Web開発概要

s1140227 横山 卓也

Webプログラミング

- Webサーバー

HTMLや画像などのデータを保持し、wwwにおいて情報通信する。
主にApache(アパッチ)と言うフリーソフトで動いている。

- クライアント(一般PC)

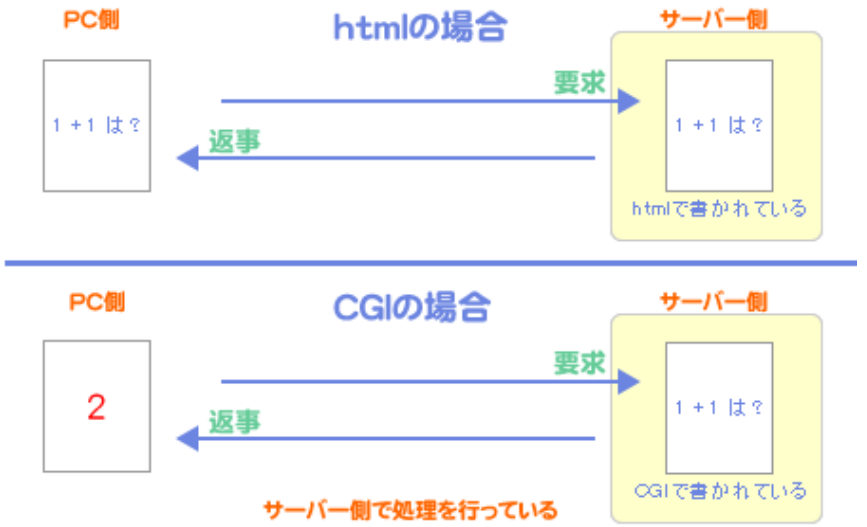
IE等のWebブラウザを利用し、Webサーバーにデータを要求する。

- サーバーサイドスクリプト

サーバー上で動作するプログラム。
JSP・ASP、CGI(Perl)やPHP

- クライアントサイドスクリプト

IE等のWebブラウザ上で動作する
プログラム。
主にjavascript。



動的(結果が変わる)ページが作れる

JSP・ASP

- JSP

JavaをそのままwebプログラミングにApacheではなくTomcatを利用するが、Apacheのプラグインで動作させるのが主流。
-メリット・デメリット
プラットフォームに依存しないため、バージョンアップの影響なし。
何でもできる反面、習得に時間がかかる。

- ASP

VBScript(Visual Basicの簡易版)。マイクロソフトのスクリプト言語。
ApacheではなくIISを利用。Windows Server2003に入っている。
-メリット・デメリット
管理ツールが豊富。初心者でも開発しやすい。
有料。サーバーが重い。バージョンアップに影響する。

JSPは大規模、主流・ASPは小規模、否主流

CGI(Perl)・PHP

- CGI(Perl)

HTML丸ごと書き出すのに適している。掲示板、アクセスカウンタ等
Perlの他にもRubyなどがある。
Apacheを利用し、PerlやRubyなどのインストールが必要。
-メリット・デメリット
JSP・ASPより動作環境構築、習得が楽。
サーバーの負担が重い。

- PHP

JSP・ASPと同じくHTML埋め込み型。
Apacheを利用し、PHPのインストールが必要。
-メリット・デメリット
CGIと同様のメリットの他、サーバーの負担が軽い、関数が多い。
他の言語より脆弱性の報告が多い。

後発のPHPは良いとこだらけ。

HTML埋め込み型

- PHP(サーバーサイド)　sample.php

```
<html>
  <body>
    <?php echo 'この文が表示される。'; ?>
  </body>
</html>
```

- Javascript(クライアントサイド)

下記意外にも書き方がある。sample.html

```
<html>
 <SCRIPT language="JavaScript">
   <!--
     実行するスクリプト
   // -->
 </SCRIPT>
</html>
```

