理论盘点：数据分析的流程及对应的 Python 库

目录

数据分析对企业的价值

- ★ 用户行为分析可以优化业务、提升用户体验
- ★ 业务数据分析可以提前规避风险、识别机会
- ★ 合理运用数据分析技术，还能为公司内部提高投入产出比（ROI）

数据分析的一般流程

- 数据采集
- 数据挖掘
- 数据可视化

数据分析的一般流程

- 数据采集

数据采集可以基于公司积累的数据，也可以基于公开数据

通常会采用两者结合方式，让数据类别更丰富

采集数据时需遵守《中华人民共和国个人信息保护法》

数据分析的一般流程

- 数据挖掘

- 数据（业务逻辑）理解
- 数据准备
- 建立模型
- 数据清洗
- 数据存储
- 模型评估

数据分析的一般流程

- 数据可视化
  - 建立各类图表
  - 按不同的维度展示图表

数据分析用到的 Python 库

数据采集：requests

数据挖掘：re、BeautifulSoup、pandas、Scikit-learn

数据可视化：matplotlib、Seaborn

总结

1. 数据分析遵循数据采集、数据挖掘、数据展示三个主要过程

2. 数据分析的每个过程均有 Python 库支持，可以通过 Python 实现数据分析的全部技术栈

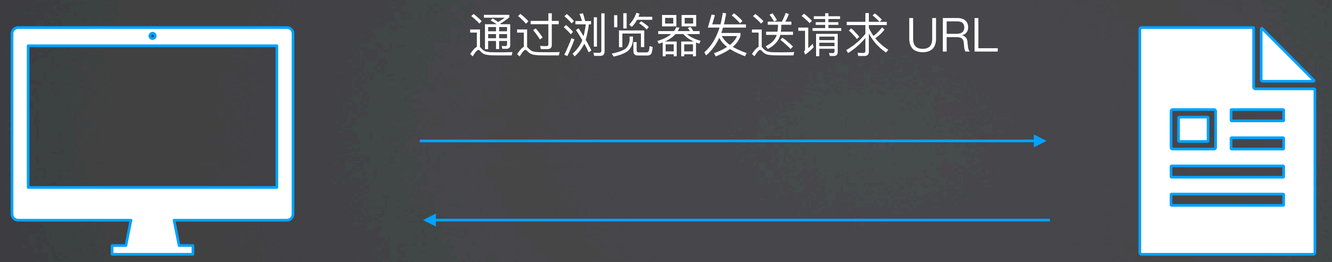
课后作业

请你根据所在地区的气温、天气，使用 Excel 工具绘制一张天气和气温变化图表

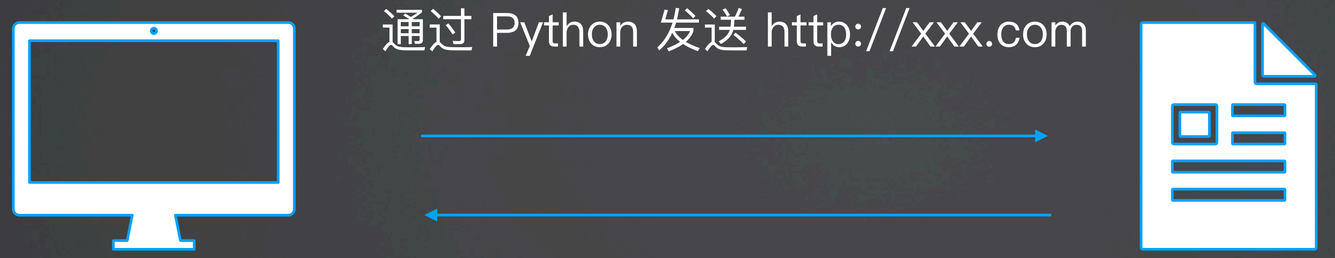
理论盘点：数据采集的方法与 HTTP 协议

目录

基于 HTTP 协议实现数据采集的原理



基于 HTTP 协议实现数据采集的原理



HTTP 协议速查

常见的客户端请求方式：GET、POST

请求和返回结构：HTTP头、HTTP主体、返回码

HTTP 协议速查

```
import requests
r = requests.get('https://time.geekbang.org', auth=('user', 'pass'))
r.status_code #200
r.headers['content-type'] #'application/json; charset=utf8'
r.encoding # ’utf-8’
r.text #’{"authenticated": true, …'
r.json() #{'authenticated': True, ...}
```

