大家好，本节在这里主要说的是URLError还有HTTPError，以及对它们的一些处理。

**1.URLError**

首先解释下URLError可能产生的原因：

* 网络无连接，即本机无法上网
* 连接不到特定的服务器
* 服务器不存在
* URLError与reason属性联系在一起

在代码中，我们需要用try-except语句来包围并捕获相应的异常。下面是一个例子，先感受下它的风骚

Python

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | import urllib2    requset = urllib2.Request('http://www.xxxxx.com')  try:      urllib2.urlopen(requset)  except urllib2.URLError, e:      print e.reason |

我们利用了 urlopen方法访问了一个不存在的网址，运行结果如下：

Python

|  |  |
| --- | --- |
| 1 | [Errno 11004] getaddrinfo failed |

它说明了错误代号是11004，错误原因是 getaddrinfo failed

2.HTTPError

HTTPError是URLError的子类，在你利用urlopen方法发出一个请求时，服务器上都会对应一个应答对象response，其中它包含一个数字”状态码”。举个例子，假如response是一个”重定向”，需[定位](http://cpro.baidu.com/cpro/ui/uijs.php?c=news&cf=1001&ch=0&di=8&fv=17&jk=273f52071a06fb13&k=%B6%A8%CE%BB&k0=%B6%A8%CE%BB&kdi0=0&luki=1&n=10&p=baidu&q=35091180_cpr&rb=0&rs=1&seller_id=1&sid=13fb061a7523f27&ssp2=1&stid=0&t=tpclicked3_hc&tu=u1959022&u=http%3A%2F%2Fcuiqingcai%2Ecom%2F961%2Ehtml&urlid=0)到别的地址获取文档，urllib2将对此进行处理。

HTTPError与code属性联系在一起。

其他不能处理的，urlopen会产生一个HTTPError，对应相应的状态吗，HTTP状态码表示HTTP协议所返回的响应的状态。下面将状态码归结如下：

100：继续 客户端应当继续发送请求。客户端应当继续发送请求的剩余部分，或者如果请求已经完成，忽略这个响应。

101： 转换协议 在发送完这个响应最后的空行后，[服务器](http://cpro.baidu.com/cpro/ui/uijs.php?c=news&cf=1001&ch=0&di=8&fv=17&jk=273f52071a06fb13&k=%B7%FE%CE%F1%C6%F7&k0=%B7%FE%CE%F1%C6%F7&kdi0=0&luki=6&n=10&p=baidu&q=35091180_cpr&rb=0&rs=1&seller_id=1&sid=13fb061a7523f27&ssp2=1&stid=0&t=tpclicked3_hc&tu=u1959022&u=http%3A%2F%2Fcuiqingcai%2Ecom%2F961%2Ehtml&urlid=0)将会切换到在Upgrade 消息头中定义的那些协议。只有在切换新的协议更有好处的时候才应该采取类似措施。

102：继续处理 由WebDAV（RFC 2518）扩展的状态码，代表处理将被继续执行。

200：请求成功 处理方式：获得响应的内容，进行处理

201：请求完成，结果是创建了新资源。新创建资源的URI可在响应的实体中得到 处理方式：爬虫中不会遇到

202：请求被接受，但处理尚未完成 处理方式：阻塞等待

204：服务器端已经实现了请求，但是没有返回新的信 息。如果客户是用户[代理](http://cpro.baidu.com/cpro/ui/uijs.php?c=news&cf=1001&ch=0&di=8&fv=17&jk=273f52071a06fb13&k=%B4%FA%C0%ED&k0=%B4%FA%C0%ED&kdi0=0&luki=2&n=10&p=baidu&q=35091180_cpr&rb=0&rs=1&seller_id=1&sid=13fb061a7523f27&ssp2=1&stid=0&t=tpclicked3_hc&tu=u1959022&u=http%3A%2F%2Fcuiqingcai%2Ecom%2F961%2Ehtml&urlid=0)，则无须为此更新自身的文档视图。 处理方式：丢弃

300：该状态码不被HTTP/1.0的应用程序直接使用， 只是作为3XX类型回应的默认解释。存在多个可用的被请求资源。 处理方式：若程序中能够处理，则进行进一步处理，如果程序中不能处理，则丢弃  
301：请求到的资源都会分配一个永久的URL，这样就可以在将来通过该URL来访问此资源 处理方式：重定向到分配的URL

