- 丰富的移动版网页开发范例：移动商品目录、访客留言板、讨论组、文件上传、在线寄信服务、电子贺卡、会员管理系统、在线投票系统、购物车、网络相册等

超强范例集

PHP&MySQL

跨设备网站开发实例精粹

使用jQuery Mobile快速打造移动版网页

陈惠贞 陈俊荣 编著

清華大學出版社

PHP&MySQL

跨设备网站开发

实例精粹

陈惠贞 陈俊荣 编著

清華大學出版社
北京

本书版权登记号：图字 01-2015-1305

内 容 简 介

本书从易学实用的角度详细讲解 PHP、HTML5 语法，MySQL 数据库存取；针对网页之间的信息传递、表单的后端处理、HTTP Header、Cookie、Session、文件存取、GD 绘图与图像处理、面向对象、使用 Ajax、访问 MySQL 数据库、SQL 查询等应用，做了鞭辟入里的讲解；让您克服初学者的迷茫，向专业的程序设计之路迈进。

为紧跟网站推出“移动版”的潮流，书中列举了如何根据网络设备，自动切换 PC 版网页和移动版网页，另辟专门章节讲解如何使用 jQuery Mobile 快速开发移动版网站，以及如何在移动版网站内使用 PHP 与 MySQL 进行数据库访问。

本书最重要的是提供了丰富的网站开发范例，包括建立 Google 地图应用网站、移动版商品目录、访客留言板、讨论组、文件上传、在线寄信服务、电子贺卡、会员管理系统、在线投票系统、购物车、网络相册等超强范例；满足您应用于各种专题的项目开发或参与程序设计技能竞赛的需求。

图书在版编目（CIP）数据

PHP&MySQL 跨设备网站开发实例精粹 / 陈惠贞，陈俊荣编著. -- 北京 : 清华大学出版社，2015
ISBN 978-7-302-40045-5

Ⅰ. ①P… Ⅱ. ①陈… ②陈… Ⅲ. ①PHP 语言－程序设计②关系数据库系统③网页制作工具
Ⅳ. ①TP312②TP311.138③TP393.092

中国版本图书馆 CIP 数据核字（2015）第 091143 号

责任编辑：夏非彼
封面设计：王　翔
责任校对：闫秀华
责任印制：李红英

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　**邮　　编**：100084
社 总 机：010-62770175　　**邮　　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：清华大学印刷厂
装 订 者：三河市溧源装订厂
经　　销：全国新华书店
开　　本：190mm×260mm　　**印　　张**：30.75　　**字　　数**：787 千字
版　　次：2015 年 8 月第 1 版　　**印　　次**：2015 年 8 月第1 次印刷
印　　数：1～3500
定　　价：79.00 元

产品编号：063814-01

关于本书

PHP 是极为普遍的服务器端的脚本语言，广泛应用于数以百万计的 Web 服务器及大规模的网站，其中一个重要的原因是 PHP 属于开放源代码（open source），具有完全免费、稳定、快速、跨平台（UNIX、FreeBSD、Windows、Linux、Mac OS……）、易学易用、面向对象等优点。

PHP 5 从 2004 年发布迄今已经有数年，中间陆续发布了 5.0、5.1、5.2、5.3、5.4、5.5、5.6 等版本，而 PHP 6 的开发亦同步进行中，相关的标准是由 PHP Group 和开放源代码社区负责维护。

本书的另一个主角 MySQL 则是由 MySQL AB 公司所开发的关系数据库管理系统，它和 PHP 一样属于开放源代码，若纯粹为个人用途，无须申请即可免费使用，而且具有快速、简单、可靠、功能齐全、跨平台等优点。

❖ 本书特点

无论您是否学习过 PHP，本书都是您的绝佳选择，除了详细解说 PHP 的语法外，更针对在网页之间传递信息、表单的后端处理、HTTP Header、Cookie、Session、文件存取、GD 绘图与图像处理、面向对象、使用 Ajax、访问 MySQL 数据库、SQL 查询等主题，做了鞭辟入里的解说，让您克服初学者的迷茫，向专业的程序设计之路迈进。

此外，为了应网站推出“移动版”的潮流，本书不仅示范了如何根据上网的设备，自动切换 PC 版网页和移动版网页，同时另辟专门章节介绍如何使用 jQuery Mobile 快速开发移动版网站，以及如何在移动版网站内使用 PHP 与 MySQL 进行数据库访问。

最重要的是本书提供了丰富范例，可以满足您制作各种专题、项目及参与技能竞赛的需求，包括建立 Google 地图应用网站、移动版商品目录、访客留言板、讨论组、文件上传、在线寄信服务、电子贺卡、会员管理系统、在线投票系统、购物车、网络相册等。

❖ 网络资源下载内容

本书提供网络下载资源文件，便于您可以运用书中的范例程序开发自己的程序，但请勿贩卖或散布：

- WampServer：利用这套开放源码软件可以快速建立 Windows + Apache + MySQL + PHP 的运行环境，详细的安装方式请参考第 1.3 节。
- 本书范例程序与数据库：包括 \samples 与 \database 文件夹，详细的安装方式请参考

第 1.5 节和第 11.3.7 节。

- 附录篇 PDF 电子书：包括附录 A（HTML 语法教学）、附录 B（HTML 标签与属性速查）、附录 C（HTML 特殊字符表）。

本书配套源代码下载地址（注意数字与字母大小写）：http://pan.baidu.com/s/1c0EYnc8，若下载有问题，请电子邮件联系 booksaga@126.com，邮件标题为“求代码，PHP&MySQL 跨设备网站开发实例精粹”。

❖ 排版惯例

本书在列出程序代码、关键词、标签、属性及语法时，遵循了下列的排版惯例：

- HTML 不会区分英文字母的大小写，本书将采用小写英文字母，至于 PHP 则是变量名称与常数名称区分英文字母的大小写。
- 斜体字表示用户自行键入的属性值、语句、表达式或名称，例如 function *function_name*(){...} 的 *function_name* 表示用户自行键入的函数名称。
- 中括号“[]”表示可以省略不写，例如 round(*num*[, *precision*]) 表示 round() 函数的第二个参数 *precision* 为选择性参数，可以指定，也可以省略不写。
- 垂直线 | 用来隔开替代选项，例如“return;|return *value*;”表示 return 关键词后面可以不加上返回值，也可以加上返回值。

编者

2015.3

目　录

以下为 PDF 电子书

第 1 章
开始编写 PHP 程序

1.1 认识动态网页技术

在介绍动态网页技术之前，我们先来说明 Web 的工作原理。Web 采用的是客户机-服务器架构（client-server model），如图 1-1 所示，其中客户端（client）可以通过网络连接访问另一台计算机的资源或服务，而提供资源或服务的计算机就叫服务器（server）。

Web 客户端只要安装了浏览器软件（例如 Internet Explorer、Google Chrome、Mozilla FireFox、Opera、Apple Safari……），就能通过该软件连上全球各地的 Web 服务器，进而浏览 Web 服务器所提供的网页（homepage）。

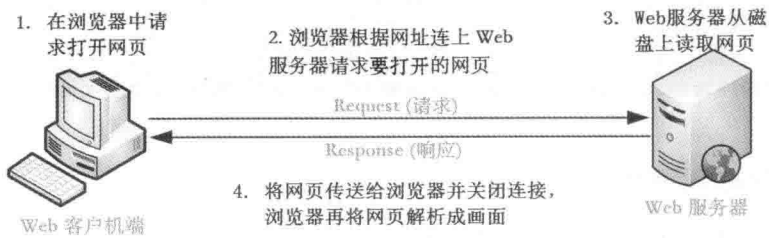


图 1-1

由上图可知，当用户在浏览器中输入网址或单击超链接时，浏览器会根据该网址连上 Web 服务器，并向 Web 服务器请求用户要打开的网页，此时，Web 服务器会从磁盘上读取该网页，然后传送给浏览器并关闭连接，而浏览器一收到该网页，就会将之解析成界面，呈现在用户面前。

事实上，当浏览器向 Web 服务器发送出请求时，它并不只是将要打开网页的网址传送给 Web 服务器，还会连同自己的浏览器类型、版本等信息一并传送过去，这些信息称为 Request Header（请求标头）。

相反的，当 Web 服务器响应浏览器的请求时，它并不只是将要打开的网页传送给浏览器，还会连同该网页的文件大小、日期等信息一并传送过去，这些信息称为 Response Header（响应标头），而 Request Header 和 Response Header 则统称为 HTTP Header（HTTP 标头）。

在因特网盛行的早期，网页只是静态的图文组合，用户可以在网页上浏览资料，但无法做进一步的查询、发表文章或进行电子商务、实时通信、在线游戏、会员管理等活动，而这显然不能满足人们日趋多元化的需求。

为此，开始有不少人提出动态网页的解决方案，“动态网页”指的是客户端和服务器可以互动，也就是服务器可以实时处理客户端的请求，然后将结果响应给客户端。动态网页通常通过“浏览器端 Scripts”和“服务器端 Scripts”两种技术来完成，以下就为您进行说明。

1.1.1 浏览器端 Scripts

浏览器端 Scripts 指的是嵌入在 HTML 源代码中的小程序，由浏览器负责执行。JavaScript 和 VBScript 均能用来编写浏览器端 Scripts，其中以 JavaScript 为主流。

图 1-2 是 Web 服务器处理浏览器端 Scripts 的过程，当浏览器向 Web 服务器请求打开包含浏览器端 Scripts 的 HTML 网页时（扩展名为 .htm 或 .html），Web 服务器会从磁盘上读取该网页，然后传送给浏览器并关闭连接，不做任何运算，而浏览器一收到该网页，就会执行里面的浏览器端 Scripts 并将结果解析成界面。

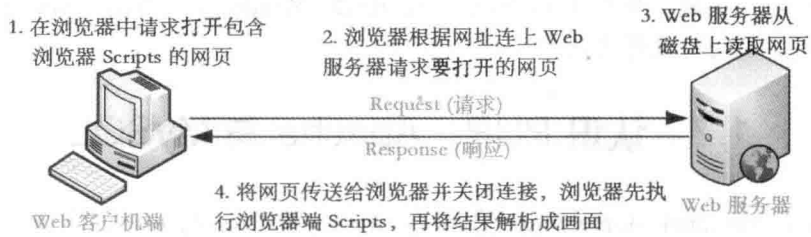


图 1-2

1.1.2　服务器端 Scripts

虽然浏览器端 Scripts 已经能够完成许多工作，但有些工作还是得在服务器端执行 Scripts 才能完成，例如访问数据库。服务器端 Scripts 也是嵌入在 HTML 源代码中的小程序，但和浏览器端 Scripts 不同的是它由 Web 服务器负责执行。

图 1-3 是 Web 服务器处理服务器端 Scripts 的过程，当浏览器向 Web 服务器请求打开包含服务器端 Scripts 的网页时（扩展名为 .php、.asp、.aspx、.jsp、.cgi 等），Web 服务器会从磁盘上读取该网页，先执行里面的服务器端 Scripts，将结果转换成 HTML 网页（扩展名为 .htm 或 .html），然后传送给浏览器并关闭连接，而浏览器一收到该网页，就会将它解析成界面。

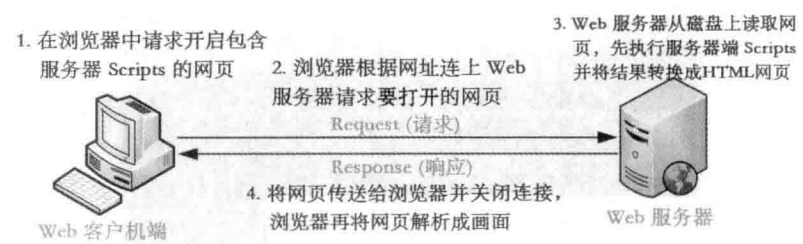


图 1-3

常见的服务器端 Scripts 有下列几种。

- CGI（Common Gateway Interface）：CGI 是在服务器端程序之间传送信息的标准接口，而 CGI 程序则是符合 CGI 标准接口的 Scripts，通常由 Perl、Python 或 C 语言所编写（扩展名为 .cgi）。
- JSP（Java Server Pages）：JSP 是 Sun 公司所提出的动态网页技术，可以在 HTML 原始文件中嵌入 Java 程序并由 Web 服务器负责执行（扩展名为 .jsp）。
- ASP（Active Server Pages）/ASP.NET：ASP 程序是在 Microsoft IIS Web 服务器上执行的 Scripts，通常由 VBScript 或 JavaScript 所编写（扩展名为 .asp），而新一代的

ASP.NET 程序则改由功能较强大的 Visual Basic、Visual C#、Microsoft J#、JavaScript.NET 等 .NET 兼容语言所编写（扩展名为 .aspx）。

- PHP（PHP:Hypertext Preprocessor）：PHP 程序是在 Apache、Microsoft IIS 等 Web 服务器上执行的 Scripts，由 PHP 语言所编写，属于开放源码（Open Source），具有免费、稳定、快速、跨平台（UNIX、FreeBSD、Windows、Linux、Mac OS……）、易学易用、面向对象等优点。

1.2 认识 PHP、Apache 与 MySQL

PHP 原本是 Personal Home Page 的简写，是由一位开发 Apache 的软件工程师 Rasmus Lerdorf 于 1994 年所设计的，目的是要维护个人网站并统计网站流量，Rasmus Lerdorf 将一些工具程序和解释器（interpreter）集成起来，称为 PHP/FI，PHP/FI 可以连接数据库和制作简单的动态网页程序，于 1995 年发布，称为 PHP 2。

随着 PHP 的用户量的快速增加，两位以色列程序设计人员 Zeev Suraski 与 Andi Gutmans 于 1997 年重新改写 PHP 的语法分析器（parser），称为 Zend Engine，它成为 PHP 3 的基础并于 1998 年发布，而 PHP 就在此时正式改名为 PHP: Hypertext Preprocessor。之后于 2000、2004 年发布 PHP 4、PHP 5，截至目前，PHP 5 已经陆续发布了 5.0、5.1、5.2、5.3、5.4、5.5、5.6 等版本，而 PHP 6 的开发亦在同步进行中，相关的标准由 PHP Group 和开放源码社区负责维护。

您可以到 PHP 官方网站 http://php.net/ 查看 PHP 的发展现况、下载 PHP 模块/PHP 文件或加入讨论，如图 1-4 所示。

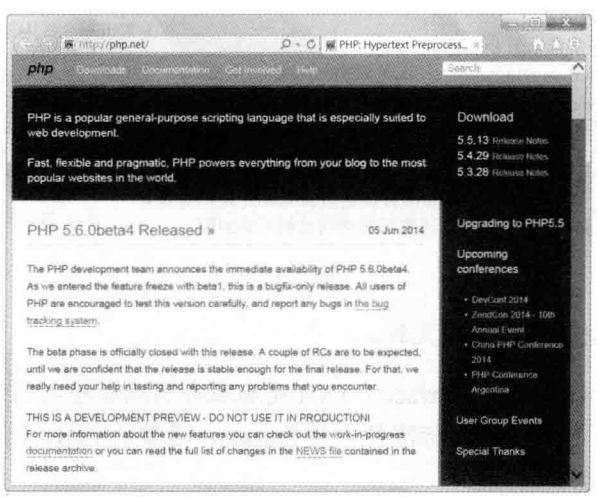


图 1-4

在看过 PHP 的介绍后，现在换 Apache 出场，本书提及的 Apache 其实指的是 Apache HTTP Server，这是 Apache 软件基金会所开发的 Web 服务器，和 PHP 一样属于开放源码，具有完全免费、稳定、快速、安全、跨平台等优点，被公认为是目前最稳定的 Web 服务器，全球有

超过半数的网站采用 Apache 作为 Web 服务器。

至于 MySQL 则是由 MySQL AB 公司开发的关系数据库管理系统，它和 PHP、Apache 一样属于开放源码，但不同的是如果纯粹为个人用途，无须申请即可免费使用。若为商业用途，则必须向 MySQL AB 公司购买授权（MySQL AB 公司已经于 2008 年被 Sun 公司收购，而 Sun 公司又于 2009 年被 Oracle 公司收购，所以 MySQL 目前是 Oracle 公司旗下的产品，其中 MySQL Enterprise Edition 须付费购买，而 MySQL Community Edition 则可免费下载）。

事实上，PHP 支持许多数据库管理系统，例如 Adabas D、DBA/DBM、dBase、IBM DB2、Informix、Microsoft Access、Microsoft SQL Server、MySQL、Oracle、PostgreSQL、SQLite、Sybase 等，本书选择使用 MySQL，原因是个人用途无须申请即可免费使用，而且它具有快速、简单、可靠、功能齐全、跨平台等优点，广泛应用于许多网站，我们可以说，PHP、Apache 与 MySQL 是开发网页最佳的组合。

您可以到 MySQL 官方网站 http://www.mysql.com/ 查看 MySQL 的发展现况、下载免费的 MySQL 社区版，如图 1-5 所示。

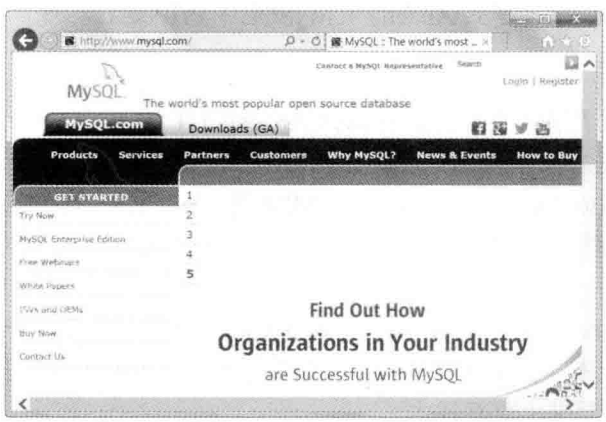


图 1-5

1.3　建立 PHP、Apache 与 MySQL 运行环境

虽然 Microsoft Windows 操作系统内建了 IIS Web 服务器，而且 PHP 可以在 IIS（Internet Information Services）上运行，但大部分 PHP 网页都是在 Apache Web 服务器上执行的，探究其主因，不外乎是 Apache 为开放源码软件，可以免费取得，同时具有稳定性高及跨平台的优势。

在 Windows 平台上建立 PHP、Apache 与 MySQL 运行环境的方式有下列几种。

- 传统安装方式：以往在建立运行环境时，必须手动安装下列软件。
  - Apache HTTP Server：这是开发 PHP 网页最受欢迎的 Web 服务器软件，您可以在 http://httpd.apache.org/ 下载最新版的软件。
  - PHP 模块：Apache HTTP Server 本身无法执行 PHP 网页，我们需要额外安装 PHP 模块，才能让 Apache HTTP Server 具有执行 PHP 网页的能力，您可以在

http://php.net/ 下载最新版的软件。

- MySQL 数据库：虽然在没有安装 MySQL 数据库的情况下，PHP 网页也能正常运行，不过，由于 PHP 网页的开发人员通常会将数据存储在 MySQL 数据库，因此，MySQL 数据库也被列为必备软件之一，您可以在 http://www.mysql.com/ 下载最新版的软件。
- 传统安装方式的优点是较有弹性，可以分别选择 Apache HTTP Server、PHP 模块与 MySQL 数据库的不同版本，缺点则是在把这些软件安装上去后，还要进行一连串的设置，不仅设置过程繁琐，而且参数容易设置错，导致功能异常，甚至造成 Apache HTTP Server 宕机，不易调试。

- WAMP 安装方式：WAMP 并不是某个软件的名称，而是 Windows + Apache + MySQL + PHP 的简写，也就是将 Apache、MySQL 与 PHP 三大功能集成在一起的软件包，若是安装在 Windows 平台，则称为 WAMP，若是安装在 Linux 平台，则称为 LAMP，若是安装在 Mac 平台，则称为 MAMP。WAMP 安装方式的优点是只要安装一套软件，即可在 Windows 平台建立 Apache + MySQL + PHP 运行环境，不需要进行繁琐的设置。

