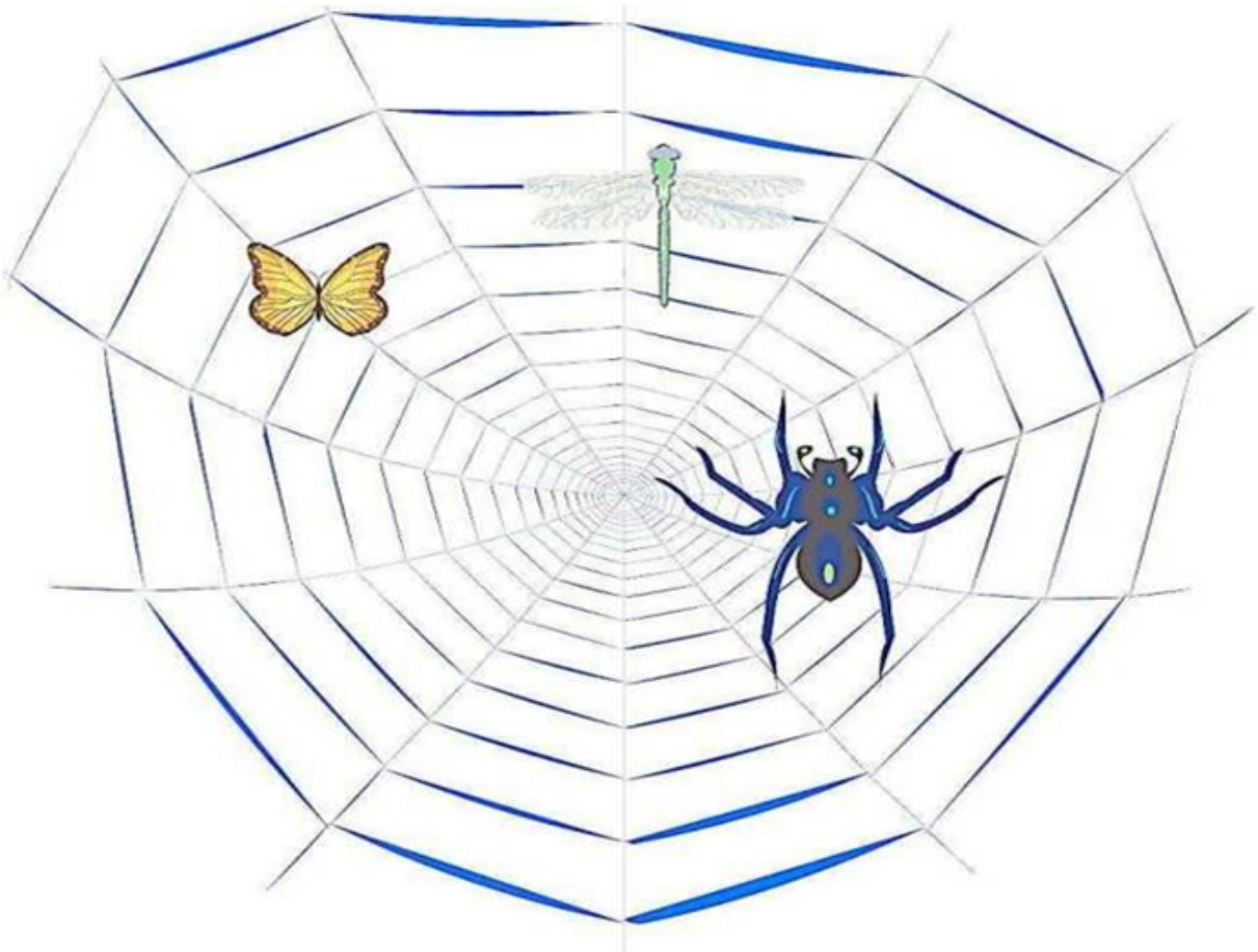
Urllib

1.什么是互联网爬虫?