网页文字的提取

- 通过 r.text 可以提取网页的内容，*大部分和使用浏览器查看源代码相同
- 网页包含样式和数据两类内容，要想实现文字内容的提取，必须将样式剔除

*注：部分网页使用了特定浏览器访问机制

网页图片的提取

- 网页中的图片采用 <img> 标签存放
- 为了方便管理，多张图片在网页源代码中存放时也有特定的规律

总结

1. 在 HTTP 主体中，数据采集文字和图片一般通过 GET 方式

2. 获取时可以借助浏览器调试工具分析，获取多段文字或多张图片时，应观察网页源代码的规律，采用 for 循环进行处理

课后作业

请使用 requests 库采集一周内的天气（气温 + 降水），存入文本文件中。

理论盘点：任务的并行执行原理

目录

并行下载原理

前提：

- 服务器上的网页能够支持并行访问
- 根据 HTTP 协议的规定，每次请求相互独立，互不影响
- 客户端能够支持并行展示或存储

多进程并行下载网页中的图片

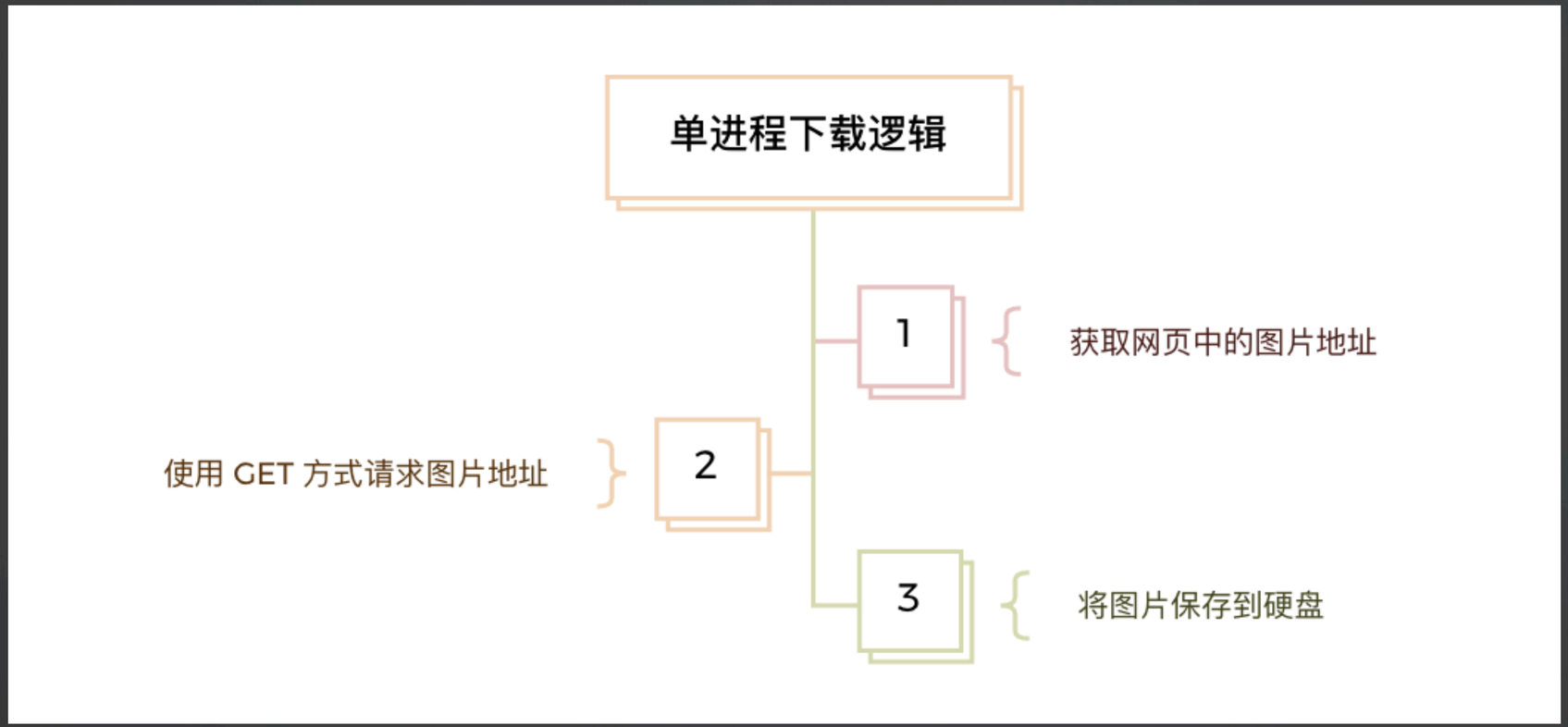
- 并行下载模型应当控制并行程序的个数
- 避免造成服务器或客户机因资源消耗过大，出现服务不可用的情况

