电子商务

第四章 PHP与网站设计

4.1 架设APACHE+PHP网站

4.2 PHP概述

4.3 PHP编程基础

4.4 PHP典型函数簇

4.5 PHP应用程序实例

4.6 PHP数据库应用程序设计

4.1架设APACHE+PHP网站

安装环境

MS Windows XP

文件下载

在下面的网站下载Apache httpd

http://httpd.apache.org/

最新版本2.2.4

在下面的网站下载php

http://www.php.net/

最新版本5.2.1

安装Apache

从教学网站上下载apache_2.0.54-x86-no_ssl.msi到你的计算机上, 在你的计算机上直接打开该文件即可以执行安装. 在安装过程中, 除必要的输入信息, 其它均可以选择默认的选项即可。

通常安装在目录C:\Program Files\Apache Group.

正常安装完成后, 不需启动计算机, 即可以通过打开浏览器检查你的安装是否成功. 你在浏览器的地址栏里输入：

http://127.0.0.1/

应该可以看到Apache默认的页面.

安装Apache

在桌面下任务栏右边的托盘上可以显示是个Apache控制图标, 用于启停Apache.

Apache httpd默认的网页文件放在

C:\Program Files\Apache Group\ Apache2\htdocs

目录下。如果要改成其它的目录，用记事本打开

C:\Program Files\Apache Group\Apache2 \conf\httpd.conf

文件，找到

DocumentRoot "C:/Program Files/Apache Group/Apache2/htdocs"

改成

DocumentRoot "C:/Program Files/Apache Group/Apache2/www"

即可

安装PHP

从教学网站上下载php-5.0.4-Win32.zip到你的计算机上, 然后把它用WinZip解压缩至你的硬盘, 目录最好是C:\php. 也可能参照解压后的文件intsall.txt.

然后在Apache的配置文件目录下找到httpd.conf(这个文件通常在C:\Program Files\Apache Group\Apache2\conf目录下)。用写字板打开httpd.conf, 加入以下几行:(最好的位置在一系列LoadModule行之后)

```
# For PHP 5 do something like this:
LoadModule php5_module
    "c:/php/php5apache2_4.dll"
AddType application/x-httpd-php .php
# configure the path to php.ini
PHPIniDir "C:/php"
```

用任务栏右边的托盘上的Apache控制来重新启动apache.

安装PHP

如要测试php是否安装就绪, 在C:\xampp\www目录下用写字板建立一个文件index.php, 文件包括如下语句:

```
<?php
echo "My php installed Ok!";
?>
```

保存后退出写字板.打开浏览器, 在浏览器的地址栏里输入：

http://127.0.0.1/index.php

那么, 在浏览器中应该显示的内容是:

My php installed Ok!

安装PHP

如果想让apache输出的默认首页是index.php,在httpd.conf中找到这样一行:

DirectoryIndex index.html index.html.var

改成如下的行:

DirectoryIndex index.php index.html index.html.var

让PHP支持更多的功能

php有很多的功能，在默认安装的时候不一定都加载了。这些扩展模块位于C:\php\ext目录下以.DLL文件的形式存在。

如果要加载这些模块，打开C:\php\php.ini(如果没有这个文件，可以从php.ini-recommended拷贝一份), 找到;extension=php_???.dll对应你所需的模块，去掉前的;, 重启apache即可载入你所指定的模块。

Linux下Apache和php安装

下载Linux环境下Apache httpd和php.

按照相关文档进行安装。

也可在互连网上查找相关文档。

4.2 PHP概述

PHP(Hypertext Preprocessor，超文本预处理器的字母缩写)是一种被广泛应用的开放源代码的多用途脚本语言，它可嵌入到 HTML中，尤其适合 web 开发。

```
<html>
   <head>
        <title>Example</title>
  </head>
    <body>

     <?php
     echo "Hello Word !";
  </body>
  ?>
</html>
```

Hello Word !

PHP概述

这个例子和其它用 C 语言写的脚本之间的区别——与用大量的命令来编写程序以输出 HTML 不同的是，我们用 PHP 编写了一个 HTML 脚本，其中嵌入了一些代码来做一些事情(例如，在本例中输出了一些文本)。

PHP 代码被包含在特殊的起始符<?php和结束符?>中，使得可以进出“PHP 模式”。 简写成<? ?>。

注：在HTML中<? ?> 之间的内容是注释!