埋め込み型は手軽に使え、小規模開発に適している。

データの保存先

- csvやテキストファイル　csvではエクセルのような表形式で保存。

-メリット・デメリット

読み書き、保存用プログラムだけで利用でき手軽。

データ量に限界あり。複雑なデータ取得ができない。動作が重い。

- DBMS　データベース管理システム

RDB(リレーショナルデータベース)サーバーを構成するシステム。

表で保存される。読み書き・更新に使うソフトはMySQLが主流。

PostgreSQL、Oracle等もある。言語はSQL。(それぞれ若干異なる)

-メリット・デメリット

多くののデータが高速で利用でき、安全性が高い。バックアップ可能。

環境構築が必要。SQLや接続用のプログラムを覚える必要あり。

データベースはどのサーバーサイド技術でも利用でき、必須スキル。

MySQLの場合phpMyAdminを利用することで、webブラウザから
データベースを管理できる。構造の変更やcsvの入出力等。

データベースの構造

phpMyAdminの画面「c_diary」テーブルを参照(DB名は伏字)

Webサーバーに DBも入っている場合
サーバー名の通常は「localhost」

テーブルを読むSQL文
SELECT * FROM 'c_diary';
phpmyadminではSQL文を書かなくても参照可能。

PHPとMysqlの連携例

```
<?php
// MySQL 接続
if (!($cn = mysql_connect("localhost", "root", “password")))
   { die; }
// MySQL DB 選択
if (!(mysql_select_db(“dbname"))) { die; }
// MySQL 問い合わせ
$sql = "select * from c_diary";
if (!($rs = mysql_query($sql))) { die; }
// データを配列にし表示
While($data=mysql_fetch_array($rs)){
echo $data[‘c_diary_id’]; echo “<br />”;}
// MySQL 切断
mysql_close($cn);  ?>
```

クライアントからのデータ受け取り

- GET送信

URLに変数を埋め込むことでデータを送信する。
改ざんされても問題ないデータのみで利用。
例）検索キーワードやリンク先に日記の固有IDを埋め込む。

diary

GET送信における変数内容取得
$_GET[“c_diary_id”]

test.phpにて固有IDが「20」の日記を閲覧できる。
(DBと連携し、固有IDの値によりデータを取得することによって。
SELECT * FROM c_diary WHERE c_diary_id=‘～’;)

手軽に値を送信できる。
リンク一覧を簡単に作成できる。

クライアントからのデータ受け取り

- POST送信

textではテキストフォームを利用しクライアントの入力を送れる。
hiddenでは手軽に固定情報を送れる。
例）参照している日記の固有ID番号を次のページに引き継ぐ
セキュア(安全性の保障)ではないため、ログイン情報等は送らない。

```
<form action="test.php" method="post">
  <input type="text" name=“name" value="">
  <input type=“hidden" name=“c_diary_id" value="20">
  <input type="submit" name="soushin" value="送信">
</form>
```

test.phpにてテキストフォームに入力した文を取得。
$_POST[“name”]

$_POST[“c_diary_id”]には20が入っている。
GET送信よりはわずかにセキュア。

ユーザー情報の保持

- クッキー

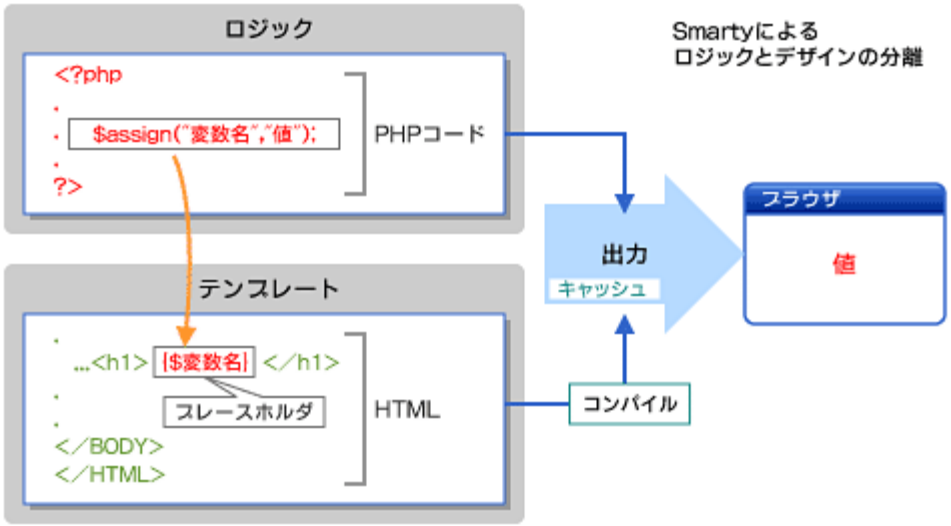
$_COOKIE[‘～’]のように取得。ログイン情報などを数日間保持する際に利用。これがXSS攻撃(詳細は後述)の対象となる。
setcookie($cookieName,$value,$timeout,$path,$domain)で保存。
$timeoutにて保存期間を設定できる。

