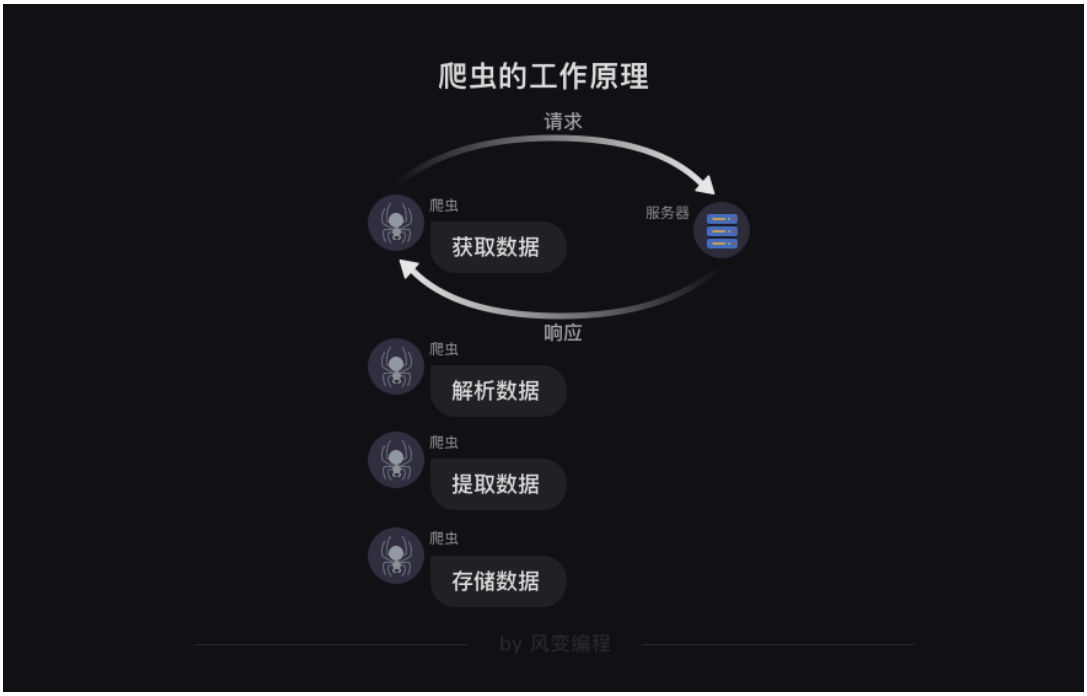
第 15 关、复习

1、爬虫总复习

（1）爬虫四步

获取数据（包含请求和响应两个动作）、解析数据、提取数据、存储数据。



（2）最简请求方式：requests.get()

```
import requests

url = ''
response = requests.get(url)
```

