urllib的使用

`urllib`库包含4个模块：

- [ ] `request`模块：HTTP请求模块；
- [ ] `error`模块：异常处理模块；
- [ ] `parse`模块：一个URL处理的工具模块；
- [ ] `robotparser`模块：用来识别网页的robots.txt文件，然后判断哪些网页可以爬取。

一、request模块

1、urlopen： `urlopen(url, data, timeout)`

- `url`：定位要爬取的网页
- `data`：要提交给服务器的参数。默认为`None`，表示使用GET请求。如果提供了数据，将**使用POST请求**，数据需要是字节类型（`bytes`）

  `data = bytes(urllib.parse.urlencode({'name':'germey'}), encoding = 'utf-8')`
- `timeout`：用于设置超时时间。

2、Request： 利用 `urlopen` 方法可以发起最基本的请求，但往往需要在请求添加一些 `headers` 信息，这就需要使用 `Request` 类来构建请求。Request的构造方法如下：

`class urllib.request.Request(url, data, headers={}, origin_req_host, unverifiable, method)`

- url：用于请求URL，是**必传参数**；
- data：如果要使用这一参数，**必须是bytes类型。**如果数据是字典，可以先使用parse.urlencode方法进行编码；
- headers：一个字典，这就是请求头，最常见的方法就是添加**User-Agent**；
- origin_req_host：请求方的host名称或者IP地址；
- unverifiable：用于指明请求是否是无法验证的，默认值为False（用户没有足够的权限来接收这一请求的结果）；
- method：一个字符串，用于指示请求使用的方法。

3、一些高级用法： 我们已经可以构建起请求了，那么对于一些**高级操作**（验证、Cookie、代理）的需求如何处理？由此引入Handler类

- HTTPBasicAuthHandler：用于管理**验证**，如果一个链接在打开时需要认证，那么可以用这个类来解决认证问题；
- HTTPCookieProcessor：用于**处理Cookie**（此时，必须要声明一个CookieJar对象，然后利用HTTPCookieProcessor来构建Handler）；
- ProxyHandler：用于**设置代理**，代理默认为空；
- HTTPPasswordMgr：用于管理密码。维护着用户密码的对照表；

- HTTPDefaultErrorHandler：用于处理HTTP响应请求，所有错误会给出HTTPError类型的异常。

4、在建起Handler之后，通常需要再**通过build_opener方法构建Opener**，以此完成网页信息爬取。

二、error模块

通过合理地捕获异常，可以更准确的做出异常判断，是程序更加稳健。

1、URLError：URLError类继承自OSError类，是error异常模块的基类，由request模块产生的异常都可以通过捕获这个类来处理。它有一个属性reason，即：返回错误的原因。

2、HTTPError：URLError类的子类，专门处理HTTP请求错误，它有三个属性：

- [ ] code：返回HTTP状态码
- [ ] reason：返回错误原因（有的时候，reason属性返回的不一定是字符串，也有可能是是对象）
- [ ] headers：返回请求头

由于HTTPError是URLError的子类，因此在捕获异常时，通常**先选择捕获子类的错误，然后再捕获父类的错误，最后用else语句处理正常的逻辑。**

三、parse模块

具体的操作见《Python3 网络爬虫开发实战》第40页

- urlparse：解析URL，按照URL标准将URL进行识别和分段；（注意：urlparse 有三个参数）
- urlunparse：合成URL，要求参数必须是可迭代对象，且**长度必须为6**
- urlsplit：与urlparse类似，但不单独解析params字段
- urlunsplit：与urlunparse类似，要求参数必须是可迭代对象，且**长度必须为5**
- urljion：对URL的拼合
- urlencode：用于将字典转化为URL参数，广泛应用于构造GET请求参数时（序列化）
- parse_qs：反序列化，将URL参数转回字典
- parse_qsl：将URL转为由元组组成的列表
- quote：将内容转化为URL编码格式，通常用于URL带有中文参数时
- unquote：对URL进行解码

四、Robots协议与robotparser模块

1、Robots协议：

Robots协议又称网络爬虫排除协议（Robots Exclusion Protocol），用于告诉爬虫哪些网页可以抓取、哪些不可以。通常是一个叫作robots.txt文件，存放在网页的根目录下。

在搜索爬虫访问一个页面时，**首先会检查这个站点根目录下是否存在robots.txt文件**。如果存在，则根据其中定义的爬取范围进行爬取；如果不存在，则访问索引可直接访问的页面。

2、robotparser模块：

该模块可根据 robots.txt 文件判断爬虫是否有权限爬取这个网页。

- set_url：设置 robots.txt 的链接
- read：读取 robots.txt 文件并进行分析，这个方法是在声明 RobotFileParser 类之后**必须调用**的。
- can_fetch：有两个参数——`（Uer-Agent，URL）`，用于判断 User-Agent 是否可以抓取这个 URL。
- parse：用于解析 robots.txt 文件。
- mtime：返回上次抓取和分析 robot.txt 文件的时间。
- modified：将当前时间设置为上次抓取和分析 robots.txt 文件的时间。