使用php编写的脚本文件一般以.php结尾。

使用 PHP 的一大好处是它对于初学者来说极其简单，同时也给专业的程序员提供了各种高级的特性。当看到 PHP 长长的特性列表时，请不要害怕。可以很快的入门，只需几个小时就可以自己写一些简单的脚本。

PHP概述

和客户端的 JavaScript 不同的是，PHP 代码是运行在服务端的。如果在服务器上建立了如上例类似的代码，则在运行该脚本后，客户端就能接收到其结果，但他们无法得知其背后的代码是如何运作的。甚至可以将 web 服务器设置成让 PHP 来处理所有的 HTML 文件，这么一来，用户就无法得知服务端到底做了什么。

PHP和JavaScript可以同时使用。一个作用于服务器端， 一个作用于浏览器端。

PHP 能做什么?

PHP 能做任何事。PHP 主要是用于服务端的脚本程序，因此可以用 PHP 来完成任何其它的 CGI 程序能够完成的工作，例如收集表单数据，生成动态网页，或者发送 / 接收 Cookies。但 PHP 的功能远不局限于此。

PHP 脚本主要用于以下三个领域：

服务端脚本。这是 PHP 最传统，也是最主要的目标领域。

命令行脚本。

编写桌面应用程序。

PHP概述

类C语言：Php是一个有些类似于C语言的脚本语言，它支持面向对象功能。

Php不仅有变量，还有类似于C语言的控制流程。

Php不仅提供了丰富的内置函数，而且也可以自定义函数，实现结构化编程。

Php提供了面向对象的机制，以实现封装和继承。Php内定义了很多类，编程者也可以定义自己的类。

4.3 PHP编程基础

- 基本语法
- 数据类型
- 变量、常量、表达式、运算符
- 控制结构
- 函数
- 类与对象
- 异常处理

从 HTML 中分离

当 PHP 解析一个文件时，会寻找开始和结束标记，标记告诉 PHP 开始和停止解释其中的代码。此种方式的解析可以使 PHP 嵌入到各种不同的文档中，凡是在一对开始和结束标记之外的内容都会被 PHP 解析器忽略。大多数情况下 PHP 都是嵌入在 HTML 文档中的，如下例所示。

```
<p>This is going to be ignored.</p>
    <?php echo “This will be parsed.”; ?>
<p>This will also be ignored.</p>
```

PHP只解析<? ?>之间的内容(称之为php的语句)，而将<? ?>之外的内容将直接发送给浏览器。

语句和注释

语句以;结构

注释

单行注释：// 或#开头的行，但不能注释掉?>

多行注释：/*　　*/

PHP的数据类型

PHP 支持八种原始类型。

四种标量类型：
boolean（布尔型）
integer（整型）
float（浮点型double）
string（字符串）

两种复合类型：
array（数组）
object（对象）

两种特殊类型：
resource（资源）
NULL

- 为了代码的易读性，引入伪类型：
  mixed
  number
  callback
- 变量的类型通常不是由程序员设定的，确切地说，是由 PHP 根据该变量使用的上下文在运行时决定的。

PHP的数据类型

PHP 中的类型强制转换和 C 中的非常像：在要转换的变量之前加上用括号括起来的目标类型。

允许的强制转换有：

(int)，(integer) - 转换成整型
(bool)，(boolean) - 转换成布尔型
(float)，(double)，(real) - 转换成浮点型
(string) - 转换成字符串
(array) - 转换成数组
(object) - 转换成对象

PHP变量

PHP 中的变量用一个美元符号后面跟变量名来表示。变量名是区分大小写的。

变量名与 PHP 中其它的标签一样遵循相同的规则。一个有效的变量名由字母或者下划线开头，后面跟上任意数量的字母，数字，或者下划线。按照正常的正则表达式，它将被表述为：‘[a-zA-Z_\x7f-\xff][a-zA-Z0-9_\x7f-\xff]*’。

变量在使用时，可以传值赋值，也可以引用赋值。但只有有名字的变量才可以引用赋值。

虽然在 PHP 中并不需要初始化变量，但这是个好习惯。未初始化的变量具有其类型的默认值 - FALSE，零，空字符串或者空数组。

PHP变量

PHP 提供了大量的预定义变量。

从 PHP 4.1.0 开始，PHP 提供了一套附加的预定数组，这些数组变量包含了来自 web 服务器（如果可用），运行环境，和用户输入的数据。这些数组非常特别，它们在全局范围内自动生效，例如，在任何范围内自动生效。因此通常被称为自动全局变量（autoglobals）或者超全局变量（superglobals）。

