ビギナー向けの真珠みがき2

お手軽CGIとHTMLテンプレート

Kansai.pm 2周年記念イベント

舟木 隆康 2002/5/11

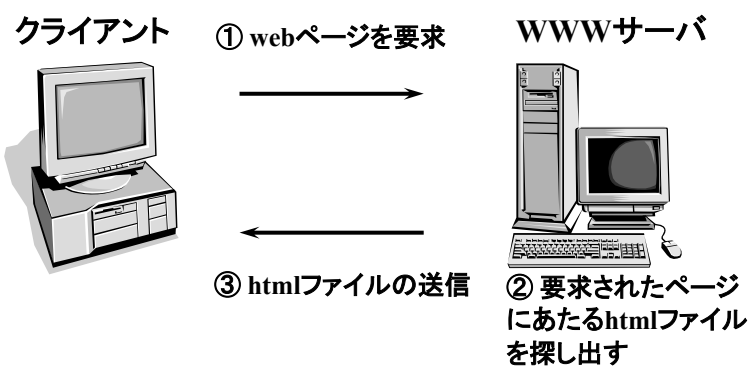
本日の真珠みがき

- CGIとは何か
- CGI.pmの使い方
- HTML::Templateの使い方
- 構造体とLoop処理
- etc...

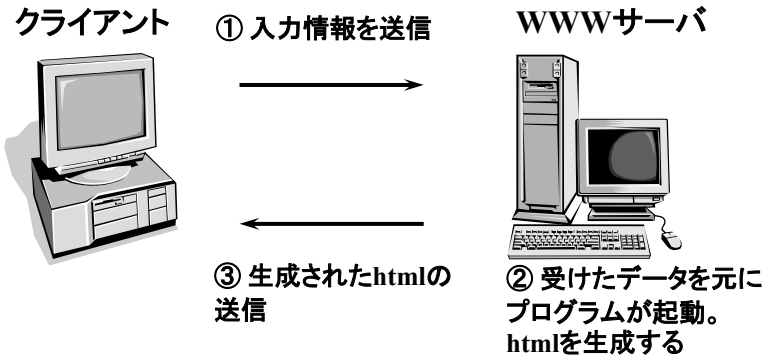
インターネットを見る？

- もう少し正確に言うと…
- インターネット上のWWWサービスを利用してWebページをブラウズする。
- WebページはWWWサーバにより提供される。

webページのブラウズ



CGIって？



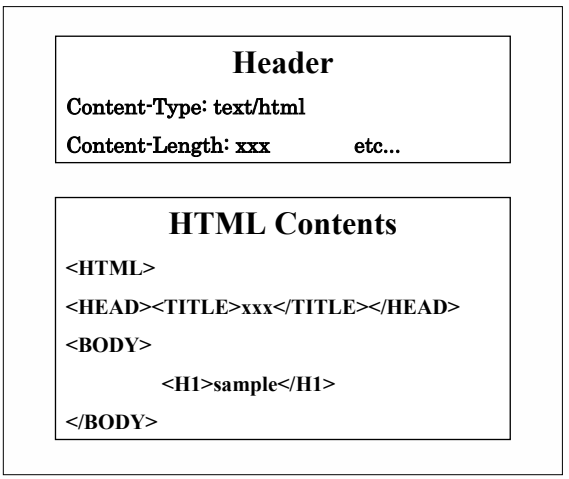
CGIとは何か？

- 簡単に説明すると、入力された情報を元にWWWサーバがプログラムを実行、htmlの結果を出力する仕組みの事。
- 情報を入力しなくてもプログラムを実行させてhtmlを出力させることもできる。

CGIには何が必要か

- WWWサーバ
- CGIプログラム
  - 言語は動くなら何でもOK
  - ただ、Perlが広く使われているようだ
- という訳でPerlでやってみよう！

WWWサーバの送信内容



CGI.pmを呼び出す

- CGI.pmモジュールはVer5.004以降のPerlに標準装備されている、とても奥が深いモジュール
- 呼び出し方
  - use CGI qw(:standard);
- 最近のVersionのCGI.pmがあればついでに…
  - use CGI::Carp qw(fatalsToBrowser);
    - CGI::Carpはエラーページを自動で生成してくれる。エラー発生時に便利