若您是初学者，或者您并不坚持自行选择 Apache HTTP Server、PHP 模块与 MySQL 数据库的版本，那么建议采用 WAMP 安装方式。常见的 WAMP 软件包有 WampServer、Zend Server CE（Community Edition）、AppServer、XAMPP 等，在本书中，我们选择使用 WampServer，因为 WampServer 为开放源码软件，可以免费使用，而且 WampServer 的用户相当多，遇到问题时容易找到解决办法。

您可以到 WampServer 的官方网站 http://www.wampserver.com/en/ 下载 Wamp Server 安装程序并查看有无更新版本，由于 PHP 每隔一段时间就会发布更新版本，例如 5.5.x 之类，所以 WampServer 会不定期推出更新版本。

1.3.1 安装 WampServer

请按照如下步骤安装 WampServer。

步骤 01 连接到 WampServer 的官方网站 http://www.wampserver.com/en/ 下载安装程序，利用资源管理器找到 WampServer 安装程序 wampserver2.5-Apache-2.4.9-Mysql-5.6.17-php5.5.12-32b.exe，如图 1-6 所示。此为 32 位版本，若您的系统为 64 位，请到官方网站下载 64 位版本。

步骤 02 执行 WampServer 安装程序，屏幕上会出现安装程序精灵，单击 “Next” 按钮。这个版本的 WampServer 主要包含 Apache 2.4.9（Apache HTTP Server）、PHP 5.5.12（PHP 模块）、MySQL 5.6.17（MySQL 数据库）、PhpMyAdmin 4.1.14（管理 MySQL 数据库的程序）等软件，如图 1-7 所示。

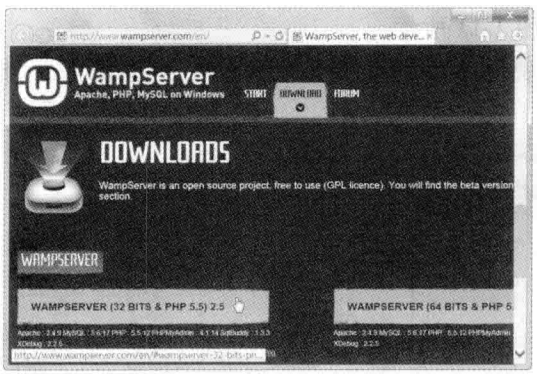


图 1-6

图 1-7

步骤 03　出现许可协议对话框，请选中 “I accept the agreement” 复选框，然后单击 “Next” 按钮，如图 1-8 所示。

步骤 04　随即要选择程序安装路径，建议您直接单击 “Next” 按钮，将 WampServer 安装在默认的路径 C:\wamp，如图 1-9 所示。

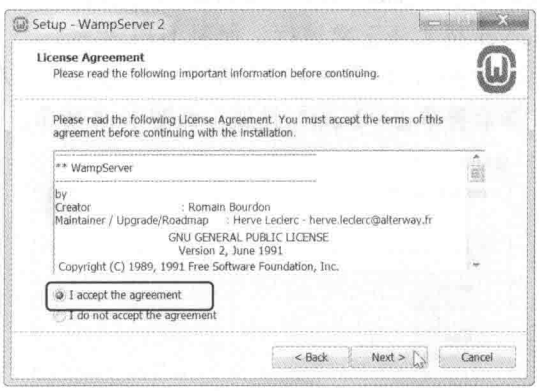


图 1-8

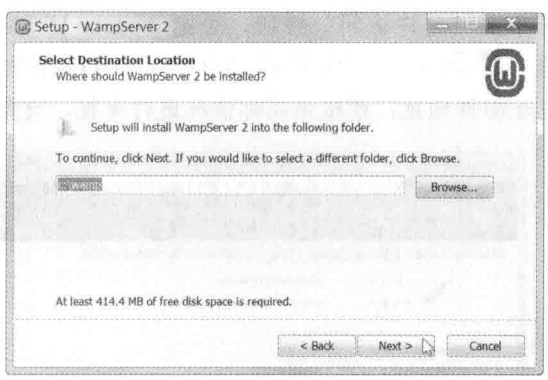


图 1-9

步骤 05　接下来会询问您是否创建快速启动图标及桌面建立程序图标，请按照自己的喜好，决定是否选择图 1-10 中的两个复选框，然后单击 “Next” 按钮。

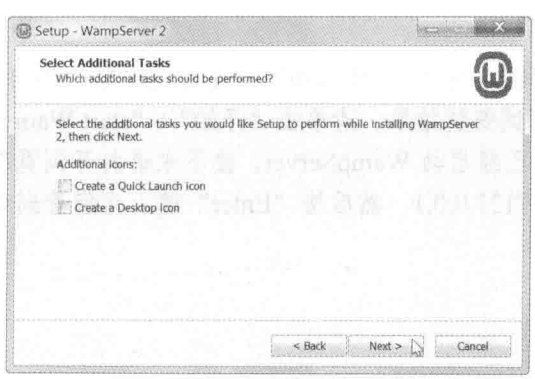


图 1-10

步骤 06 单击“Install”按钮开始安装，如图 1-11 所示。

步骤 07 继续选择要预设的浏览器，如果要使用 Windows 预设的浏览器，可直接单击“打开”按钮，如图 1-12 所示。

图 1-11

图 1-12

步骤 08 此时，若屏幕上出现 Windows 防火墙封锁 Apache HTTP Server 的对话框，请务必单击“允许访问”按钮，否则 Apache HTTP Server 将无法正常运行，如图 1-13 所示。

步骤 09 随即要设置 PHP 使用 mail() 函数发送邮件时，所要使用的 SMTP 服务器与发件人电子邮件地址，请根据实际情况进行设置，设置完毕后再单击“Next”按钮，如图 1-14 所示。

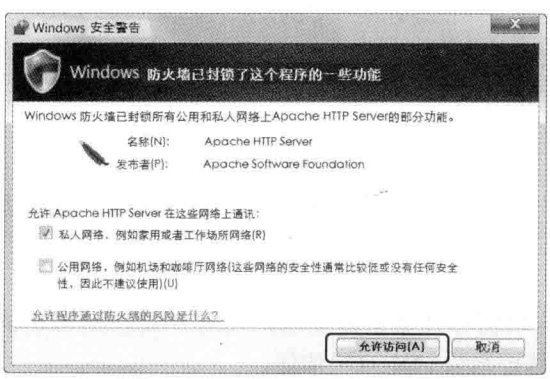


图 1-13

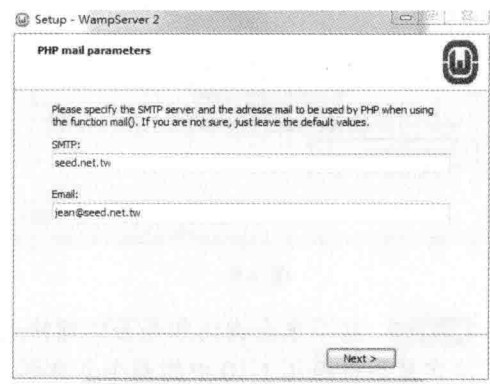


图 1-14

步骤 10 出现如下界面，请单击“Finish”按钮完成安装，如图 1-15 所示。

步骤 11 现在要来测试安装结果，请单击“开始”\“start WampServer”，此时，任务栏会出现一个图标，表示已经启动 WampServer，接下来请打开浏览器，在网址栏输入本机网址 http://localhost 或 http://127.0.0.1，然后按“Enter”键，当您看到如下界面时，表示安装成功，如图 1-16 所示。

图 1-15

图 1-16

步骤 12　在步骤 11 的界面中单击 phpmyadmin 超链接，或在网址栏输入 http://localhost/phpmyadmin/，当您看到如图 1-17 所示的界面时，表示用来管理 MySQL 数据库的 phpMyAdmin 程序也能正常运行，恭喜您，PHP 开发环境已经安装成功了！

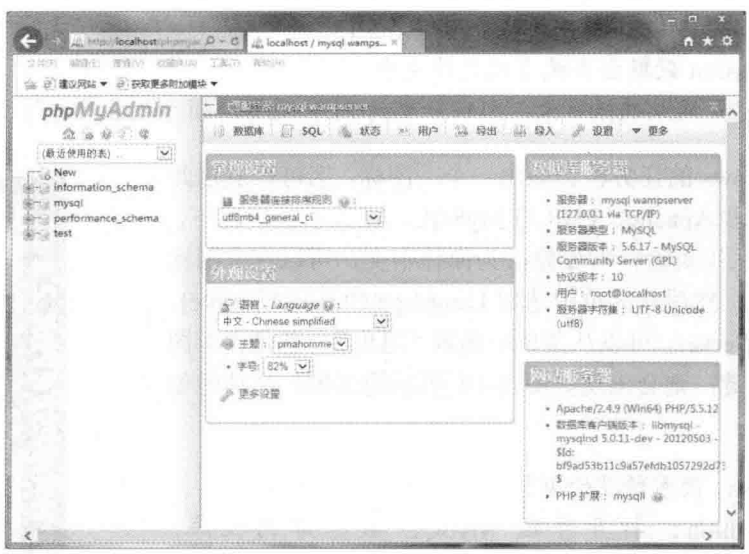


图 1-17

请注意，若您无法顺利启动 WampServer，而是看到如图 1-18 所示的错误信息，则表示系统缺少一个名称为 MSVCR110.dll 的文件，那么您可以连接到 http://www.microsoft.com/en-us/download/details.aspx?id=30679。32 位系统请下载并运行 vcredist_x86.exe，64 位系统请下载并运行 vcredist_x64.exe，这是 Microsoft 公司提供的 Visual C++ Redistributable for Visual Studio 2012 Update 4，安装完毕后，用鼠标单击任务栏中的图标，从菜单中选取 Apache\Service\Install Service，即可启动 WampServer。

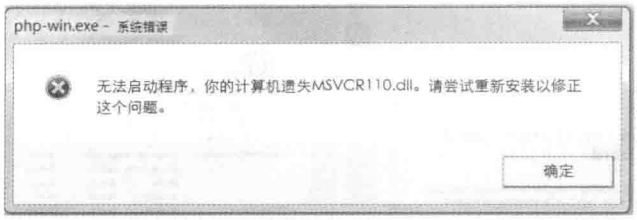


图 1-18

1.3.2 设置 WampServer

WampServer 安装完毕后，C 盘的根目录下会多出一个 wamp 文件夹，里面有下列几个重要的文件夹。

- apps：管理程序均安装在此文件夹中，例如管理 MySQL 数据库的 phpMyAdmin 程序安装目录为 C:\wamp\apps\phpmyadmin4.1.14，4.1.14 为 phpMyAdmin 的版本。
- bin：Apache、PHP 与 MySQL 的主程序均安装在此文件夹中，其中 C:\wamp\bin\php 里包含 php5.5.12 文件夹，代表 PHP 5.5.12。
- tmp：Session 数据会存放在此文件夹中。
- www：这是网站的根目录，您所开发的 PHP 网页均须存放在此文件夹中。

此外，Windows 的任务栏会多出一个图标，您可以通过此图标来设置并管理 Apache、PHP 与 MySQL。在预设的情况下，WampServer 会采用英语，若要修改为简体中文，可以在图标上单击鼠标右键，然后从菜单中选取 Language\Chinese[Simple]，若要结束 WampServer，可以从菜单中选取“退出”；若是在图标处单击鼠标左键，则会出现如图 1-19 所示的菜单，其功能如下。

图 1-19

- Localhost：使用预设的浏览器打开 http://localhost/。
- phpMyAdmin：打开管理 MySQL 数据库的程序 phpMyAdmin。
- www 目录：打开网站的根目录，默认为 C:\wamp\www。
- Apache：用来管理 Apache HTTP Server，其子功能如下。
  - Version：用来切换不同的 Apache 版本（如有安装的话）。
  - Service：用来启动、停止、重新启动 Apache 服务。
  - Apache 模块：用来启用或停用 Apache 模块，其实就是修改 Apache 的配置文件 httpd.config。
  - httpd.conf：使用文本编辑器打开 Apache 的配置文件 httpd.conf，您可以修改设置值，然后重新启动 Apache HTTP Server，以使设置生效。
- PHP：用来管理 PHP 模块，其子功能如下。

  - Version：用来切换不同的 PHP 版本（如有安装的话）。
  - PHP 设置：用来设置 PHP 的功能。
  - PHP 扩展：用来启用或停用 PHP extension。
  - php.ini：使用文本编辑器打开 PHP 的配置文件 php.ini，您可以修改设置值，然后重新启动 Apache HTTP Server，以使设置生效。
- MySQL：用来管理 MySQL 数据库，其子功能如下。
  - Version：用来切换不同的 MySQL 版本（如有安装的话）。
  - Service：用来启动、停止、重新启动 MySQL 服务。
  - my.ini：使用文本编辑器打开 MySQL 的配置文件 my.ini，您可以修改设置值，然后重新启动 MySQL，以使设置生效。
- 启动所有服务：用来启动 Apache HTTP Server 及 MySQL 数据库。
- 停止所有服务：用来停止 Apache HTTP Server 及 MySQL 数据库。
- 重新启动所有服务：用来重新启动 Apache HTTP Server 及 MySQL 数据库。

1.3.3　查看 PHP 文件

PHP 官方网站提供了两种查看 PHP 文件的方式。

- 在线文件：您只要通过浏览器连接到 http://www.php.net/manual/en/，就可以在线浏览英文版的 PHP 文件，如图 1-20 所示。

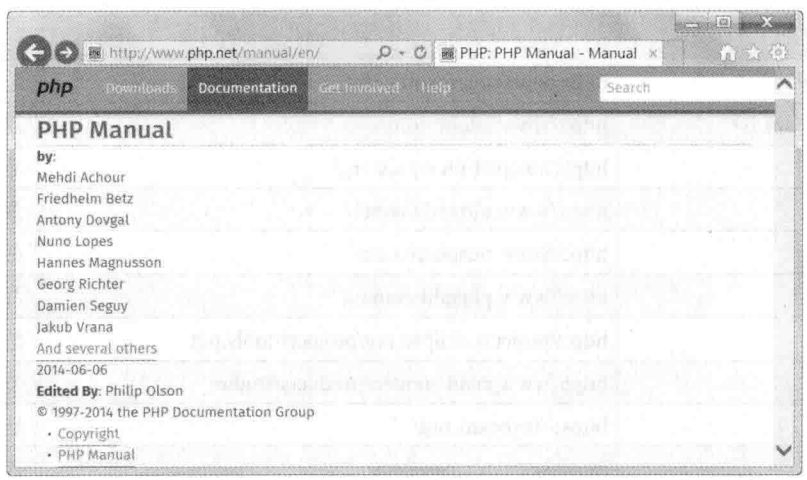


图 1-20

- 下载文件：若您想将 PHP 文件下载到自己的计算机，可以通过浏览器连接到 http://www.php.net/download-docs.php，然后选择适合的格式进行下载，我们的建议是选择英文版的 HTML Help file 格式，如图 1-21 所示。

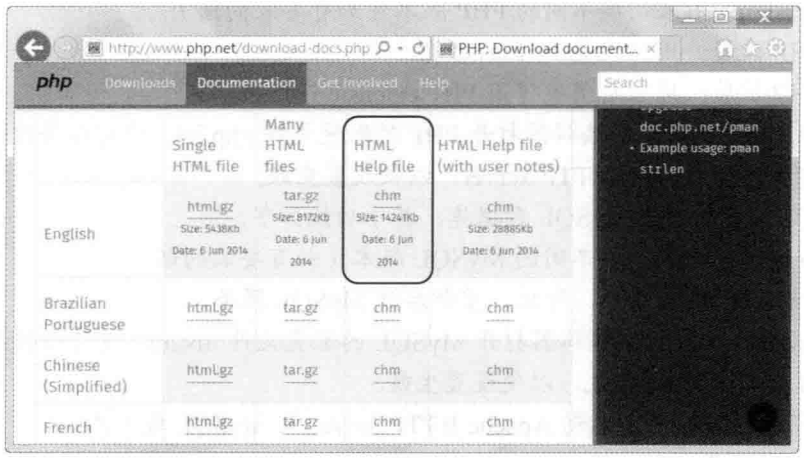


图 1-21

1.4 PHP 程序的编辑工具

PHP 程序其实是一个纯文本文件，只是扩展名为 .php，而不是我们平常惯用的 .txt。原则上，任何能够用来输入纯文本文件的编辑工具，都可以用来编写 PHP 程序，表 1-1 列出了一些常见的编辑工具。

表 1-1

编辑工具名称	网址	是否免费
记事本、WordPad	http://www.microsoft.com/	是
Dreamweaver、GoLive	http://www.adobe.com/	否
NotePad++	http://notepad-plus-plus.org/	是
UltraEdit-32	http://www.ultraedit.com/	否
PhpED	http://www.nusphere.com/	否
PHPEdit	http://www.phpedit.com/en	否
Eclipse PDT	http://projects.eclipse.org/projects/tools.pdt	是
Zend Studio	http://www.zend.com/en/products/studio/	否
NetBeans	https://netbeans.org/	是
TextPad	http://www.textpad.com/	否
HTML-Kit	http://htmlkit.com/	否

上面的编辑工具大致上可以分为两类：

- 所见即所得网页设计软件（WYSIWYG，What You See Is What You Get）：这指的是诸如 Adobe Dreamweaver、Adobe GoLive、HTML-Kit 等网页设计软件，其优点是让您通过图形界面进行网页设计，然后它会自动产生对应的 HTML、CSS、JavaScript

或 PHP 等源代码。

- 文字编辑软件：这指的是 Windows 内置的记事本、WordPad、NotePad++、UltraEdit、PhpED、PHPEdit、Eclipse PDT、Zend Studio、NetBeans 等编辑工具，使用这类软件开发 PHP 程序时，您必须自己编写 HTML、CSS、JavaScript 或 PHP 等源代码，其优点是所有源代码都是您自己输入的，所以不会产生额外的垃圾码，可读性较好，同时网页占用的空间也会相对较小。

文字编辑软件又分为单纯的编辑器和集成开发环境（Integrated Development Environment，IDE），诸如记事本、WordPad、NotePad++、UltraEdit 等属于前者，它们的功能很简单，就是编辑与保存文字。而诸如 PhpED、PHPEdit、Eclipse PDT、Zend Studio、NetBeans 等则属于后者，它们不仅能够编辑与保存文字，还会提供语法标记、拼字检查、显示行号、FTP 存取、PHP 程序代码自动完成、连接数据库、程序代码折叠、网页预览等功能（详细的功能因软件而异），方便用来编写 PHP 程序。

对于想快速设计网页，暂时不想深入学习程序语法的读者来说，所见即所得网页设计软件是较好的选择，因为它隔绝了用户与程序语法，即便用户不具备程序设计知识，也能够设计出图文并茂的网页。

相反的，对于想学习程序语法的读者来说，使用文字编辑软件来设计网页则是较好的选择，因为它可以让用户专注于程序语法，不像所见即所得网页设计软件会产生多余的程序代码，造成困扰。

❖ 使用 NotePad++ 编辑并存储 PHP 程序

在过去，有不少人使用 Windows 内置的记事本来编辑 PHP 程序，因为记事本随手可得且完全免费，但使用记事本会遇到一个问题，就是当我们采用 UTF-8 编码方式保存时，记事本会自动在文件的前端插入 BOM（Byte-Order Mark，字节顺序记号），用来识别文件的编码方式，例如 UTF-8 的 BOM 为 EF BB BF（十六进制）、UTF-16（BE）的 BOM 为 FE FF、UTF-16（LE）的 BOM 为 FF FE 等。

