     一、urllib2模块回顾

       urllib2模块中最重要的函数是urlopen()函数，用于获取URLs资源（Uniform Resorce Locators）。urlopen函数不仅可以用于简单的情况，还可以进行复杂情况下的资源获取如认证(authentication)、cookies、代理等。urlopen支持多种协议，如http、ftp、file等。

        HTTP是基于请求、响应的协议，客户端发出请求、服务器端作出响应。urllib2通过Request对象反映发出的HTTP请求，调用urlopen()时就会发出请求，函数返回值就是相应的响应对象。

     1、POST数据

import urllib,urllib2

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

datas={"email":user,"password":password}

req=urllib2.Request(url,urllib.encode(datas))

response=urllib2.urlopen(req)

     2、增加Header头部

     由于一些网站不希望被程序访问，或网站会发送不同的内容给不同的浏览器类型，因此需要修改HTTP头部来将程序伪造成相应的浏览器，而浏览器通常通过头部的User-Agent来识别，因此通常只改User-Agent即可。方法是传递一个headers头部字典给Request对象。

import urllib2

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

headers={"User-Agent":"Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1"}

req=urllib2.Request(url,headers=headers)

response=urllib2.urlopen(req)

**二、异常处理**

          当urlopen()不能处理响应时会引起URLError异常。HTTPError异常是URLError的一个子类，只有在访问HTTP类型的URL时才会引起。

1、URLError异常

    通常引起URLError的原因是：无网络连接（没有到目标服务器的路由）、访问的目标服务器不存在。在这种情况下，异常对象会有reason属性（是一个（错误码、错误原因）的元组）。

#! /usr/bin/env python

#coding=utf-8

import urllib2

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

try:

response=urllib2.urlopen(url)

except urllib2.URLError,e:

print e.reason

2、HTTPError

    每一个从服务器返回的HTTP响应都有一个状态码。其中，有的状态码表示服务器不能完成相应的请求，默认的处理程序可以为我们处理一些这样的状态码（如返回的响应是重定向，urllib2会自动为我们从重定向后的页面中获取信息）。有些状态码，urllib2模块不能帮我们处理，那么urlopen函数就会引起HTTPError异常,其中典型的有404/401。

   HTTPError异常的实例有整数类型的code属性，表示服务器返回的错误状态码。

    urllib2模块默认的处理程序可以处理重定向（状态码是300范围），而且状态码在100-299范围内表示成功。因此，能够引起HTTPError异常的状态码范围是：400-599.

    当引起错误时，服务器会返回HTTP错误码和错误页面。你可以将HTPError实例作为返回页面，这意味着，HTTPError实例不仅有code属性，还有read、geturl、info等方法。

#! /usr/bin/env python

#coding=utf-8

import urllib2

url="http://cs.scu.edu.cn/~duanlei"

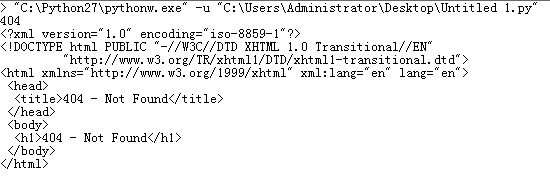
try:

response=urllib2.urlopen(url)

except urllib2.HTTPError,e:

print e.code

print e.read()



3、总结

    如果想在代码中处理URLError和HTTPError有两种方法，代码如下：

#! /usr/bin/env python

#coding=utf-8

import urllib2

url="xxxxxx" #需要访问的URL

try:

response=urllib2.urlopen(url)

except urllib2.HTTPError,e: #HTTPError必须排在URLError的前面

print "The server couldn't fulfill the request"

print "Error code:",e.code

print "Return content:",e.read()

except urllib2.URLError,e:

print "Failed to reach the server"

print "The reason:",e.reason

else:

#something you should do

pass #其他异常的处理

#! /usr/bin/env python

#coding=utf-8

import urllib2

url="http://xxx" #需要访问的URL

try:

response=urllib2.urlopen(url)

except urllib2.URLError,e:

if hasattr(e,"reason"):

print "Failed to reach the server"

print "The reason:",e.reason

elif hasattr(e,"code"):

print "The server couldn't fulfill the request"

print "Error code:",e.code

print "Return content:",e.read()

else:

pass #其他异常的处理

   相比较而言，第二种异常处理方法更优。