目次へ

WEB編 何が起こっているか

直前に見てもらった「hello WEB APP world」がどういう理屈で動いたのかを確認していきます。

webサーバー（ここでは AN HTTPD）が、webブラウザ（Internet Explorer とか Firefox）に「hello WEB APP world」というメッセージを送りました。これは目で見て確認できる通りです。今回はwebサーバーを自分のコンピュータの上で動かしましたが、実際は、ネットワークコンセントの向こうにある、どこか離れたコンピュータ（サーバー）と通信しているので、そこはご注意を。

webサーバーってのは、一番の基本は、すでに出来上がったページとか画像をwebブラウザに渡すというのがその役目です。下のようなテキストファイルを作って、先のスクリプトを置いた cgi-bin フォルダに一緒に置いてください。

hello.txt

```
print "hello"
```

そしたら下のURLにアクセスしてみましょう。（もちろん、AN HTTPD を動かしている状態で試してね）

http://localhost/cgi-bin/hello.txt [http://localhost/cgi-bin/hello.txt]

そのままの内容がブラウザ上には出てきましたね。別にこれがpythonスクリプトのように扱われて、hello とのみ表示されるわけではありません。あるがままの内容が渡されます。

次に、下の内容を持つファイルを同じフォルダに作って置いてください。今度は hello.html です。

hello.html

```
<html>
<body>
<h1>hello</h1>
</body>
</html>
```

で、アクセスするURLは、たぶん予想できると思いますが、下の通り。

http://localhost/cgi-bin/hello.html [http://localhost/cgi-bin/hello.html]

今度は内容がちゃんとHTMLとみなされて、太字で hello と出てきましたね。これも、HTML自体はあるがままにサーバーから送られてきて、ブラウザのほうはこれを「こっち側で」整形したというだけのことです。

.cgi で終わるファイルだけは、あるがままにブラウザに渡されません。かわりに、それはwebサーバーの中で「実行」され、その出力結果がブラウザに送られるという仕掛けになっているのです。

AN HTTPD の設定で、.cgi だったら云々、という設定を行いましたね。あれがその仕掛けをした部分です。別に .cgi じゃなくても、.app だったら実行、とか、.xyz だったら実行、とか、そんな風に決めたって構わないのですが、まあ、慣習的に .cgi が「実行」すべき対象、ということにするのが普通ですので、これに従っています。

一般にこういう仕組みをCGIスクリプトって呼んだりもしますしね。

で、CGIスクリプトのもうひとつの特徴は、一行目にあります。前回動かしたのは、こんな風じゃなかったですか。

```
#!C:¥python26¥python.exe
```

「#」で始まるのだから、pythonにとっては意味のないコメントに過ぎませんが、AN HTTPDなどのウェブサーバーにとっては、この一行目は大事なものです。これが、このファイルをCGIスクリプトとみなしたときに、何で実行するかを指定する機能を持つからです。

今まで、.py で終わるスクリプトは、windowsが自動的に python.exe という「スクリプト実行用プログラム」によって実行していました。この自習テキスト内のやりかたに従ううちは、特に意識することはなかったですけどね。.cgi では何で実行したらいいのかすぐには分かりませんから、明示的に「このスクリプトはpythonで実行してね」と指定したというわけです。

「#!」までが、「実行するプログラムは」といった意味の目印で、それ以降の C:\python26\python.exe がそのプログラムの具体的な場所です。python以外にも、perl とか ruby とかいろんなスクリプト言語が世の中にはあるんですが、それらがインストールされているなら、そいつの場所をこの一行目に書けば、その言語で動くCGIスクリプトも書けるんですよ。（一昔前は、perl でCGIを書くのが圧倒的な流行でした。今は選択肢が増えたので、昔よりは少ないね。perlも使い慣れるとそれなりの楽しさがあるよ。）

さて、こういうわけで、前回書いた hello.cgi がスクリプトとして「webサーバーの中で」実行されて、その実行結果がwebブラウザに届くんだというところを説明したつもりですが、よろしいでしょうか。

hello.cgi をpythonスクリプトして実行すると、（試すまでもないでしょうが）下のような実行結果になります。

```
Content-Type: text/plain
hello WEB APP world
```

しかるに、ブラウザには、hello… の部分だけが出てきて、最初の二行分（Content… というところと、それにつづく空行）は表示されませんね。これもまた、CGIスクリプトを書く上で注意しなければいけない部分のひとつです。

Content… のような部分を、「レスポンスヘッダー」と呼びます。電子メールの話を読んでくれた方は、ああ、メールヘッダーに似た感じだな、と思いついてくれてもいいですね。CGIスクリプトは、実行した結果として、必ずこのレスポンスヘッダーを最初に print しなくてはいけません。（ちょっと専門用語を使うと、レスポンスヘッダーを標準出力に出力しなくてはいけません。うん、この用語は今のところ忘れてもいいですけど）