在多数情况下，PHP 程序的文件头部被自动插入 BOM 并不会影响其执行，但少数 PHP 程序可能会导致错误，例如调用 header() 函数输出标头信息的 PHP 程序或启用 Session 功能的 PHP 程序。

本书的范例程序统一采用 UTF-8 编码方式，同时为了避免少数 PHP 程序因为文件头部被插入 BOM 导致错误，我们将使用免费软件 NotePad++ 来编辑 PHP 程序，而不再使用 Windows 自带的记事本，因为 NotePad++ 支持以 UTF-8 无 BOM 编码方式保存。您可以到 NotePad++ 的官方网站 http://notepad-plus-plus.org/ 下载安装程序，以下就为您说明如何使用 NotePad++ 编辑并保存 PHP 程序。

在第一次使用 NotePad++ 编辑 PHP 程序之前，我们要做如下的基本设置。

步骤 01　从菜单栏选择“设置”\“首选项”(英文版为 Setting\Preferences)，然后在“常用”页面中选择 NotePad++ 的语言为“中文简体”，如图 1-22 所示。

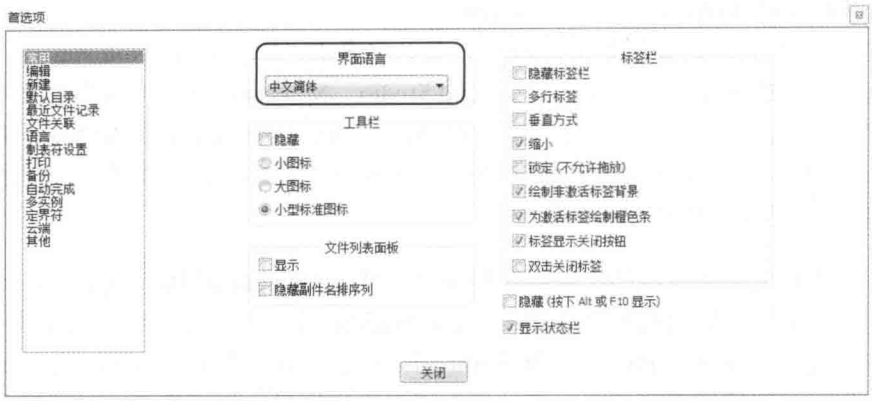


图 1-22

步骤 02 在“新建文档”标签页中设置编码为“UTF-8（无 BOM）”。默认程序设计语言为“PHP”，然后单击“关闭”按钮，如图 1-23 所示。

图 1-23

由于默认的程序设计语言已经设置为 PHP，因此，当我们编辑 PHP 程序时，NotePad++ 就会根据 PHP 的语法，以不同的颜色标记 PHP 程序代码和 HTML 元素，如图 1-24 所示。

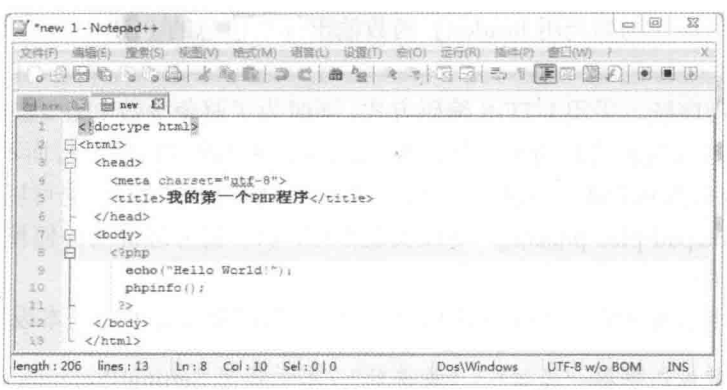


图 1-24

此外，当我们保存文件时，NotePad++ 也会采用“UTF-8（无 BOM）”编码方式，且“保存类型”预设为 PHP（即扩展名为 .php），若要保存为其他类型，例如 HTML，可以在“保存类型”字段中选择 HTML，此时扩展名将变为 .htm 或.html，如图 1-25 所示。

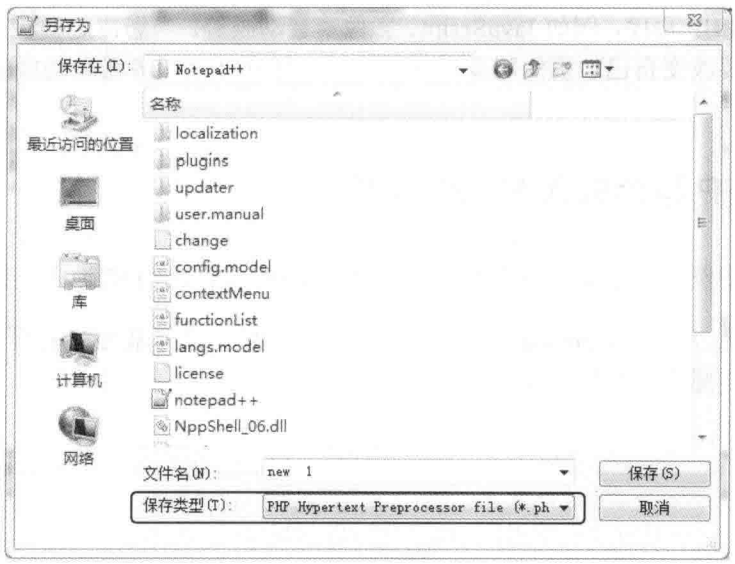


图 1-25

1.5　安装本书范例程序

在您开始动手编写自己的 PHP 程序之前，请按照如下步骤安装本书范例程序，以帮助您获得最佳的学习效果。

步骤 01　将下载\samples 文件夹内的所有文件夹与文件复制到网页主目录，由于我们使用 WampServer 软件包，故网页主目录默认为 C:\wamp\www（请注意，不同的安装方式有不同的网页主目录，例如 AppServer 软件包的网页主目录默认为 C:\AppServer\www）。

步骤 02　取消这些文件夹与文件的只读属性，举例来说，假设我们要在 Microsoft Windows 7 中取消 ch01 文件夹的只读属性，那么可以在 ch01 文件夹上单击鼠标右键，选择“属性”，然后在属性对话框中取消“只读”，再单击“确定”按钮，如图 1-26 所示。

步骤 03　之后您就可以在浏览器的网址栏中输入类似 http://localhost/ch01/hello.php 的网址来运行第 1 章的范例程序 hello.php 了。

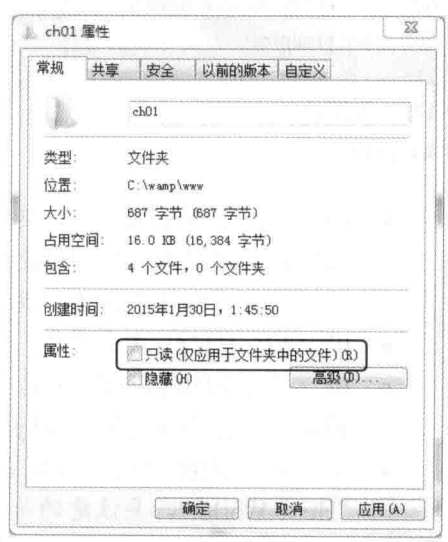


图 1-26

1.6 编写第一个 PHP 程序

PHP 程序可以嵌入 HTML 文件，也可以放在外部文件中，客户端中任何与 HTML 兼容的技术亦兼容于 PHP 程序，例如 JavaScript、影像声音动画等。当您决定要在网页上加入 PHP 程序时，并不需要改变自己一贯的网页设计方式，只要在涉及服务器端功能的部分加入 PHP 程序即可。

1.6.1 将 PHP 程序嵌入 HTML 文件

我们以实际的例子示范如何将 PHP 程序嵌入 HTML 文件，请按照如下步骤操作。

步骤 01 首先，打开 Notepad++，然后编写如下文件，注意最左边的行号和冒号是为了方便介绍，不要实际输入到程序代码中。

\ch01\hello.php

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:    <title>我的第一个 PHP 程序</title>
06:  </head>
07:  <body>
08:    <?php
09:      echo("Hello World!");
10:      phpinfo();
11:    ?>
12:  </body>
13:</html>
```

HTML 文件头

PHP 程序代码段

HTML 文件主体

- 03 ~ 06：HTML 文件头，其中第 04 行是指定网页的编码方式为 UTF-8，若网页有中文输出到浏览器，就一定要加上这行语句，否则中文会变成乱码，而第 05 行是在浏览器的索引标签中显示“我的第一个 PHP 程序”。
- 07 ~ 12：HTML 文件主体，您可以在此放置网页内容。
- 08 ~ 11：PHP 程序代码段，前后以第 08、11 行的 <?php 和 ?> 标记起来，PHP 的语法分析器会去解析 <?php 和 ?> 之间的程序代码。
- 09：调用 PHP 的内部函数 echo()，在网页上显示参数所指定的字符串，例如此处的 "Hello World!"，要注意的是 PHP 程序的语句结尾必须加上分号（;）。
- 10：调用 PHP 的内部函数 phpinfo()，在网页上显示 PHP 的相关信息，包括 PHP 的版本、操作系统、Apache 环境设置、MySQL 支持、ODBC 支持等。

步骤 02　接着，从 NotePad++ 的菜单栏中选择“文件”\“保存”或“文件”\“另存为”，屏幕上会出现“另存为”对话框，请按照图 1-27 进行操作，要注意的是编码方式为 UTF-8（无 BOM）、文件名为 hello.php，并存放在网页主目录下的 ch01 文件夹中，即 C:\wamp\www\ch01（注：编码方式可以在“编码”子菜单中查看，本书的范例程序均采用“编辑成 UTF-8 码（文件头无 BOM）”）。

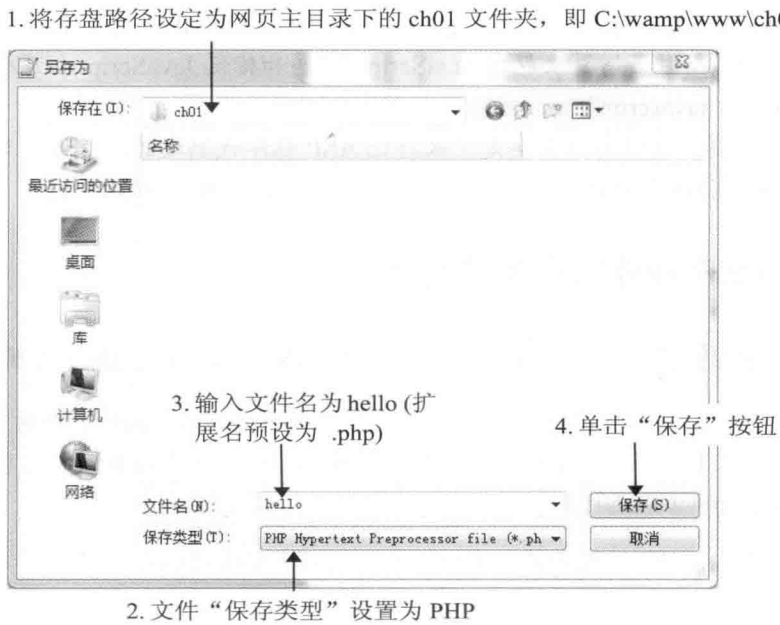


图 1-27

步骤 03　最后，打开浏览器，在网址栏输入 http://localhost/ch01/hello.php 并按 Enter 键，就会得到如图 1-28 所示的执行结果。

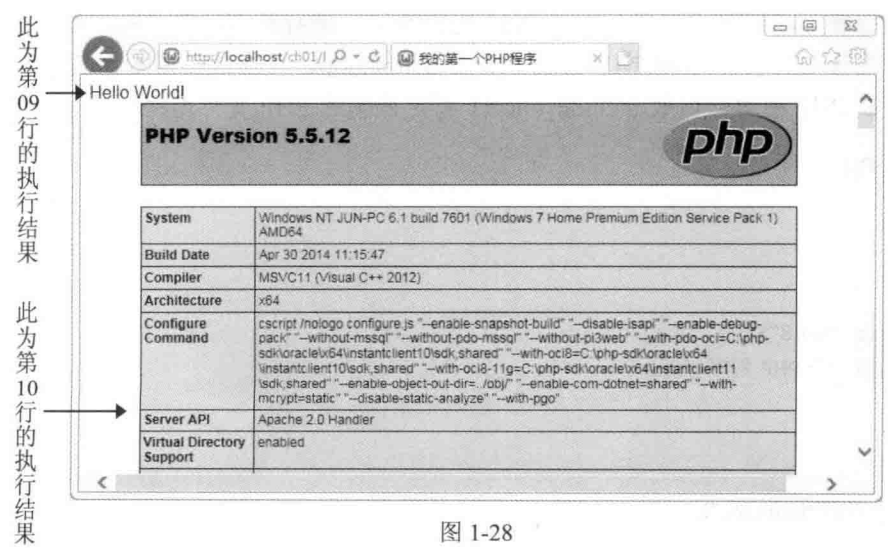


图 1-28

除了以 <?php 和 ?> 标记 PHP 程序代码段之外，您也可以采用下列几种方式，但我们建议您还是以 <?php 和 ?> 的方式为主。

- <? 和 ?>：若您要采用 <? 和 ?> 标记 PHP 程序代码段，那么必须将 php.ini 文件内的 short_open_tag 设置为 on，虽然 <? 和 ?> 比 <?php 和 ?> 简洁，但却与 XML 不兼容。
- <script language="php">...</script>：这种写法以 HTML 元素标记 PHP 程序代码段，不仅冗长，而且若网页上包含 JavaScript，容易和标记 JavaScript 程序代码段的 <script language="javascript">...</script> 混淆。
- <% 和 %>：这种写法其实是用来标记 ASP 程序代码段的，过去，为了鼓励 ASP 网页开发人员改用 PHP，允许以 <% 和 %> 标记 PHP 程序代码段。

1.6.2 将 PHP 程序放在外部文件中

我们以实际的例子示范如何将 PHP 程序放在外部文件，请按照如下步骤操作。

步骤 01 首先，将 PHP 程序放在外部文件中。请打开 NotePad++，然后编写如下 PHP 程序，编码方式为 UTF-8（无 BOM）、文件名为 demo.inc，并存放在网页主目录下的 ch01 文件夹（即 C:\wamp\www\ch01）。

\ch01\demo.inc

```
<?php
  echo("Hello World!");
  phpinfo();
?>
```

步骤 02 接着，在网页内指定外部的 PHP 文件路径。请打开 NotePad++，然后编写如下文件，编码方式为 UTF-8（无 BOM）、文件名为 hello2.php，并存放在网页主目录下的 ch01 文件夹。此处以 PHP 的内部函数 include_once() 指定外部的 PHP 文件路径。

\ch01\hello2.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>我的第一个 PHP 程序</title>
  </head>
  <body>
    <?php
      include_once("demo.inc");
    ?>
```

```
</body>
</html>
```

步骤 03　最后，打开浏览器，在网址栏输入 http://localhost/ch01/hello2.php 并按 Enter 键，就会得到如图 1-29 所示的执行结果。

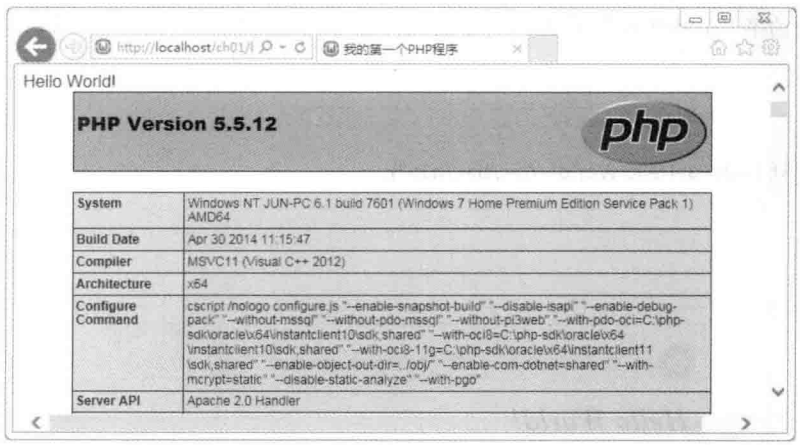


图 1-29

PHP 提供了下列 4 个内部函数可以指定外部的 PHP 文件路径：

- include("*/path/filename*")
- require("*/path/filename*")
- include_once("*/path/filename*")
- require_once("*/path/filename*")

其差异如下：

- 当有错误发生时，include() 和 include_once() 仅会造成警告(warning)，而 require() 和 require_once() 却会造成程序终止执行 (fatal error)。
- include_once() 和 require_once() 仅会载入外部的 PHP 文件一次，不会重复载入，这两个内部函数非常实用，因为在 PHP 程序中重复定义函数，将会导致严重错误 (fatal error)。

既然如此，那么 include() 和 require() 何时才派得上用场呢？当您希望自己编写出完美无瑕的 PHP 程序时，就可以采用这两个比较严格的内部函数，一旦重复定义函数，程序将无法执行，直到找出错误。

在本节的最后，我们来补充说明一点，就是 PHP 的内部函数 echo() 除了能够输出字符串到网页，也能够输出 HTML 元素到网页，下面是一个例子，其中第 09 行的 echo() 函数除了输出字符串“Hello World!”到网页，还输出了<h1>、<b>、<i> 三个 HTML 元素到网页，因此，这个字符串的浏览结果将会呈现标题 1、加粗、斜体的效果，如图 1-30 所示。

\ch01\hello3.php

```
01:<!doctype html>
02:<html>
03:   <head>
04:      <meta charset="utf-8">
05:      <title>我的第一个 PHP 程序</title>
06:   </head>
07:   <body>
08:      <?php
09:         echo("<h1><b><i>Hello World!</i></b></h1>");
10:      ?>
11:   </body>
12:</html>
```

图 1-30

1.7 PHP 程序代码的编写惯例

在说明 PHP 程序代码的编写惯例之前，我们先来讨论什么叫“程序”（program），所谓程序由一行一行的“叙述”或“语句”（statement）所组成，而叙述或语句则是由“保留字”、“特殊字符”或“标识符”所组成。

- 保留字（reserved word）：又称为“关键词”（keyword），它是由 PHP 所定义，包含特定的意义与用途，程序设计人员必须遵守 PHP 的规定来使用保留字，否则会发生错误。举例来说，function 是 PHP 用来定义函数的保留字，所以您不能使用 function 定义一般的变量或常数。
- 特殊字符（special character）：PHP 常用的特殊字符不少，例如定义函数时所使用的小括号（ ）、用来表示程序代码段开头与结尾的大括号 { }、用来表示程序语句结尾的分号 “;”、用来表示变量名称的 $、用来表示多行注释的 /* */ 等。

- 标识符（identifier）：除了保留字、特殊字符之外，程序设计人员可以自行定义新字，作为变量或常数的名称，例如 myName、myAddress……，这些新词语就称为标识符，标识符不一定要合乎英文文法，但要合乎 PHP 命名规则，而且必须区分英文字母大小写，稍后我们会做进一步的讨论。

原则上，叙述或语句是程序内最小的可执行单元，而多个语句可以构成函数（function）、流程控制（flow control）、类（class）等较大的可执行单元。PHP 程序代码的编写惯例涵盖了命名规则、注释、缩排、格式等，虽然不是硬性规定，但遵循这些惯例却可以提高程序的可读性，让程序更容易排错与维护。

❖ 英文字母大小写

HTML 不会区分标签与属性的英文字母大小写，但 PHP 会区分变量名称与常数名称的英文字母大小写，不会区分内部函数或 define、function、if、else、do、for、while 等保留字的英文字母大小写，举例来说，$myName 和$myname 是两个不同的变量，因为大写的 N 和小写的 n 不同，而 if 和 IF 则指的都是 if 判断结构。

❖ 空格符

PHP 会自动忽略多余的空格符，例如下面几个语句的意义均相同：

```
$x = 10;
$x     =     10;
$x   =     10;
```

