Interactuar con Webs

Antonio Espín Herranz

Interactuar con Webs

Módulos

urllib.request

urllib.error

Interactuar con Webs

- **urllib.request**
  - Es el módulo de python que dispone para conectar con URLs.

- **Funciones**:
  - **urllib.request.urlopen()**
    - Crear un objeto parecido a un fichero con el que leer de la URL.
    - Este objeto cuenta con métodos como read, readline, readlines y close, los cuales funcionan exactamente igual que en los objetos file, (por debajo trabaja con un socket).
    - Lo que recibimos viene en bytes si queremos indexar o preguntar por esos datos, aplicar el método decode('utf-8')

```
import urllib.request
with urllib.request.urlopen('http://python.org/') as response:
   html = response.read()
   print(html[:1000].decode('utf-8')+' ...')
   print("\nURL:", response.geturl())
```

Interactuar con Webs

- **urllib.request.urlretrieve**
  - Podemos recuperar un recurso y almacenarlo en una ubicación temporal.
  - Consultar las cabeceras.

```
import urllib.request
local_filename, headers =
urllib.request.urlretrieve('http://python.org/')
html = open(local_filename)
print(headers)
```

Interactuar con Webs

- **urllib.parse.urlencode()**
  - Codificar los parámetros de una URL.
  - Cuando necesitemos enviar parámetros a un URL

```
import urllib.parse
import urllib.request

url = 'http://www.someserver.com/cgi-bin/register.cgi'
values = {'name' : 'Michael Foord',
          'location' : 'Northampton',
          'language' : 'Python‘ }
data = urllib.parse.urlencode(values)
data = data.encode('ascii') # data should be bytes
req = urllib.request.Request(url, data)
with urllib.request.urlopen(req) as response:
    the_page = response.read()
```

Interactuar con Webs

- Para manejo de excepciones tenemos el módulo **urllib.error**

- Excepciones:
  - URLError:
    - Se lanza cuando no hay conexión a la red.
    - O una URL no existe.
    - Ejemplo:

```
>>> req = urllib.request.Request('http://www.pretend_server.org')
>>> try:  urllib.request.urlopen(req)
... except urllib.error.URLError as e:
 ...  print(e.reason)
...  (4, 'getaddrinfo failed')
```

Interactuar con Webs

- HTTPError:
  - Cada respuesta HTTP del servidor contiene un código de estado que indica un posible problema en el server, o si todo ha ido bien.
  - Dispone de un diccionario con todos los códigos de error de la RFC 2616
  - http.server.BaseHTTPRequestHandler.responses
  - responses = {
  - 100: ('Continue', 'Request received, please continue'),
  - 101: ('Switching Protocols', 'Switching to new protocol; obey Upgrade header'),
  - **200**: ('OK', 'Request fulfilled, document follows'),
  - **404**: ('Not Found', 'Nothing matches the given URI'), …

Ejemplo

```
from urllib.request import Request, urlopen
from urllib.error import URLError, HTTPError
req = Request(someurl)

try:
    response = urlopen(req)
except HTTPError as e:
    print('The server couldn\'t fulfill the request.')
    print('Error code: ', e.code)
except URLError as e:
    print('We failed to reach a server.')
    print('Reason: ', e.reason)
else:
    # everything is fine
```

info / geturl

- Métodos del objeto que devuelve **urllib.request.urlopen:**
  - **geturl** ()
    - Devuelve la url real (puede que hubiera redirección).
  - **info()**:
    - Devuelve información de las cabeceras enviadas por el servidor.
    - Van en un diccionario.

Handlers

- Instalación de **handlers**:
  - Para personalizar la forma en que trabaja **urllib.request** podemos **instalar un grupo de manejadores** (handlers) agrupados en un objeto de la clase **OpenerDirector** , que será el que se utilice a partir de ese momento al llamar a urlopen.
  - **OpenerDirector** cuenta con handlers que se encargan de manejar los esquemas disponibles (**HTTP, HTTPS, FTP**), **manejar la autenticación**, **manejar las redirecciones**, etc.
  - Llamar al método **add_handler**(“…”)

Handlers

- *# Ejemplo:* ***HTTPBasicAuthHandler***
- ***# Crear un gestor de password***
- password_mgr = urllib.request.HTTPPasswordMgrWithDefaultRealm()

- *# Añadir usuario y password*
- *# Indicar el realm si lo conocemos, en otro caso utilizar* ***None****.*
- top_level_url = "http://example.com/foo/"
- password_mgr.add_password(**None**, top_level_url, username, password)
- handler = urllib.request.HTTPBasicAuthHandler(password_mgr)

- *# Crear el opener*
- opener = urllib.request.build_opener(handler)

- *# Utilizar el opener para acceder a la URL*
- opener.open(a_url)

- *# Instalar el opener.*
- *# Ahora llama a urllib.request.urlopen utilizando el openner.*
  urllib.request.install_opener(opener)