✅ 推荐进程池模型

❎ 不推荐基于采集的图片数量，设置并发数量

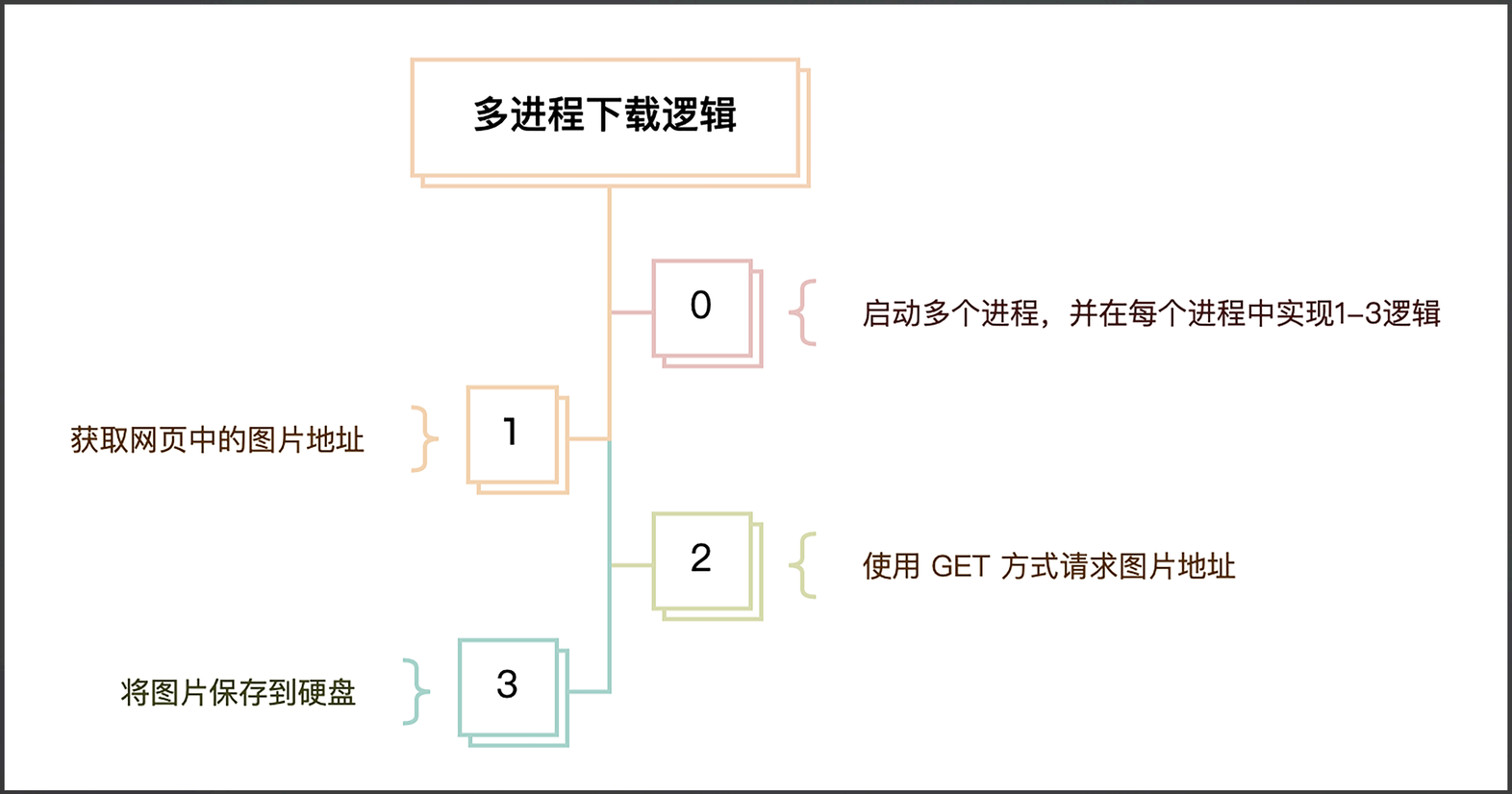
多进程并行下载网页中的图片

单进程下载逻辑



多进程并行下载网页中的图片

多进程下载逻辑



多线程并行下载图片

- 多线程并行下载，同样推荐线程池模型
- 将代码改为多线程下载后，分别比较单进程、多进程和多线程下载程序的运行时间
- ★ 结论：多进程和多线程比单进程下载速度快；但多进程和多线程之间没有明显差别

总结

1. 并行下载图片要比单进程方式下载更快，将多进程和多线程模型用于数据采集有助于提高工作效率

2. 并行下载图片可以使用多进程和多线程模型，它们在图片并行下载的业务场景中性能没有差异

课后作业

在演示将单进程下载图片应用改为多进程下载图片应用时，我将多进程模型应用到了下载和保存文件两个函数中。

请你根据今天所学的内容判断，如果将下载和文件保存两个函数分别放在两个进程池中，能否提高下载效率?

请你通过代码实现并观察执行时间是否有变化。

理论盘点：非规范化数据处理的基础与正则表达式

目录

非规范化数据分类

- 完全无格式数据
- 有特定规律的数据：json、以特定符号分隔的数据
- 规范数据：数据中有空数据、空格、特殊字符、乱码

类型转换库

- 常见的结构化文本格式有 json 和 xml 两种
  - 可以采用 json.dumps() 将 Python 对象编码成 JSON 字符串
  - 可以采用 json.loads 将已编码的 JSON 字符串解码为 Python 对象
