小象学院

Parser

主　讲: 杨真

小象学院

Part 1 基础

- 环境搭建
- HTML 基础
- 第一个10行代码的爬虫
- 内容抽取及解析
- HTTP 协议
- POSTMAN 工具详解

Part 2 爬虫

- 网站结构分析
- 抓取方案
- 多线程并行及排重
- 用 MySQL 信息存储

Part 3 进阶

- 网站服务结构
- Cookie 及 登录处理
- 控制抓取的节奏
- 日志
- 守护进程

Part 4 实战

- 网站结构分析
- 网页抓取方案
- 数据提取
- 存储方案

- requests
- 字符编码
- 文件命名

requests

```
>>> r = requests.get('https://api.github.com/user’, header=headers)
>>> r.status_code 200
>>> r.headers['content-type'] 'application/json; charset=utf8’
>>> r.encoding 'utf-8’
>>> r.text u'{"type":"User"...’
>>> r.content b'{"type":"User"...’
>>> r.json() {u'private_gists': 419, u'total_private_repos': 77, ...}
```

字符编码

```
⊞ Frame 366: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
⊞ Ethernet II, Src: Anritsu_05:6c:5b (00:00:91:05:6c:5b), Dst: AdlinkTe_25:6f:f4 (00:30:64:25:
⊟ Internet Protocol Version 4, Src: 192.168.1.31 (192.168.1.31), Dst: 192.168.1.32 (192.168.1.
    Version: 4
    Header length: 20 bytes
  ⊟ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capab
      0000 00.. = Differentiated Services Codepoint: Default (0x00)
      .... ..00 = Explicit Congestion Notification: Not-ECT (Not ECN-Capable Transport) (0x00)
    Total Length: 60
    Identification: 0x5887 (22663)
  ⊟ Flags: 0x00
      0... .... = Reserved bit: Not set
      .0.. .... = Don't fragment: Not set
      ..0. .... = More fragments: Not set
    Fragment offset: 0
    Time to live: 128
    Protocol: ICMP (1)
  ⊞ Header checksum: 0x5eaa [correct]
    Source: 192.168.1.31 (192.168.1.31)
    Destination: 192.168.1.32 (192.168.1.32)

0000  00 30 64 25 6f f4 00 00  91 05 6c 5b 08 00 45 00   .0d%o... ..l[..E.
0010  00 3c 58 87 00 00 80 01  5e aa c0 a8 01 1f c0 a8   .<X..... ^.......
0020  01 20 08 00 4d 5a 00 01  00 01 61 62 63 64 65 66   . ..MZ.. ..abcdef
0030  67 68 69 6a 6b 6c 6d 6e  6f 70 71 72 73 74 75 76   ghijklmn opqrstuv
0040  77 61 62 63 64 65 66 67  68 69                    wabcdefg hi
```

一个字符串在计算机里，以二进制的方式存储。字符串的编码，就是用制定的编码字节来表示这个字符

你好

utf8：　　b'\xe4\xbd\xa0\xe5\xa5\xbd'
Unicode: u'\u4f60\u597d'

ISO8859-1 是一个8bit编码格式

仅支持英文字符、数字及常见的符号

3.6% 的全球网站使用 ISO 8859-1

ISO/IEC 8859-1

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A 10	_B 11	_C 12	_D 13	_E 14	_F 15
0_ 0																
1_ 16																
2_ 32	SP 0020	! 0021	" 0022	# 0023	$ 0024	% 0025	& 0026	' 0027	(0028	) 0029	* 002A	+ 002B	, 002C	- 002D	. 002E	/ 002F
3_ 48	0 0030	1 0031	2 0032	3 0033	4 0034	5 0035	6 0036	7 0037	8 0038	9 0039	: 003A	; 003B	< 003C	= 003D	> 003E	? 003F
4_ 64	@ 0040	A 0041	B 0042	C 0043	D 0044	E 0045	F 0046	G 0047	H 0048	I 0049	J 004A	K 004B	L 004C	M 004D	N 004E	O 004F
5_ 80	P 0050	Q 0051	R 0052	S 0053	T 0054	U 0055	V 0056	W 0057	X 0058	Y 0059	Z 005A	[005B	\ 005C	] 005D	^ 005E	_ 005F
6_ 96	` 0060	a 0061	b 0062	c 0063	d 0064	e 0065	f 0066	g 0067	h 0068	i 0069	j 006A	k 006B	l 006C	m 006D	n 006E	o 006F
7_ 112	p 0070	q 0071	r 0072	s 0073	t 0074	u 0075	v 0076	w 0077	x 0078	y 0079	z 007A	{ 007B	| 007C	} 007D	~ 007E	
8_ 128																
9_ 144																
A_ 160	NBSP 00A0	¡ 00A1	¢ 00A2	£ 00A3	¤ 00A4	¥ 00A5	¦ 00A6	§ 00A7	¨ 00A8	© 00A9	ª 00AA	« 00AB	¬ 00AC	SHY 00AD	® 00AE	¯ 00AF
B_ 176	° 00B0	± 00B1	² 00B2	³ 00B3	´ 00B4	µ 00B5	¶ 00B6	· 00B7	¸ 00B8	¹ 00B9	º 00BA	» 00BB	¼ 00BC	½ 00BD	¾ 00BE	¿ 00BF
C_ 192	À 00C0	Á 00C1	Â 00C2	Ã 00C3	Ä 00C4	Å 00C5	Æ 00C6	Ç 00C7	È 00C8	É 00C9	Ê 00CA	Ë 00CB	Ì 00CC	Í 00CD	Î 00CE	Ï 00CF
D_ 208	Ð 00D0	Ñ 00D1	Ò 00D2	Ó 00D3	Ô 00D4	Õ 00D5	Ö 00D6	× 00D7	Ø 00D8	Ù 00D9	Ú 00DA	Û 00DB	Ü 00DC	Ý 00DD	Þ 00DE	ß 00DF
E_ 224	à 00E0	á 00E1	â 00E2	ã 00E3	ä 00E4	å 00E5	æ 00E6	ç 00E7	è 00E8	é 00E9	ê 00EA	ë 00EB	ì 00EC	í 00ED	î 00EE	ï 00EF
F_ 240	ð 00F0	ñ 00F1	ò 00F2	ó 00F3	ô 00F4	õ 00F5	ö 00F6	÷ 00F7	ø 00F8	ù 00F9	ú 00FA	û 00FB	ü 00FC	ý 00FD	þ 00FE	ÿ 00FF

