1.HTTP:超文本传输协议

·以明文的形式传输

·效率更高，但是不安全

2.HTTPS:HTTP + SSL(安全套接字层)

·传输之前数据先加密，之后解密获取内容,效率较低，但是安全。

·get请求和post请求的区别:

·get请求没有请求体，post有，get请求把数据放到url地址中

·post请求常用于登录注册

·post请求携带的数据量比get请求大，多，常用于传输大文本的时候

3.HTTP协议之请求

1.请求行

2.请求头

·User-Agent: 用户代理:对方服务器能够通过user\_agent知道当前请求对方资源的是什么浏览器

·如果我们需要模拟手机版的浏览器发送请求，对应的,就需要user\_agent改成手机版

·Cookie: 用来存储用户信息的，每次请求会被携带上发送给对方的服务器

用于：要获取登录后才能访问的页面

对方的服务器会通过cookie来判断是我们是一个爬虫。

4.请求体

·携带数据

·get请求没有请求体

·post请求有请求体

5.HTTP协议之响应

·响应头

Set-Cookie:对方服务器通过该字段设置cookie到本地。

·响应体

url地址对应的响应

6.使用前requests模块：

pip install requests

#get,post请求

response = requests.get(url) #发送get请求

response = requests.post(url,data={请求体字典) #发送post请求

#response的方法

response.text

该方式往往会出现乱码，出现乱码使用response.encoding= "utf-8 "

response.content.decode() #把响应的二进制字节流转化为str类型

resposne.request.url #发送请求的url地址

response.url #response # response响应的url地址

resposne.request.header #请求头

resposne.headers #响应请求

#获取网页源码的正确方式(通过下面三种方式一定能获取到网页的正确解码方式)

1． response.content.decode()

实例：

import requests

url = "http://www.baidu.com"

response = requests.get(url)

print(response.content.decode())

2. response.content.decode("gbk ")

实例：

import requests

url = "http://www.baidu.com"

response = requests.get(url)

#获取网页的html字符串

response.encoding = "utf-8"

print(response.text)

3. response.text

实例：

import requests

url = "http://www.baidu.com"

response = requests.get(url)

print(response.text)

#发送带header的请求

为了模拟浏览器，获取浏览器一摸一样的内容

headers = {"User-Agent":" Mozilla/5.0 (Linux; Android 8.0.0; Pixel 2 XL Build/OPD1.170816.004) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/65.0.3325.181 Mobile Safari/537.36 "}

###超时参数

##使用超时参数

requests.get(url.headers=headers,timeout=3) #3秒之内必须返回响应，否则报错

## retrying模块的学习

pip install retrying

##处理cookie相关的请求

1. 直接携带cookie请求url地址
   1. cooki放在headers中

headers = {“User-Agent”:”…”,”Cookie”:”字符串”}

1. cookie字典传给cookie参数

requests.get(url,cookies=cookie\_dict)

先发送post请求，获取cookie，带上cookie请求登陆的页面

session = requests.session()#会话保持,seession具有的方法和requests一样

session.post(url,data,headers) #服务器设置在本地的cookie会保存在session中

session.get(url)#会带上之前保存在session中cookie，能够请求成功

数据提取方法：

### json

数据交换格式，看起来像python的字符串

使用json前导入 import json

哪里会返回json数据

浏览器切换为手机版

抓包app

json.loads

把json字符串转化为python类型

json.loads(json字符串)

json.dumps

把python类型转化为json字符串

json.dumps({})

json.dumps(ret1,ensure\_ascii=False,indent=2)

ensure\_ascii=False,indent=2

ensure\_ascii :让中文显示中文

indent：能够让下一行在上一行的基础上空格