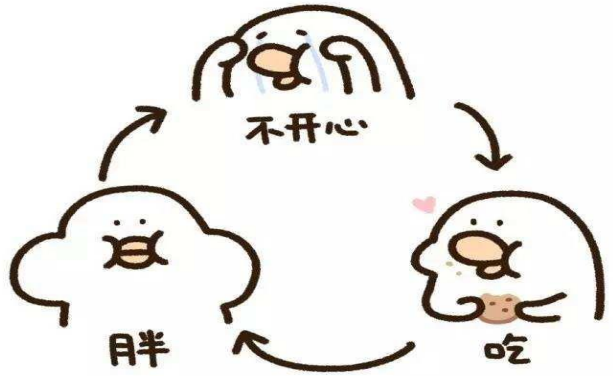
示例：在pygame.quit()语句之前调用update()函数。

```
...
def main():
    pygame.init()                                          # 初始化所有模块
    WINSET = pygame.display.set_mode((WINWIDTH, WINHEIGHT))
    WINSET.fill(BGCOLOR)                                   # 填充背景颜色
    pygame.display.set_caption('小游戏')                    # 设置窗口标题
    pygame.display.update()                                # 刷新窗口
    pygame.quit()                                          # 卸载所有模块
...
```

10.5.3 pygame库

游戏循环

- 游戏启动后一般由玩家手动关闭。
- 若要使游戏保持运行，需要在程序中添加一个无限循环。

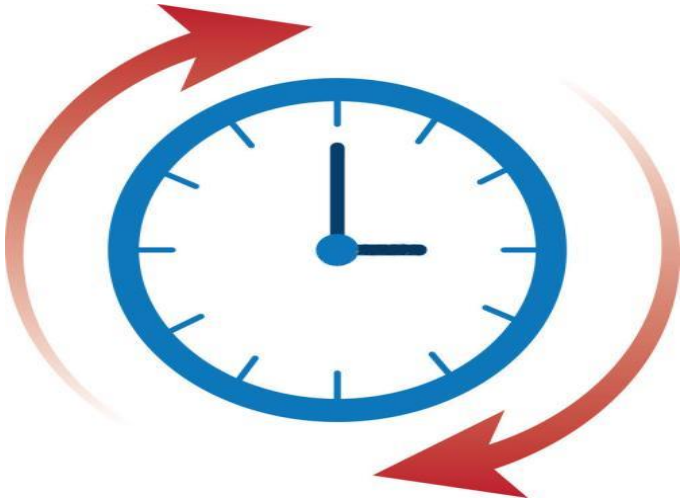


示例：

```
while True:
    pass
```

10.5.3 pygame库

游戏时钟



- 游戏时钟用于控制帧率，利用视觉暂留现象形成动画。
- 帧率>60帧/s，就能实现连续、高品质动画效果。
- 游戏时钟用于解决帧率过高的问题。

Clock类与tick()方法

通过Clock类的tick()方法可以方便地设置游戏循环的执行频率，具体操作如下：

```
FPSCLOCK = pygame.time.Clock()  # 创建Clock对象
FPSCLOCK.tick(FPS)              # 为Clock对象设置帧率
```

示例：修改程序代码，为其添加帧率控制语句。

10.5.3 pygame库

图形和文本绘制

图形化窗口是绘制文本和图形的前提，创建窗口之后方可在其中绘制文本、图形等元素。通过前面的讲解可知，pygame中的图形窗口是一个Surface对象，在窗口中进行绘制实质上就是在Surface对象之上进行绘制。