❖ 分号

PHP 程序的每行语句结尾要加上分号（;），例如：

```
echo（"Hello World!");
```

❖ 注释

PHP 提供了“//”和“#”两种单行注释符号，例如：

```
//这是第一种单行批注符号
#这是第二种单行批注符号
```

PHP 亦提供了“/* */”这种多行注释符号，例如：

```
/* 这是
     多行批注符号 */
```

要注意的是切勿使用嵌套注释，以免发生错误，例如：

```
/*
  xxxxx /* 这是嵌套注释，将发生错误 */
*/
```

另外要提醒您，HTML 的注释元素为 <!-- -->，请勿与 PHP 的注释符号混淆。

❖ 保留字一览

除了下面列出的保留字之外，PHP 的保留字尚有一些预定义的类名称、函数名称、变量名称、常数名称等，由于这些名称非常多，无法一一列举，有需要的读者，可以自行参考 PHP 文件（第 1.3.3 小节介绍过如何查看 PHP 文件）。

and	or	xor	__FILE__
exception	php_user_filter	__LINE__	array()
as	break	case	cfunction
class	const	continue	declare
default	die()	do	echo()
else	elseif	empty()	enddeclare
endfor	endforeach	endif	endswitch
endwhile	eval()	exit()	extends
for	foreach	function	global
if	include()	include_once()	isset()
list()	new	old_function	print()
require()	require_once()	return()	static
switch	unset()	use	var
while	__FUNCTION__	__CLASS__	__METHOD__

一、选择题

( ) 1. 下列哪一项属于服务器端 Scripts？

A. VBScript　　B. JavaScript

C. Java Applet　　D. PHP

( ) 2. 下列哪一项可以用来配置 Web Server？

A. PHP　　B. Apache

C. MySQL　　D. Notepad++

（　）3. 下列哪一项不能用来在 HTML 文件中标记 PHP 程序代码段？

A. < 和 />　　B. <?php 和 ?>

C. <? 和 ?>　　D. <% 和 %>

（　）4. 下列哪个函数可以用来加载外部的 PHP 文件一次？

A. phpinfo()　　B. echo()

C. require()　　D. include_once()

（　）5. 下列关于 PHP 的叙述哪一个是错误的？

A. PHP 会自动忽略多余的空格符

B. PHP 不会区分变量名称和常数名称的英文字母大小写

C. PHP 程序的每行语句结尾要加上分号（;）

D. PHP 属于开放源码软件

二、简答题

1. 简单说明什么是动态网页。它和静态网页有何不同？
2. 简单说明什么是客户端 Scripts。
3. 简单说明什么是服务器端 Scripts。
4. 简单说明什么是保留字。举出三个 PHP 保留字作为例子。
5. PHP 提供了哪些注释符号？

第 2 章
类型、变量、常数与运算符

2.1　类型

和多数程序设计语言一样，PHP 也将数据分成多种“类型”（type），这些类型决定了数据将占用的内存空间、能够表示的范围及程序处理数据的方式。

但和诸如 C、C++、C#、Java 等“强类型”（strongly typed）程序设计语言不同，PHP 属于“弱类型”（weakly typed）程序设计语言，又称为“动态类型”（dynamically typed）程序设计语言，也就是说数据在使用之前无须声明类型，同时可以在运行期间视实际情况动态转换类型。举例来说，PHP 会将 "1 + 23" 视为字符串，而 1 + "23" 则会被视为整数 24，也就是字符串 "23" 会先转换成整数 23，然后和整数 1 相加，于是得到整数 24。

PHP 支持下列 8 种类型，在本节中，我们将依次为您介绍前 6 种，至于数组和对象，则留待第 3 章和第 7 章再做说明：

- 标量类型（scalar type）
  - 整数（integer）
  - 浮点数（float、double）
  - 布尔（boolean）
  - 字符串（string）
- 特殊类型（special type）
  - NULL
  - 资源（resource）
- 复合类型（compound type）
  - 数组（array）
  - 对象（object）

2.1.1　整数（integer）

整数（integer）是最简单的类型，PHP 所支持的整数范围取决于计算机平台的字长（word size），以 32 位平台为例，其整数范围为±（2^{31} - 1），即±2147483647。

PHP 接受十进制、八进制、十六进制整数，诸如 123、-4567……均属于十进制整数（中间不能加上逗号），而八进制整数的前面要加上 0 作为区分，十六进制整数的前面是加上 0x 作为区分，例如：

```
echo(10 + 10);                  //十进制整数相加，会显示十进制整数 20
echo(010 + 010);                //八进制整数相加，会显示十进制整数 16
echo(0x10 + 0x10);              //十六进制整数相加，会显示十进制整数 32
echo PHP_INT_SIZE;              //32 位平台会显示 4，表示为 4 字节
echo PHP_INT_MAX;               //32 位平台会显示 2147483647
```

请注意，当您使用超过范围的整数时（integer overflow），例如2147483648或 -2147483648，PHP 会自动将类型转换成浮点数。

此外，PHP 不支持无符号整数（unsigned integer），至于当前计算机平台的字长及 PHP 所支持的最大整数则可以通过 PHP_INT_SIZE 和 PHP_INT_MAX 两个常数来获得。

2.1.2 浮点数（float、double）

浮点数（float、double）指的是实数，PHP 所支持的浮点数范围亦取决于计算机平台的字长，以 64 位平台为例，最大浮点数范围约为 1.8E+308，有效位数约 14 位。我们可以使用小数点或科学符号表示浮点数，其中科学符号的 E 或 e 没有大小写之分（只有变量名称和常数名称才有大小写之分），例如：

```
echo(-123.456);                    //会显示 -123.456 (符号“-”表示负数)
echo(+12.3);                       //会显示 12.3 (符号“+”或没有符号表示正数)
echo(0.123456789012345);           //会显示 0.123456789012，多出的位数被四舍五入
echo(1.2345E+2);                   //会显示 123.45
echo(-123.45e-3);                  //会显示 -0.12345
```

2.1.3 布尔（boolean）

布尔（boolean）只能表示 TRUE（真）或 FALSE（假）两种值（没有大小写之分），当您要表示的数据只有 TRUE/FALSE、YES/NO 两种选择时，就可以使用布尔类型，换句话说，布尔类型通常用来表示表达式是否成立或某个情况是否满足。

当我们将布尔数据转换成值类型时，TRUE 会转换成 1，FALSE 会转换成 0；当我们将布尔数据转换成字符串类型时，TRUE 会转换成字符串 "1"，FALSE 会转换成空字符串 ""；当我们将非布尔数据转换成布尔类型时，只有下列数据会转换成 FALSE，其他数据均会转换成 TRUE，包括所有负数及任何有效资源：

- 整数 0
- 浮点数 0.0
- 空字符串 "" 与字符串 "0"
- 没有元素的数组
- 没有成员的对象
- 特殊类型 NULL（包括尚未设置的变量）

2.1.4 字符串（string）

字符串（string）指的是由字母、数字、文字、符号所组成的单字、词组或句子。过去，PHP 的字符串只能由 256 种不同的字符组成，而 PHP 6 则支持 Unicode，所有字符串都默认

为 Unicode 格式，不再受限于 256 种字符。我们可以使用下列 4 种方法指定字符串：

- 单引号字符串（single quoted string）
- 双引号字符串（double quoted string）
- heredoc 语法
- nowdoc 语法

❖ 单引号字符串（single quoted string）

在这种表示法中，字符串的前后必须加上单引号（'），例如 'happy'、'快乐'，换码字符（escaped character）有 \\ 和 \' 两个，分别会被解释为 \ 和 '，例如：

```
echo('生日快乐！');          //会显示“生日快乐！” (字符串可以包含中文和全角标点符号)
echo('C:\\Win');             //会显示“C:\Win” (\\ 为换码字符，会被解释为 \)
echo('I am \'Jean\'.');      //会显示“I am 'Jean'.” (\' 为换码字符，会被解释为 ')
```

❖ 双引号字符串（double quoted string）

在这种表示法中，字符串的前后必须加上双引号（"），例如 "小美"、"happy"。双引号字符串和单引号字符串的不同之处在于它支持更多换码字符，而且会进行变量解析（以变量的值取代变量），例如：

```
echo("I am \"Jean\".");      //会显示 I am "Jean". (\" 为换码字符，会被解释为 ")
$str = "Mary";               //将变量 str 设置为字符串 "Mary" (变量的名称前面必须加上 $)
echo("Hi, $str.");           //会显示 Hi, Mary. (变量 str 会被解释为字符串 "Mary")
```

至于 PHP 的语法分析器（parser）如何取得变量名称呢？原则上，语法分析器一碰到 $ 符号，就会取得 $ 符号后面到下一个不是英文字母、阿拉伯数字及下划线（_）的字符之间的字符串，将它当成变量名称，若程序中没有这个变量，就会自动忽略。

换码字符	会被解释为
\n	line feed（换行，ASCII 码为 10）
\r	carriage return（换行，ASCII 码为 13）
\t	horizontal tab（Tab 键，ASCII 码为 9）
\v	vertical tab（Tab 键，ASCII 码为 11）
\f	form feed（换行，ASCII 码为 12）
\\	\（反斜杠）
\$	$（美元符号）
\"	"（双引号）
\[0-7]{1,3}	八进制数字表示法
\x[0-9A-Fa-f]{1,2}	十六进制数字表示法

❖ heredoc 语法

这种表示法有固定的格式，一开始是 <<< 运算符，接着是一个标识符和换行，然后是字符串，最后是以同一个标识符结尾（标识符的命名规则和变量相同），下面是一个例子<\ch02\heredoc.php>。

```
<body>
  <?php
    echo <<< STR1
My name is Jean.<br>
Happy birthday to You!
STR1;
  ?>
</body>
```

运行结果如图 2-1 所示。

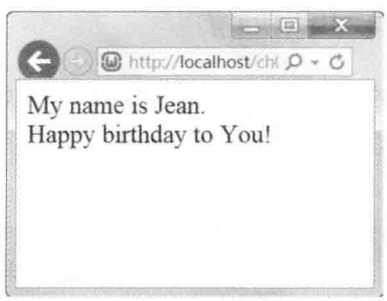


图 2-1

❖ nowdoc 语法

heredoc 语法是针对双引号字符串的，nowdoc 语法则是针对单引号字符串的，即 heredoc 语法会进行变量解析，而 nowdoc 语法不会。下面是一个例子 <\ch02\nowdoc.php>，由于采用 nowdoc 语法，故不会以变量 name 的值取代变量 name。若要进行变量解析，必须改用 heredoc 语法，即去掉标识符 'STR1' 前后的单引号。

```
<body>
  <?php
    $name = "Jean";
    echo <<< 'STR1'
My name is $name.<br>
Happy birthday to You!
STR1;
  ?>
</body>
```

运行结果如图 2-2 所示。

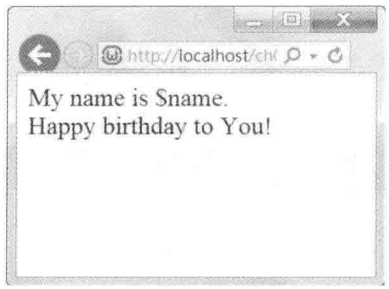


图 2-2

我们可以使用“数组”（array）的概念存取字符串中的字符，而且“键”（key）的起始值为 0，举例来说，假设“$str = "Day";”，那么第一个字符为 $str[0]，即 D，第二个字符为 $str[1]，即 a，其他依次类推，如要更改字符串中的字符，假设要将第一个字符变更为 P，那么可以写成“$str[0] = "P";”，$str 的值将变成 "Pay"。

2.1.5　NULL

凡是类型为 NULL 的变量，就只有一种值——常数 NULL（没有大小写之分），所代表的意义是没有值（no value），例如：

```
$var = NULL;                    //将变量 var 的值设置为 NULL
```

凡是尚未设置值的变量、值被赋值为常数 NULL 的变量或值被 unset() 函数清除的变量均会被视为 NULL。

2.1.6　资源（resource）

资源（resource）类型代表的是一种特殊值，用来指向 PHP 程序的外部资源，例如数据库、文件、图形图像等。通常 resource 类型是在调用函数存取外部资源时自动建立的，例如下面的语句会建立一个 MySQL 数据库连接，然后将资源赋值给变量 my_resource：

```
$my_resource = mysql_connect();
```

一般来说，我们并不需要手动释放资源，PHP Zend Engine 会自动管理除了数据库之外的所有资源。当我们要手动释放资源时，可以将指向资源的变量设置为 NULL，例如：

```
$my_resource = NULL;
```

2.2 类型转换

PHP 会视实际情况自动转换类型，其规则如表 2-1 所示。

表 2-1 类型转换规则

原始类型	目的类型	说明
float	integer	取出整数部分，小数部分无条件舍去，例如浮点数 5.6、-5.6 会分别转换成整数 5、-5
boolean	integer、float	FALSE 会转换成 0，TRUE 会转换成 1
string	integer	从字符串开头取出整数，没有的话，就是 0，例如字符串 "3A"、"8.2bc"、"x12" 会分别转换成整数 3、8、0
string	float	从字符串开头取出浮点数，没有的话，就是 0.0，例如字符串 "3A"、"8.2bc"、"x12" 会分别转换成浮点数 3.0、8.2、0.0
integer	float	在整数后面加上小数点和 0，例如整数 5 会转换成浮点数 5.0
integer、float	boolean	0 会转换成 FALSE，非 0 会转换成 TRUE
String	boolean	空字符串 "" 或字符串 "0" 会转换成 FALSE，其他会转换成 TRUE
integer、float	string	将所有数字（包括小数点）转换成字符串，例如整数 123 会转换成字符串 "123"，浮点数 123.456 会转换成字符串 "123.456"
Boolean	string	FALSE 会转换成空字符串 ""，TRUE 会转换成字符串 "1"
NULL	integer、float	0
NULL、resource	boolean	FALSE
Resource	string	类似 "Resource id #1" 的字符串
resource、array、object	integer、float	没有定义
integer、float	boolean	0 会转换成 FALSE，非 0 会转换成 TRUE
array	boolean	没有元素的数组会转换成 FALSE，否则会转换成 TRUE
array	string	字符串 "Array"
object	boolean	没有成员的对象会转换成 FALSE，否则会转换成 TRUE
object	string	字符串 "Object"
integer、float、boolean、string	array	建立一个新的数组，而且第一个元素就是该整数、浮点数、布尔或字符串

2.2.1 检查类型

PHP 提供了几个函数可以检查数据的类型，比较实用的如表 2-2 所示。

表 2-2　检查数据类型的函数

函数	说明
gettype(*arg*)	返回参数 *arg* 的类型，返回值有 "integer"、"double"、"boolean"、"string"、"NULL"、"resource"、"array"、"object"、"unknown type"
is_integer(*arg*) 、 is_int(*arg*) 、 is_long(*arg*)	若参数 *arg* 为整数类型，就返回 TRUE，否则返回 FALSE
is_float(*arg*)、is_real(*arg*)	若参数 *arg* 为浮点数类型，就返回 TRUE，否则返回 FALSE
is_bool(*arg*)	若参数 *arg* 为布尔类型，就返回 TRUE，否则返回 FALSE
is_string(*arg*)	若参数 *arg* 为字符串类型，就返回 TRUE，否则返回 FALSE
is_null(*arg*)	若参数 *arg* 为 NULL 类型，就返回 TRUE，否则返回 FALSE
is_resource(*arg*)	若参数 *arg* 为 resource 类型，就返回 TRUE，否则返回 FALSE
is_array(*arg*)	若参数 *arg* 为 array 类型，就返回 TRUE，否则返回 FALSE
is_object(*arg*)	若参数 *arg* 为 object 类型，就返回 TRUE，否则返回 FALSE
is_numeric(*arg*)	若参数 *arg* 为数值或数值字符串，就返回 TRUE，否则返回 FALSE
is_scalar(*arg*)	若参数 *arg* 为数值、布尔或字符串等标量类型，就返回 TRUE，否则返回 FALSE

2.2.2　明确转换类型

PHP 提供了下列几种方式可以明确转换数据的类型。

- 使用转型表达式将数据的类型转换成指定类型，例如：

```
echo(int)TRUE;                    //转型表达式 (int) 将 TRUE 转换成整数 1，故显示 1
```

表 2-3　转型表达式

转型表达式	说明
(int)、(integer)	将数据的类型转换成 integer 类型
(float)、(double)、(real)	将数据的类型转换成 float 类型
(bool)、(boolean)	将数据的类型转换成 boolean 类型
(string)	将数据的类型转换成 string 类型
(array)	将数据的类型转换成 array 类型
(object)	将数据的类型转换成 object 类型

- 使用 settype(*var*, *type*) 函数设置类型，第一个参数 *var* 是要设置类型的变量，第二个参数 *type* 是要指定的类型（"integer"、"double"、"boolean"、"string"、"NULL"、"array"、"object"），例如：

```
$var = TRUE;                          //将变量 var 设置为 TRUE (布尔类型)
settype($var, "integer");             //调用 settype() 函数将变量 var 的类型设置为 integer 类型
echo $var;                            //TRUE 转换成 integer 类型会得到 1，故显示 1
```

- 使用 intval(*var*)、floatval(*var*)、strval(*var*) 函数将参数 *var* 分别转换成 integer、float、string 类型，例如：

```
$var = TRUE;                          //将变量 var 设置为 TRUE (布尔类型)
intval($var);                         //调用 intval() 函数将变量 var 的类型转换成 integer 类型
echo $var;                            //TRUE 转换成 integer 类型会得到 1，故显示 1
```

2.3 变量

“变量”（variable）是我们在程序中所使用的一个“名称”（name），计算机会提供预留的内存空间给这个名称，然后我们可以使用它来存放整数、浮点数、字符串、布尔、资源、NULL、数组、对象等，称为变量的“值”（value），而且值可以重新设置或经由运算更改。

2.3.1 变量的命名规则

PHP 规定变量名称的前面必须加上美元符号（$），其命名规则如下：

- 第一个字符可以是英文字母或下划线（_），其他字符可以是英文字母、下划线或阿拉伯数字，而且英文字母有大小写之分。
- 诸如 integer、double、boolean、TRUE、FALSE、if、do、while、echo 等 PHP 的保留字均无大小写之分，但变量名称和常数名称则有大小写之分，所以像 $myName 和 $myname 是不同的变量，因为大写的 N 和小写的 n 不同。
- 不能使用保留字、内置变量的名称、内部函数的名称、内部对象的名称等。
- 避免在内部范围使用与外部范围相同的名称，以免存取错误。

下面是几个例子：

```
$My_Variable1                         //合法的变量名称
$_MyVariable2                         //合法的变量名称
$3MyVariable                          //非法的变量名称，不能以数字开头
$My@Variable4                         //非法的变量名称，不能包含@符号
```

此外，PHP 内置了许多预定义的变量（predefined variable），例如服务器变量（$_SERVER）、环境变量（$_ENV）、HTTP Cookies（$_COOKIE）、HTTP GET 变量（$_GET）、HTTP POST 变量（$_POST）、HTTP 文件上传变量（$_FILES）、Request 变量（$_REQUEST）、Session 变量（$_SESSION）、Global 变量（$GLOBALS）、前一个错误信息（$php_errormsg）……我们会在相关章节中进行介绍。

2.3.2　变量的访问方式

PHP 属于动态类型的程序设计语言，变量在使用之前无须声明类型，同时可以在运行期间视实际情况动态转换类型。我们通常使用等于符号（=）给变量赋值，而且是以最近一次赋的值为主，例如下面的语句是将变量 myName 的值设置为字符串"小丸子"：

```
$myName = "小丸子";      //变量 myName 的类型为 string
```

由于变量 myName 的值为字符串 "小丸子"，故 PHP 会自动将变量 myName 的类型视为 string，若我们中途改变它的值，例如当 PHP 碰到下面的语句时，会自动将变量 myName 的类型转换成 boolean：

```
$myName = TRUE;          //变量 myName 的类型转换成 boolean
```