オブジェクト指向？

- CGI.pmは標準的な構文とオブジェクト指向構文がある。
- オブジェクト指向構文の方が動作が若干早い。
  - 今回のサンプルはオブジェクト指向構文。

ヘッダ部分を出力

```
use CGI qw(:standard);
use CGI::Carp qw(fatalsToBrowser);
my $q = new CGI;
print    $q->header(“text/html”);
print    << ‘EOF’;
<html><head><title>test</title></head>
<body>Just Testing</body> </html>
EOF
```

ヘッダ出力結果

■ 実行結果

```
Content-Type: text/html

<html>
<head><title>test</title></head>
<body>Just Testing</body>
</html>
```

- （見た目の都合のため、改行しています。）

便利なヘッダ

■ メディアタイプ

```
print $q->header(“text/plain”);
print $q->header();  など
```

– 空だとtext/htmlになる

■ リダイレクト

```
print $q->redirect(“http://hoge/hoge.html”)
```

– 対応しているブラウザであれば、自動でリダイレクトしてくれて、webページでリダイレクトするより早い
– 利用する際は他のヘッダと併用しないこと

Html出力をさせるには

```
use CGI qw(:standard);
my $q = new CGI;
print    $q->start_html(“Hello World!!”),
         $q->p(“Welcome to CGI.pm”),
         $q->end_html;
```

- (CGI::Carpは省略しました)

Html出力結果

■ 実行結果

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN">

<HTML>
<HEAD><TITLE>Hello World!!</TITLE></HEAD>
<BODY>
<P>Welcome to CGI.pm</P>
</BODY>
</HTML>
```

CGI.pmの代表的なタグ集

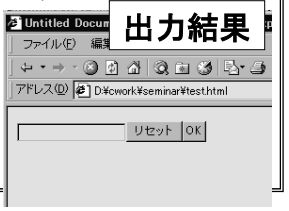
<html>…	$q->start_html;
…</html>	$q->end_html;
<hr>タグ	$q->hr;
<p>タグ	$q->p(“xxx”);
<a href>タグ	$q->a({-href => “xxx”}, “download”);
<table>タグ	$q->table(省略);
<form>タグ	$q->start_form(省略); ⇔ $q->end_form
<input type =“text”>	$q->textfield(省略)
<input type = “submit”>	$q->submit(省略)
<input type = “hidden”>	$q->hidden(省略)
	etc…

CGI.pmで入力画面を作る

– start_form時に値の送信先を指定する

```
use CGI qw(:standard);

my $q = new CGI;

print $q->header();
print $q->start_html();
print $q->start_form(-method => “post”,
                     -action => “値の送信先アドレス”);
print $q->textfield(-name => “user_id”);
print $q->reset ();
print $q->submit (-name => “submit”,
                  -value => “OK”);
print $q->end_form();
print $q->end_html();
```

出力結果



CGI.pmで値の取得

```
$hoge = $q->param(“パラメータ名”);
```

- この方法でGetやPostされた全ての情報が取得できる。
  - text、hiddenフィールド、radio-box、check-box、button、クリッカブルマップなど
  - ちなみにcookieは別。リクエストヘッダで返って来るので。

CGI.pmで動的ページ生成

```
use CGI qw(:standard);
my $q = new CGI;
my $hoge = $q->param(“user_id”);
print $q->header();
print $q->start_html();
print $q->p(“Your user ID is $hoge!”);
print $q->end_html();
```

HTML::Templateとは

- htmlでデザインされたファイルをベースに情報を埋め込んでhtml出力させる。
  - デザインとプログラムが分離できるので便利
  - htmlファイルにコメントとして埋め込む

コマンド一覧

```
<!-- TMPL_VAR      NAME=“変数名” -->
<!-- TMPL_LOOP     NAME=“変数名” -->
<!-- TMPL_IF       NAME=“変数名” -->
<!-- TMPL_UNLESS NAME=“変数名” -->
<!-- TMPL_ELSE -->
```

Templateファイル

