# ウェブ上の情報を抽出する方法

パイソンは、ウェブサイトにあるデータを抽出するために、″ urlibライブラリ″ を使用します。 このライブラリを使うと、HTTPまたはFTPを使ってデータをダウンロードすることができます。 urllibはURLを扱うモジュールを集めたパッケージです。 その中でもurllib.requestモジュールはウェブサイトにあるデータにアクセスする機能を提供します。その中でもurllib.requestモジュールはウェブサイトにあるデータにアクセスする機能を提供します。 また、認証、リダイレクト、クッキー(Cookie)のようにインターネットを利用した様々な要請と処理をサポートします。

# urllib.requestを利用したダウンロード

まず、ウェブサイトからファイルをダウンロードする方法を確認しましょう。

ファイルをダウンロードする際はurllib.requestモジュールにあるurlretrieve()関数を使います。

この関数を用いると、直接ファイルをダウンロードできます。

次のコードはウェブにあるPNGファイルを"test.png"という名前のファイルとして保存した例です。

In [4]:
```
#ライブラリの読み込み
import urllib.request

#URLと保存経路を指定する
url = "http://uta.pw/shodou/img/28/214.png"
savename = "test.png"

#ダウンロード
urllib.request.urlretrieve(url, savename)
print("保存されました")
```

保存されました

パイソンライブラリを使うためにはimport構文を使ってライブラリを読み込まなければならないです。 プログラムではurllib.requestモジュールを読み込みます。 "urllib.request"のように、" "で区分されたモジュールを指定したのは、"urllibパッケージ内部にあるrequestモジュール"という意味です。

プログラムの2番からファイルをダウンロードします。 urlretrieve()の最初の媒介変数にURLを、2番目の媒介変数に保存するファイルの経路を指定します。パイソンを使ったらファイルをダウンロードする時、このように数行のコードだけあれば良いです。

# urlopen()でファイルに保存する方法

先の例題ではrequest.urlretrieve()関数を使ってファイルにすぐに保存しましたが、今回はrequest.urlopen()の使い方を紹介します。 request.urlopen()を使えば、すぐにファイルとして保存するのではなく、データをパイソンメモリーの上に乗せることができます。

それではrequest.urlopen()を使ってメモリーの上にデータをアップロードして、その後ファイルを保存してみます。 この過程はデータを抽出し、ファイルに保存する流れに従って進められます。

In [6]:
```
import urllib.request

 #URLと保存経路指定する
url = "http://uta.pw/shodou/img/28/214.png"
savename = "test1.png"

 #ダウンロード
mem = urllib.request.urlopen(url).read()

 #ファイルに保存する
with open(savename, mode = "wb") as f:
    f.write(mem)
    print("保存されました")
```

保存されました

1) ではurlopen() 関数でPNG ファイルのURL リソースを開きます。 続いてread()メソッドでデータを読み込みます。 そして(2)では、ファイルを開くopen()関数でファイルを開きます。 この時、ファイルの読み書きモードを表すmodeを"wb"でファイルを開きます。 "w"は書き込みモード、"b"はバイナリモードを意味します。

そしてwrite()メソッドでダウンロードしたバイナリデータをファイルに保存します。

# ウェブでデータを抽出

続いて、ウェブからXMLまたはHTMLなどのテキストベースのデータをダウンロードする方法をご紹介します。 今回は簡単なダウンロード例として、筆者が運用しているウェブAPIを使ってみましょう。

## http://api.aoikujira.com/ (http://api.aoikujira.com/)

# クライアント接続情報を出力

まず、基本的な使い方からみてみましょう。次はIPアドレス、UserAgentなどのクライアント接続情報を出力する"IP確認API"にアクセスして情報を抽出するプログラムです。

In [10]
```
#IP確認APIにアクセスして結果を出力する
#モジュール読み込み - 1
import urllib.request

#データ読み込み - 2
url = "http://api.aoikujira.com/ip/ini"
res = urllib.request.urlopen(url)
data = res.read()

#バイナリ文字列に変換する - 3
text = data.decode("utf-8")
print(text)
```

```
[ip]
API_URI=http://api.aoikujira.com/ip/get.php
REMOTE_ADDR=223.39.188.57
REMOTE_HOST=223.39.188.57
REMOTE_PORT=56344
HTTP_HOST=api.aoikujira.com
HTTP_USER_AGENT=Python-urllib/3.7
HTTP_ACCEPT_LANGUAGE=
HTTP_ACCEPT_CHARSET=
SERVER_PORT=80
FORMAT=ini
```