图形绘制

1. 加载图片
2. 绘制图片

10.5.3 pygame库

1.加载图片

- 加载图片即将图片读取到程序中。
- 通过pygame中image类的load()方法可以向程序中加载图片，生成Surface对象。

load()方法

```
load(filename) -> Surface
```

方法说明：

- 参数filename：被加载的文件名。
- 返回值Surface：load()方法返回一个Surface对象。

示例：使用load()方法加载名为“bg.jpg”的图片。

2.绘制图片

- 绘制图像即将一个Surface对象叠加在另一个Surface对象之上。
- 通过Surface对象的blit()方法可以实现图像绘制。

blit()方法

```
blit(source, dest, area=None, special_flags = 0) -> Rect
```

参数说明：

- source：接收被绘制的Surface对象。
- dest：接收一个表示位置的元组（left,top），或接收一个表示矩形的元组(left,top,width,height)，将矩形的位置作为绘制的位置。
- area：是一个可选参数，通过该参数可设置矩形区域。若设置的矩形区域小于source所设置Surface对象的区域，那么仅绘制Surface对象的部分内容。
- special_flags：标志位。

2.绘制图片

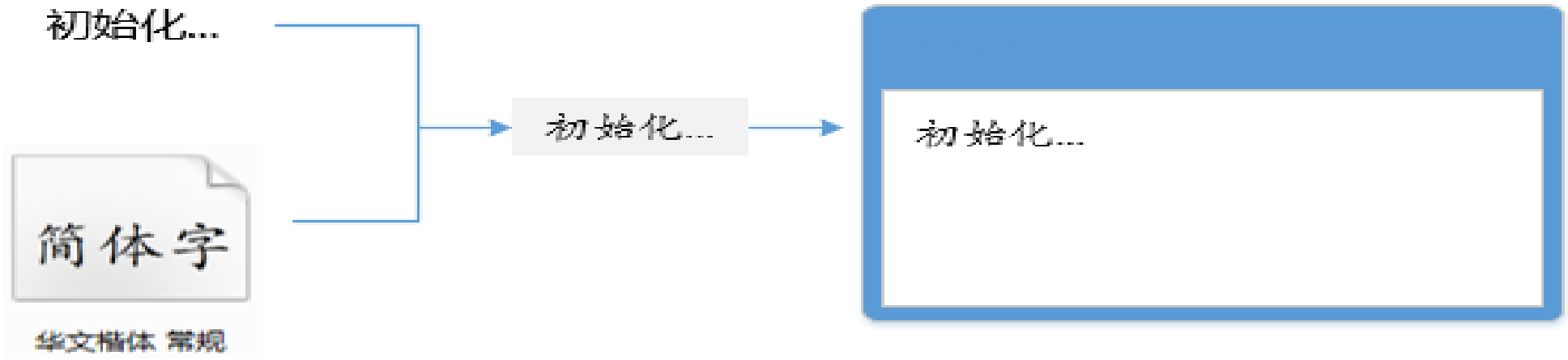
示例： 使用blit()方法将加载生成的imgSurf对象绘制到窗口WINSET中。

10.5.3 pygame库

文本绘制

1. 创建字体对象
2. 渲染文本内容，生成一张图像
3. 将生成的图像绘制到游戏主窗口中

文本绘制实际上也是图片的叠加，只是在绘制之前需要先结合字体，将文本内容制作成图片。



10.5.3 pygame库

1.创建字体对象

Font()函数

调用font模块的Font()函数可以创建一个字体对象，Font()函数的声明如下：

```
Font(filename, size) -> Font
```

参数说明：

- filename：用于设置字体对象的字体。
- size：用于设置字体对象的大小。

示例：

```
BASICFONT = pygame.font.Font('STKAITI.TTF', 25)
```

10.5.3 pygame库

1.创建字体对象

SysFont()函数

调用font模块的SysFont()函数可以创建一个字体对象，SysFont()函数的声明如下：

```
SysFont(name, size, bold=False, italic=False) -> Font
```

参数说明：

- name：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()函数可以获取当前系统的所有可用字体列表。该参数亦可接收字体路径名。
- size：字体大小。
- bold：是否设置为粗体，默认为否。
- italic：是否设置为斜体，默认为否。