302：请求到的资源在一个不同的URL处临时保存 处理方式：重定向到临时的URL

304：请求的资源未更新 处理方式：丢弃

400：非法请求 处理方式：丢弃

401：未授权 处理方式：丢弃

403：禁止 处理方式：丢弃

404：没有找到 处理方式：丢弃

500：服务器内部错误 服务器遇到了一个未曾预料的状况，导致了它无法完成对请求的处理。一般来说，这个问题都会在[服务器端](http://cpro.baidu.com/cpro/ui/uijs.php?c=news&cf=1001&ch=0&di=8&fv=17&jk=273f52071a06fb13&k=%B7%FE%CE%F1%C6%F7%B6%CB&k0=%B7%FE%CE%F1%C6%F7%B6%CB&kdi0=0&luki=5&n=10&p=baidu&q=35091180_cpr&rb=0&rs=1&seller_id=1&sid=13fb061a7523f27&ssp2=1&stid=0&t=tpclicked3_hc&tu=u1959022&u=http%3A%2F%2Fcuiqingcai%2Ecom%2F961%2Ehtml&urlid=0)的源代码出现错误时出现。

501：服务器无法识别 服务器不支持当前请求所需要的某个功能。当服务器无法识别请求的方法，并且无法支持其对任何资源的请求。

502：错误网关 作为网关或者[代理](http://cpro.baidu.com/cpro/ui/uijs.php?c=news&cf=1001&ch=0&di=8&fv=17&jk=273f52071a06fb13&k=%B4%FA%C0%ED&k0=%B4%FA%C0%ED&kdi0=0&luki=2&n=10&p=baidu&q=35091180_cpr&rb=0&rs=1&seller_id=1&sid=13fb061a7523f27&ssp2=1&stid=0&t=tpclicked3_hc&tu=u1959022&u=http%3A%2F%2Fcuiqingcai%2Ecom%2F961%2Ehtml&urlid=0)工作的服务器尝试执行请求时，从上游服务器接收到无效的响应。

503：服务出错 由于临时的[服务器](http://cpro.baidu.com/cpro/ui/uijs.php?c=news&cf=1001&ch=0&di=8&fv=17&jk=273f52071a06fb13&k=%B7%FE%CE%F1%C6%F7&k0=%B7%FE%CE%F1%C6%F7&kdi0=0&luki=6&n=10&p=baidu&q=35091180_cpr&rb=0&rs=1&seller_id=1&sid=13fb061a7523f27&ssp2=1&stid=0&t=tpclicked3_hc&tu=u1959022&u=http%3A%2F%2Fcuiqingcai%2Ecom%2F961%2Ehtml&urlid=0)维护或者过载，服务器当前无法处理请求。这个状况是临时的，并且将在一段时间以后恢复。

HTTPError实例产生后会有一个code属性，这就是是服务器发送的相关错误号。  
因为urllib2可以为你处理重定向，也就是3开头的代号可以被处理，并且100-299范围的号码指示成功，所以你只能看到400-599的错误号码。

下面我们写一个例子来感受一下，捕获的异常是HTTPError，它会带有一个code属性，就是错误代号，另外我们又打印了reason属性，这是它的父类URLError的属性。

Python

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | import urllib2    req = urllib2.Request('http://blog.csdn.net/cqcre')  try:      urllib2.urlopen(req)  except urllib2.HTTPError, e:      print e.code      print e.reason |

运行结果如下

Python

|  |  |
| --- | --- |
| 1  2 | 403  Forbidden |

错误代号是403，错误原因是Forbidden，说明服务器禁止访问。

我们知道，HTTPError的父类是URLError，根据[编程](http://cpro.baidu.com/cpro/ui/uijs.php?c=news&cf=1001&ch=0&di=8&fv=17&jk=273f52071a06fb13&k=%B1%E0%B3%CC&k0=%B1%E0%B3%CC&kdi0=0&luki=4&n=10&p=baidu&q=35091180_cpr&rb=0&rs=1&seller_id=1&sid=13fb061a7523f27&ssp2=1&stid=0&t=tpclicked3_hc&tu=u1959022&u=http%3A%2F%2Fcuiqingcai%2Ecom%2F961%2Ehtml&urlid=0)经验，父类的异常应当写到子类异常的后面，如果子类捕获不到，那么可以捕获父类的异常，所以上述的代码可以这么改写

Python

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | import urllib2    req = urllib2.Request('http://blog.csdn.net/cqcre')  try:      urllib2.urlopen(req)  except urllib2.HTTPError, e:      print e.code  except urllib2.URLError, e:      print e.reason  else:      print "OK" |

如果捕获到了HTTPError，则输出code，不会再处理URLError异常。如果发生的不是HTTPError，则会去捕获URLError异常，输出错误原因。

另外还可以加入 hasattr属性提前对属性进行判断，代码改写如下

Python

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | import urllib2    req = urllib2.Request('http://blog.csdn.net/cqcre')  try:      urllib2.urlopen(req)  except urllib2.URLError, e:      if hasattr(e,"code"):          print e.code      if hasattr(e,"reason"):          print e.reason  else:      print "OK" |

首先对异常的属性进行判断，以免出现属性输出报错的现象。

以上，就是对URLError和HTTPError的相关介绍，以及相应的错误处理办法，小伙伴们加油！