如果我们把互联网比作一张大的蜘蛛网，那一台计算机上的数据便是蜘蛛网上的一个猎物，而爬虫程序就是一只小蜘蛛，沿着蜘蛛网抓取自己想要的数据

```
解释1：通过一个程序，根据Url(http://www.taobao.com)进行爬取网页，获取有用信息
解释2：使用程序模拟浏览器，去向服务器发送请求，获取响应信息
```

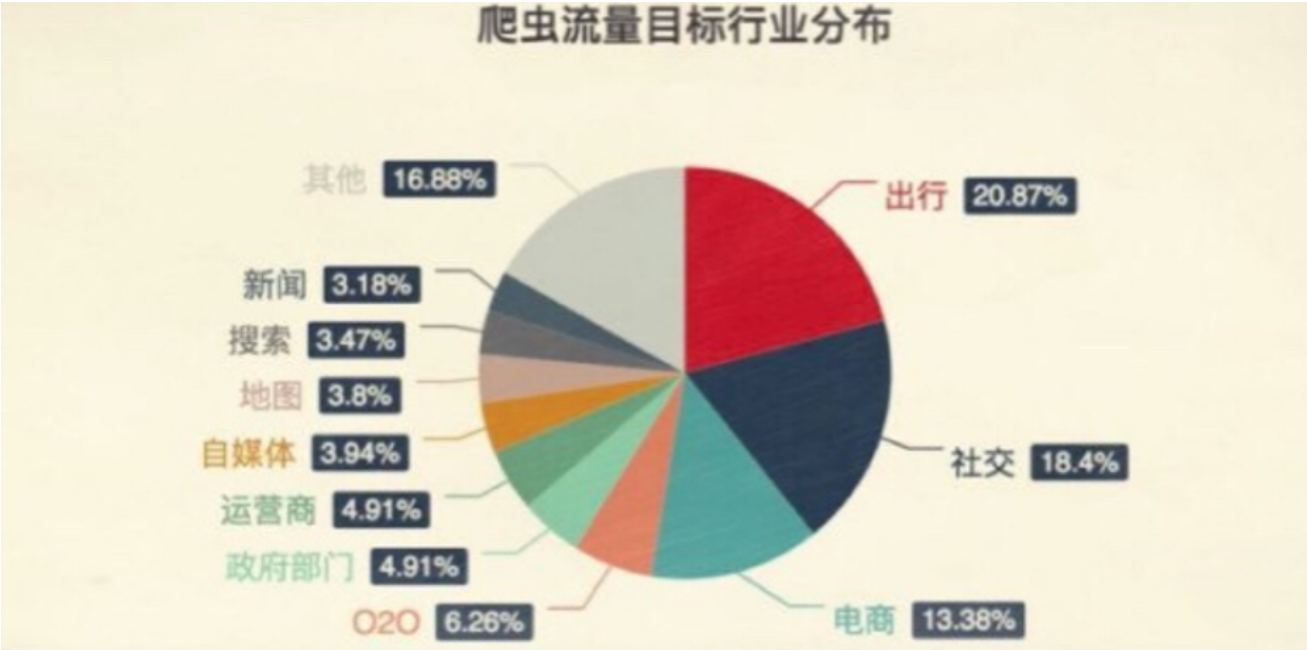
2.爬虫核心?

```
1.爬取网页：爬取整个网页  包含了网页中所有得内容
2.解析数据：将网页中你得到的数据 进行解析
3.难点：爬虫和反爬虫之间的博弈
```

3.爬虫的用途?

- 数据分析/人工数据集
- 社交软件冷启动
- 舆情监控

- 竞争对手监控



4.爬虫分类?

通用爬虫:

实例

百度、360、google、sougou等搜索引擎---伯乐在线

功能

访问网页->抓取数据->数据存储->数据处理->提供检索服务

robots协议

一个约定俗成的协议，添加robots.txt文件，来说明本网站哪些内容不可以被抓取，起不到限制作用

自己写的爬虫无需遵守

网站排名(SEO)

1. 根据pagerank算法值进行排名（参考个网站流量、点击率等指标）
2. 百度竞价排名

缺点

1. 抓取的数据大多是无用的
2. 不能根据用户的需求来精准获取数据

聚焦爬虫

功能

根据需求，实现爬虫程序，抓取需要的数据

设计思路

1.确定要爬取的url

如何获取Url

2.模拟浏览器通过http协议访问url，获取服务器返回的html代码

如何访问

3.解析html字符串（根据一定规则提取需要的数据）

如何解析

5.反爬手段?