1.创建字体对象

Font和SysFont()函数的区别

- SysFont()对系统依赖度较高，Font()则可以在设置字体时将字体文件存储到程序路径中，使用自定义的字体。
- Font()函数更加灵活，也更利于游戏程序的打包和移植。

2.渲染文本内容

render()方法

pygame模块中可通过字体对象的render()方法进行渲染，该方法的声明如下：

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

参数说明：

- text：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：文字颜色。
- background：背景颜色，默认为 None，表示无颜色。

示例：以调用Font()函数生成的字体对象BASICFONT为例，通过render()方法渲染文本内容。

```
YELLOW = (255, 255, 193)                    # 颜色预设
MSGCOLOR = DARKTURQUOISE                    # 设置字体颜色
MSGBGCOLOR = YELLOW                         # 按钮背景颜色
msgSurf = BASICFONT.render('初始化...',True,MSGCOLOR,MSGBGCOLOR)
```

2.渲染文本内容

save()方法

通过image类的save()方法可以将渲染生成的Surface对象作为图片存储到本地，save()方法的语法格式如下：

```
save(Surface, filename) -> None
```

示例：使用save()方法将msgSurf对象保存到本地，并命名为msg.png。

初始化...

设置透明背景

通过image类的save()方法可以将渲染生成的Surface对象作为图片存储到本地，save()方法的语法格式如下：

```
msgSurf = BASICFONT.render('初始化...', True, MSGCOLOR, None)
pygame.image.save(msgSurf, 'msg.png')
```

