Web (2)

○ HTML (HyperText Markup Language)
タグによるマークアップ
   ...
attribute
  <img src="http://...../.../logo.gif" ALT="LOGO">
hyperlink
   anchor text
ブラウザの view source を利用した観察
進化
  HTML 1.0 -> ... -> HTML 5.0

○ CSS (Cascading Style Sheets)
本来, HTMLは文書の構造を指示するものであるべき (見栄えではなく)
しかしデザイン上の要求から詳細な指示が必要になった
それを本来の目的どおり分離する仕組み - スタイルシート
  別ファイル(たとえば mystyle.css) でスタイルを指定
  .htmlファイルからそれを読み込むように指定
    <link rel="stylesheet" type="text/css" href="mystyle.css" />

-特定のタグの見栄えを指定
  h1 {font-size:200%;}
-設定に名前をつける
  div.command {
        background-color: #d0ffff;
        border: none;
        margin: 10px;
        padding: 10px;
        font: x-large monospace;
  }
使うときは タグ や タグで範囲を指定
   テキスト

○ Input and Forms
双方向通信 (ユーザからの入力)
- HTTP
  GET/POST メソッドを使用
- HTML
  Form (ユーザインタフェースのためのボタン, ボックス)
  HTML 2.0 から

<form ACTION="http://..../action.cgi" method="POST">
 Please enter your name: <input type="text" name="name">
 Please enter your age: <input type="text" name="age">
<input type="submit">
</form>

<input> タグで入力欄やボタンを作成 (type属性で区別)
submitボタンが押されるとデータがサーバに返される
  データはACTIONで指定したURLに送られる
  その時に使うのが POSTメソッド

データを返す通信を観察
formタグでPOSTメソッドを指定した場合, HTTPのボディでフォームの内容を一行で返信

POST http://pr.ice.uec.ac.jp/~terada/learn/php/action.cgi HTTP/1.1
Host: pr.ice.uec.ac.jp
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:17.0) Gecko/20130917 Firefox/17.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ja,en-us;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Proxy-Connection: keep-alive
Referer: http://pr.ice.uec.ac.jp/~terada/learn/php/test1.html
Content-Type: application/x-www-form-urlencoded
Content-Length: 18

name=Terada&age=54

```
---------------------------------------

下は <form ACTION="http://..../action.cgi" method="GET"> と指定した場合.
GETの場合, 要求するURLのあとに ? でフォーム内容を続ける.
(したがってボディは空)
---------------------------------------
GET http://pr.ice.uec.ac.jp/~terada/learn/php/action.cgi?name=Terada&age=54 HTTP/1.1
Host: pr.ice.uec.ac.jp
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:17.0) Gecko/20130917 Firefox/17.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ja,en-us;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Proxy-Connection: keep-alive
Referer: http://pr.ice.uec.ac.jp/~terada/learn/php/test2.html

---------------------------------------

返信されたフォームの情報をサーバ側でどのように受けとるか

○ Server-Side Dynamic Web Page Generation
form -> POST/GET -> server
処理に二つの方法 - CGI, PHP

CGI (Common Gateway Interface; RFC 3875)
Webサーバが別のプログラムを起動し, フォームの情報を渡す
CGIのプログラムは何言語でもOK
  しかし通常は文字列処理に向いたスクリプト言語 (Perl, Ruby, Python, ...)
CGIプログラムの出力は(サーバ経由で)そのままブラウザに返される
  普通はHTMLによるWebページの形式にする
CGIへのフォームデータの渡り方
  POST - CGIプログラムの標準入力として
  GET - 環境変数 QUERY_STRING の値として
生データを分解してくれるライブラリ/モジュールがある
-----------------------------
#! /usr/bin/perl
use CGI;

$query = new CGI;
$person = $query->param("name");        (フォームデータから取り出したパラメータ)
$age = $query->param("age") + 1;        (フォームデータから取り出したパラメータ)

print << "EOL";                          (ここからブラウザに返すページを出力)
content-type: text/html

<html><body>
Hello $person.<br>
Prediction: next year you will be $age.
</body></html>
EOL

exit;
-----------------------------

PHP (PHP: Hypertext Preprocessor)
HTML中に処理プログラムを埋め込んでおき, WEBサーバがそれを実行
<?php プログラム ?> という形式
-----------------------------
<html>
<body>
<h1> Reply: </h1>
Hello <?php echo $_POST["name"]; ?>.
Prediction: next year you will be <?php echo $_POST["age"] + 1; ?>
</body>
</html>
-----------------------------
フォームの値は $_POST[] で得る

そのほか
```

JSP (JavaServer Pages)
  PHPのかわりに Java コードを埋め込む
ASP.NET
  Microsoft

○ Client-Side Dynamic Web Page Generation
PHP, CGIの提供する対話は HTTP の往復が必要
ユーザの操作にただちに反応するのは困難

HTMLのなかにスクリプト(プログラム)を埋め込み, クライアント(ブラウザ)で実行
<script> タグ (HTML 4.0)
dynamic HTML

JavaScript
  関数 response の定義
    form の値の取り出し
    結果ページ(document)の作成
  submit ボタンにおける onclick=...

```
<html>
<head>
<script language="javascript" type="text/javascript">   (プログラムの始まり)
function response(test_form) {                          (関数の定義)
var person = test_form.name.value;
var years = parseInt(test_form.age.value) + 1;
document.open();                                        (現在表示しているページを書き直す)
document.writeln("<html> <body>");
document.writeln("Hello " + person + ".<br>");
document.writeln("Prediction: next year you will be " + years + ".");
document.writeln("</body> </html>");
document.close();
}
</script>                                               (ここまでプログラム)
</head>
<body>
<form>
Please enter your name: <input type="text" name="name">
<p>
Please enter your age: <input type="text" name="age">
<p>
<input type="button" value="submit" onclick="response(this.form)">
</form>
</body>
</html>
```