1.User-Agent:

User Agent中文名为用户代理，简称 UA，它是一个特殊字符串头，使得服务器能够识别客户使用的操作系统及版本、CPU 类型、浏览器及版本、浏览器渲染引擎、浏览器语言、浏览器插件等。

2.代理IP

西次代理

快代理

什么是高匿名、匿名和透明代理？它们有什么区别？

1.使用透明代理，对方服务器可以知道你使用了代理，并且也知道你的真实IP。

2.使用匿名代理，对方服务器可以知道你使用了代理，但不知道你的真实IP。

3.使用高匿名代理，对方服务器不知道你使用了代理，更不知道你的真实IP。

3.验证码访问

打码平台

云打码平台

超级鹰

4.动态加载网页 网站返回的是js数据 并不是网页的真实数据

selenium驱动真实的浏览器发送请求

5.数据加密

分析js代码

6.urllib库使用

```
urllib.request.urlopen() 模拟浏览器向服务器发送请求
response    服务器返回的数据
    response的数据类型是HttpResponse
    字节-->字符串
            解码decode
    字符串-->字节
            编码encode
    read()      字节形式读取二进制    扩展：rede(5)返回前几个字节
    readline()  读取一行
    readlines()  一行一行读取 直至结束
    getcode()    获取状态码
    geturl()     获取url
    getheaders() 获取headers
urllib.request.urlretrieve()
    请求网页
    请求图片
    请求视频
```

7.请求对象的定制

UA介绍：User Agent中文名为用户代理，简称 UA，它是一个特殊字符串头，使得服务器能够识别客户使用的操作系统及版本、CPU 类型、浏览器及版本。浏览器内核、浏览器渲染引擎、浏览器语言、浏览器插件等

语法：request = urllib.request.Request()

扩展：编码的由来

```
'''编码集的演变---
由于计算机是美国人发明的，因此，最早只有127个字符被编码到计算机里，也就是大小写英文字母、数字和一些符号，
这个编码表被称为ASCII编码，比如大写字母A的编码是65，小写字母z的编码是122。
但是要处理中文显然一个字节是不够的，至少需要两个字节，而且还不能和ASCII编码冲突，
所以，中国制定了GB2312编码，用来把中文编进去。
你可以想得到的是，全世界有上百种语言，日本把日文编到Shift_JIS里，韩国把韩文编到Euc-kr里，
各国有各国的标准，就会不可避免地出现冲突，结果就是，在多语言混合的文本中，显示出来会有乱码。
因此，Unicode应运而生。Unicode把所有语言都统一到一套编码里，这样就不会再有乱码问题了。
Unicode标准也在不断发展，但最常用的是用两个字节表示一个字符（如果要用到非常偏僻的字符，就需要4个字节）。
现代操作系统和大多数编程语言都直接支持Unicode。'''
```

8.编解码

1.get请求方式：urllib.parse.quote（）

```
eg:
import urllib.request
import urllib.parse

url = 'https://www.baidu.com/s?wd='

headers = {
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/74.0.3729.169 Safari/537.36'
}

url = url + urllib.parse.quote('小野')

request = urllib.request.Request(url=url,headers=headers)

response = urllib.request.urlopen(request)

print(response.read().decode('utf-8'))
```

2.get请求方式：urllib.parse.urlencode（）

```
eg:
import urllib.request
import urllib.parse
url = 'http://www.baidu.com/s?'
data = {
    'name':'小刚',
    'sex':'男',
}
data = urllib.parse.urlencode(data)
url = url + data
print(url)
headers = {
    'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/74.0.3729.169 Safari/537.36'
}

request = urllib.request.Request(url=url,headers=headers)
```

```
response = urllib.request.urlopen(request)
print(response.read().decode('utf-8'))
```