变量范围

变量的范围即它定义的上下文背景(也就是它的生效范围)。大部分 PHP 变量只有一个单独的范围。这个单独的范围跨度同样包含了 include 和 require 引入的文件。

global 关键字 用于申明全局变量。

PHP超全局变量

$GLOBALS：包含一个引用指向每个当前脚本的全局范围内有效的变量。该数组的键名为全局变量的名称。

$_SERVER：由 web 服务器设定或者直接与当前脚本的执行环境相关联。

$_GET：由 URL 请求提交至脚本的变量。

$_POST：由 HTTP POST 方法提交至脚本的变量。

$_COOKIE：由 HTTP Cookies 方法提交至脚本的变量。

$_FILES：由 HTTP POST 文件上传而提交至脚本的变量。

$_ENV：执行环境提交至脚本的变量。

$_REQUEST：由 GET，POST 和 COOKIE 机制提交至脚本的变量。

来自 PHP 之外的变量

当一个表单体交给 PHP 脚本时，表单中的信息会自动在脚本中可用。有很多方法访问此信息，例如：

```
<form action="foo.php" method="POST">
      Name:  <input type="text" name="username"><br />
      Email: <input type="text" name="email"><br />
      <input type="submit" name="submit" value="Submit
   me!" />
</form>
```

在PHP中以如下方式引用这些变量

```
<?php
      echo $_POST['username'];
      echo $_REQUEST['username'];
      echo $HTTP_POST_VARS['username'];
      echo $username;
?>
```

PHP控制结构

if elseif else
while
do-while
for
break
continue
switch

- foreach
- declare
- Ticks
- return
- require
- include
- require_once
- include_once

PHP函数

一个函数可由以下的语法来定义：

```
<?php
function foo($arg_1, $arg_2, ..., $arg_n)
{
    echo "Example function.\n";
    return $retval;
}
?>
```

任何有效的 PHP 代码都有可能出现在函数内部，甚至包括其它函数和类定义。

函数名和 PHP 中的其它标签命名规则相同。有效的函数名以字母或下划线打头，后面跟字母，数字或下划线。可以用正则表达式表示为：[a-zA-Z_\x7f-\xff][a-zA-Z0-9_\x7f-\xff]*。

PHP函数

PHP可以在函数中定义函数

```
<?php
function foo()
{
 function bar()
 {
  echo "I don't exist until foo() is called.\n";
 }
}

/* We can't call bar() yet
  since it doesn't exist. */

foo();

/* Now we can call bar(),
  foo()'s processesing has
  made it accessable. */
bar();
?>
```

PHP函数

PHP 中的所有函数和类都具有全局域，可以在内部定义外部调用，反之亦然。

PHP 不支持函数重载，也不可能取消定义或者重定义已声明的函数。

函数名是非大小写敏感的，不过在调用函数的时候，通常使用其在定义时相同的形式。

在 PHP 中可以调用递归函数。但是要避免递归函数 / 方法调用超过 100-200 层，因为可能会破坏堆栈从而使当前脚本终止。

PHP函数

函数的参数

通过参数列表可以传递信息到函数，即以逗号作为分隔符的表达式列表。PHP 支持按值传递参数（默认），通过引用传递以及默认参数。支持可变长度参数列表。

```
<?php
function takes_array($input)
{
  echo "$input[0] + $input[1] = ", $input[0]+$input[1];
}
?>
```

PHP函数

通过引用传递参数

缺省情况下，函数参数通过值传递（因而即使在函数内部改变参数的值，它并不会改变函数外部的值）。如果希望允许函数修改它的参数值，必须通过引用传递参数。

如果想要函数的一个参数总是通过引用传递，可以在函数定义中该参数的前面预先加上符号 &：

```
<?php
function add_some_extra(&$string)
{
  $string .= 'and something extra.';
}
$str = 'This is a string, ';
add_some_extra($str);
echo $str;    // outputs 'This is a string, and something extra.'
?>
```

PHP函数

默认参数的值

函数可以定义 C++风格的标量参数默认值

```
<?php
function makecoffee($type = "cappuccino")
{
   return "Making a cup of $type.\n";
}
echo makecoffee();
echo makecoffee("espresso");
?>
```