  - 可以采用 xml.etree 包对 xml 格式进行解析

正则表达式工作过程

原始字符串 AABBCC

目标字符串 AACC

正则表达式 BB

正则表达式工作过程

- 更复杂的情况：

原始字符串 AAB…BCC

目标字符串 AACC

正则表达式 B+

正则表达式工作过程

元字符

- 正则表达式包含了很多用于特殊的匹配功能符号，称作元字符
- 常用的有：

. 1个字符

* 0个或多个字符

？0个或1个字符

+ 多个字符

() 分组符号

利用正则表达式处理非规范数据

- Python 使用 re 库实现正则表达式的匹配操作，并在匹配指定字符后，能够实现数据的查找和删除功能

```
import re
re.split(r'\W+', 'Words, words, words.', 1)  # ['Words', 'words, words.']
```

- 更复杂的功能数据处理可以使用：

```
re.sub(pattern, repl, string, count=0, flags=0)
```

总结

1. 数据处理工作中，如果数据有 json、xml 等规范格式时，可以使用 Python 的同名库处理数据

2. 当数据中出现有规律的分隔符，但无法之间处理时，可以使用正则表达式库 re 实现内容的拆分和分隔符替换

课后作业

已知有一大批原始数据为以下格式：

string = ' "{" hello ":" world "}" '

请根据这节课学习的正则表达式和相关的库，将这个数据转换为 Python 的字典并在终端进行输出。

如何获取网页上的数据并存储到文件?