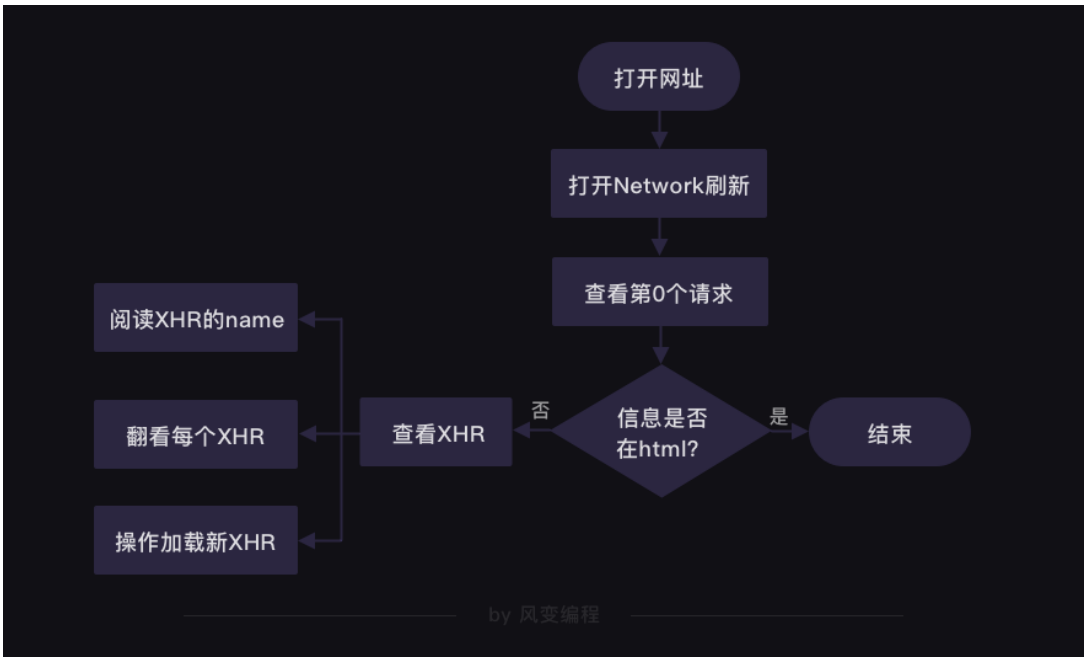
1-1、工具

1-1-1、Network

Network 能够记录浏览器的所有请求。我们最常用的是：ALL（查看全部）/XHR（仅查看XHR）/Doc（Document，第0个请求一般在这里），有时候也会看看：Img（仅查看图片）/Media（仅查看媒体文件）/Other（其他）。最后，JS 和 CSS，则是前端代码，负责发起请求和页面实现；Font 是文字的字体；而理解 WS 和 Manifest，需要网络编程的知识，倘若不是专门这个，我们不需要了解。

1-1-2、XHR和Doc

我们能在 Doc 里找到一个网页的源代码，而在网页源代码里找不到的信息，通常都能在 XHR 里找到，XHR 帮我们实现了异步请求。



1-2、解析与提取（一）

BeautifulSoup，它能提供一套完整的数据解析、数据提取解决方案。

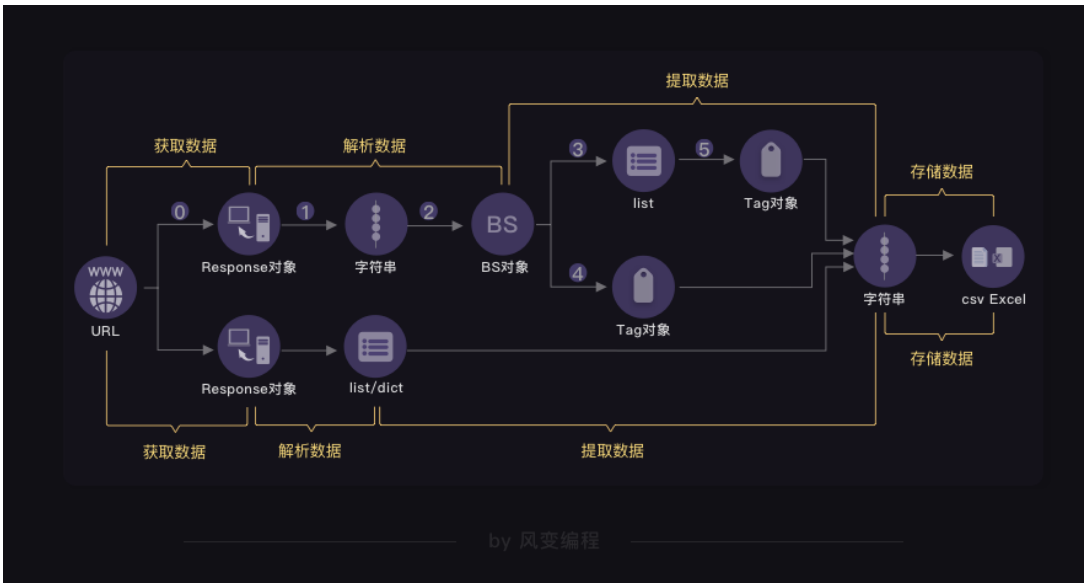
当 response.text 自动解码出问题，可使用 response.encoding='' 来对编码进行修改。

