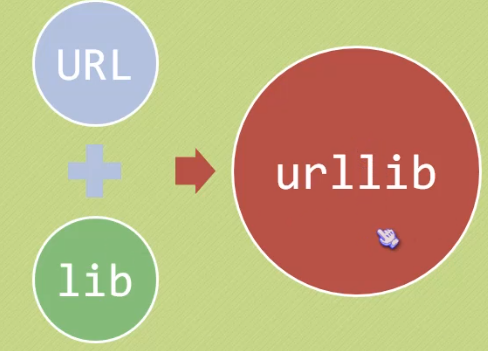
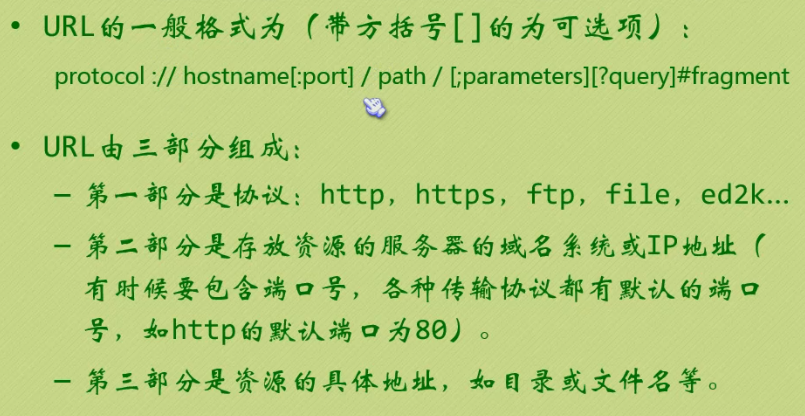
1. 网络爬虫

PY如何访问互联网







urllib是python3.x中提供的一系列操作URL的库，他可以轻松模拟用户使用浏览器访问网页

使用步骤：（1）导入urllib库的request模块from urllib import request

(2)请求URL resp = request.urlopen('http://www.baidu.com')

(3)使用响应对象输出数据 print(resp.read().decode("utf-8"))

1. Eg1：从placekitten上下载一张图片



程序执行之后 还有geturl（）方法： 返回具体访问的url

info（）方法： 得到一个httpMessage的对象

getcode（）方法：http的状态 200 表示正常响应



下面两张图片是等价的：





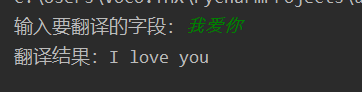
1. Eg2：模拟有道词典翻译

Py提交post表单

如果data参数默认值为None 则为Get提交方式 若给data赋值，则为POST方式 data 必须是网页源代码中POST项里面Form Data 中的所有项的值

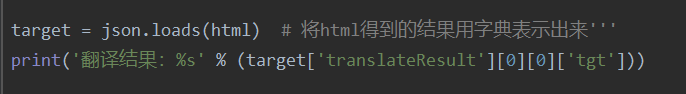






Target的具体步骤：





1. Encode 和 decode

字符串在py中是unicode编码 在做编码转换时，通常需要以unicode作为中间编码， 即先将其他编码的字符串解码（decode）成unicode，再从unicode编码（encode）成另一种编码。

decode的作用是将其他编码的字符串转换成unicode编码，如str1.decode('gb2312')，表示将gb2312编码的字符串str1转换成unicode编码。

encode的作用是将unicode编码转换成其他编码的字符串，如str2.encode('gb2312')，表示将unicode编码的字符串str2转换成gb2312编码。

总得意思:想要将其他的编码转换成utf-8必须先将其解码成unicode然后重新编码成utf-8,它是以unicode为转换媒介的 如：s='中文' 如果是在utf8的文件中，该字符串就是utf8编码，如果是在gb2312的文件中，则其编码为gb2312。这种情况下，要进行编码转换，都需要先用 decode方法将其转换成unicode编码，再使用encode方法将其转换成其他编码。通常，在没有指定特定的编码方式时，都是使用的系统默认编码创建的代码文件