❖ 参照赋值（assign by reference）

PHP 提供了另一种叫做“参照赋值”的方式，也就是让新变量指向原变量，一旦新变量的值发生改变，原变量的值也会随着改变。以下面的语句为例，由于新变量 var2 指向原变量 var1（前面加上 & 符号），当新变量 var2 的值变为 "Mary" 时，原变量 var1 的值也会随着变为 "Mary"：

```
$var1 = "John";              //原变量 var1 的值为 "John"
$var2 = &$var1;              //新变量 var2 指向原变量 var1 (前面加上 & 符号)
$var2 = "Mary";              //新变量 var2 的值变更为 "Mary"
echo $var2;                  //新变量 var2 的值变更为 "Mary"，故显示 "Mary"
echo $var1;                  //原变量 var1 的值随着新变量 var2 变更为 "Mary"，故显示 "Mary"
```

❖ 可变变量（variable variables）

“可变变量”指的是我们可以动态设置变量的名称，以下面的语句为例，这是一个名称为 var、值为 "Happy" 的变量：

```
$var = "Happy";
```

接着，我们要使用可变变量，以下面的语句为例，这是一个以变量 var 的值为名称的变量，也就是名称为 Happy、其值为 "Birthday" 的变量：

```
$$var = "Birthday";
```

于是下面的语句将分别显示 Happy、Birthday、Birthday：

```
echo $var;                   //显示变量 var 的值，即 "Happy"
echo $$var;                  //显示变量 Happy 的值，即 "Birthday"
```

```
echo $Happy;                    //显示变量 Happy 的值，即 "Birthday"
```

2.3.3 变量的有效范围

“有效范围”（scope）指的是程序中哪些程序段能够访问变量的值，大部分的 PHP 变量都只有一种有效范围，就是程序中的所有程序段皆能访问变量的值，称为“全局变量”（global variable），例外的是在函数（function）内定义的变量，称为“局部变量”（local variable），只有函数内的语句能够访问局部变量的值，我们会在第 4.5 节说明两者的差别。

2.3.4 变量处理函数

除了第 2.2.1 小节所介绍的检查类型函数外，PHP 还提供其他变量处理函数，比较实用的如表 2-4 所示。

表 2-4 变量处理函数

函数	说明
isset(*arg*)	若参数 *arg* 的值不是 NULL，就返回 TRUE，否则返回 FALSE
unset(*arg*)	清除参数 *arg* 的值，使之成为 NULL
empty(*arg*)	若参数 *arg* 的值是空的，就返回 TRUE，否则返回 FALSE。所谓空的指的是整数 0、浮点数 0.0、空字符串 ""、字符串 "0"、空数组、NULL、FALSE、var $var;（在类内声明且尚未设置值的变量）
intval(*arg*)	返回参数 *arg* 的整数值，例如 intval(4.2) 会返回整数值 4
floatval(*arg*)	返回参数 *arg* 的浮点数，例如 floatval('12.345ab') 会返回浮点数 12.345

2.4 常数

“常数”（constant）是一个有意义的名称，它的值不会随着程序的运行而改变，同时程序设计人员亦无法改变常数的值。PHP 提供了“用户自定义常数”和“预定义的常数”两种，以下就为您进行介绍。

2.4.1 用户自定义常数

我们可以使用 define() 函数建立常数，其语法如下：

```
define(name, value[, case_insensitive])
```

- *name*：第一个参数为 string 类型，代表常数的名称，命名规则和变量相同，默认有大

小写之分，一般的惯例是以全部大写来表示。

- *value*：第二个参数为标量类型，代表常数的值。
- *case_insensitive*：第三个参数为 boolean 类型，若省略不写，表示常数的名称有大小写之分，如果要设置成没有大小写之分，可以赋值为 TRUE。

例如下面的语句用于定义一个名称为 PI、其值为 3.14、有大小写之分的常数：

```
define("PI", 3.14);
```

此外，PHP 允许我们根据其他已经定义的常数定义新的常数，例如：

```
define("X", 10 * 5);                    //定义名称为 X、值为 50 的常数，* 为乘法运算符
define("Y", X + 2);                     //定义名称为 Y、值为 52 的常数，+ 为加号
```

请小心不要产生循环引用，以免造成非预期的结果，下面是一个例子：

```
define("X", Y * 5);                     //常数 X 的值根据常数 Y 的值去做定义
define("Y", X * 2);                     //常数 Y 的值又根据常数 X 的值去做定义
echo X;                                 //循环引用导致常数 X 的值变成 0，而这可能是非预期的结果
```

2.4.2　预定义的常数

PHP 内置了几个预定义的常数（predefined constant），如表 2-5 所示。

表 2-5　PHP 预定义常数

预定义的常数	说明
__LINE__	文件的行数
__FILE__	文件名与完整路径
__DIR__	文件所在的目录
__FUNCTION__	函数名
__CLASS__	类名
__METHOD__	方法名
__NAMESPACE__	命名空间的名称

下面是一个例子，它会显示当前网页的文件名与完整路径，以及当前网页所在的目录。

\ch02\constants.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
```

```
</head>
<body>
  <?php
    echo __FILE__;
    echo "<br>";
    echo __DIR__;
  ?>
</body>
</html>
```

运行结果如图 2-3 所示。

图 2-3

2.5 运算符

运算符（operator）可以针对一个或多个元素进行运算，PHP 提供了下列运算符。

- 算术运算符（arithmetic operator）：+、-、*、/、%
- 字符串运算符（string operator）：.
- 递增/递减运算符（incrementing/decrementing operator）：++、--
- 比较运算符（comparison operator）：==、!=、<>、<、>、<=、>=、===、!==
- 位运算符（bitwise operator）：~（NOT）、&（AND）、|（OR）、^（XOR）、<<、>>
- 逻辑运算符（logical operator）：!（NOT）、and、or、xor、&&（AND）、||（OR）
- 赋值运算符（assignment operator）：=、+=、-=、*=、/=、%=、&=、|=、^=、<<=、>>=、.=
- 条件运算符（conditional operator）：?:
- 错误控制运算符（error control operator）：@
- 执行运算符（execution operator）：``
- 数组运算符（array operator）：+、==、===、!=、<>、!==

运算符又分成下列三种类型。

- 单元运算符（unary operator）：这种运算符只有一个操作数（operand），使用前置记法（prefix notation，例如 -x）或后置记法（postfix notation，例如 x++）。
- 二元运算符（binary operator）：这种运算符有两个操作数，使用中置记法（infix notation，例如 x + y）。
- 三元运算符（ternary operator）：这种运算符只有 ?: 一种，它有三个操作数，使用中置记法（例如 c? x : y）。

2.5.1　算术运算符

PHP 提供的算术运算符如表 2-6 所示，其用途是进行算术运算。

表 2-6　PHP 提供的算术运算符

运算符	语法	说明	范例	返回值
+	*操作数 1* + *操作数 2*	*操作数 1* 加上 *操作数 2*	12 + 3	15
-	*操作数 1* - *操作数 2*	*操作数 1* 减去 *操作数 2*	12.8 - 5	7.8
*	*操作数 1* * *操作数 2*	*操作数 1* 乘以 *操作数 2*	8.1 * 3	24.3
/	*操作数 1* / *操作数 2*	*操作数 1* 除以 *操作数 2*	1 / 3	0.333333333333
%	*操作数 1* % *操作数 2*	*操作数 1* 除以 *操作数 2* 的余数	20 % 3	2

- 加号也可以用来表示正值，例如 +5 表示正整数 5；减法运算符也可以用来表示负值，例如 -5 表示负整数 5。
- 若算术运算符左右两边的任一操作数或两个操作数不是值类型，那么 PHP 会先根据第 2.2 节的原则，将操作数转换成值类型，再进行算术运算，例如 5 + TRUE 会得到 6，因为 TRUE 会先转换成整数 1；5 + FALSE 会得到 5，因为 FALSE 会先转换成整数 0。

2.5.2　字符串运算符

PHP 提供的字符串运算符为小数点（.），其用途是连接字符串，下面是几个例子，若字符串运算符左右两边的任一操作数或两个操作数不是 string 类型，那么 PHP 会先根据 2.2 节的原则，将操作数转换成 string 类型，再进行运算。

```
$a = "PHP" . "5";          //将变量 a 的值设置为字符串 "PHP5"
$b = "PHP" . 5;            //将变量 b 的值设置为字符串 "PHP5" (数字 5 会先转换成字符串 "5")
$c = "PHP" . TRUE;         //将变量 c 的值设置为字符串 "PHP1" (TRUE 会先转换成字符串 "1")
$d = 5 . "PHP";            //将变量 d 的值设置为字符串 "5PHP" (注意数字和小数点之间须隔开)
```

2.5.3 递增/递减运算符

递增运算符（++）的语法如下，其用途是将操作数的值加 1，第一种形式的递增运算符出现在操作数的前面，表示运算结果为操作数递增之后的值，第二种形式的递增运算符出现在操作数的后面，表示运算结果为操作数递增之前的值：

```
++操作数
操作数++
```

例如：

```
$X = 10;                    //将变量 X 的值设定为 10
echo(++$X);                 //先将变量 X 的值递增 1，之后再显示变量 X 的值为 11
echo($X);                   //变量 X 的值在前一个语句中递增为 11，因而显示 11
$Y = 5;                     //将变量 Y 的值设定为 5
echo($Y++);                 //先显示变量 Y 的值为 5，之后再将变量 Y 的值递增 1
echo($Y);                   //变量 Y 的值在前一个语句中递增为 6，因而显示 6
```

递减运算符（--）的语法如下，其用途是将操作数的值减 1，第一种形式的递减运算符出现在操作数的前面，表示运算结果为操作数递减之后的值，第二种形式的递减运算符出现在操作数的后面，表示运算结果为操作数递减之前的值：

```
--操作数
操作数--
```

例如：

```
$X = 10;                    //将变量 X 的值设定为 10
echo(--$X);                 //先将变量 X 的值递减 1，之后再显示变量 X 的值为 9
echo($X);                   //变量 X 的值在前一个语句中递减为 9，因而显示 9
$Y = 5;                     //将变量 Y 的值设定为 5
echo($Y--);                 //先显示变量 Y 的值为 5，之后再将变量 Y 的值递减 1
echo($Y);                   //变量 Y 的值在前一个语句中递减为 4，因而显示 4
```

2.5.4 比较运算符

PHP 提供的比较运算符如表 2-7 所示，其用途是比较左右两边的操作数，如果正确，返回 TRUE（真），如果错误，返回 FALSE（伪）。例如 10000 < 20000 会返回 TRUE，而 10000 > 20000 会返回 FALSE。程序设计人员可以根据比较运算符的返回值，做不同的处理。

表 2-7　PHP 提供的比较运算符

运算符	说明	范例	返回值
==	等于	21 + 5 == 18 + 8	TRUE
		"abc" == "ABC"	FALSE
		1 == "1"	TRUE
		1 == TRUE	TRUE
		"1" == TRUE	TRUE
		FALSE == 0	TRUE
!=	不等于	21 + 5 != 18 + 8	FALSE
		"abc" != "ABC"	TRUE
<>	不等于	21 + 5 <> 18 + 8	FALSE
		"abc" <> "ABC"	TRUE
<	小于	18 + 3 < 18	FALSE
>	大于	18 + 3 > 18	TRUE
<=	小于等于	18 + 3 <= 21	TRUE
>=	大于等于	18 + 3 >= 21	TRUE
===	等于且相同类型	1 === "1"	FALSE
		1 === TRUE	FALSE
		"1" === TRUE	FALSE
!==	不等于且/或不同类型	1 !== "1"	TRUE
		1 !== TRUE	TRUE
		"1" !== TRUE	TRUE

2.5.5　位运算符

PHP 提供的位运算符如表 2-8 所示，其用途是进行位运算。

表 2-8　PHP 提供的位运算符

运算符	语法	说明
~ (位 NOT)	~ *操作数*	将*操作数*进行位取反运算（即“非”运算），若位为 1，就返回 0，否则返回 1，例如 ~10 会得到 -11，因为 10 的二进制值是 1010，~10 的二进制值是 0101，而 0101 在二进制补码表示法中是 -11
& (位 AND)	*操作数 1* & *操作数 2*	将*操作数 1* 和*操作数 2* 进行位的“与”运算，若两者对应的位均为 1，位“与”就是 1，否则是 0，例如 10 & 6 会得到 2，因为 10 的二进制值是 1010，6 的二进制值是 0110，而 1010 & 0110 会得到 0010，即 2

（续表）

运算符	语法	说明
|（位 OR）	*操作数 1* | *操作数 2*	将*操作数 1* 和*操作数 2* 进行位的“或”运算，若两者对应的位均为 0，位“或”是 0，否则是 1，例如 10|6 会得到 14，因为 1010|0110 会得到 1110，即 14
^（位 XOR）	*操作数 1* ^ *操作数 2*	将*操作数 1* 和*操作数 2* 进行位“异或”运算，若两者对应的位一个为 1 一个为 0，位“异或”就是 1，否则是 0，例如 10^6 会得到 12，因为 1010^0110 会得到 1100，即 12
<<（向左移位）	*操作数 1*<<*操作数 2*	将*操作数 1* 向左移动*操作数 2* 所指定的位数，例如 1 << 2 表示向左移位 2 个位，会得到 4；-1 << 2 表示向左移位 2 个位，会得到 -4
>>（向右移位）	*操作数 1*>>*操作数 2*	将*操作数 1* 向右移动*操作数 2* 所指定的位数，例如 16 >> 1 表示向右移位 1 个位，会得到 8；-16 >> 1 表示向右移位 1 个位，会得到 -8

2.5.6 逻辑运算符

PHP 提供的逻辑运算符如表 2-9 所示，其用途是进行逻辑运算。请注意，逻辑 AND 运算符有 and 和 && 两个，而逻辑 OR 运算符也有 or 和 || 两个，差别在于优先级不同，后面的第 2.5.11 小节会列表比较。

表 2-9 PHP 提供的逻辑运算符

运算符	语法	说明
!（逻辑 NOT）	! *操作数*	将*操作数*进行逻辑“非”运算，若*操作数*的值为 TRUE，就返回 FALSE，否则返回 TRUE，例如 !（5 > 4）会返回 FALSE，!（5 < 4）会返回 TRUE
and（逻辑 AND）	*操作数 1* and *操作数 2*	将*操作数 1* 和*操作数 2* 进行逻辑“与”运算，若两者的值均为 TRUE，就返回 TRUE，否则返回 FALSE，例如（5 > 4）and（3 > 2）会返回 TRUE，（5 > 4）and（3 < 2）会返回 FALSE
or（逻辑 OR）	*操作数 1* or *操作数 2*	将*操作数 1* 和*操作数 2* 进行逻辑“或”运算，若两者的值均为 FALSE，就返回 FALSE，否则返回 TRUE，例如（5 > 4）or（3 < 2）会返回 TRUE，（5 < 4）or（3 < 2）会返回 FALSE

（续表）

运算符	语法	说明
xor (逻辑 XOR)	*操作数 1* xor *操作数 2*	将*操作数 1* 和*操作数 2* 进行逻辑“异或”运算，若两者的值一个为 TRUE，一个为 FALSE，就返回 TRUE，否则返回 FALSE，例如（5 > 4）xor（3 > 2）会返回 FALSE，（5 > 4）xor（3 < 2）会返回 TRUE
&& (逻辑 AND)	*操作数 1* && *操作数 2*	用途和 and 运算符相同，但 && 运算符的优先级较高
|| (逻辑 OR)	*操作数 1* || *操作数 2*	用途和 or 运算符相同，但 || 运算符的优先级较高

2.5.7　赋值运算符

PHP 提供的赋值运算符如表 2-10 所示，其用途是进行赋值运算。

表 2-10　PHP 提供的赋值运算符

运算符	范例	说明
=	$a = 3;	将 = 右边的值或表达式赋值给 = 左边的变量
+=	$a += 3;	相当于“$a = $a + 3;”，+ 为加号
-=	$a -= 3;	相当于“$a = $a - 3;”，- 为减法运算符
*=	$a *= 3;	相当于“$a = $a * 3;”，* 为乘法运算符
/=	$a /= 3;	相当于“$a = $a / 3;”，/ 为除法运算符
%=	$a %= 3;	相当于“$a = $a % 3;”，% 为余数运算符
&=	$a &= 3;	相当于“$a = $a & 3;”，& 为位 AND 运算符
|=	$a |= 3;	相当于“$a = $a | 3;”，| 为位 OR 运算符
^=	$a ^= 3;	相当于“$a = $a ^ 3;”，^ 为位 XOR 运算符
<<=	$a <<= 3;	相当于“$a = $a << 3;”，<< 为向左移位运算符
>>=	$a >>= 3;	相当于“$a = $a >> 3;”，>> 为向右移位运算符
.=	$a .= "str";	相当于“$a = $a . "str" ;”，. 为字符串运算符

2.5.8　条件运算符

条件运算符 ?: 是一个三元运算符，其语法如下，若条件表达式的结果为 TRUE，就返回第一个表达式的值，否则返回第二个表达式的值：

```
条件表达式 ? 表达式1 : 表达式2
```

例如下面的语句会得到 "YES"：

```
10 > 2? "YES" : "NO"
```

而下面的语句会得到 8：

```
FALSE? 10 + 2 : 10 - 2
```

2.5.9 错误控制运算符

当我们在表达式的前面加上错误控制运算符 @ 时，表达式可能产生的错误信息将会被忽略。以下面的语句为例，我们试图打开不存在的文件 C:\hello.php，正常的情况下会显示警告信息，但只要在表达式的前面加上 @，就不会显示警告信息。请注意，@ 不能放在函数定义、类定义或流程控制等语句的前面。

```
$a = @file("C:\hello.php");
```

2.5.10 执行运算符

执行运算符（` `）的用途是执行 shell 命令，下面是一个例子，它会执行 dir 命令，显示当前路径的目录（如图 2-4 所示）。

\ch02\exe_op.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <?php
      $a = `dir`;
      echo "<pre>$a</pre>";
    ?>
  </body>
</html>
```

若在网页上看到乱码，只要在浏览器画面单击鼠标右键，从快捷菜单中选择“编码”\“简体中文(GB2312)”，就能显示正确的结果。

图 2-4

2.5.11　运算符的优先级

当表达式中有一种以上的运算符时，PHP 会按照如表 2-11 所示的优先级执行运算符，若要改变预设的优先级，可以加上小括号（ ），PHP 就会优先执行小括号内的表达式。

表 2-11　运算符优先级

	分类	运算符
高	递增/递减运算符	++，--
	单元运算符	~，-（int）（float）（string）（array）（object）（bool）@
	逻辑 NOT 运算符	!
	乘除余数运算符	*，/，%
	加减运算符、字符串运算符	+，-，.
	移位运算符	<<，>>
	比较运算符	<，<=，>，>=，<>
	等于运算符	==，!=，===，!==
	位 AND 运算符	&
	位 XOR 运算符	^
	位 OR 运算符	|
	逻辑 AND 运算符	&&
	逻辑 OR 运算符	||
	条件运算符	?:
	赋值运算符	=，+=，-=，*=，/=，%=，&=，|=，^=，<<=，>>=，.=
	逻辑 AND 运算符	and
	逻辑 XOR 运算符	xor
低	逻辑 OR 运算符	or

2.6　PHP 的输出函数

PHP 提供了几个输出函数，比较常用的有 echo、print、var_dump() 等。

❖ echo

```
echo str1[, str2[, str3...]]
```

这个函数可以输出一个或多个字符串（*str1*、*str2*、*str3*…），下面是一个例子，运行结果如图 2-5 所示。

\ch02\output1.php

```
01: <!doctype html>
02:<html>
03:   <head>
04:     <meta charset="utf-8">
05:    </head>
06:   <body>
07:     <?php
08:       echo '<i>Hello!</i><br>';
09:       echo '生日', '快乐', '<br>';
10:       echo '<a href="defualt.htm">回首页</a>';
11:     ?>
12:   </body>
13:</html>
```