3.绘制渲染结果

示例： 将创建的文本对象msgSurf绘制到WINSET的(0,0)位置。

10.5.3 pygame库

元素位置控制要点

- pygame图形窗口的坐标体系
- pygame的Rect类
- 位置控制

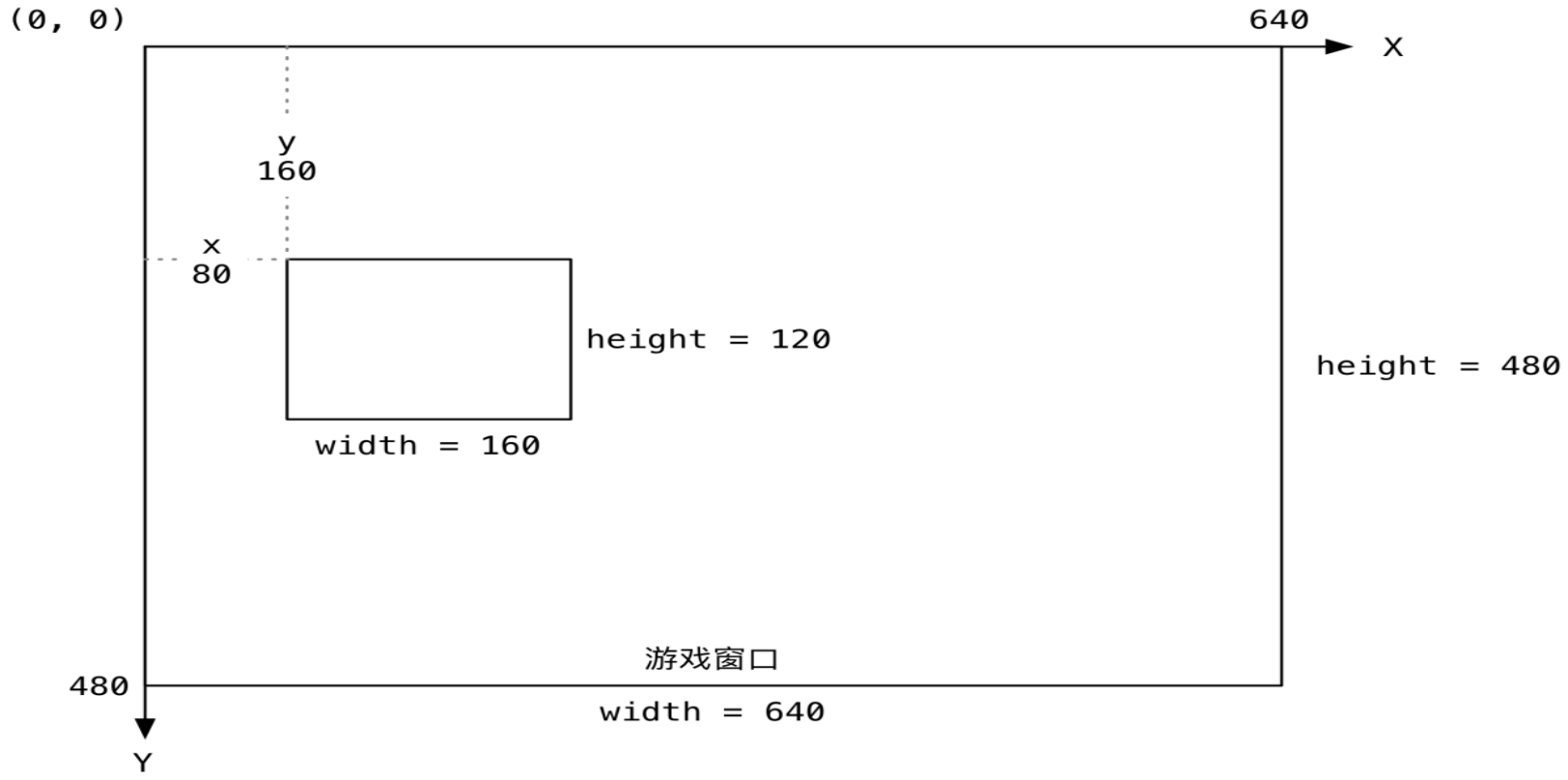
1. pygame图形窗口的坐标体系

pygame图形窗口坐标体系的定义如下：

- 坐标原点在游戏窗口的左上角。
- x轴与水平方向平行，以向右为正。
- y轴与垂直方向平行，以向下为正。

10.5.3 pygame库

示例： 将分辨率为160×120的矩形放置在分辨率为640×480的pygame窗口的(80,160)位置。



2.Rect类

Rect类用于描述、控制可见对象（文本、图片等）在pygame窗口中的位置，该类定义在pygame模块之中，它的构造函数如下：

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

示例： 创建坐标为(80,160)、分辨率为160×120的矩形对象。

```
rect = pygame.Rect(80,160,160,120)
```

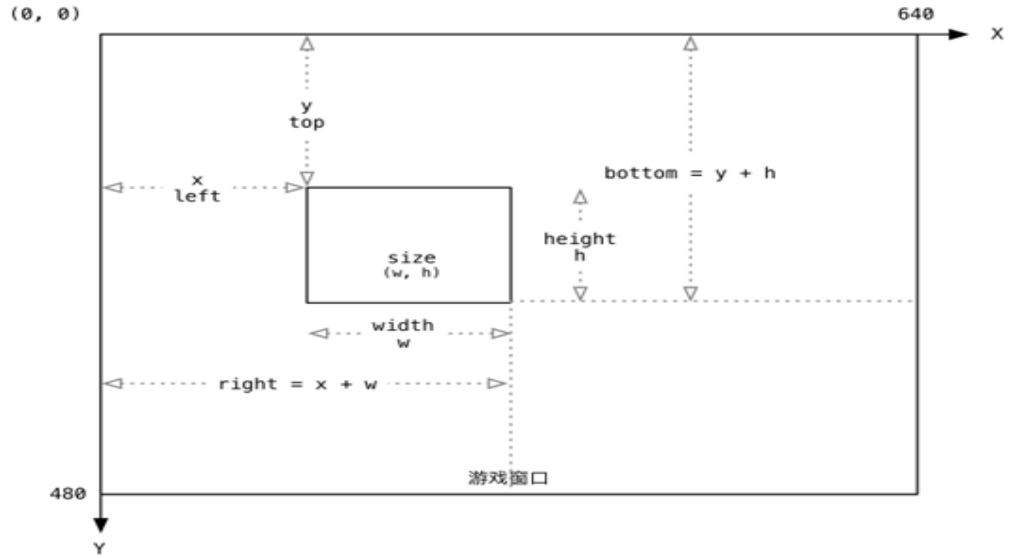
2.Rect类

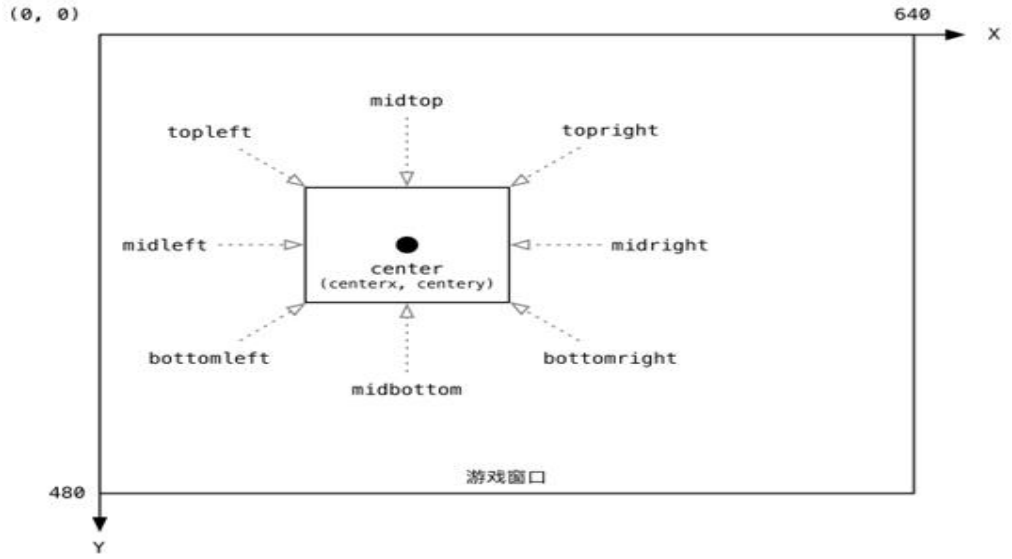
除坐标、宽、高之外，矩形还具有许多用于描述与坐标系相对关系的属性，下面将给出矩形对象的常见属性，并以矩形 rect = Rect(10, 80, 168, 50) 为例对这些属性进行说明，具体如表所示。

属性	说明	示例
x、left	水平方向和 Y 轴的距离	rect.x = 10、rect.left = 10
y、top	垂直方向和 X 轴的距离	rect.y = 80、rect.top = 80
width、w	宽度	rect.width = 168、rect.w = 168
height、h	高度	rect.height = 50、rect.h = 50
right	右侧 = x + w	rect.right = 178
bottom	底部 = y + h	rect.bottom = 130
size	尺寸 (w, h)	rect.size = (168, 50)
topleft	(x, y)	rect.topleft = (10, 80)
bottomleft	(x, bottom)	rect.bottomleft = (10, 130)
topright	(right, y)	rect.topright = (178, 80)
bottomright	(right, bottom)	rect.bottomright = (178, 130)
centerx	中心点 x = x + 0.5 * w	rect.centerx = 94
centery	中心点 y = y + 0.5 * h	rect.centery = 105
center	(centerx, centery)	rect.center = (94, 105)
midtop	(centerx, y)	rect.midtop = (94, 80)
midleft	(x, centery)	rect.midleft = (10, 105)
midbottom	(centerx, bottom)	rect.midbottom = (94, 130)
midright	(right, centery)	rect.midright = (178, 105)