1. headers



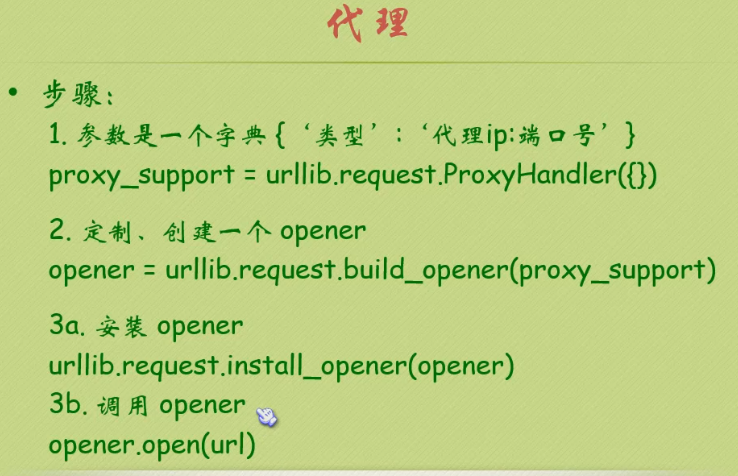
Headers 必须是字典形式 在网页中 找 Request Headers 里面的User-Agent写入字典中

添加headers：

1. 在传参之前定义headers
2. 在urllib.request.Request生成之后调用 add\_header(key,value) 添加 
3. 代理

有些网站会设置 下载多少之后进行验证是否人为操作 普通爬虫解决不了这个问题 就会使用 延迟提交的时间 或 使用 代理 进行

延迟提交时间： 使用time模块 即在循环后添加 time.sleep()方法 过几秒钟再出现





Opener也可以修改headers

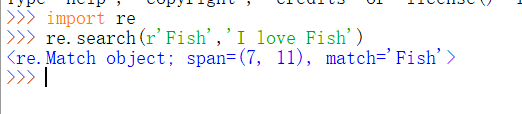
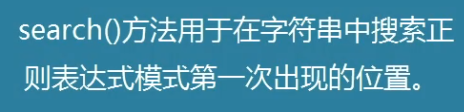


addheaders 是一个列表 里面是映射关系 也是User-Agent

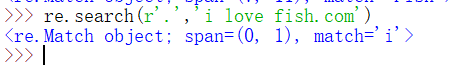
1. 正则表达式

Import re

re.search()

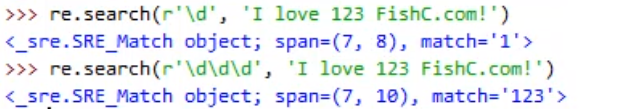


通配符



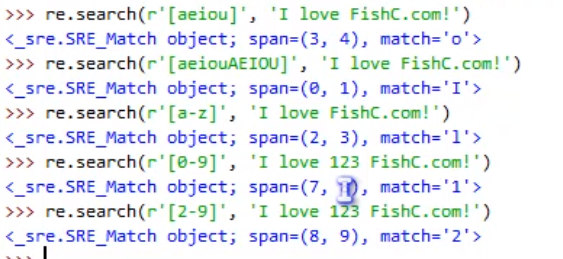
r’.’ 中的 . 表示除换行符外的任何字符

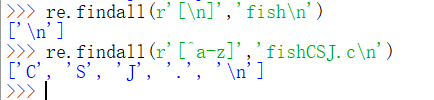
r’ \. ’ 匹配 点本身

\d 匹配数字

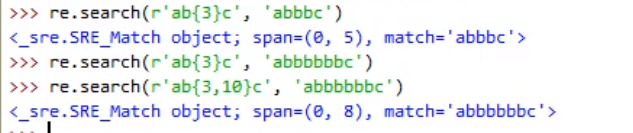
限定匹配的范围：（字符类）[ ]

将里面的字符当作普通字符 [.]表示匹配 ‘ . ’



第二个脱字符放前面表示取反： 除了a-z其他都匹配 放在最后表示匹配脱字符本身

限定匹配个数 { }

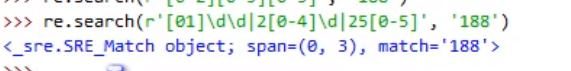


表示{ } 前的那个字符重复多少次 是几次必须多少次 可以限定范围



正则表达式匹配的是字符串 多位的数字认为是由几个单个字符组成 上图【】中表示 0 - 2 和5 即 匹配 0 1 2 5 中任何一个

例:IP地址 192.182.2.70 每一个范围都是0-255 且每一位不一定有三位



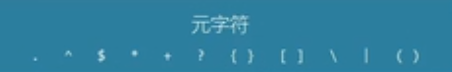
| 逻辑或 这三个任意满足，【01】\d\d表示 百位为0或1 十位个位任意数字都可以