目录

文本文件形式保存数据

- Python 一般通过 HTTP 协议的 GET 方式抓取网页的数据
- 这些被访问的原始数据往往用于多次数据分析，需要保存到磁盘
- 文本文件、数据库都可以存放抓取的数据

★ 文本文件的特点：存储简单、稳定但不易搜索

★ 数据库的特点：接口丰富，方便查找但操作比文本文件复杂

文本文件形式保存数据

- 使用文本文件存储数据的两种常见逻辑：

1. 使用 for 循环迭代网页上的元素，利用元素+时间作为每个采集到的数据的文件名
2. 采用追加模式将网站的数据写入到一个文件中

* 注意：并发写入会导致文件数据丢失

数据库形式保存数据

- MySQL 能够比文本文件提供更友好的开发接口，也能提供较好的查询性能
- 要想使用 MySQL 数据库，必须要进行数据库安装、启动服务、初始化服务这三步操作

数据库形式保存数据

以 MySQLdb 库为例，要想连接到 MySQL 数据库，你需要使用如下参数：

```
conn = MySQLdb.connect(host="127.0.0.1", port=3306, user="root", passwd="root", db="test")
```

host MySQL	服务器所在 IP
port MySQL	服务器默认端口
user MySQL	服务器的用户名
password MySQL	服务器的用户密码
db MySQL	服务器的数据库

数据库形式保存数据

以采集到的城市气温为例，可以采用以下格式存放：

城市	日期	气温（摄氏度）
Beijing	20220620	30
Shanghai	20220620	32
Beijing	20220621	31
Guangzhou	20220620	33

数据库形式保存数据

设计数据库的几个小问题：

1. 为了找到唯一的记录，需要为之前的表设计唯一的 ID
2. 为了方便并发写入数据库，需要将数据库查询操作封装为函数或类
3. 数据库写入后，需要提交才能“生效”，并发写入后提交操作是容易忽略的一个关键步骤

总结

1. 使用 HTTP 协议可以获取网页的数据，由于需要从各个维度分析数据，需要保存到本地磁盘

2. 数据可以采用文本文件和数据库两种形式保存，各有利弊。数据量较大时，建议采用数据库存储数据

课后作业

请你编写程序，将课程演示的表结构（ID，城市，日期，温度）改为：ID, 城市，日期温度三个字段，并将原有表中的数据按照新的格式存放到新的表中。例如：

原格式:

1，北京，20220620，30

2, 北京，20220621，28

新格式：

1，北京，{"20220620":30, "20220621":28 }

小试牛刀：如何将数据进行图形化展示？

目录

图形化展示的意义

一组简单的数据：25 28 30 31 31 29 26 24

你能看出它们的规律吗?

图形化展示的意义

再来一组数据：

25 28 30 31 31 29 26 24 25 28 30 31 31 29 26 24

25 28 31 33 32 27 23 22 24 28 30 31 32 26 25 24

如果没有第一组数据，你还能否发现数据的规律呢?

如果是二维、三维甚至更高维度的数据呢?