- 08: 将参数所指定的字符串 'Hello World!
' 输出至网页，其中 、
 为 HTML 元素，可以将文字加上斜体并强迫换行。
- 09: 将三个参数所指定的字符串 '生日'、'快乐'、'<br>' 输出至网页。
- 10: 将参数所指定的字符串输出至网页，其中 <a> 为 HTML 元素，可以插入超链接（附录 A 中有 HTML 语法教学，有需要的读者，请自行参阅）。

图 2-5

❖ print

```
print str
```

这个函数可以输出一个字符串（*str*），基本上，它的用法和 echo 差不多，不同之处在于 print 只能接受一个参数，而且 print 有返回值，1 表示成功，0 表示失败。我们可以使用 print

将前面的例子 <\ch02\output1.php> 改写成如下的形式，要注意的是 print 不接受多个参数，所以将第 09 行的三个参数合并成一个参数：

```
print '<i>Hello!</i><br>';
print '生日快乐<br>';                                    //将原来的三个参数合并成一个参数
print '<a href="defualt.htm">回首页</a>';
```

var_dump()

```
var_dump(var1[, var2[, var3...]])
```

这个函数可以输出一个或多个变量（*var1*、*var2*、*var3*…）的相关信息，下面是一个例子，它会显示变量 a、b、c 的相关信息，运行结果如图 2-6 所示。

\ch02\var_dump.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <?php
      $a = 1.1;
      $b = TRUE;
      $c = 'Hello!';
      var_dump($a, $b, $c);
    ?>
  </body>

</html>
```

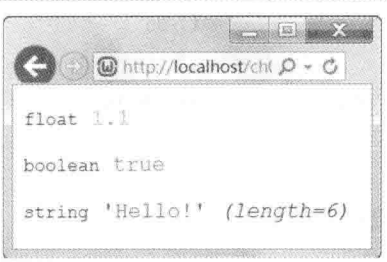


图 2-6

一、选择题

（ ）1. 下列哪一项是正确的数值表示方式？

A. "100"　　B. &O98　　C. 1.23E+5　　D. &H89AB

（ ）2. 下列哪一项是正确的字符串表示方式？

A. 5.67E-5　　B. FALSE　　C. "Ha'p'py"　　D. @"Ha""p""py"

（ ）3. PHP 规定变量名称的前面必须加上下列哪个符号作为识别？

A. $　　B. #　　C. &　　D. @

（ ）4. 假设“$a = 'Merry';”、“$$a = 'Christmas';”，那么“echo $Merry;”会显示下列哪一项？

A. 空字符串　　B. Christmas　　C. Merry　　D. MerryChristmas

（ ）5. 我们可以使用下列哪一项来定义常数？

A. define　　B. const　　C. var　　D. $

（ ）6. 下列哪个运算符可以连接两个字符串？

A. @　　B. .　　C. +　　D. &

（ ）7. 下列哪个运算符可以对两个布尔表达式进行逻辑“或”运算？

A. ?:　　B. !　　C. ||　　D. &&

（ ）8. 下列哪一项的结果为 FALSE？

A. 13 === "13"　　B. x < y　　C. !("ab" == "AB")　　D.（1 < 4）||（3 > 5）

（ ）9. (10 > 20）xor（50 > 80）的结果是什么？

A. TRUE　　B. FALSE

（ ）10. (12 | 7）的结果是什么？

A. TRUE　　B. FALSE　　C. 15　　D. 7

（ ）11. 下列哪个运算符可以用来执行 shell 命令？

A. ``　　B. .　　C. +　　D. &

（ ）12. -1024 >> 2=?

A. 256　　B. -256　　C. -2048　　D. -512

二、简答题

写出下列各个表达式的结果：

1. 0xFFF - 0xFAB
2. 2 / 3.0
3. (int)2 / 3.0
4. (int)(TRUE + TRUE)
5. 12.3 * 10 % 5

```
6.    'a' > 'Z'
7.    ~0x0005
8.    $X = 5; echo(++$X);
9.    $Y = 10; echo($Y--);
10.   50 > 30 ^ 70 > 100
11.   3 | 5
12.   Z = 5 > 7 ? "对" : "不对"
13.   128 << 3
14.   -4096 >> 5
15.   $Z = 10.5; var_dump($Z);
16.   echo(int)is_bool(1.0);
17.   echo(int)is_scalar('abc');
18.   echo(int)'10.5abc';
19.   echo intval(FALSE);
20.   echo(int)(("aBC" < "ABC") ^("abc" != "ABC"));
```

第 3 章
流程控制与数组

3.1　认识流程控制

我们在前两章所介绍的例子都是相当“单纯”的程序，所谓“单纯”指的是程序的执行方向只会由上而下，不会转弯或跳转。但事实上，大部分程序并不会这么单纯，它们会根据不同情况而转弯或跳转，以提高程序的处理能力，于是就需要“流程控制”（flow control）来帮助程序设计人员控制程序的执行方向。

PHP 的流程控制又分为下列两种类型。

- 判断结构（decision structures）：判断结构可以测试程序设计人员提供的条件，然后根据条件判断的结果执行不同的操作，PHP 支持如下判断结构。
  - if（if、if...else...、if...elseif...）
  - switch
- 循环结构（loop structures）：循环结构可以重复执行某些程序代码，PHP 支持如下循环结构。
  - for
  - foreach
  - while
  - do...while

流程控制通常需要借助于逻辑数据，以下是常见的类型与逻辑数据之间的关联：

- 等于 0 的数值会被视为 FALSE，不等于 0 的数值会被视为 TRUE。
- 空字符串 "" 与字符串 "0" 会被视为 FALSE，其他字符串会被视为 TRUE。
- 没有元素的数组和没有成员的对象会被视为 FALSE。
- NULL 会被视为 FALSE。

3.2　if

3.2.1　if：若...就...（单向选择）

```
if（condition) statement;
```

这种判断结构非常简单见图 3-1，字面翻译过来的意义是“若...就...”，属于单向选择。*condition* 是一个条件判断式，它的结果为布尔类型，若 *condition* 返回 TRUE，就执行后面的 *statement*（语句）；若 *condition* 返回 FALSE，就跳过整个 if 判断结构，不会执行后面的 *statement*（语句）。

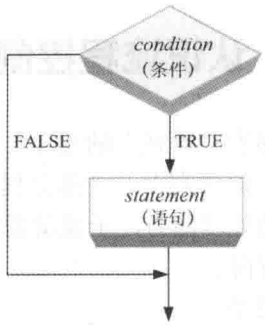


图 3-1

请注意，若 if 后面的 *statement*（语句）有很多行，那么要加上大括号 { } 标记 *statement*（语句）的开头与结尾，如下所示：

```
if (condition)
{
   statement1;
   statement2;
   ...
   statementN;
}
```

编写如下网页，然后启动浏览器执行网页，看看执行结果如何。

\ch03\if1.php

```
<!doctype html>
<html>
   <head>
      <meta charset="utf-8">
   </head>
   <body>
      <?php
         $a = 20;
         $b = 10;
         if（$a > $b) echo '$a 比$b 大';
      ?>
   </body>
</html>
```

【解答】

这个网页的执行结果如图 3-2 所示，由于变量 a 的值（20）大于变量 b 的值（10），故条件（$a > $b）会返回 TRUE，进而执行 if 后面的语句“echo '$a 比$b 大';”，在网页上显示 $a 比$b 大，此处使用单引号字符串的原因是为了不要进行变量解析。您不妨试着交换变量 a 和变量 b 的值，看看执行结果有何不同。

图 3-2

3.2.2 if...else...：若...就...否则...（双向选择）

```
if (condition)
{
  statements1;
}
else
{
  statements2;
}
```

这种格式和前一节所介绍格式的不同之处在于多了 else 语句，字面翻译过来的意义是“若...就...否则...”，属于双向选择。*condition* 是一个条件判断语句，它的结果为布尔类型，若 *condition* 返回 TRUE，就执行 if 后面的 *statements1*（语句 1），然后跳出整个 if...else...判断结构，否则执行 else 后面的 *statements2*（语句 2），它之所以称为双向选择，就是因为比前一节的格式多了一种变化见图 3-3。

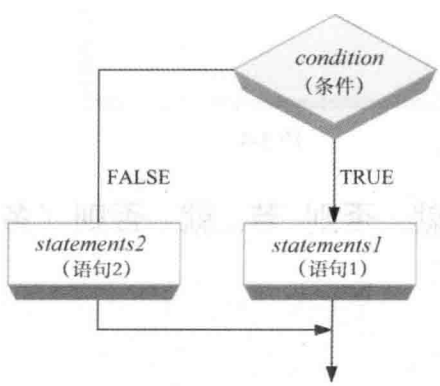


图 3-3

编写如下网页，然后启动浏览器执行网页，看看执行结果如何。

\ch03\if2.php

```
<!doctype html>
<html>
  <head><meta charset="utf-8"></head>
  <body>
    <?php
      $score = 59;
      if（$score > 60)
        echo '及格！';
      else
        echo '不及格！';
    ?>
  </body>
</html>
```

【解答】

这个网页的执行结果如图 3-4 所示，由于变量 score 的值为 59，小于 60，故条件（$score > 60）会返回 FALSE，进而执行 else 后面的语句“echo '不及格！';”，在网页上显示“不及格！”；相反的，当变量 score 的值大于 60 时，条件（$score > 60）会返回 TRUE，进而执行条件后面的语句“echo '及格！';”，在网页上显示“及格！”。

图 3-4

3.2.3 if...elseif...：若...就...否则 若...就...否则（多向选择）

```
if (condition1)
{
  statements1;
}
```

```
elseif (condition2)
{
   statements2;
}
elseif (condition3)
{
   statements3;
}
...
else
{
   statementsN+1;
}
```

这种格式最复杂但实用性也最高，字面翻译过来的意义是“若...就...否则 若...就...否则...”，属于多向选择，前两节所介绍的格式都只能处理一个条件，而这种格式可以处理多个条件，如图 3-5 所示。

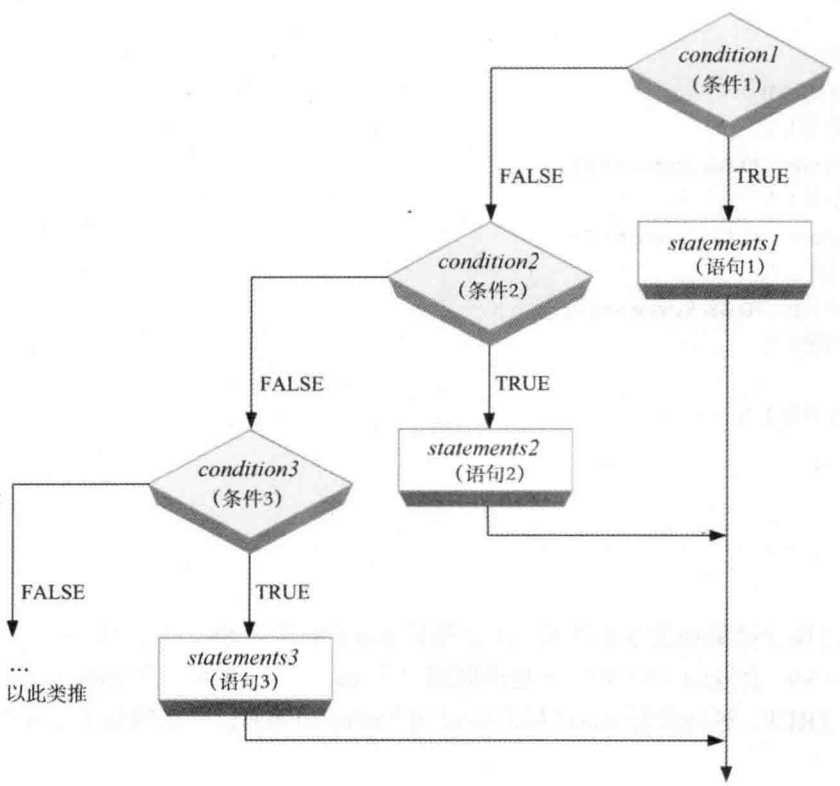


图 3-5

程序执行时会先检查 *condition1*（条件 1），若 *condition1* 返回 TRUE，就执行 *statements1*（语句 1），然后跳出整个 if...elseif...判断结构；若 *condition1* 返回 FALSE，就检查 *condition2*

（条件 2），若 *condition2* 返回 TRUE，就执行 *statements2*（语句 2），然后跳出整个 if...elseif... 判断结构，否则继续检查 *condition3*……以此类推，若所有条件皆不成立，就执行 else 后面的 *statementsN+1*，故 *statements1* 至 *statementsN+1* 只有一个会被执行。

编写如下网页，然后启动浏览器执行网页，看看执行结果如何？

\ch03\if3.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <?php
      $score = 85;
      if（$score >= 90)
        echo '优等！ ';
      elseif（$score < 90 && $score >= 80)
        echo '甲等！ ';
      elseif（$score < 80 && $score >= 70)
        echo '乙等！ ';
      elseif（$score < 70 && $score >= 60)
        echo '丙等！ ';
      else
        echo '不及格！ ';
    ?>
  </body>
</html>
```

【解答】

这个网页的执行结果如图 3-6 所示，由于变量 score 的值为 85，小于 90 大于等于 80，故条件（$score >= 90）会返回 FALSE，于是跳到第一个 elseif，此时条件（$score < 90 && $score >= 80）会返回 TRUE，进而执行 elseif 后面的语句“echo '甲等！ ';”，在网页上显示“甲等！”。

图 3-6

同理，当变量 score 的值小于 80 大于等于 70 时，条件（$score >= 90）会返回 FALSE，于是跳到第一个 elseif，此时条件（$score < 90 && $score >= 80）会返回 FALSE，于是又跳到第二个 elseif，此时条件（$score < 80 && $score >= 70）会返回 TRUE，进而执行 elseif 后面的语句“echo '乙等！';”，在网页上显示“乙等！”，其他数字可依次类推。

3.3　switch

switch 判断结构可以根据变量的值而有不同的执行方向，您可以将它想象成一个有多种车位的车库，这个车库根据车辆的种类来分配停靠位置，若进来的是小客车，就会进入小客车专用的车位，若进来的是大货车，就会进入大货车专用的车位，其语法如下：

```
switch(expression)
{
   case value1:
      statements1;
      break;
   case value2:
      statements2;
      break;
   ...
   default:
      statementsN+1;
}
```

我们必须先给 switch 判断结构一个表达式 *expression* 当作判断的对象，就好像上面比喻的车库以车辆的种类当作判断的对象，接下来的 case 则是要写出这个表达式可能的结果，就好像车辆可能有几个种类。

switch 判断结构会从第一个值 *value1* 开始做比较，看看是否和表达式 *expression* 的值相等，若相等，就执行其下的语句 *statements1*，执行完毕后，break 语句会令其跳离 switch 判断结构；相反的，若不相等，就和第二个值 *value2* 做比较，看看是否和表达式 *expression* 的值相等，若相等，就执行其下的语句 *statements2*，执行完毕后，break 语句会令其跳离 switch 判断结构……依次类推；若没有任何值和表达式 *expression* 的值相等，就执行 default 下的程序代码 *statementsN+1*，如图 3-7 所示。

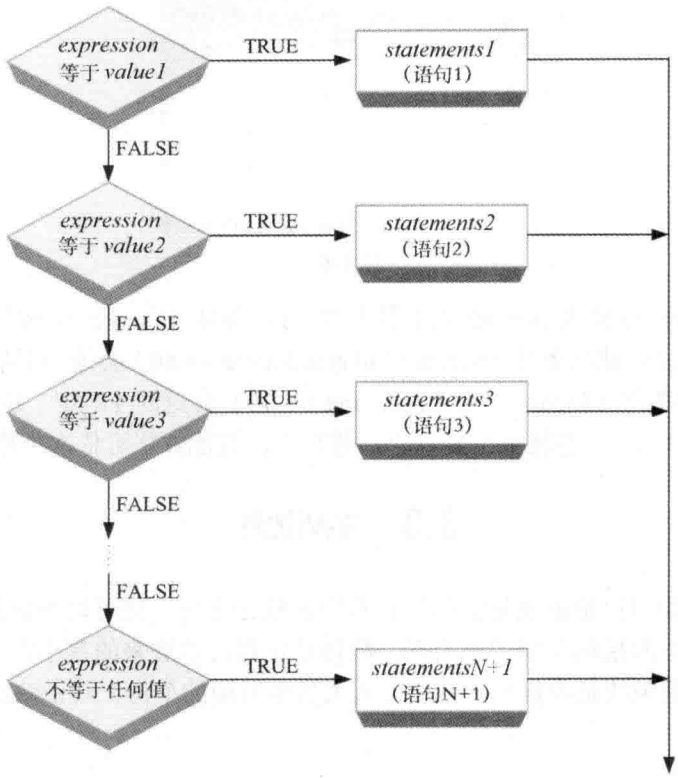


图 3-7

请注意，若在 switch 判断结构中省略了 break 语句，将会执行其下的所有程序代码，直到抵达 switch 判断结构的结尾。

随堂练习

编写一个 PHP 网页，令它使用 switch 判断结构，根据变量 number 的值显示对应的英文，当变量 number 的值为 1、2、3、4、5 时，分别显示“ONE”、“TWO”、“THREE”、“FOUR”、“FIVE”，当变量 number 的值为其他数值时，显示“数值超过范围！”，如图 3-8 所示。

当变量 number 的值为 3 时，会显示“THREE”，您不妨试着修改它的值，看看执行结果有何不同。

图 3-8

【解答】

\ch03\switch.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <?php
      $number = 3;                    //假设变量 number 的值为 3
      switch($number)
      {
        case 1:                       //当变量 number 的值为 1 时
          echo 'ONE';
          break;
        case 2:                       //当变量 number 的值为 2 时
          echo 'TWO';
          break;
        case 3:                       //当变量 number 的值为 3 时
          echo 'THREE';
          break;
        case 4:                       //当变量 number 的值为 4 时
          echo 'FOUR';
          break;
        case 5:                       //当变量 number 的值为 5 时
          echo 'FIVE';
          break;
        default:                      //当变量 number 的值为 1-5 以外的数字时
          echo '数值超过范围！';
      }
    ?>
  </body>
</html>
```

3.4　for（计数循环）

重复执行某项功能是计算机的专长之一，若每执行一次，就要编写一次程序代码，那么大部分程序必然非常冗长，而 for（计数循环）就是用来解决重复执行的问题，例如要打印一个班级的成绩单，只要使用 for 循环逐一取出每个学生的成绩发送给打印机，程序代码编写一次即可，其他例子还有“算出 1 加 2 加 3 一直加到 100 的总和”、“显示九九表”等，如图 3-9 所示。

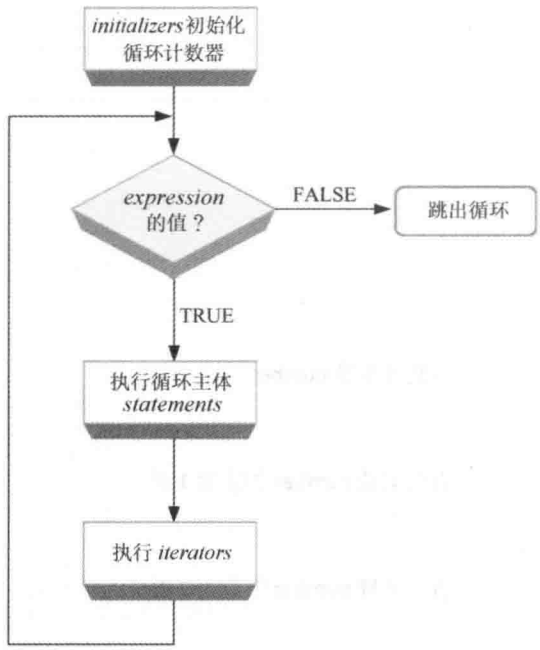


图 3-9