具体的に、Content-Type: text/plain がどういう意味のかというと、これは、「.txt」がついているときと同じようなテキストファイルとみなしてください、というメッセージを表しています。CGIスクリプトは、出力結果が「テキストファイル」なのか「HTMLファイル」なのか、または他の形式のファイルなのか、といった説明を、レスポンスヘッダで行います。

ちなみに、この出力結果はHTMLとして扱ってくれ、というときには、 Content-Type: text/html と出力します。出力結果がJPEG画像だぞ、というとき（ありえない話じゃないんですよ）は、Content-Type: image/jpeg です。まあ、普通はテキストかHTMLのどっちかでしょうか。

今回はこんなところです。CGIスクリプトというものの説明をしました。CGIスクリプトは、下の特徴を持ったものです。

- 特定の拡張子（典型的には .cgi）を持っていて、webサーバーがこれをCGIとみなすように設定してある
- 一行目に、自分自身が何を使って実行すればいいのかが指定してある
- 出力結果の先頭に、レスポンスヘッダと呼ばれるものが付加される

CGIをpythonで書けるとすると、pythonでできることは何でもWEBアプリケーションの一部として使えることになります。今まで練習問題でやってきたことは、ほとんどすべてCGIとして動くように書き直すことができます。だから、HTMLの自動生成みたいなやつをこの枠組みの中で行うと、それはとりもなおさずWEBアプリですね。

とはいえ、これだけで実用的なCGIが書けるかというと、実はまだ足りません。次の回は、CGIスクリプトそのものに、「引数」を与えて色々な動作をさせられるようにします。でもこの時点でひとつ練習問題をやってみましょう。

練習問題W01

【練習問題:W01】「九九の表」を表示するCGIスクリプト(kuku.cgi)を書け。出力結果はHTMLとして整形されること。

HTMLを自動生成するのはすでにいろいろとこなしてくれていますから、ちょっと頭をひねれば書けると思います。あとはこれに適当なレスポンスヘッダもくっつければいいということですね。

スクリプトそのものの実行結果は、下のようなものになるはずですよ。

```
Content-Type: text/html

<html>
<head>
<title>9 x 9</title>
<body>
...
```

スクリプトを書きながら、その途中経過もWEBブラウザの上で試しながら進めていくといいと思います。つまり、最初はレスポンスヘッダとテスト出力だけを行うCGIスクリプトを書いて、それを実際のCGIとしてwebサーバーの上で実行する。で、ちょっとスクリプトを書き換えて、セーブして、ブラウザの上でリロードボタン（またはF5キー）を押して最新の実行結果を確認する。こんな感じ。そうすると、コマンドプロンプトの狭い画面で確認するよりも広々とした出力結果が見られますからね。

あ、最初のうちは、レスポンスヘッダを Content-Type: text/plain ではじめるといいかも知れませんね。そこでまともそうなHTMLが表示できるようになってきたと思ったら、ある時点で、Content-Type: text/html に切り替えて実際の表示を試す、という手順が踏めますからね。

- Windowsで使えるショートカットキーで、[Alt] + [Tab]をご存知ですか。Altキーを押しながらTabを数回押すと、マウスを使わずに使うウィンドウを切り替えられます。エディタとブラウザをこれで行ったり来たりしながらコードを書き進めると、きっと軽快ですよ。

あ、あと、日本語を使いたいときに、今まで書いていた # coding:c932 みたいなのをどこに書けばいいんだ、と迷ったりするかも知れませんね。今回は別に日本語が入ってなくてもいいですけど、でも使いたいんだ、というときは、スクリプトの一行目はCGI用のアレを書いて、二行目にエンコードを示す coding:云々 ってのを書けばいいです。スクリプトの例で示すなら、下のような感じにすればいいですよ。

```
#!C:¥python26¥python.exe
# coding: cp932
print "なんちゃらかんちゃら…"
```

もうひとつだけ。今回の問題では多分二重ループを使うことになるんじゃないかと思います。なんのことはない、ループの中身でさらにループを書くわけですが、たしかまだこの技法が必要な問題ってなかったんじゃないかと思います。下のサンプルを見てください。

```
for i in xrange(9):
  for j in xrange(9):
    print "%d * %d = %d" % ((i+1), (j+1), (i+1)*(j+1))
```

外側のループは i を使ってループしています。その内側には、j を使ったループを書きます。そのループの中は、i と j を使って処理を書けますよ。

出題した後のヒントが今回は多いね。