結果を確認したらプログラムを見てみましょう。 プログラム(1)ではurllib.requestモジュールを読み込む。

続いてプログラムの(2)ではrequest.urlopen()メソッドを呼び出す。 そしてread()メソッドを使ってデータを読み込む。read()メソッドで読み込んだデータはバイナリデータである。 従って、プログラム(3)でdecode()メソッドを使ってバイナリを文字列に変換する。 文字列に変換した後にprint()関数で標準出力にデータを出力する。

もしFTP上のリソースを抽出したければ、request.urlopen()に指定したURLを"http://"から"http://"から"ftp://"に変更すればよいです。

# 媒介変数を追加してリクエストを送信する方法

続いてURLに媒介変数を追加してリクエストを送信する方法を確認してみましょう。 今回の例題では気象庁のRSSサービスを使ってみます。 気象庁RSSは、以下のURLに市外局番を指定すると、該当地域の情報を提供してくれます。
**http://www.kma.go.kr/weather/forecast/mid-term-rss3.jsp (http://www.kma.go.kr/weather/forecast/mid-term-rss3.jsp)**

この時、市外局番は媒介変数に指定します。

パイソンでリクエスト専用の媒介変数を作る時は、 urllib.parserモジュールのurlencode()関数を使って媒介変数をURLエンコードする。では、実際に市外局番を使って気象情報を持ってくるプログラムを作ってみましょう。

In [15]:
```
import urllib.request
import urllib.parse

API = "http://www.kma.go.kr/weather/forecast/mid-term-rss3.jsp"
 # 媒介変数をURLエンコードする - 1
values = {


}
params =  urllib.parse.urlencode(values)
 # リクエスト専用URLを作成する - 2
url = API + "?" + params
print("url=", url)

  #ダウンロードする - 3
data = urllib.request.urlopen(url).read()
text = data.decode("utf-8")
print(text)
```

```
url= http://www.kma.go.kr/weather/forecast/mid-term-rss3.jsp?
<?xml version="1.0" encoding="utf-8" ?>
<rss version="2.0">
<channel>
<title> 気象庁陸上中期予報 </title>
<link>http://www.kma.go.kr/weather/forecast/mid-term_01.jsp</link>
<description> 気象庁天気ウェブサービス </description>
<language>ko</language>
<generator>気象庁</generator>
<pubDate>2020年 05月 09日 (土)曜日 06:00</pubDate>
 <item>
<author>気象庁</author>
<category>陸上中期予報</category>
<title>全国陸上中期予報 - 2020年 05月 09日 (土)曜日 06:00 発表</title>
```

```
<link>http://www.kma.go.kr/weather/forecast/mid-term_01.jsp</link>
<guid>http://www.kma.go.kr/weather/forecast/mid-term_01.jsp</guid>
<description>
        <header>
                <title> 全国陸上中期予報 </title>
                <tm>202005090600</tm>
                <wf><![CDATA[○(降水)15日(金)は済州島と南部地方、忠清道に雨が始まり16日(土)は全国で雨が降ります。 <br />○(気温)19日(火)までの日中の気温は、昨日(17~29度)と同じ18~28度の分布となります。<br />                一方、13日(水)~15日(金)と17日(日)~19日(火)の内陸を中心に25度以上上がるところが多く、暑いですが、 <br />                16日(土)は、雨により日中の気温が上がらず、20~24度の分布となります。]]></wf>
        </header>
        <body>


                <location wl_ver="3">
                        <province> ソウル·仁川·京畿道 </province>
                        <city>ソウル</city>

                        <data>
                                <mode>A02</mode>
                                <tmEf>2020-05-12 00:00</tm
Ef>
                                <wf>晴れ</wf>
                                <tmn>12</tmn>
                                <tmx>21</tmx>
                                <reliability></reliability
>
                                <rnSt>10</rnSt>
                        </data>


                        <data>
                                <mode>A02</mode>
                                <tmEf>2020-05-12 12:00</tm
Ef>
                                <wf>晴れ</wf>
                                <tmn>12</tmn>
                                <tmx>21</tmx>
                                <reliability></reliability
>
                                <rnSt>10</rnSt>
                        </data>


                        <data>
                                <mode>A02</mode>
                                <tmEf>2020-05-13 00:00</tm
Ef>
                                <wf>晴れ</wf>
                                <tmn>12</tmn>
                                <tmx>24</tmx>
                                <reliability></reliability
```

```
                                        <data>
                                                <mode>A01</mode>
                                                <tmEf>2020-05-18 00:00</tm
        Ef>
                                                <wf>曇り</wf>
                                                <tmn>17</tmn>
                                                <tmx>22</tmx>
                                                <reliability></reliability
        >
                                                <rnSt>30</rnSt>
                                        </data>


                                        <data>
                                                <mode>A01</mode>
                                                <tmEf>2020-05-19 00:00</tm
        Ef>
                                                <wf>曇り</wf>
                                                <tmn>17</tmn>
                                                <tmx>22</tmx>
                                                <reliability></reliability
        >
                                                <rnSt>40</rnSt>
                                        </data>


                        </location>



                </body>
        </description>
        </item>
        </channel>
        </rss>
```