```
for (initializers; expression; iterators)
{
   statements;
   [break;]
   statements;
}
```

PHP 在开始执行 for 循环时，会先通过 *initializers* 初始化循环计数器，*initializers* 使用逗点分隔的表达式列表或赋值语句，然后计算表达式 *expression* 的值，若 *expression* 返回 FALSE，就跳出 for 循环，否则执行 for 循环内的 *statements*（语句），完毕后跳回 for 处执行 *iterators*，再计算表达式 *expression* 的值，若 *expression* 返回 FALSE，就跳出 for 循环，否则执行 for 循环内的 *statements*（语句），完毕后跳回 for 处执行 *iterators*，再计算表达式 *expression* 的值……如此周而复始，直到表达式 *expression* 返回 FALSE。

原则上，在我们编写 for 循环后，PHP 就会忠实地将 for 循环执行完毕，不会中途离开循环。不过，有时我们可能需要在 for 循环内检查某些条件，一旦条件符合便强制离开循环，此时可以使用 break 语句强制离开循环。此外，若 for 循环省略了 *initializers*、*expression*、*iterators*，也就是 for（;;），则会得到一个无限循环（infinite loop）。

下面是一个例子 <\ch03\for1.php>，它会使用 for 循环在网页上显示 1 到 10，如图 3-10 所示。

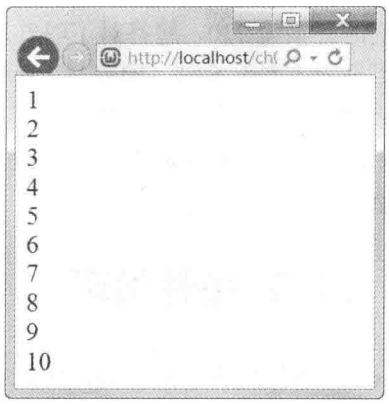


图 3-10

```
<?php
   for ($i = 1; $i <= 10; $i++)
      echo $i.'<br>';
?>
```

❖ break 语句的妙用

原则上，在我们编写 for 循环后，程序就会将 for 循环执行完毕，不会中途跳出循环。不过，有时我们可能需要在 for 循环内检查某些条件，一旦条件符合便强制离开循环，此时可以使用 break 语句强制离开循环，下面是一个例子。

\ch03\for4.php

```
01:<!doctype>
02:<html>
03:   <head>
04:      <meta charset="utf-8">
05:   </head>
06:   <body>
07:      <?php
08:         $result = 1;
09:         for（$i = 1; $i <= 10; $i++)
10:         {
11:            if（$i > 6) break;
12:            $result = $result * $i;
13:         }
14:         echo $result;
15:      ?>
16:   </body>
17:</html>
```

猜猜看结果是多少呢？正确的答案为 720，您答对了吗？事实上，这个 for 循环并没有执行到 10 次，一旦第 11 行检查到变量 i 大于 6 时（即变量 i 等于 7），就会执行 break 语句强制离开循环，故 result 的值为 1 * 2 * 3 * 4 * 5 * 6 = 720。

注意，break 语句不仅可以用来强制离开 for 循环，还可以用来强制离开 while 循环、do...while 循环、函数等程序代码段。

3.5 条件循环

条件循环（conditional loops）以条件是否成立作为循环执行与否的根据，有别于 for 循环是以计数器作为循环执行与否的根据。

3.5.1 while

```
while(condition)
{
   statements;
   [break;]
   statements;
}
```

这种循环结构在执行到 while 时，会先检查 *condition*（条件）是否成立，即是否为 TRUE，若为 FALSE，就跳出循环，若为 TRUE，就执行 *statements*（语句），如图 3-11 所示，碰到 while 循环的结尾时又回到 while 循环的开头，再度检查 *condition* 是否成立。若要在中途强制离开循环，可以加上 break 语句。此处的 *condition* 弹性很大，只要 *condition* 的返回值为 FALSE，就会结束循环，而不必限制循环执行的次数，使用范围比 for 循环广泛。

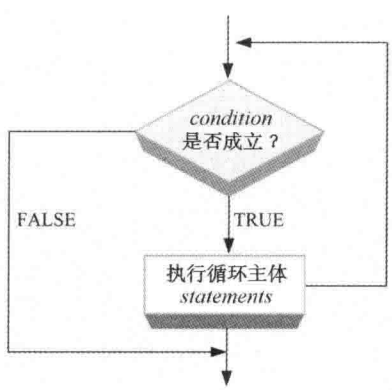


图 3-11

下面是一个例子 <\ch03\while1.php>，它会使用 while 循环在网页上显示 1 到 10，见图 3-12 所示。

图 3-12

```
<?php
   $i = 1;
   while ($i <= 10)
      echo $i++.'<br>';
?>
```

3.5.2　do...while

```
do
{
   statements;
   [break;]
   statements;
}while(condition);
```

这种循环结构在执行到 do 时，会先执行 *statements*（语句），完毕后碰到 while，再检查 *condition*（条件）是否成立，即是否为 TRUE，若为 FALSE，就跳离循环，若为 TRUE，就再回到 do，继续执行 *statements*，如此可以确保 *statements* 至少执行一次，见图 3-13 所示。

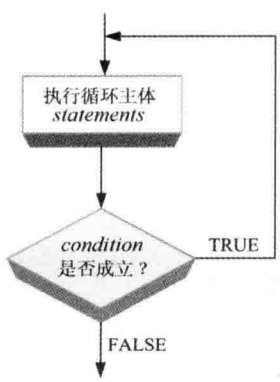


图 3-13

以下面的程序代码为例，它使用 do...while 循环改写前一个例子 <\ch03\while1.php> 的

while 循环，执行结果同样是在网页上显示 1 到 10。

```
<?php
  $i = 1;
  do
    echo $i++.'<br>';
  while ($i <= 10)
?>
```

3.5.3 break 与 continue 语句

正如前面所言，break 语句可以用来强制离开 for 循环、while 循环、do...while 循环、函数等程序代码段，此处不再重复讲解。

至于 continue 语句则可以用来在循环内跳过后面的语句，直接返回循环的开头，例如下面的程序代码只会在网页上显示 11 到 15，见图 3-14 所示。因为在执行到“if($i <= 10)continue;”语句时，只要变量 i 小于等于 10，就会跳过“continue;”后面的语句，直接返回循环的开头，直到变量 i 大于 10，才会执行“echo $i.'
';”语句，在网页上显示变量 i 的值。

\ch03\while2.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <?php
      for（$i = 1; $i <= 15; $i++)
      {
        if（$i <= 10) continue;
        echo $i.'<br>';
      }
    ?>
  </body>
</html>
```

图 3-14

3.5.4　exit() 函数

我们知道，break 语句可以用来强制离开程序代码段，但若我们希望强制离开程序终止执行呢？此时，必须改用 exit() 函数或其同名函数 die()，这个函数有一个参数，当参数为字符串时，表示会强制终止程序并在网页上显示字符串，而当参数为 0～254 的整数时，表示会强制终止程序但不会在网页上显示信息，整数参数代表的是错误码。例如，下面的语句会强制离开程序并在网页上显示“文件打开失败，程序终止执行！”，让浏览者了解程序终止执行的原因：

```
exit('文件打开失败，程序终止执行！');
```

3.6　foreach

foreach 循环是专门设计给“数组”(array)使用的 for 循环，其格式有下列两种，*array_name* 为数组名，$*value* 为数组内当前元素的值，$*key* 为数组内当前元素的键，foreach 循环每重复一次，就会移往数组的下一个元素，若中途要强制离开 foreach 循环，可以加上 break 语句。

```
foreach (array_name as $value)
{
   statements;
   [break;]
   statements;
}

foreach (array_name as $key => $value)
{
   statements;
   [break;]
   statements;
}
```

数组和变量一样是用来存放数据的，不同的是数组虽然只有一个名称，却可以用来存放多个数据。数组所存放的每个数据称为“元素”（element），每个元素有各自的“值”（value）。那么数组是如何区分它所存放的元素呢？答案是通过“键”（key），在默认的情况下，数组内第一个元素的键为 0，第二个元素的键为 1……依次类推，第 n 个元素的键为 n - 1，稍后我们会对数组做进一步的介绍。

以下面的程序代码为例，<\ch03\foreach1.php> 的第 08 行定义了一个名称为 city 且包含三个元素（'东京'、'台北'、'纽约'）的数组，而 <\ch03\foreach2.php> 的第 08 行也定义了一个名称为 city 且包含三个元素（'东京'、'台北'、'纽约'）的数组，见图 3-15 所示，不同的是它还赋值了这三个元素的键（'Japan'、'Taiwan'、'USA'），如图 3-16 所示。

\ch03\foreach1.php

```
01:<!doctype html>
02:<html>
03:   <head>
04:      <meta charset="utf-8">
05:   </head>
06:   <body>
07:      <?php
08:         $city = array('东京', '台北', '纽约');
09:         foreach（$city as $value)
10:            echo $value.'<br>';
11:      ?>
12:   </body>
13:</html>
```

\ch03\foreach2.php

```
01:<!doctype html>
02:<html>
03:   <head>
04:      <meta charset="utf-8">
05:   </head>
06:   <body>
07:      <?php
08:         $city = array('Japan' => '东京', 'Taiwan' => '台北', 'USA' => '纽约');
09:         foreach（$city as $key => $value)
10:            echo '键：'.$key.'；值：'.$value.'<br>';
11:      ?>
12:   </body>
13:</html>
```

图 3-15

图 3-16

3.7　认识数组

我们知道，计算机可以执行重复的动作，也可以处理大量的数据，但到目前为止，我们都只是定义了极小量的数据，若想定义成百上千个字符或数字，该怎么办呢？难道要写出成百上千个语句吗？当然不是！此时，您应该使用“数组”（array）定义大量的数据，而接下来的内容就是要告诉您如何建立及存取数组。

数组和变量一样是用来存放数据的，不同的是数组虽然只有一个名称，却可以用来存放多个数据。数组所存放的每个数据称为“元素”（element），每个元素有各自的“值”（value）。那么数组是如何区分它所存放的元素呢？答案是通过“键”（key），在默认的情况下，数组内第一个元素的键为 0，第二个元素的键为 1，……依次类推，第 n 个元素的键为 n - 1。

当数组的元素个数为 n 时，表示数组的“长度”（length）为 n，而且除了“一维”（one-rank、one-dimension）数组之外，PHP 也允许我们使用“多维”（multi-rank、multi-dimension）数组。

PHP 规定数组的“键”（key）必须为整数或字符串（但不能是数组或对象），例如下面的语句是将元素的值 '樱樱美代子' 存放在一维数组 arr 内键为 0 的位置：

```
$arr[0]= '樱樱美代子';
```

而下面的语句是将元素的值 '樱樱美代子' 存放在一维数组 arr 内键为 '姓名' 的位置：

```
$arr['姓名']= '樱樱美代子';
```

我们也可以使用多维数组，例如下面的语句是将元素的值 '牡丹花' 存放在二维数组 arr 内键为 1、2 的位置：

```
$arr[1][2]= '牡丹花';
```

而下面的语句是将元素的值 '牡丹花' 存放在二维数组 arr 内键为 'flower'、'red' 的位置：

```
$arr['flower']['red']= '牡丹花';
```

事实上，PHP 所支持的数组属于“关联数组”（associative array），有别于 C、C++、Java、C# 等程序设计语言所支持的“向量数组”（vector array），在向量数组中，数组的大小通常得事先声明，每个元素的类型必须相同，同时只能通过 0、1、2、3 等整数键进行存取，其优点是存取效率较佳，因为编译程序已经事先根据数组的大小及元素的类型分配内存空间给数组，自然可以快速计算出所要存取的元素位置。

相反的，在关联数组中，数组的大小无须事先声明，每个元素的类型不一定要相同，同时可以通过整数键或字符串键进行存取，因此，编译程序不会事先分配内存空间给数组，而是在用户新增元素时，再分配内存空间给该元素，然后放到数组后面。

此外，还有下列注意事项：

- 除了中括号[]之外，您也可以使用大括号 {} 存取数组的元素，但建议还是使用中括号[]。
- 当您使用布尔数据作为数组的键时，TRUE 会转换成 1，而 FALSE 会转换成 0。
- 当您使用 NULL 作为数组的键时，NULL 会转换成空字符串 ""。
- 当您将 integer、float、boolean、string、resource 等类型的数据转换为数组时，会得到一个包含一个元素的数组，而且元素的键为 0，值为该数据。
- 当您将 NULL 转换为数组时，会得到一个空数组。
- 当数组的键为常数或变量时，不能在其前后加上单引号，否则 PHP 将不会去解析该常数或变量。

3.8 一维数组

3.8.1 建立一维数组

1. 直接赋值

建立一维数组最简单的方式就是直接赋值它的键（key）与值（value），键必须为整数或字符串，值则无此限制。以下面的语句为例，若一维数组 my_array 内没有键为 0，就将元素的值 100 新增至数组后面并令其键为 0；相反的，若一维数组 my_array 内已经有键为 0，就将元素的值 100 覆写至数组内键为 0 的位置：

```
$my_array[0]= 100;
```

2. 使用 array() 函数

除了直接赋值之外，我们也可以使用 array() 函数建立数组，举例来说，假设要建立一个空数组，然后赋值给变量 my_array，可以写成如下的形式：

```
$my_array = array();
```

假设要建立一个包含三个元素（'台北'、'纽约'、'东京'）的一维数组，然后赋值给变量 my_array，可以写成如下的形式，由于这个语句没有赋值键，故预设为 0、1、2：

```
$my_array = array('台北', '纽约', '东京');
```

若要自行赋值键为 'Taiwan'、'USA'、'Japan'，可以将这个语句改写成如下的形式：

```
$my_array = array('Taiwan' => '台北', 'USA' => '纽约', 'Japan' => '东京');
```

请注意，当您给元素赋了值但没有给元素的键赋值时，例如“my_array[]= 100;”，那么默认的键为数组内最大的键加 1，若数组内没有正整数键（负数或字符串），那么默认的键

为 0。

3.8.2　存取一维数组

存取一维数组最简单的方式就是通过键指定所要存取的元素，以下面的一维数组为例，若要存取第一、二、三个元素，可以分别写成 $my_array['Taiwan']、$my_array['USA']、$my_array['Japan']：

```
$my_array = array('Taiwan' => '台北', 'USA' => '纽约', 'Japan' => '东京');
```

另外，PHP 还提供了一个 list() 函数用于存取一维数组，以下面的程序代码为例，第 09 行是使用 list() 函数将变量 tour1、tour2 的值分别赋值给一维数组 my_array 内第一、二个元素，换句话说，变量 tour1、tour2 的值分别为 '台北'、'纽约'，故第 10、11 行会在网页上显示 '台北'、'纽约'，如图 3-17 所示。

\ch03\arr1.php

```
01:<!doctype html>
02:<html>
03:   <head>
04:      <metacharset="utf-8">
05:   </head>
06:   <body>
07:      <?php
08:         $my_array = array('台北', '纽约', '东京');
09:         list($tour1, $tour2) = $my_array;
10:         echo $tour1.'<br>';
11:         echo $tour2.'<br>';
12:      ?>
13:   </body>
14:</html>
```

图 3-17

事实上，list() 就像 array() 的反函数，不过，您在 list() 中所指定的变量个数不一定要和数组的元素个数相同，例如第 08 行虽然赋值三个元素给数组，但第 09 行的 list() 却只有指定

的两个变量。

随堂练习

编写一个 PHP 网页，里面有一个名称为 Scores、包含 6 个元素的一维数组，用来存放 6 个学生的分数（分别为 85、60、54、91、100、77），而且程序执行完毕后会在网页上显示最高分与最低分，如图 3-18 所示。

图 3-18

【解答】

\ch03\arr2.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <?php
      $Scores = array(85, 60, 54, 91, 100, 77);
      $MaxScore = 0;
      $MinScore = 100;

      //使用循环找出最高分
      foreach($Scores as $Value)
        if（$Value > $MaxScore) $MaxScore = $Value;

      //使用循环找出最低分
      foreach($Scores as $Value)
        if（$Value < $MinScore) $MinScore = $Value;

      echo "最高分为$MaxScore<br>";
      echo "最低分为$MinScore<br>";
    ?>
```

```
</body>
</html>
```

3.9　多维数组

3.9.1　建立多维数组

1. 直接赋值

建立多维数组最简单的方式就是直接给它的键（key）与值（value）进行赋值，键必须为整数或字符串，值则无此限制。以下面的语句为例，若三维数组 3dim_array 内没有键为 1、2、'name'，就将元素的值 '小丸子' 新增至数组后面并令其键为 1、2、'name'；相反的，若三维数组 3dim_array 内已经有键为 1、2、'name'，就将元素的值 '小丸子' 覆写至数组内键为 1、2、'name' 的位置：

```
$3dim_array[1][2]['name']= '小丸子';
```

使用多维数组时要多留意，例如下面第一个语句是将元素的值 '玫瑰' 存放在二维数组 2dim_array 内键为 0、'red' 的位置，也就是说，键为 0 的位置用来存放另一个包含字符串 '玫瑰' 的数组，而下面第二个语句是将元素的值 100 存放在二维数组 2dim_array 内键为 1 的位置，也就是说，键为 1 的位置用来存放整数 100：

```
$2dim_array[0]['red'] = '玫瑰';
$2dim_array[1] = 100;
```

由于二维数组 2dim_array 内键为 0 的位置是用来存放另一个数组的，所以类似下面的语句企图将它用来存放其他数组是合法的：

```
$2dim_array[0]['white']= '兰花';
```

相反的，由于二维数组 2dim_array 内键为 1 的位置是用来存放整数的，所以类似下面的语句企图将它用来存放另一个数组是不合法的：

```
$2dim_array[1]['white']= '兰花';
```

2. 使用 array() 函数

假设要建立一个如下的二维数组，然后赋值给变量 my_array，可以写成如下的形式：

'玫瑰'	'兰花'	'菊花'	
'苹果'	'白凤'	'香蕉'	'葡萄'

```
$my_array = array(array('玫瑰', '兰花', '菊花'), array('苹果', '白凤', '香蕉', '葡萄'));
```

由于这个语句没有给键赋值，所以预设的键如下：

[0][0]	[0][1]	[0][2]	
[1][0]	[1][1]	[1][2]	[1][3]

若要自行赋值如下的键，可以将这个语句改写成如下的形式：

['flower']['red']	['flower'][white]	['flower'][yellow]	
['fruit']['red']	['fruit']['white']	['fruit']['yellow']	['fruit']['purple']