图形化展示的意义

- 通过图形化更容易找到规律
- 图形化更加直观

图形化库 Matplotlib

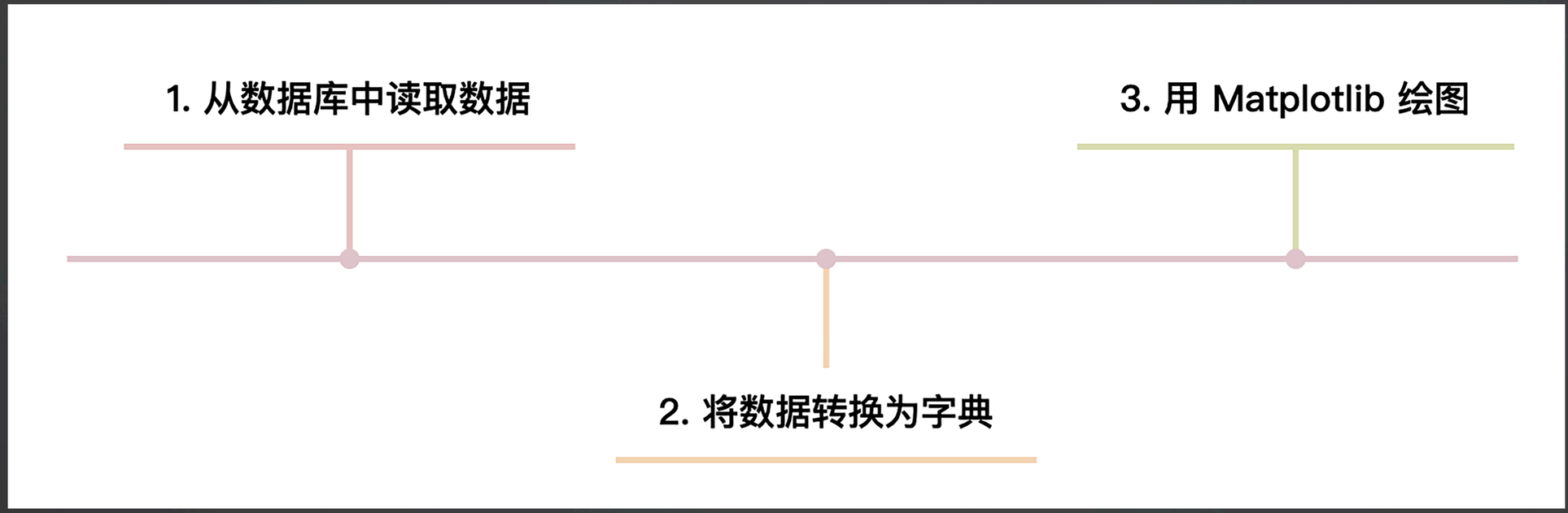
- Matplotlib 是 Python 里最常用的 2D 绘图库
- 主要由画布、坐标系、坐标轴组成
- 一块画布上可以有一个或多个坐标系，每个坐标系上面有一个坐标轴

图形化库 Matplotlib

- Notebook 不会像 Python 终端一样自动展示图表
- 如果你用 Notebook 运行 Matplotlib 库，需要在绘图时增加函数 figure.show()