- セッション

$_SESSION[‘～’]のように取得。ログイン情報をページ間で保持する際に利用。ショッピングカート内のデータでも利用される。ブラウザを閉じると値は消える。
session_start(); が書かれたページ間において、情報を保持できる。
$_SESSION[‘～’]=値　で保存。

テンプレートエンジンSmartyによる中規模開発

プログラム処理を書くファイルと、デザインするファイルをわける事ができる。それによりデザイナーと分業できたり、テンプレートソースの可視性が向上する。複数のファイルをインポートする際に、変数の領域を変えることで、変数名の重複を防ぐことができる。(カプセル化)また、関数も豊富で簡単なプログラムを書くこともできる。それによりプログラム処理ファイルの内容を簡素化できる。



PHPの普及に伴う問題点

簡単にwebアプリケーションが作れるようになった反面、セキュリティの問題が深刻である。対策を学ばずに作れば、必ず穴ができる。

- XSS(クロスサイドスクリプティング)

ユーザーがテキスト入力をする際、文中にjavascriptを埋め込み攻撃。
ログイン情報を盗み取り、管理者になりすます事などができる。
-対策　サニタイジング(無害化)を行う。
$name=htmlspecialchars($_POST[`name`], ENT_QUOTES);

- SQLインジェクション

ログインフォーム等にSQL文を不正に入れることで、DBのデータを破壊したり、認証をすり抜けたりする攻撃。
-対策
SQLの読み書き前後は特殊文字のエスケープ処理を行う。
$id=addslashes($_GET[c_diary_id]);　DB設定によっては不必要。

最も良い対策方法は、「ホワイトリスト」詳細は検索で。
攻撃や対策はこれだけではないので注意。

フレームワークによるセキュアな開発

前述したように、セキュアなアプリを作る際には、手間が余計にかかる。しかし、バリデーション(ユーザーが入力したデータの型が正しいかの判別)やhtmlspecialchars、addslashes等はルーチン化できる。普通に作成してもセキュアなアプリができ、Smartyのようなテンプレートエンジンなどを含む開発環境をフレームワークと呼ぶ。大規模向けのsymfonyや、小規模向けのCakePHPなどそれぞれ特徴がある。また、ほとんどがMVCやRoRのような考え方を取り入れている。

- MVC Model-View-Controllerの３つにプログラムを分担し、開発生産性や保守性を向上させる。

- RoR(Ruby on Rails) 元々Railsのフレームワーク。O/Rマッピング(オブジェクト指向とリレーショナルデータベースの連携)やCRUD(データの作成、読み込み、更新、削除)画面の自動生成などがある。

これから開発に取り組むにあたって

０から作成するより、オープンソースのWebアプリを利用する方がよい。データベースの設計や、プログラムファイルの分離方法、セキュリティ対策などがわかるからだ。自分の場合SNSのOpenPNEを利用しているが、Smartyの利用や、セキュリティ対策のためのルーチン化がなされているので、機能追加が楽にできる。

開発環境構築

- XAMPP Windowsにローカルサーバー(外部からアクセスできないサーバー。)を構築する際利用。これはApache、MySQL、PHP、Perl、phpMyAdmin、SQLite(簡易DB)が同時にインストールできる。

- サーバー 無料レンタルサーバーではすべての環境がそろわないケースが多い。いらなくなったPCをサーバーにすると良い。その際FedoraやCentos、Ubuntuなどのディストリビューションを利用。