プログラムの(1)ではディクショナリー資料型の媒介変数をURLエンコードします。 URLエンコードのためurllib.parseモジュールを使います。 続いて、プログラムの(2)では、リクエスト専用URLを作成し、標準出力にURLを出力します。URLエンコード結果が上記の実行結果の最初に出力されたものです。 このように要請を送信するときは次のようなURLをお送りします。

URLの最後に"?"を入力し、"="の形式で媒介変数を作成します。 複数の媒介変数を使用するときは、"+"を使用して区別してくれます。

実は今回の例題では媒介変数として簡単な数字を使用するので、 複雑な処理を使わず、媒介変数を直接URLに指定して使ってもいいです。しかし、媒介変数に韓国語などが含まれている場合、URLエンコードが必要です。

# 媒介変数をコマンドラインで指定

しかし、現在のプログラムは媒介変数をコードに入力しなければならないので、他の地域の情報を調べたい場合はプログラムを開いて修正しなければなりません。 このような煩わしい過程を経ずに、コマンドラインからすぐに市外局番を入力して使うことができたらどうでしょうか？

In [19]:
```
#!/usr/bin/env python3

#ライブラリを読み込む--1
import sys
import urllib.request as req
import urllib.parse as  parse

#コマンドライン媒介変数抽出--2
if len(sys.argv) <= 1:
    print("USAGE: download-forecast-argv <Region Number>")
    sys.exit()
regionNumber = sys.argv[1]

#媒介変数をエンコード--3
API = "http://www.kma.go.kr/weather/forecast/mid-term-rss3.jsp"
values = {
    'stnId' : 'regionNumber'
}

params = parse.urlencode(values)
url = API + "?" + params
print("url=", url)

#ダウンロードする--4
data = req.urlopen(url).read()
text = data.decode("utf-8")
print(text)
```

```
url= http://www.kma.go.kr/weather/forecast/mid-term-rss3.jsp?stnId
=regionNumber
```

*python*3*do*r*nload* − *forecast* − *argu*108python3 download-forecast-argv 109 $python3 download-forecast-argv 184

入力したコマンドラインの媒介変数によって結果が異なります。

それではパイソンプログラムを確認してみましょう。プログラム(1)ではライブラリを読み込みます。コマンドライン媒介変数を抽出するときに使うsysモジュールを読み込みます。 また、import構文にasを指定してモジュールを好きな名前で使えるようにしました。 現在のコードではurllib.requestをreqという名前で、urllib。parseはparseという名前で宣言しました。 このようにモジュールに別称(alias)を定義すると、プログラムを作成する際にキーボードを入力する回数を減らすことができます。

続いてプログラムの(2)を見てみましょう。 ここでは、コマンドラインの媒介変数を調整します。 コマンドライン媒介変数は、sys.argvにリスト形式で入ります。 sys.argv[0]にはスクリプトの名前、sys.argv[1]以降はコマンドライン媒介変数が設定されます。 いくつかの媒介変数が指定されているかは、len(sys.argv)のようにlen()関数を使って確認することができます。 そして、パイソンプログラムを中断するときは、sys.exit()またはquit()関数を使います。

プログラムの (3) では、URL 媒介変数を生成するために、URL エンコードします。 実は数字で入力するのであまり必要はありませんが、韓国語などを使う時は必要なので覚えておいた方がいいです。 この部分以外は以前に作ったプログラムとまったく同じプログラムです。 プログラムの(4)からデータを抽出し、結果を標準出力に出力します。