3.post请求方式

```
eg:百度翻译
import urllib.request
import urllib.parse
url = 'https://fanyi.baidu.com/sug'
headers = {
    'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/74.0.3729.169 Safari/537.36'
}
keyword = input('请输入您要查询的单词')
data = {
    'kw':keyword
}
data = urllib.parse.urlencode(data).encode('utf-8')
request = urllib.request.Request(url=url,headers=headers,data=data)
response = urllib.request.urlopen(request)
print(response.read().decode('utf-8'))
```

总结：post和get区别?

```
1：get请求方式的参数必须编码，参数是拼接到url后面，编码之后不需要调用encode方法
2：post请求方式的参数必须编码，参数是放在请求对象定制的方法中，编码之后需要调用encode方法
```

案例练习：百度详细翻译

```
import urllib.request
import urllib.parse

url = 'https://fanyi.baidu.com/v2transapi'
headers = {
    # ':authority': 'fanyi.baidu.com',
    # ':method': 'POST',
    # ':path': '/v2transapi',
    # ':scheme': 'https',
    # 'accept': '*/*',
    # 'accept-encoding': 'gzip, deflate, br',
    # 'accept-language': 'zh-CN,zh;q=0.9',
    # 'content-length': '119',
    # 'content-type': 'application/x-www-form-urlencoded; charset=UTF-8',









    'cookie': 'REALTIME_TRANS_SWITCH=1; FANYI_WORD_SWITCH=1; HISTORY_SWITCH=1;
```

```
SOUND_SPD_SWITCH=1; SOUND_PREFER_SWITCH=1; PSTM=1537097513;
BIDUPSID=D96F9A49A8630C54630DD60CE082A55C; BAIDUID=0814C35D13AE23F5EAFA8E0B24D9B436:FG=1;
to_lang_often=%5B%7B%22value%22%3A%22en%22%2C%22text%22%3A%22%u82F1%u8BED%22%7D%2C%7B%22value%22
%3A%22zh%22%2C%22text%22%3A%22%u4E2D%u6587%22%7D%5D;
from_lang_often=%5B%7B%22value%22%3A%22zh%22%2C%22text%22%3A%22%u4E2D%u6587%22%7D%2C%7B%22value%
22%3A%22en%22%2C%22text%22%3A%22%u82F1%u8BED%22%7D%5D; BDORZ=B490B5EBF6F3CD402E515D22BCDA1598;
delPer=0; H_PS_PSSID=1424_21115_29522_29519_29099_29568_28835_29220_26350; PSINO=2; locale=zh;
Hm_lvt_64ecd82404c51e03dc91cb9e8c025574=1563000604,1563334706,1565592510;
Hm_lpvt_64ecd82404c51e03dc91cb9e8c025574=1565592510;
yjs_js_security_passport=2379b52646498f3b5d216e6b21c6f1c7bf00f062_1565592544_js',
    # 'origin': 'https://fanyi.baidu.com',
    # 'referer': 'https://fanyi.baidu.com/translate?
aldtype=16047&query=&keyfrom=baidu&smartresult=dict&lang=auto2zh',
    # 'sec-fetch-mode': 'cors',
    # 'sec-fetch-site': 'same-origin',
    # 'user-agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/76.0.3809.100 Safari/537.36',
    # 'x-requested-with': 'XMLHttpRequest',
}
data = {
    'from': 'en',
    'to': 'zh',
    'query': 'you',
    'transtype': 'realtime',
    'simple_means_flag': '3',
    'sign': '269482.65435',
    'token': '2e0f1cb44414248f3a2b49fbad28bbd5',
}
#参数的编码
data = urllib.parse.urlencode(data).encode('utf-8')
# 请求对象的定制
request = urllib.request.Request(url=url,headers=headers,data=data)
response = urllib.request.urlopen(request)
# 请求之后返回的所有的数据
content = response.read().decode('utf-8')
import json
# loads将字符串转换为python对象
obj = json.loads(content)
# python对象转换为json字符串  ensure_ascii=False  忽略字符集编码
s = json.dumps(obj,ensure_ascii=False)
print(s)
```