1-3、解析与提取（二）

XHR 所传输的数据，最重要的一种是用 json 格式写成的，和 html 一样，这种数据能够有组织地存储大量内容。json 的数据类型是“文本”，在 Python 语言当中，我们把它称为字符串。我们能够非常轻易地将 json 格式的数据转化为列表/字典，也能将字典/列表转为 json 格式的数据。

1-3-1、解析 json 数据

1-3-2、提取数据



1-4、更厉害的请求

1-4-1、params

可以让我们带着参数来请求数据：我想要第几页？我想要搜索的关键词？我想要多少个数据？

1-4-2、headers

请求头。它告诉服务器，我的设备/浏览器是什么？我从哪个页面而来？

1-4-3、post

post 区别于 get 的是：get 是明文显示参数，post 是非明文显示参数。

1-4-4、cookies

cookies，的作用是让服务器“记住你”，当下一次，浏览器带着cookies访问博客，服务器会知道你是何人，你不需要再重复输入账号密码，就能直接访问。

1-5、存储

最常见两种存储数据的方法：csv 和 excel。

1-5-1、csv

（1）csv 写入步骤

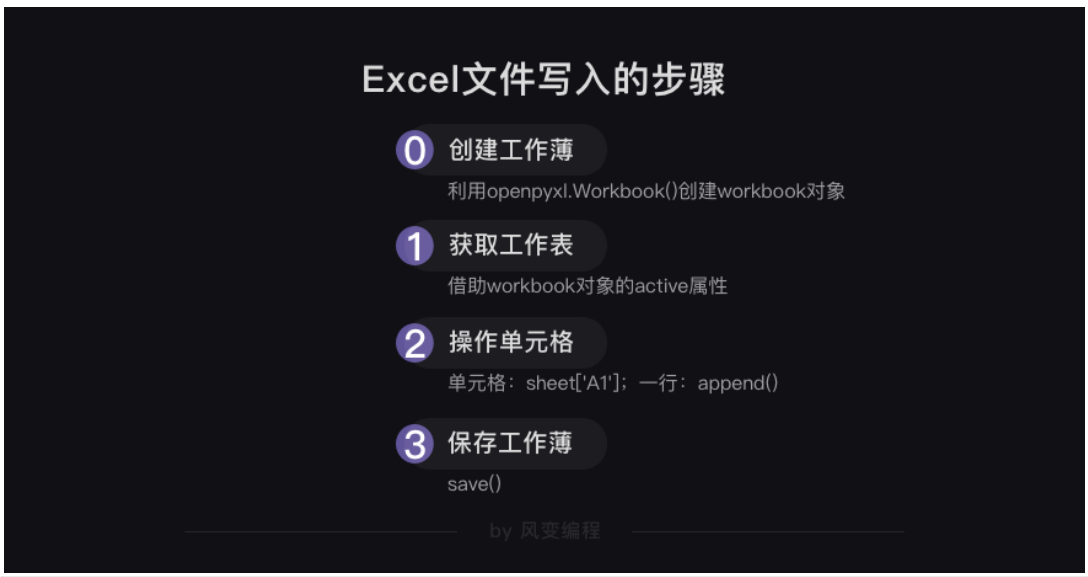

```
#csv写入的代码：

import csv
csv_file=open('demo.csv','w',newline='')
writer = csv.writer(csv_file)
writer.writerow(['电影','豆瓣评分'])
csv_file.close()
```

（2）csv 读取步骤


```
#csv读取的代码：

import csv
csv_file=open('demo.csv','r',newline='')
reader=csv.reader(csv_file)
for row in reader:
    print(row)
```

1-5-2、Excel

（1）Excel 写入步骤




```
#Excel写入的代码：

import openpyxl
wb=openpyxl.Workbook()
sheet=wb.active
sheet.title='new title'
sheet['A1'] = '漫威宇宙'
rows= [['美国队长','钢铁侠','蜘蛛侠','雷神'],['是','漫威','宇宙', '经典','人物']]
for i in rows:
    sheet.append(i)
```

```
print(rows)
wb.save('Marvel.xlsx')
```