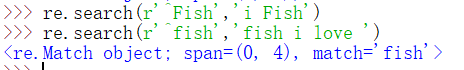
2【0-4】\d 百位为2，十位0-4 个位任意

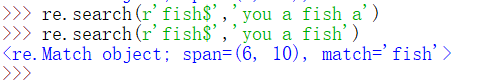
下图是IP地址的匹配 {0,1}重复0次或1次 表示百位 十位 是可选的：



1. 元字符



^ 表示 该字符是整个字符串的开始则匹配

$ 表示该字符是整个字符串的结尾就匹配

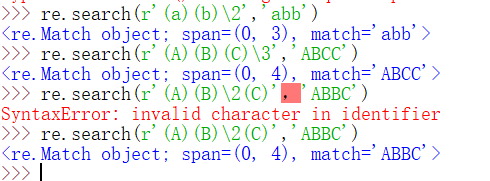
\

1. 加普通字符变成特殊字符

2）解除原字符的特殊功能

3） \+数字

1-99表示对应的第几个组所匹配的字符串 数字为0 或三位数字表示对应的八进制所对应的ASCLL码对应的字符

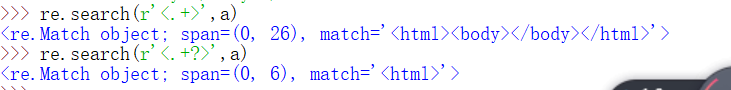


\* 表示匹配表达式 0 次或多次（0，∞】

+ 表示匹配表达式1次或多次 （1，∞】 重复前面任何内容

？ 表示匹配表达式 0次或1次（0，1）

匹配字符串默认启用贪婪模式 在符合条件的情况下尽可能多的匹配



<.+>点 表示消耗字符 +重复前面的内容（重复点的内容）

<.+?> ?表示去除贪婪模式 启用非贪婪 只匹配一个符合条件的字符串

1. \A ： \+普通字符构成特殊字符

\A 匹配字符串的开始位置

\Z 匹配字符串的结束位置

\b 匹配单词的边界 符号 小括号都算作单词的边界

单词： 字母 数字 下划线

其中\_被当作一个单词不符合单词边界

\B 匹配非单词边界

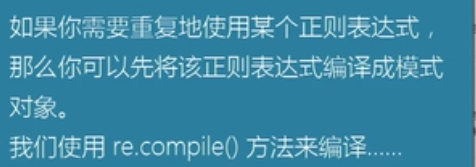
\d 匹配0-9任意数字

\D 匹配非数字

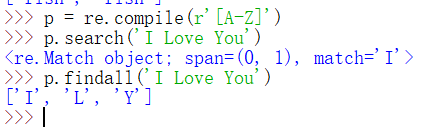
\s 匹配空白字符

\w 匹配单词字母：字母 数字 下划线

1. 编译正则表达式



编译后匹配时就不用将正则表达式当参数传入了 直接输入要匹配的字符串



模块级正则表达式 ：将正则表达式作为第一个参数传入

模式对象： 正则表达式通过编译，匹配时不用再传入正则表达式

需要多次使用 则使用模式对象

1. 编译标志

可以修改正则表达式的工作方式

编译标志有两个名字：完整名和简写

ASCII----A 使转义符号如\w \b \s只匹配ASII字符

DOTALL----S 使 . 可以匹配任何符号 包括换行符

IGNORECASE----I 匹配时不区分大小写

LOCALE----L 支持当前的语言设置

MULTILINE----M 多行匹配，影响 ^ 和 $

VERBOSE----X 启用详细的正则表达式 支持换行 空格等

1. 正则表达式的使用方法和扩展语法

模块级别的search（r‘正则表达式’，‘匹配的字符串’，‘flags=0匹配的模式’）

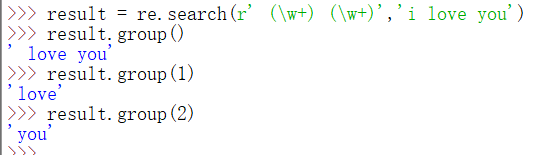
模式对象已编译的正则表达式（‘匹配字符串’，起始位置，结束位置） 后两个参数可选

Search方法返回的是匹配对象 并不是字符串

就需要使用一些方法获取需要的字符串

例： result = re.search(r‘ (\w+) (\w+) ’ , ‘i love you ’)

group()方法 如果正则表达式中存在子组 则在group中指定序号 进行对应的子组捕获



findall()方法 找到所有符合条件的元素 如果其中正则表达式含有子组，则会单独返回子组的内容 若有多个子组 则会将结果组成元组返回