2.Rect类

矩形属性示意图如图所示。





3.位置控制

位置控制方式

- 方式1：将Surface对象绘制到窗口时，以元组(x,y)的形式将坐标传递给参数dest。
- 方式2：使用get_rect()方法获取Surface对象的矩形属性，重置矩形横纵坐标后，再将矩形属性传递给参数dest以设置绘制位置。

示例：以在小游戏窗口右下角的功能绘制“自动”按钮为例，使用第二种位置控制方式在窗口中绘制文本。

10.5.3 pygame库

动态效果原理

动态效果分类

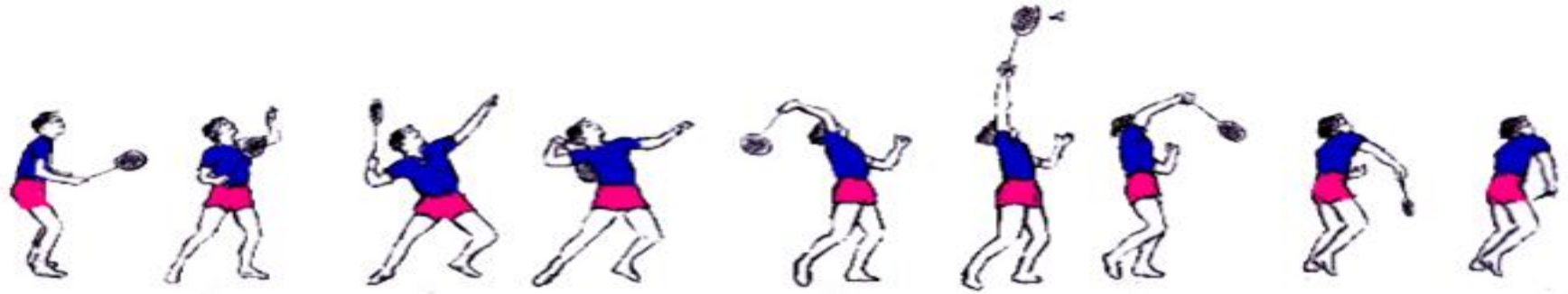
1. 移动。多次修改Surface对象绘制的位置并连续绘制刷新。
2. 动画。在同一位置绘制不同的Surface对象。
3. 移动的动画。连续绘制不同Surface对象的同时，修改绘制的位置。