PHP との動作の違い
  どちらもHTMLにコードを埋め込んでいるが
  - PHPでは, フォームの結果がサーバに送信され, それを受けてPHPを実行し
    ブラウザに返信している
  - JavaScriptでは, サーバとのやりとりなくブラウザがコード実行

VBScript, applets, ActiveX controls
  いずれもブラウザで実行

例をもう一つ

```
<html>
<head>
<script language="javascript" type="text/javascript">
function response() {
var x = parseInt(document.getElementById("input_x").value);
var y = parseInt(document.getElementById("input_y").value);
var result = document.getElementById("result");
result.textContent = x + y;
}
</script>
</head>
<body>
```

```
<input type="text" id="input_x" oninput="response()">
+
<input type="text" id="input_y" oninput="response()">
=
<span id="result"></span>
</body>
</html>
-----------------------------
二つ表示されているinputボックスの内容が変化するたびに
合計を計算して表示を更新する

DOM (Document Object Model)
ブラウザが保持しているページの内容を、
木構造のオブジェクトとしてアクセスできるようにしたモデル
     html--body--form--input--...

上例では、合計を表示するspanオブジェクト
（HTMLソースでは <span></span>タグでくくられている）
をプログラムから getElementById で取得し、
その内容を設定することで結果を表示している

-「javascript dom reference」で検索
-ブラウザで木構造を表示して吟味することも可能
        開発ツールの inspector とか

○ AJAX - Asynchronous JAvaScript and XML
ブラウザ上での高度な対話性と, サーバのデータへのアクセスを結合
Google Gmail, Maps, Docs
"not a language; a set of technologies"
1. HTML and CSS
2. DOM
3. XML (eXtensible Markup Language)
   ブラウザ上のプログラムとサーバとの間の構造をもつデータのやりとり
4. asynchronous way to send and receive XML data
5. JavaScript

○ WEB API

ここまでの説明では
  ブラウザ->サーバ: ページ要求, フォーム送信
  サーバ->ブラウザ: HTMLで記述したページ

AJAXをはじめとするクライアント側でのプログラム実行では
  ブラウザ内での描画やコンテンツ生成もあるが
  サーバからの情報取得の必要
  「サーバ上にあるライブラリ関数の呼び出し」モデルで
  Application Program Interface

関数呼び出しのインタフェースとは
  関数名
  引数 - 型と個数
  返り値 - 型

引数
  フォームと同じようにGET/POSTメソッドで送信することが多い
  文字列, 数字など

返り値
  構造を持ったデータ(オブジェクト)を返す必要もある
  ネットワークで構造を持ったデータを渡す方法が必要
    プログラム言語でいうと
      構造体(struct / record / object / hash ...)
        key-value pair
      配列
  WEBの世界では XML, JSON などの形式がよく使われる
  それらへの形式変換をデータの serialize / marshalling / ... という

XML - Extensible Markup Language
```

  <要素名 属性="値" 属性="値" ...>内容</要素名>

JSON - JavaScript Object Notation
  配列: [値, 値, ...]
  オブジェクト: { "キー": 値, "キー": 値, ... }

WEBサービスごとにリクエストと応答の形式が決まっている

例:
Google Places API (https://developers.google.com/places/documentation/)
    "Google Places API は、この API 内で施設、地理的位置、有名なスポットとして定義されているプレイスについての情報を HTTP リクエストを使用して返すサービスです。プレイス リクエストでは、場所を緯度と経度の座標として指定します。"

検索要求 - あるURLへのGETリクエスト
  引数に相当するパラメータはフォームのときと同様に ? と & でエンコード
  下の例は
    「調布駅周辺100mの登録された地点」を要求
    JSON形式での応答を指定している
以下の実行例はシェルからwgetコマンドで直接サーバに要求を送っている
(シェルの$xxは変数xxの値に置きかわる)
----
output=json
key=検索に必要となるキー(登録すると発行してもらえる)
location=35.652191,139.543945
radius=100
sensor=false

wget -nd "https://maps.googleapis.com/maps/api/place/nearbysearch/$output?location=$location&radius=$radius&key=$key&sensor=$sensor"

応答

```
{
   "html_attributions" : [],
   中略
   "results" : [
     中略
      {
         "geometry" : {
            "location" : {
               "lat" : 35.65165,
               "lng" : 139.544476
            }
         },
         "icon" : "http://maps.gstatic.com/mapfiles/place_api/icons/restaurant-71.png",
         "id" : "bc88b53b4b5c1e3a4f1bc96224afd06f4915f760",
         "name" : "会津喜多方ラーメン坂内調布店",
         "opening_hours" : {
            "open_now" : true
         },
         "photos" : [
           中略
         ],
         "place_id" : "ChIJNdpqSmvwGGARooAPHJsYDf0",
         "price_level" : 1,
         "rating" : 3.6,
         "reference" : 中略,
         "scope" : "GOOGLE",
         "types" : [ "restaurant", "food", "establishment" ],
         "vicinity" : "4 Chome-2-1 Fuda, Chofu"
      },
      以下略
```

XMLでの応答を指定した場合

```
--------------
 前略
 <result>
  <name>会津喜多方ラーメン坂内調布店</name>
  <vicinity>4 Chome-2-1 Fuda, Chofu</vicinity>
  <type>restaurant</type>
  <type>food</type>
  <type>establishment</type>
  <geometry>
   <location>
    <lat>35.6516500</lat>
    <lng>139.5444760</lng>
   </location>
  </geometry>
  <rating>3.6</rating>
  <icon>http://maps.gstatic.com/mapfiles/place_api/icons/restaurant-71.png</icon>
  以下略
--------------
```