PHP函数

内部（内置）函数

PHP 有很多标准的函数和结构。

还有一些函数需要和特定地 PHP 扩展模块一起编译，否则在使用它们的时候就会得到一个致命的“未定义函数”错误。

同时还应该注意，很多扩展库默认就是有效的。

PHP类与对象

类：类是变量与作用于这些变量的函数的集合。

使用下面的语法定义一个类：

```
<?php
  class Cart {
    var $items;

    function add_item($artnr, $num) {
      $this->items[$artnr] += $num;
    }

    function remove_item($artnr, $num) {
      if ($this->items[$artnr] > $num) {
        $this->items[$artnr] -= $num;
        return true;
      } elseif ($this->items[$artnr] == $num) {
        unset($this->items[$artnr]);
        return true;
      } else {
        return false;
      }
    }
  }
  ?>
```

- 这个例子定义了一个 Cart 类，这个类由购物车中的商品构成的数组和两个用于从购物车中添加和删除商品的函数组成。

PHP类与对象

继承：通常需要这样一些类，这些类与其它现有的类拥有相同变量和函数。实际上，定义一个通用类用于所有的项目，并且不断丰富这个类以适应每个具体项目将是一个不错的练习。为了使这一点变得更加容易，类可以从其它的类中扩展出来。扩展或派生出来的类拥有其基类的所有变量和函数，并包含所有派生类中定义的部分(这称为“继承”）。类中的元素不可能减少，就是说，不可以注销任何存在的函数或者变量。

一个扩充类总是依赖一个单独的基类，也就是说，PHP不支持多重继承。

使用关键字“extends”来扩展一个类。

PHP类与对象

构造函数：构造函数是类中的一个特殊函数，当使用 new 操作符创建一个类的实例时，构造函数将会自动调用。当函数与类同名时，这个函数将成为构造函数。如果一个类没有构造函数，则调用基类的构造函数，如果有的话。

```
<?php
class Auto_Cart extends Cart {
  function Auto_Cart() {
     $this->add_item ("10", 1);
  }
}
?>
```

PHP5可以用__construct()作为构造函数。

PHP类与对象

可以使用 @ 操作符来*抑制*发生在构造函数中的错误。例如 @new。

PHP 4 不会从派生类的构造函数中自动调用基类的构造函数。PHP4 中也没有析构函数。

PHP 5 引入了析构函数的概念，这类似于其它面向对象的语言，如 C++。析构函数会在到某个对象的所有引用都被删除或者当对象被显式销毁时执行。

PHP5的析构函数是__destruct().

析构函数在脚本关闭时调用，此时所有的头信息已经发出。

PHP类与对象

范围解析操作符（::）

有时，在没有声明任何实例的情况下访问类中的函数或者基类中的函数和变量很有用处。而 :: 运算符即用于此情况。

PHP 5 引入了析构函数的概念，这类似于其它面向对象的语言，如 C++。析构函数会在到某个对象的所有引用都被删除或者当对象被显式销毁时执行。

4.4 PHP典型函数簇

- Apache 特有函数
- PHP 选项和资讯函数
- 标准PHP库函数
- Session 处理函数
- URL 函数
- Variable 变量函数
- Array函数
- Character Functions
- Unicode Functions
- String 字符串处理函数
- Date/Time 日期 / 时间函数
- Object函数
- Math 数学函数
- hash Functions
- Directory 目录函数
- Filesystem 函数
- Stream Functions
- Miscellaneous Functions杂项
- Error Handling and Logging Functions
- Multibyte String

PHP典型函数簇-网络

HTTP 函数
FTP 函数
TCP函数
MAIL函数
Net_Gopher
IMAP, POP3
NNTP函数
SOCKET函数
SOAP函数

- SNMP函数
- OpenSSL函数
- Mimetype函数
- SimpleXML函数
- XML 语法解析函数
- XML-RPC 函数
- XMLReader函数
- libxml函数

PHP典型函数簇-数据库接口

- DB2
- ORACLE
- MySQL
- Dbase
- Informix
- Ingres
- Lotus Notes
- Paradox
- SQLite
- SQL Server
- Sybase
- ODBC