9.ajax的get请求

案例：豆瓣电影

```
# 爬取豆瓣电影前10页数据
# https://movie.douban.com/j/chart/top_list?
type=20&interval_id=100%3A90&action=&start=0&limit=20
# https://movie.douban.com/j/chart/top_list?
type=20&interval_id=100%3A90&action=&start=20&limit=20

# https://movie.douban.com/j/chart/top_list?
```

```
type=20&interval_id=100%3A90&action=&start=40&limit=20

import urllib.request
import urllib.parse

# 下载前10页数据
# 下载的步骤：1.请求对象的定制  2.获取响应的数据 3.下载

# 每执行一次返回一个request对象
def create_request(page):
    base_url = 'https://movie.douban.com/j/chart/top_list?type=20&interval_id=100%3A90&action=&'
    headers = {
            'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/76.0.3809.100 Safari/537.36'
    }
    data={
        #  1 2  3  4
        #  0 20 40 60
        'start':(page-1)*20,
        'limit':20
    }
    # data编码
    data = urllib.parse.urlencode(data)
    url = base_url + data
    request = urllib.request.Request(url=url,headers=headers)
    return request
# 获取网页源码
def get_content(request):
    response = urllib.request.urlopen(request)
    content = response.read().decode('utf-8')
    return content

def down_load(page,content):
#    with open（文件的名字，模式，编码）as fp:
#        fp.write(内容)
    with open('douban_'+str(page)+'.json','w',encoding='utf-8')as fp:
        fp.write(content)

if __name__ == '__main__':
    start_page = int(input('请输入起始页码'))
    end_page = int(input('请输入结束页码'))
    for page in range(start_page,end_page+1):
        request = create_request(page)
        content = get_content(request)
        down_load(page,content)
```

10.ajax的post请求

案例：KFC官网

11.URLError\HTTPError

简介:1.HTTPError类是URLError类的子类
2.导入的包urllib.error.HTTPError urllib.error.URLError
3.http错误：http错误是针对浏览器无法连接到服务器而增加出来的错误提示。引导并告诉浏览者该页是哪里出了问题。
4.通过urllib发送请求的时候，有可能会发送失败，这个时候如果想让你的代码更加的健壮，可以通过try-except进行捕获异常，异常有两类，URLError\HTTPError

eg:

```
import urllib.request
import urllib.error
url = 'https://blog.csdn.net/ityard/article/details/102646738'

# url = 'http://www.goudan11111.com'

headers = {
        # 'Accept':
'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,applicati
on/signed-exchange;v=b3',
        # 'Accept-Encoding': 'gzip, deflate, br',
        # 'Accept-Language': 'zh-CN,zh;q=0.9',
        # 'Cache-Control': 'max-age=0',
        # 'Connection': 'keep-alive',
        'Cookie': 'uuid_tt_dd=10_19284691370-1530006813444-566189;
smidV2=2018091619443662be2b30145de89bbb07f3f93a3167b80002b53e7acc61420;
_ga=GA1.2.1823123463.1543288103; dc_session_id=10_1550457613466.265727;
acw_tc=2760821d15710446036596250e10a1a7c89c3593e79928b22b3e3e2bc98b89;
Hm_lvt_e5ef47b9f471504959267fd614d579cd=1571329184;
Hm_ct_e5ef47b9f471504959267fd614d579cd=6525*1*10_19284691370-1530006813444-566189;
__yadk_uid=r0LSXrcNYgymXooFiLaCGt1ahSCSxMCb;
Hm_lvt_6bcd52f51e9b3dce32bec4a3997715ac=1571329199,1571329223,1571713144,1571799968;
acw_sc__v2=5dafc3b3bc5fad549cbdea513e330fbbbee00e25; firstDie=1; SESSION=396bc85c-556b-42bd-
890c-c20adaaa1e47; UserName=weixin_42565646; UserInfo=d34ab5352bfa4f21b1eb68cdacd74768;
UserToken=d34ab5352bfa4f21b1eb68cdacd74768; UserNick=weixin_42565646; AU=7A5;
UN=weixin_42565646; BT=1571800370777; p_uid=U000000; dc_tos=pzt4xf;
Hm_lpvt_6bcd52f51e9b3dce32bec4a3997715ac=1571800372;
Hm_ct_6bcd52f51e9b3dce32bec4a3997715ac=1788*1*PC_VC!6525*1*10_19284691370-1530006813444-
566189!5744*1*weixin_42565646;
announcement=%257B%2522isLogin%2522%253Atrue%252C%2522announcementUrl%2522%253A%2522https%253A%2
52F%252Fblogdev.blog.csdn.net%252Farticle%252Fdetails%252F102605809%2522%252C%2522announcementCo
unt%2522%253A0%252C%2522announcementExpire%2522%253A3600000%257D',
        # 'Host': 'blog.csdn.net',
        # 'Referer': 'https://passport.csdn.net/login?code=public',
        # 'Sec-Fetch-Mode': 'navigate',
        # 'Sec-Fetch-Site': 'same-site',
        # 'Sec-Fetch-User': '?1',
        # 'Upgrade-Insecure-Requests': '1',
        'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/77.0.3865.120 Safari/537.36',
    }
try:

    request = urllib.request.Request(url=url,headers=headers)
```

```
    response = urllib.request.urlopen(request)

    content = response.read().decode('utf-8')
    print(content)
except urllib.error.HTTPError:
    print(1111)

except urllib.error.URLError:
    print(2222)
```