10.5.3 pygame库

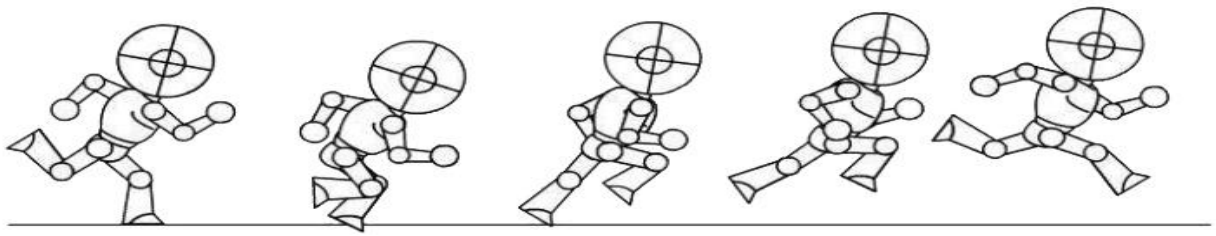
动态效果分类

移动

动画



移动的动画



10.5.3 pygame库

注意：

在实现移动效果之前应先区分动态元素与其他元素，将其他元素作为背景，制作背景的副本覆盖原始窗口，实现动态元素的“消失”，再着手重新绘制要移动的元素并刷新窗口。

copy()方法

pygame的Surface类中定义了copy()方法，使用该方法可以拷贝Surface对象，实现方块的消失。

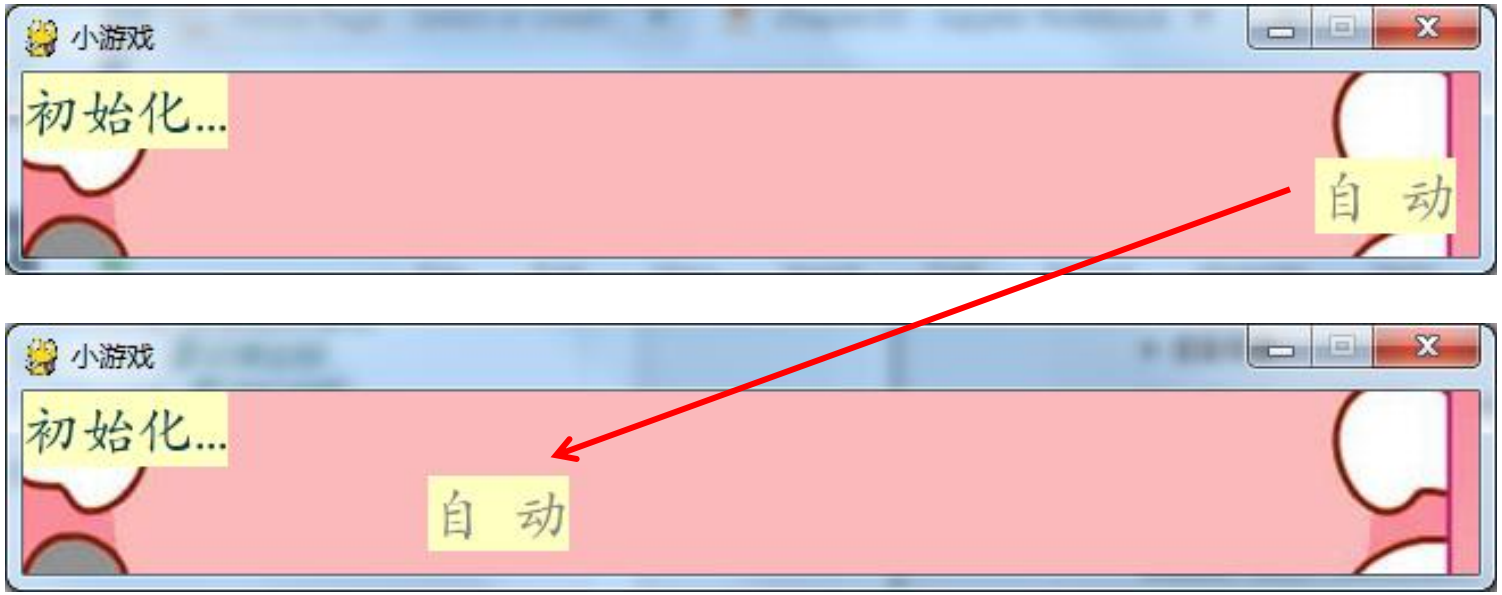
实现步骤

STEP1 在初次向窗口对象WINSET上绘制方块之前，先调用copy()方法创建WINSET的备份。

STEP2 在第二次绘制按钮之前，将备份覆盖到WINSET之上。

10.5.3 pygame库

示例：实现小游戏中“自动”按钮的移动



10.5.3 pygame库

事件

程序开发中将玩家会对游戏进行的操作称为事件（Event），根据输入媒介的不同，游戏中的事件分为：

键盘事件

鼠标事件

手柄事件

10.5.3 pygame库

pygame常见事件列表

pygame在子模块locals中对事件进行了更加细致的定义，键盘事件、鼠标事件及其产生途径和参数如表所示。

事件	产生途径	参数
KEYDOWN	键盘被按下	unicode,key,mod
KEYUP	键盘被放开	key,mod
MOUSEMOTION	鼠标移动	pos,rel,button
MOUSEBUTTONDOWN	鼠标按下	pos,button
MOUSEBUTTONUP	鼠标放开	pos,button