PHP典型函数簇-其它

Calendar 日历函数

ClibPDF函数

PDF 函数

Image 图像函数

Mcrypt加解密函数

Mhash 函数

hash 函数

- LZF压缩函数
- Bzip2 压缩函数
- Rar压缩函数
- Zip压缩函数
- Zlib压缩函数

4.5 PHP应用程序实例

- Session
- 用PHP处理表单
- GET方法与POST方法
- 文件上传
- 发送电子邮件的类

4.5.1 Session

会话Session与全局变量

```
<?
    session_start();
    session_register( $username);
    session_register( $user_pwd);
?>
```

用session_register注册一个全局变量

session_unregister取消

session_is_registered判断一个变量是否注册

使用前必须第一个PHP语句为session_start()

4.5.2 用PHP处理表单

一个用户登录的表单

Loginform.php

```
<form name="fmlog" method="post" action="logcheck.php">
用户名:<input type="text" name="username" /><br>
密　码:<input type="password" name="password"/><br>
<input type="hidden" name="action" value="login" />
<input type="submit" value="登　录" />
</form>
```

处理用户登录的php代码

Logcheck.php

```
<?
      session_start();
        if (isset($action) && $_POST["action"] == "login"){
          // 处理登录的代码
          $user=$_POST["username"];
          $pwd =$_POST["password"];
          // 记录用户id和password
          $_SESSION["USERID"]   = strtoupper($user);
          $_SESSION["USERPSW"] = $psw;
          echo "$user logged in. Your are welcome. ";
          echo '<form name="fmlog" method="post">';
          echo '<input type="hidden" name="action"
           value="logout" />';
          echo '<input type=“submit” value=“注销" />';
          echo '</form>';
        }
   ?>
```

4.5.3 GET方法与POST方法

GET方法:

http://is.hust.edu.cn/userlogin.php?action=login&userid=abc&pwd=def

POST方法

就如上面的例子中一样

GET方法和POST方法在PHP脚本中变量传递和值的取得

$_GET[“变量名”]

$_POST[“变量名”]

4.5.4 文件上传的例子

```
<?php
//filename:upldfile.php
if($_POST["ifupload"]=="1") {
 $path=AddSlashes(dirname(__FILE__))."\\\\upload\\\\";
 for($i=1;$i<=8;$i++) {
      $files="afile$i";
  if (is_uploaded_file($_FILES[$files]['tmp_name'])) {
       $filename = $_FILES[$files]['name'];
      $localfile = $path.$filename;
      move_uploaded_file($_FILES[$files]['tmp_name'], $localfile);
  }
 }
  echo "<br><b>You have uploaded files successfully</b><br>";

}
?>
```

文件上传的例子

```
<body>
<table id="divContainer" style="HEIGHT: 100%; WIDTH: 380" border="0">
 <form method="post" name="frmUpload" enctype="multipart/form-data" >
 <tr height="35"><td align="right" valign="bottom">多文件上传 </td></tr>
 <tr><td align="center" valign="top">
  <table class="Transex" border="1" cellspacing="0" cellpadding="0" >
     <tr>
     <td nowrap><label class="FieldLabel">  文件1</label></td>
        <td><input type="file" class="TxtInput" tabindex="1" name="afile1" ></td>
  </tr>
     …
     <tr>
  <td colspan="2" align="right">
        <button name="btnOk" id="btnOk" >确定</button>
        <input type="hidden" name="ifupload" value=1>  
        <button name="btnCancel" id="btnCancel" onclick="window.close();">取消</button>
     </td>
  </tr>
 </table>
 </td>
 </tr>
</form>
</table>
</body>
```

4.5.5 发送电子邮件类的例子

使用Mail函数

bool mail (string to, string subject, string message [, string additional_headers [, string additional_parameters]])

类源代码

发邮件的例子

4.6 PHP数据库应用设计

mysql_connect - 打开一个到 MySQL的连接

mysql_pconnect - 打开到 MySQL的持久连接

mysql_close - 关闭 MySQL 连接

mysql_db_query - 发送一条 MySQL 查询

mysql_db_name - 取得结果数据

mysql_free_result - 释放结果内存

mysql_query - 发送一条 MySQL 查询

mysql_info - 取得最近一条查询的信息

mysql_fetch_array - 从结果集中取得一行作为关联数组，或数字数组，或二者兼有

4.6 PHP数据库应用设计

- oci_connect
- oci_pconnect
- oci_close
- oci_execute
- oci_parse
- oci_fetch
- oci_result
- oci_commit
- oci_rollback
- OCI-Lob 对象

Oracle OCI接口类的例子

访问接口

列表显示类的例子