12.cookie登录

使用案例：
        1.weibo登陆
        作业：qq空间的爬取

13.Handler处理器

为什么要学习handler?
      urllib.request.urlopen(url)
          不能定制请求头
      urllib.request.Request(url,headers,data)
          可以定制请求头
      Handler
          定制更高级的请求头（随着业务逻辑的复杂 请求对象的定制已经满足不了我们的需求（动态cookie和代理不能使用请求对象的定制）

```
eg:
import urllib.request
url = 'http://www.baidu.com'
headers = {
        'User - Agent': 'Mozilla / 5.0(Windows NT 10.0;Win64;x64) AppleWebKit / 537.36(KHTML, likeGecko) Chrome / 74.0.3729.169Safari / 537.36'
    }
request = urllib.request.Request(url=url,headers=headers)

handler = urllib.request.HTTPHandler()

opener = urllib.request.build_opener(handler)

response = opener.open(request)

print(response.read().decode('utf-8'))
```

14.代理服务器

1.代理的常用功能?

1.突破自身IP访问限制，访问国外站点。

2.访问一些单位或团体内部资源

扩展：某大学FTP(前提是该代理地址在该资源的允许访问范围之内)，使用教育网内地址段免费代理服务器，就可以用于对教育网开放的各类FTP下载上传，以及各类资料查询共享等服务。

3.提高访问速度

扩展：通常代理服务器都设置一个较大的硬盘缓冲区，当有外界的信息通过时，同时也将其保存到缓冲区中，当其他用户再访问相同的信息时， 则直接由缓冲区中取出信息，传给用户，以提高访问速度。

4.隐藏真实IP

扩展：上网者也可以通过这种方法隐藏自己的IP，免受攻击。

2.代码配置代理

创建Reuqest对象

创建ProxyHandler对象

用handler对象创建opener对象

使用opener.open函数发送请求

```
eg:
import urllib.request
url = 'http://www.baidu.com/s?wd=ip'
headers = {
        'User - Agent': 'Mozilla / 5.0(Windows NT 10.0;Win64;x64) AppleWebKit / 537.36(KHTML, likeGecko) Chrome / 74.0.3729.169Safari / 537.36'
    }
request = urllib.request.Request(url=url,headers=headers)
proxies = {'http':'117.141.155.244:53281'}
handler = urllib.request.ProxyHandler(proxies=proxies)
opener = urllib.request.build_opener(handler)
response = opener.open(request)
content = response.read().decode('utf-8')
with open('daili.html','w',encoding='utf-8')as fp:
    fp.write(content)
```

扩展：1.代理池

2.快代理