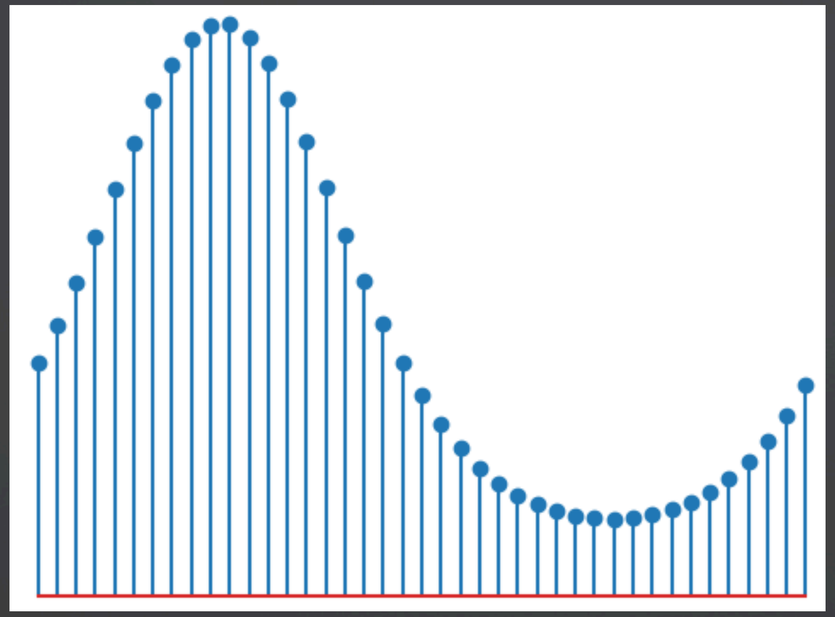
展示天气数据

基本流程



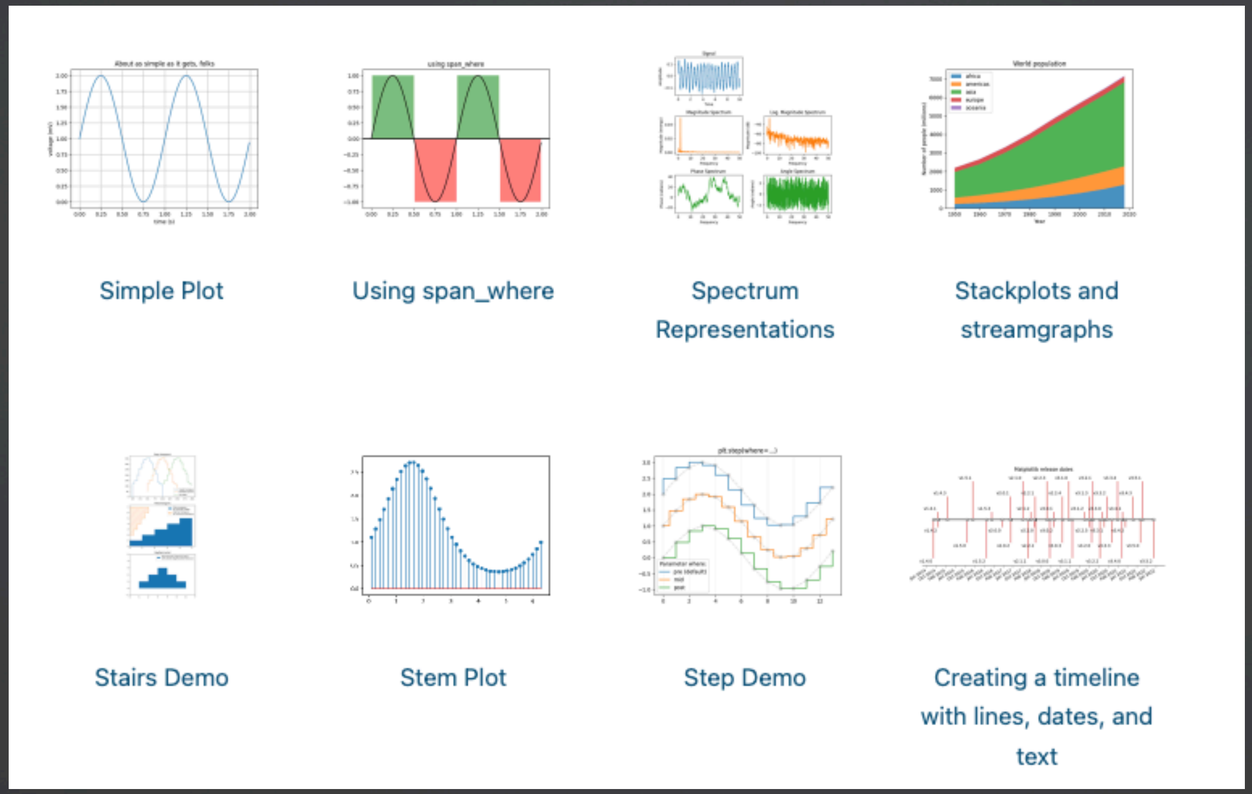
展示天气数据

基于该城市当年每月气温变化，绘制图表如下：



展示天气数据

更多图表样式参考：https://matplotlib.org/stable/gallery/index



总结

1. 图形化可以更直观地展示数据，从而更容易发现数据中的规律
2. Matplotlib 是 Python 最常用的图形化展示工具

课后作业

请你编写程序，利用 Matplotlib 展示北京、上海两个城市一周内的气温变化曲线。

THANKS