Letter Number Punctuation Symbol Other undefined undefined in the first release of ECMA-94 (1985).[5]

ISO8859-1 不能表示全球所有字符

Unicode 基本能把全球所有字符表示完，欧洲的所有字符（涵盖西班牙语、葡萄牙语、英语、德语等）、阿拉伯字符等全部加起来只有几百个，汉字有大约8000个

Python 2.7 默认UTF8，3.6 默认是Unicode

注意：Python2.7默认使用系统的编码，win默认是GBK，mac默认是UTF8。

1、因为当前老师用的是mac，所以Python2.7默认会是UTF8。

2、请看《在不同系统平台下Python2.7中文编解码问题》视频。

对于绝大多数语言来说，只需要1个字节就能编码，如果都采用 Unicode 会极大浪费，于是出现了变长的编码格式 UTF8

一般来说，Unicode 2个字节的，在UTF8需要3个字节

Python 2.7 默认 UTF8

'你好' ->'\xe4\xbd\xa0\xe5\xa5\xbd'

'你好'.decode('utf8') -> 转换为 Unicode u'\u4f60\u597d'

Python 3.6 默认 Unicode

'你好'.encode('utf8') -> 转换为 UTF8 编码

注意：Python2.7默认使用系统的编码，win默认是GBK，mac默认是UTF8。

1、因为当前老师用的是mac，所以Python2.7默认会是UTF8。

2、请看《在不同系统平台下Python2.7中文编解码问题》视频。

文件存储

表达式	描述
r	读文件
w	写文件
b	二进制的方式打开
a	在文件结尾继续添加的方式写入，区别：w默认为覆盖
+	不存在的时候，创建这个文件

上下文管理器

打开文件的时候，为了能正常释放文件的句柄，都要加个try，然后再finally里把f close掉，但是这样的代码不美观，finally就像个尾巴，一直托在后面，尤其是当try里面的语句时几十行

除了文件外，with还支持 threading、decimal等模块，当然我们也可以自己定义可以给with调用的上下文管理器

```python
class CNDTextIOWrapper():
    def __init__(self):
        print('__init__')
        pass

    def __enter__(self):
        print('__enter__')
        return self

    def __exit__(self, type, value, trace):
        print("type:", type)
        print("value:", value)
        print("trace:", trace)

    def do_something(self):
        print('do_something')

def demo():
    return CNDTextIOWrapper()

with demo() as d:
    d.do_something()
```

使用类定义上下文管理器需要在类上定义__enter__和__exit__方法，执行with A() as a: 语句时会先执行__enter__方法，这个方法的返回值会赋值给后面的a变量，当with里面的语句产生异常或正常执行完时，都好调用类中的__exit__方法

- 取最后一个斜线后的部分

https://news.sina.com.cn/c/2018-12-10/doc-ihprknvu0188659.shtml

文件名：doc-ihprknvu0188659.shtml

- 取最后两个部分

https://baike.baidu.com/item/姚明/9949328

https://baike.baidu.com/item/姚明/28

/ 用 _ 代替，并加上 .html 的后缀

文件名：姚明_28.html

- 自定义

网站域名+文件：baike_baidu_姚明_28.html

URL 的参数，除了 ？ 后面的部分，地址部分也可能是参数

一般规则，会把 ？ 之前的部分统一作为访问地址来对待，而 ？ 之后的作为这个请求的具体参数

```
url[:url.find('?')]
```

小象学院

只为遇见明天更优秀的你