```
<html><title>グラフサンプル</title>
<body bgcolor="#CCCCFF">
<h3>テンプレートでグラフを作ってみる</h3>
<hr width="400" align="left" size="1">

<!-- TMPL_IF NAME="graph_name" -->
<b>
<!-- TMPL_VAR NAME="graph_name" -->
</b>
<!-- /TMPL_IF -->

<table width="400" border="0">
 <tr bgcolor="#FFFFFF">
  <td>
   <table width="350" border="1" align="center" bordercolor="#FFFFFF">
    <!-- TMPL_LOOP NAME="graph_data"-->
    <tr>
     <td><!-- TMPL_VAR NAME="place" --></td>
     <td width="300">
      <table border="0" bgcolor="#0000FF" width="<!-- TMPL_VAR NAME="ratio" -->%">
       <tr>
.
.
.
```

長いので添付資料を参照

構造体

- HTML::TemplateのLoop処理で必須。
- Tableの一行を表現するとき使ったりする。

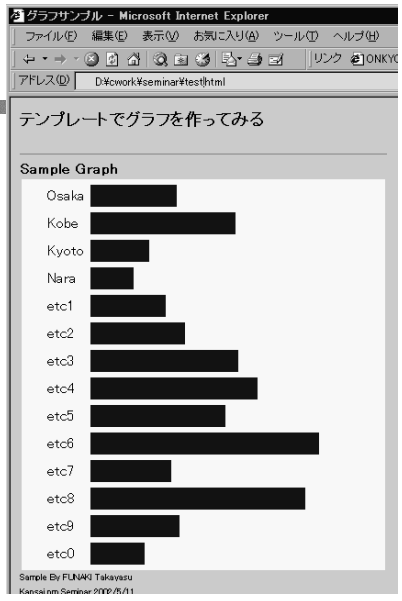
```
@table =(
        {       place   =>      “Osaka”,
                ratio   =>      “32”    },
        {       place   =>      “Kobe”,
                ratio   =>      “54”    },
        {       place   =>      “Kyoto”,
                ratio   =>      “22”    },
        etc…
);
```

HTML::Template利用サンプル

```
#
# graph.pl
#
# Creates graph using HTML::Template
# [ FUNAKI Takayasu 2002/05/10 ]
# --- Last Update 2002/05/10 ---

use strict;
use HTML::Template;
use FindBin qw($Bin);

my $tmpl_file =  $Bin . "/graph.tmpl";
my $tmpl = new HTML::Template ( filename => $tmpl_file );

my $graph_name = "Sample Graph";

my @data = qw( Osaka:32
        Kobe:54
        Kyoto:22
```

添付資料を参照

結果



余談

- Perlの出力は一時的にバッファにためられる。データを即出力させたいなら
  - $| = 1;
- CGI.pmをuseしているスクリプトをコマンドラインで実行させるとパラメータの入力を求められる。この場合EOFを送ってやること
  - Unix系:Ctrl-D, Win系:Ctrl-Z

余談

- CGI::Carpはユーザにエラーが見えるのでマズイ。専用エラーサブルーチンを作成したほうが良いかも。
  - 「CGIプログラミング」の「5.5.3 エラーサブルーチン」参照
- 一つのCGIプログラム内で入力画面の生成とデータ受け取りをさせる場合は画面ごとに$q->delete_all等しなければフィールド内容を引きずるケースがあるような無いような…

参考資料

- CGIプログラミング　第2版
  - Scott Guelich, Shishir Gundavaram, Gunther Birznieks著 田辺 茂也 監訳　大川 佳織 訳
  - オライリー・ジャパン
- CGIモジュール
  - Lincoln D. Stein著　川合孝典 訳
    - http://member.nifty.ne.jp/hippo2000
- HTML::Templateモジュール
  - Sam Tregar著 川合孝典 訳
    - http://member.nifty.ne.jp/hippo2000

みがきあげ

- れたのか？
- ご静聴ありがとうございました。