（2）Excel 读取步骤


```
#Excel读取的代码：

import openpyxl
wb = openpyxl.load_workbook('Marvel.xlsx')
sheet=wb['new title']
sheetname = wb.sheetnames
print(sheetname)
A1_value=sheet['A1'].value
print(A1_value)
```

1-6、更多的爬虫

多协程，是一种非抢占式的异步方式。使用多协程能让多个爬取任务用异步的方式交替执行。

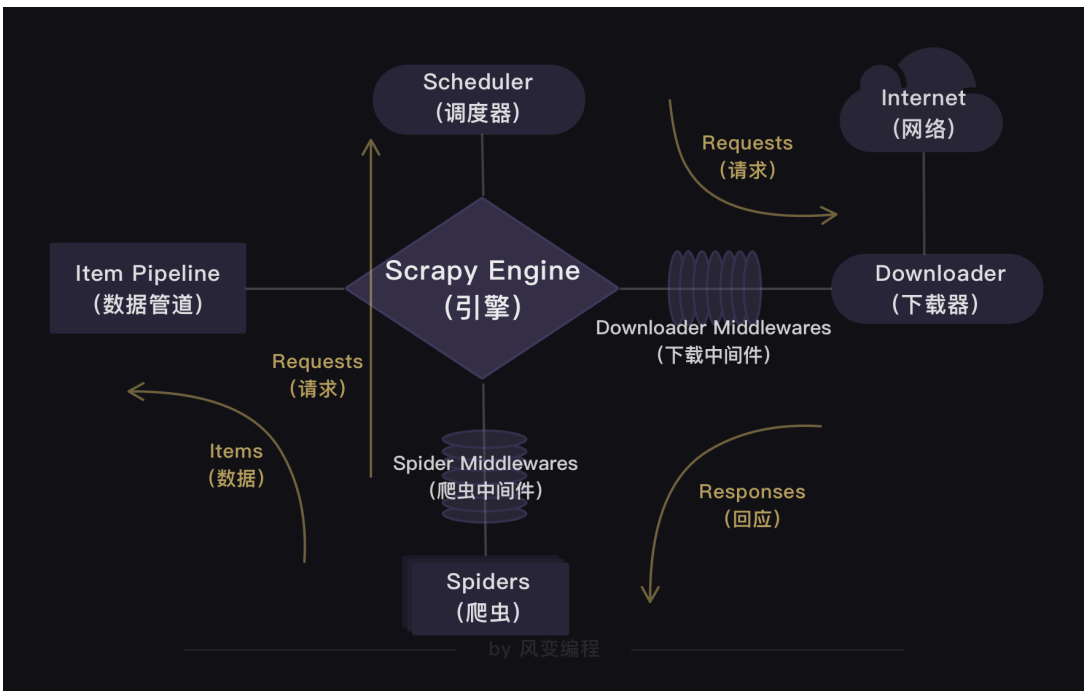
示例代码：

```
import gevent,time,requests
from gevent.queue import Queue
from gevent import monkey
monkey.patch_all()

start = time.time()

url_list = ['https://www.baidu.com/',
'https://www.sina.com.cn/',
'http://www.sohu.com/',
'https://www.qq.com/',
'https://www.163.com/',
'http://www.iqiyi.com/',
'https://www.tmall.com/',
'http://www.ifeng.com/']

work = Queue()
for url in url_list:
    work.put_nowait(url)

def crawler():
```

```
    while not work.empty():
        url = work.get_nowait()
        r = requests.get(url)
        print(url,work.qsize(),r.status_code)

tasks_list  = [ ]

for x in range(2):
    task = gevent.spawn(crawler)
    tasks_list.append(task)
gevent.joinall(tasks_list)

end = time.time()
print(end-start)
```