键盘事件

KEYDOWN

KEYUP

参数介绍

- unicode：记录按键的Unicode值。
- key：按下或放开的键的键值（K_xx）。
- mod：包含组合键信息。

10.5.3 pygame库

鼠标事件

- MOUSEMOTION
- MOUSEBUTTONDOWN
- MOUSEBUTTONUP

事件参数介绍

- pos：鼠标操作的位置(x,y)。
- rel：当前位置与上次产生鼠标事件时鼠标位置间的距离。
- buttons：一个含有三个数字的元组，元组中数字的取值只能为0或1，三个数字依次表示左键、滚轮和右键。
- button：整型数值，表示具体操作。

事件相关属性与函数

- pygame.event.type——判断事件类型
- pygame.event.get()——获取当前时刻产生的所有事件的列表

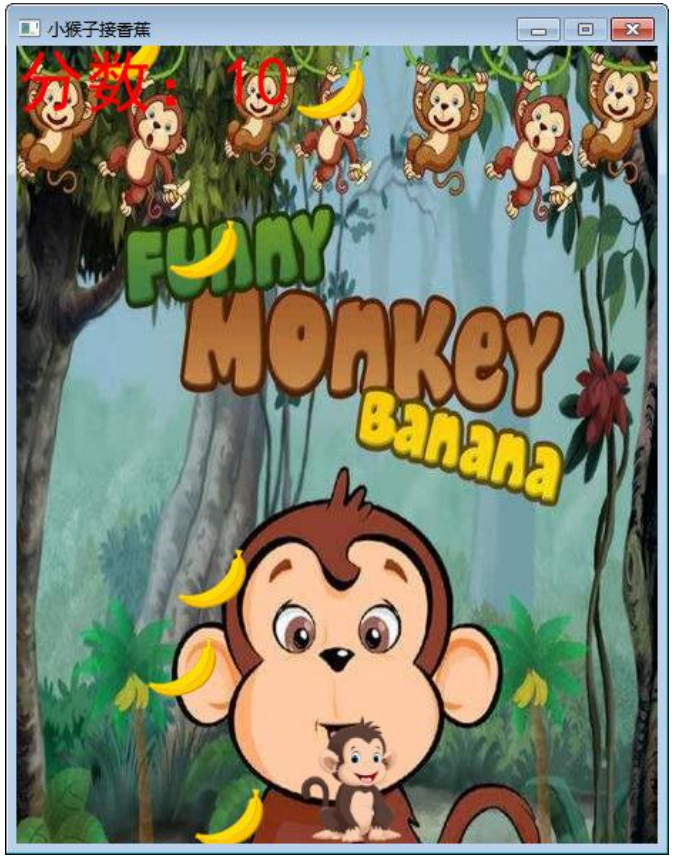
示例： 在程序中添加事件处理代码。

目录页

10.6.1 出场人物统计

《西游记》篇幅巨大、出场人物繁多本实例要求编写程序，统计提取《西游记》小说中的关键人物的出场次数，判断谁是主角。

10.6.2 小猴子接香蕉



小猴子接香蕉游戏是一个根据游戏得分判定玩家反应力的游戏，该游戏的设定非常简单，游戏主体为香蕉和猴子：香蕉从屏幕顶端随机位置出现，垂直落下，玩家用鼠标左右键控制猴子左右移动，接住下落的香蕉，猴子每接到一个香蕉加10分。

本实例要求编写程序，实现一个小猴子接香蕉游戏。

10.7 本章小结

本章简单介绍了Python计算生态、演示了如何构建与发布Python生态库，并介绍了常用的内置Python库和有趣的第三方库，包括time库、random库、turtle库、jieba库、wordcloud库和pygame库。通过本章的学习，希望读者能对Python计算生态涉及的领域所使用的Python库有所了解，掌握构建Python库的方式和random库、turtle库、jieba库的使用，熟悉time库、wordcloud库和pygame库。