1-7、更强大的爬虫 — 框架

1-7-1、Scrapy 结构



1-7-2、Scrapy 工作原理

1-7-3、Scrapy 用法

1-8、给爬虫加上翅膀

三个有力工具：selenium，邮件通知和定时。

1-8-1、selenium

（1）selenium 提取数据

（2）对象转换

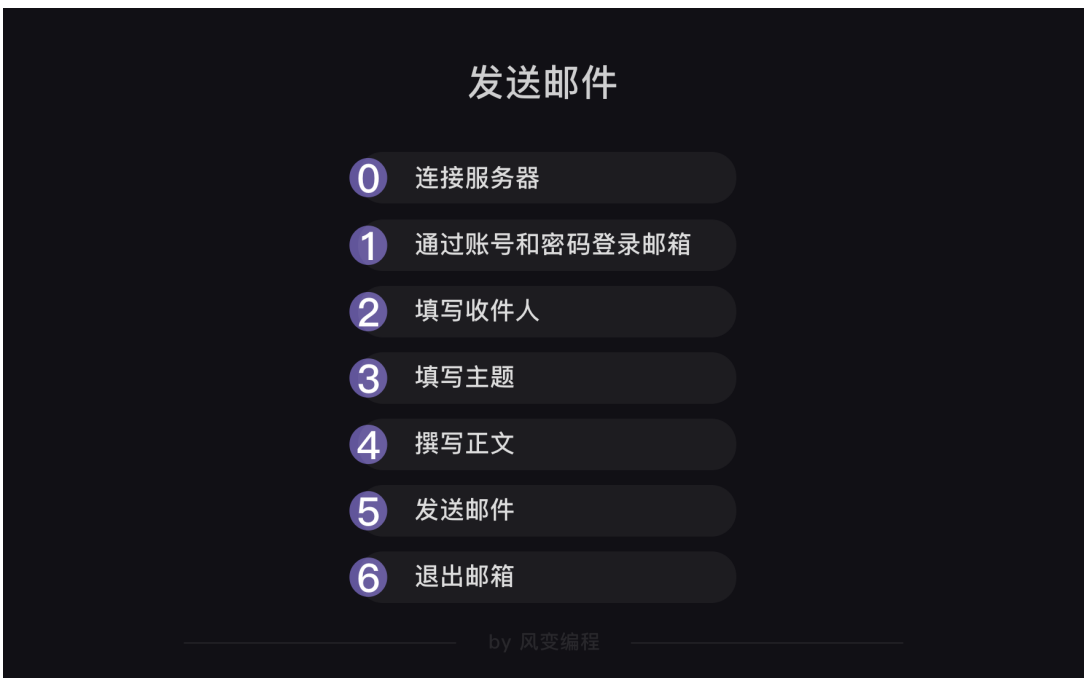
（3）搭配 BeautifulSoup 解析提取数据

前提是先获取字符串格式的网页源代码：

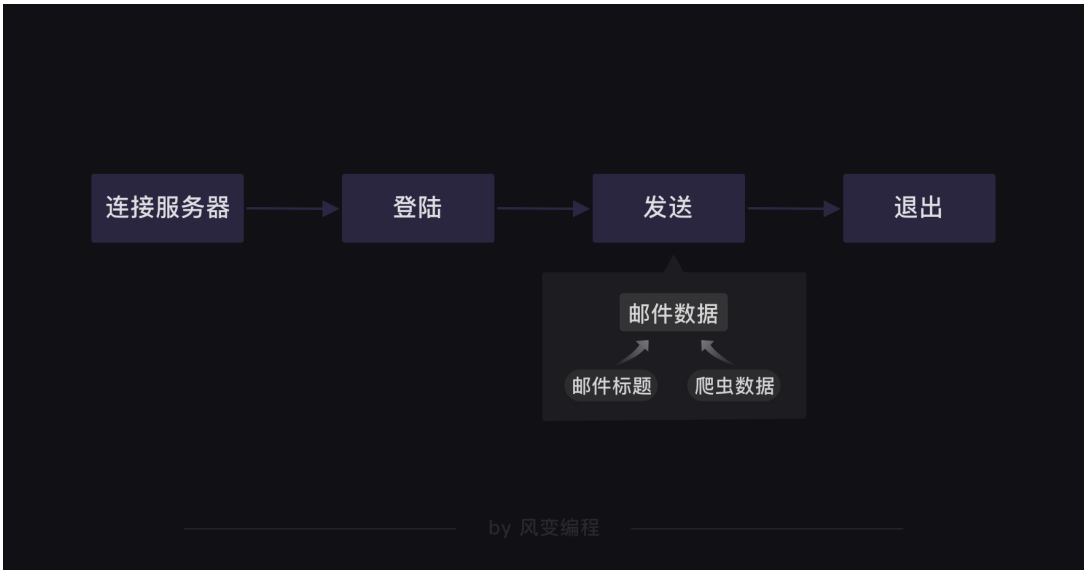
```
HTML源代码字符串 = driver.page_source
```

（4）自动操作浏览器的方法

1-8-2、邮件通知

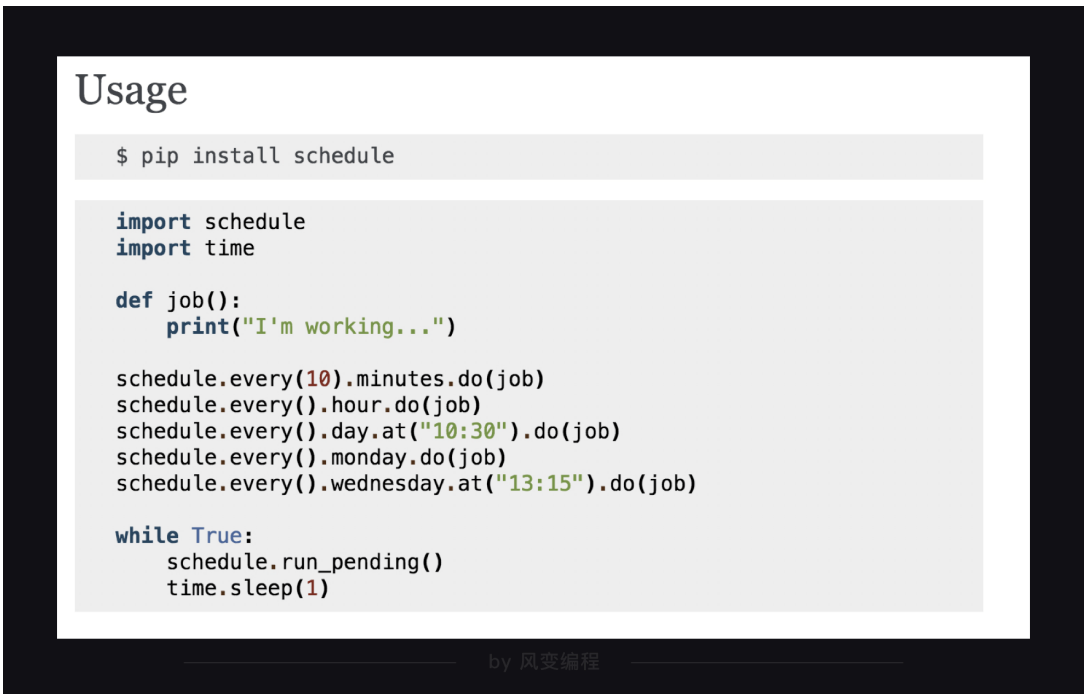


所需模块：smtplib 和 email。前者负责连接服务器、登录、发送和退出的流程；后者负责填输邮件的标题与正文。



1-8-3、定时

schedule 模块：



示例代码：

```
import schedule
import time
#引入schedule和time

def job():
    print("I'm working...")
#定义一个叫job的函数，函数的功能是打印'I'm working...'

schedule.every(10).minutes.do(job)        #部署每10分钟执行一次job()函数的任务
schedule.every().hour.do(job)             #部署每×小时执行一次job()函数的任务
```

```
schedule.every().day.at("10:30").do(job) #部署在每天的10:30执行job()函数的任务
schedule.every().monday.do(job)           #部署每个星期一执行job()函数的任务
schedule.every().wednesday.at("13:15").do(job)#部署每周三的13：15执行函数的任务

while True:
    schedule.run_pending()
    time.sleep(1)
#13-15都是检查部署的情况，如果任务准备就绪，就开始执行任务。
```