```
$my_array = array('flower' =>
                    array('red' => '玫瑰', 'white' => '兰花', 'yellow' => '菊花'),
                  'fruit' =>
                    array('red' => '苹果', 'white' => '白凤', 'yellow' => '香蕉', 'purple' => '葡萄'));
```

3.9.2 存取多维数组

存取多维数组最简单的方式就是通过键指定所要存取的元素，以前一节的二维数组 my_array 为例，假设要存取键为 'flower'、'red' 的元素（即'玫瑰'），可以写成 $my_array['flower']['red']；同理，假设要存取键为 'fruit'、'white' 的元素（即 '白凤'），可以写成 $my_array['fruit']['white']。

假设有 8 位学生各自举行三轮比赛，得分如下。

	第 1 轮	第 2 轮	第 3 轮
学生 1	5	7.7	8
学生 2	8.8	5.8	8
学生 3	6	9	8.1
学生 4	7.6	8.5	9.5
学生 5	9	9	9.2
学生 6	4	6.3	7.9
学生 7	8.2	7	9.6
学生 8	9.1	8.5	8.9

编写一个 PHP 网页，令它使用二维数组存放每位学生在每一轮比赛的得分及总得分，然

后在网页上显示每位学生的总得分，如图 3-19 所示。

图 3-19

【解答】

\ch03\arr3.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <?php
      //使用二维数组 Scores 存放每位学生在每一轮比赛的得分及总得分
      $Scores = array(array(5, 7.7, 8, 0),   array(8.8, 5.8, 8, 0),
                      array(6, 9, 8.1, 0),   array(7.6, 8.5, 9.5, 0),
                      array(9, 9, 9.2, 0),   array(4, 6.3, 7.9, 0),
                      array(8.2, 7, 9.6, 0), array(9.1, 8.5, 8.9, 0));
      //计算每位学生的总得分并存放在二维数组
      for($i = 0; $i <= 7; $i++)
      {
        $subTotal = 0;                          //将用来暂存每位学生总得分的变量 subTotal 归零
        for($j = 0; $j <= 2; $j++)              //将每一轮比赛的得分累计暂存在变量 subTotal 中
          $subTotal += $Scores[$i][$j];
        $Scores[$i][3]= $subTotal;              //将累计出来的总得分存放在二维数组
      }
      $Result = '';                             //变量 Result 是要显示在网页上的字符串
      for($i = 0; $i <= 7; $i++)
        $Result = $Result.'第'.($i + 1).'个学生的总得分为'.$Scores[$i][3].'<br>';
      echo $Result;
    ?>
  </body>
</html>
```

这个网页使用了一个 8×4 的二维数组，除了用来存放 8 位学生在 3 轮比赛中的得分之外，还多加了一个字段用来存放各自的总得分，因此，嵌套循环内的“$Scores[$i][3]= $subTotal;”语句就是将第 i + 1 位学生的总得分存放在二维数组的$Scores[$i][3]位置。

3.10 数组运算符

PHP 提供了如表 3-1 所示的数组运算符，其中 + 运算符会将右边数组的元素加入左边数组，遇到相同的键时不做覆写，下面是一个例子。

表 3-1 数组运算符

运算符	语法	说明
+	$a + $b	返回数组 a 和数组 b 的合集
==	$a == $b	若数组 a 和数组 b 有相同的元素，就返回 TRUE，否则返回 FALSE
===	$a === $b	若数组 a 和数组 b 有相同的元素且顺序相同，就返回 TRUE，否则返回 FALSE
!=	$a != $b	若数组 a 和数组 b 不相等，就返回 TRUE，否则返回 FALSE（“相等”指的是相同的键与相同的值）
<>	$a <> $b	若数组 a 和数组 b 不相等，就返回 TRUE，否则返回 FALSE
!==	$a !== $b	若数组 a 和数组 b 不相等，就返回 TRUE，否则返回 FALSE

\ch03\arr4.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <?php
      $a = Array(0 => '台北', 1 => '纽约' );
      $b = Array(0 => '巴黎', 1 => '罗马', 2 => '东京' );
      $c = $a + $b;
      foreach($c as $Value)
        echo $Value.'<br>';
    ?>
  </body>
</html>
```

程序执行结果如图 3-20。

图 3-20

3.11　数组相关的函数

PHP 提供了许多用来处理数组的函数，但由于篇幅有限，无法一一列举，此处仅列出比较实用的部分，如表 3-2 所示，其他的部分请您自行参考 PHP 文件（http://www.php.net/manual/en/）。

表 3-2　处理数组的函数

函数	说明
is_array(*arg*)	若 *arg* 为 array 类型，就返回 TRUE，否则返回 FALSE
count(*arr*)、sizeof(*arr*)	返回数组 *arr* 包含几个元素
in_array(*value*, *arr*)	若 *value* 存在于数组 *arr*，就返回 TRUE，否则返回 FALSE，例如“in_array('Jean', array('Bob', 'Mary', 'Jean'));”会返回 TRUE，“in_array('jean', array('Bob', 'Mary', 'Jean'));”会返回 FALSE，因为有大小写之分
unset(*value*)	清除数组内的元素 *value*，例如“$a = array('Bob', 'Mary', 'Jean', 'Tom');”，则“unset($a[1]);”将使数组 a 剩下 'Bob'、'Jean' 和 'Tom'
current(*arr*)、pos(*arr*)	数组内部有一个指向当前元素的指针，其初始位置为数组的第一个元素，而这两个函数会返回数组 *arr* 内部指针当前所指向的元素，例如“$a = array('Bob', 'Mary', 'Jean');”，则“current($a);”和“pos($a);”会返回 'Bob'
next(*arr*)	将数组 *arr* 内部指针指向下一个元素并返回该元素，例如“$a = array('Bob', 'Mary', 'Jean');”，则“next($a);”会返回 'Mary'
prev(*arr*)	将数组 *arr* 内部指针指向前一个元素并返回该元素，例如“$a = array('Bob', 'Mary', 'Jean');”，则先调用“next($a);”再调用 prev($a) 会返回 'Bob'
end(*arr*)	将数组 *arr* 内部指针指向最后一个元素并返回该元素
reset(*arr*)	将数组 *arr* 内部指针指向第一个元素并返回该元素
array_walk(*arr*, *func* [, *arg*,...])	对数组 *arr* 的每个元素进行 *func* 所指定的函数运算，若 *func* 所指定的函数有参数，可以在“*arg*,...”指定

（续表）

函数	说明
each(*arr*)	返回数组 *arr* 内部指针当前所指向之元素的键与值，然后将内部指针移到下一个元素，通常可以和 list(*arg1*, *arg2*,…) 函数搭配使用，例如“$a = array('Bob', 'Mary', 'Jean');”，而“list($key, $value) = each($a);”会将内部指针当前所指向之元素的键 0 与值 'Bob' 赋值给变量 key、value
list(*arg1*[, *arg2*,…])	将数组的元素赋值给变量 *arg1*、*arg2*……，例如 list($name1, $name2, $name3) = array('Bob', 'Mary', 'Jean', 'John'); 会将数组的第一、二、三个元素 'Bob'、'Mary'、'Jean' 赋值给变量 name1、name2、name3
array_combine(*arr1*, *arr2*)	将数组 *arr1* 的元素当成新数组的键，将数组 *arr2* 的元素当成新数组的值，例如“$a = array('a', 'b', 'c');、$b = array('红', '绿', '蓝');”，则“array_combine($a, $b);”会返回 array('a' => '红', 'b' => '绿', 'c' => '蓝')
array_diff(*arr1, arr2,…*)	返回第一个数组 *arr1* 和其他数组 *arr2* 等的不同元素，例如“$a = array(1, 2, 3, 4, 5);”、“$b = array(1, 2);”、“$c = array(2, 4);”，则“array_diff($a, $b, $c);”会返回 array(3, 5)
array_fill(*key*, *num*, *value*)	在数组内键为 *key* 处填入 *num* 个 *value* 所指定的值，例如 array_fill(2, 4, 'a') 会返回 array(2 => 'a', 3 => 'a', 4 => 'a', 5 => 'a')
array_keys(*arr*[, *value*])	返回数组 *arr* 内的键，或值等于 *value* 的键，例如“$a = array(1 => 'a', 'x' => 'b', 'y' => 'b');”，则“array_keys($a);”会返回 array(1, 'x', 'y')，而“array_keys($a, 'b');”会返回 array('x', 'y')
array_values(*arr*)	返回数组 *arr* 内的值，例如“$a = array(1 => 'a', 'x' => 'b', 'y' => 'b');”，则“array_values($a);”会返回 array('a', 'b', 'b')
array_reverse(*arr*, [*preserve_keys*])	将数组 *arr* 内的元素顺序颠倒过来，若 *preserve_keys* 为 TRUE，表示要保留键的顺序，例如“$a = array('a', 'b', 'c');”，则 array_reverse($a) 会返回 array(0 => 'c', 1 => 'b', 2 => 'a')，而 array_reverse($a, TRUE) 会返回 array(2 => 'c', 1 => 'b', 0 => 'a')
array_flip(*arr*)	将数组 *arr* 内的键与值交换，例如“$a = array(1 => 'a', 'x' => 'b');”，则“array_flip($a);”会返回 array('a' => 1, 'b' => 'x')
array_merge(*arr1* [, *arr2*,…])	将数组进行合并，遇到相同的键时不做覆写，例如“$a = array(1 => 'a', 'x' => 'b');”、“$b = array(2 => 'c', 'x' => 'd');”，则 array_merge($a, $b) 会返回 array(1 => 'a', 'x' => 'b', 2 => 'c')
array_pad(*arr*, *size*, *value*)	将数组 *arr* 的大小设置为 *size*，不足的元素填入 *value*，例如“$a = array('a', 'b', 'c');”，则“array_pad($a, 5, 'x');”会返回 array('a', 'b', 'c', 'x', 'x')
array_search(*value*, *arr*)	若数组 *arr* 内有值为 *value* 的元素，就返回元素的键，否则返回 FALSE，例如“$a = array('a', 'b', 'c');”，则“array_search('c', $a);”会返回 2，而“array_search('C', $a);”会返回 FALSE

（续表）

函数	说明
array_slice(*arr*, *offset* [, *length*])	根据 *offset* 和 *length* 指定的条件从数组 *arr* 内返回一串元素，若 *offset* 为正整数，就从数组的开头位移 *offset* 个元素开始，否则从数组的结尾位移 *offset* 个元素开始，若 *length* 为正整数，就返回 *length* 个元素，否则返回的元素到数组的结尾的 *length* 个元素时停止，例如“$a = array('a', 'b', 'c', 'd', 'e');”，则“array_slice($a, 2);”会返回 array('c', 'd', 'e')，“array_slice($a, 2, -1);”会返回 array('c', 'd')
array_splice(*arr*, *offset* [, *length*[, *replace*]]	根据 *offset* 和 *length* 指定的条件从数组 *arr* 内删除一串元素，若 *offset* 为正整数，就从数组的开头位移 *offset* 个元素开始，否则从数组的结尾位移 *offset* 个元素开始，若 *length* 为正整数，就删除 *length* 个元素，否则删除的元素到数组的结尾的 *length* 个元素时停止，当有指定 *replace* 时，表示以 *replace* 取代被删除的元素，例如“$a = array('a', 'b', 'c', 'd');”，则“array_splice($a, 2);”会返回 array('a', 'b')，“array_splice($a, 1, -1);”会返回 array('a', 'd')，“array_splice($a, -1, 1, array('e', 'f'));”会返回 array('a', 'b', 'c', 'e', 'f')
array_sum(*arr*)	返回数组 *arr* 内各个元素的总和（整数或浮点数），例如 $a = array(1, 2, 3, 4, 5)，则“array_sum($a);”会返回 15
array_unique(*arr*)	删除数组 *arr* 内重复的元素，例如“$a = array('a' => 'green', 'red', 'b' => 'green', 'blue', 'red');”，则“array_unique($a);”会返回 array('a' => 'green', 0 => 'red', 1 => 'blue')
array_push(*arr*, *arg1* [, *arg2*,…])	将 *arg1*[, *arg2*…]等元素加入数组 *arr* 的尾端，例如“$a = array('a', 'b', 'c');”，则“array_push($a, 'd', 'e');”会得到 $a 为 array('a', 'b', 'c', 'd', 'e')
array_pop(*arr*)	从数组 *arr* 的尾端删除一个元素，例如“$a = array('a', 'b', 'c');”，则“array_pop($a);”会得到 $a 为 array('a', 'b')。array_push() 和 array_pop() 通常用来处理堆栈（stack）
array_unshift(*arr*, *arg1* [, *arg2*…])	将 *arg1*[, *arg2*…]等元素加入数组 *arr* 的前端，例如“$a = array('a', 'b', 'c');”，则“array_unshift($a, 'd', 'e');”会得到 $a 为 array('d', 'e', 'a', 'b', 'c')
array_shift(*arr*)	从数组 *arr* 的前端删除一个元素，例如“$a = array('a', 'b', 'c');”，则“array_shift($a);”会得到 $a 为 array('b', 'c')。array_unshift() 和 array_shift() 通常用来处理队列（queue）
range(*arg1*, *arg2*[, *arg3*])	产生一个下限为 *arg1*、上限为 *arg2*、间隔为 *arg3* 的数组，若没有指定 *arg3*，表示间隔为 1，例如“range(1, 5);”会产生“array(1, 2, 3, 4, 5);”
asort(*arr*)	将数组 *arr* 内元素的值进行排序（由小到大），并维持所关联的键，例如“$a = array('c' => 'red', 'a' => 'green', 'b' => 'blue');”，则“asort($a);”会得到 $a 为 array('b' => 'blue', 'a' => 'green', 'c' => 'red')

（续表）

函数	说明
arsort(*arr*)	将数组 *arr* 内元素的值进行反向排序（由大到小），并维持所关联的键，例如“$a = array('c' => 'red', 'a' => 'green', 'b' => 'blue');”，则“arsort($a);”会得到 $a 为 array('c' => 'red', 'a' => 'green', 'b' => 'blue')
ksort(*arr*)	将数组 *arr* 内元素的键进行排序（由小到大），例如“$a = array('d' => 'red', 'a' => 'green', 'b' => 'blue', 'c' => 'yellow');”，则“ksort($a);”会得到 $a 为 array('a' => 'green', 'b' => 'blue', 'c' => 'yellow', 'd' => 'red')
krsort(*arr*)	将数组 *arr* 内元素的键进行反向排序（由大到小），例如“$a = array('d' => 'red', 'a' => 'green', 'b' => 'blue', 'c' => 'yellow');”，则“krsort($a);”会得到 $a 为 array('d' => 'red', 'c' => 'yellow', 'b' => 'blue', 'a' => 'green')
sort(*arr*[, *flag*])	将数组 *arr* 内元素的值进行排序（由小到大），参数有 *flag* 三种值，即 SORT_REGULAR（正常比较）、SORT_NUMERIC（数值比较）、SORT_STRING（字符串比较），例如“$a = array(100, 85.2, 77, 93, 60);”，则“sort($a, SORT_NUMERIC);”会得到 $a 为 array(60, 77, 85.2, 93, 100)
rsort(*arr*[, *flag*])	将数组 *arr* 内元素的值进行反向排序（由小到大），参数 *flag* 有三种值，即 SORT_REGULAR（正常比较）、SORT_NUMERIC（数值比较）、SORT_STRING（字符串比较），例如“$a = array(100, 85.2, 77, 93, 60);”，则“rsort($a, SORT_NUMERIC);”会得到 $a 为 array(100, 93, 85.2, 77, 60)
usort(*arr*, *func*)	将数组 *arr* 内元素的值根据 *func* 所指定的函数进行排序
uasort(*arr*, *func*)	将数组 *arr* 内元素的值根据 *func* 所指定的函数进行排序，并维持所关联的键
uksort(*arr*, *func*)	将数组 *arr* 内元素的键根据 *func* 所指定的函数进行排序
shuffle(*arr*)	将数组 *arr* 内元素的顺序弄乱，每次调用都会有不同的结果

这个随堂练习的目的是要示范 usort() 函数的应用，首先，程序第 08 ~ 11 行定义了一个 compare() 函数，调用 strcmp() 函数对数组元素进行字符串比较；接着，第 13 ~ 15 行定义了一个名称为 colors 的二维数组；第 17 行调用 usort() 函数，将二维数组 colors 内元素的值根据 compare() 函数进行排序；最后，第 19 ~ 20 行使用 while 循环在网页上显示二维数组各个元素的键与值。

\ch03\usort.php

```
01:<!doctype html>
02:<html>
03:   <head>
```

```
04:     <meta charset="utf-8">
05:   </head>
06:   <body>
07:     <?php
08:         function compare($a, $b)
09:         {
10:             return strcmp($a['color'], $b['color']);
11:         }
12:
13:         $colors[0]['color']= 'red';
14:         $colors[1]['color']= 'green';
15:         $colors[2]['color']= 'blue';
16:
17:         usort($colors, 'compare');
18:
19:         while（list($key, $value) = each($colors))
20:             echo "\$colors[$key]: ".$value['color'].'<br>';
21:     ?>
22:   </body>
23:</html>
```

程序执行结果见图 3-21。

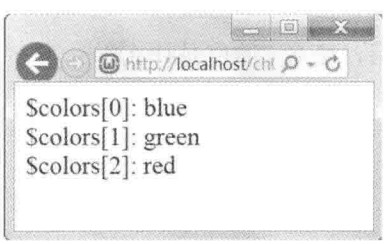


图 3-21

一、选择题

（　）1. 下列哪种流程控制可以根据变量的值而有不同的执行方向？

A. while　　B. do　　C. switch　　D. foreach

（　）2. 下列哪种流程控制最适合用来计算连续数字的累加？

A. if...else　　B. switch　　C. for　　D. foreach

（　）3. 在 for($i = 100; $i <= 200; $i += 3) 循环执行完毕时，变量 i 的值是什么？

A. 200　　B. 202　　C. 199　　D. 201

（ ）4. 假设“$i= 0; while($i < 100) $i++;”，那么在离开循环后，变量 i 的值是什么？

A. 99　　B. 100　　C. 101　　D. 0

（ ）5. 假设“$a = array(0 => 'a', 1 => 'b');”、“$b = array(1 => 'b', 0 => 'a');”，则$a === $b 会返回下列哪一项？

A. TRUE　　B. FALSE

（ ）6. “in_array('a', array('A', 'B', 'C'));”会返回下列哪一项？

A. TRUE　　B. FALSE

（ ）7. 假设“$a = array('a', 'b', 'c', 'd');”，则依次调用“next($a);”、“next($a);”、“next($a);”、“prev($a);”后，会返回下列哪一项？

A. 'a'　　B. 'b'　　C. 'c'　　D. 'd'

（ ）8. 假设“list($X, $Y) = array(100, 999, 50, 1);”，则 $X 的值为下列哪一项？

A. 100　　B. 999　　C. 50　　D. 1

（ ）9. 下列哪个函数可以将数组内元素的顺序颠倒过来？

A. array_reverse()　　B. array_walk()　　C. array_flip()　　D. array_pad()

（ ）10.假设“$a = array(0 => 5, 1 => 10);”、“$b = array(1 => 15, 2 => 20);”、“$c = $a + $b;”，则 $c 等于下列哪一项？

A. array(0 => 5, 1 => 10, 2 => 20)　　B. array(0 => 5, 1 => 15, 2 => 20)

C. array(1 => 10, 2 => 20)　　D. array(1 => 15, 2 => 20)

二、实验题

1. 下列程序代码在跳出循环后，变量 i 的值是什么？

(a)

```
$i = 0;
do
{
   $i += 7;
}while($i < 100);
```

(b)

```
$i = 500;
do
{
    $i -= 11;
}while($i > 0);
```

(c)

```
$i = 20;
while($i < 650)
{
    $i += 9;
}
```

2. 编写一个 PHP 网页，令它计算如下公式的结果并显示出来。

$$(1/2)^1 + (1/2)^2 + (1/2)^3 + (1/2)^4 + (1/2)^5 + (1/2)^6 + (1/2)^7 + (1/2)^8$$

3. 编写一个 PHP 网页，令它计算 4096 是 2 的几次方并显示出来。

4. 编写一个 PHP 网页，令它根据数字 1 ~ 12，在网页上显示对应的英文月份简写，例如 Jan.、Feb.、Mar.、Apr.、May.、Jun.、Jul.、Aug.、Sep.、Oct.、Nov.、Dec.。

5. 编写一个 PHP 网页，令它声明一个数组（5, 8, 2, 3, 7, 6, 9, 1, 4, 8, 3, 0），然后计算这些元素的平均值并显示出来。

第 4 章
函数

4.1　认识函数

函数（function）将一段具有某种功能的语句写成独立的程序单元，然后给予特定名称，以提高程序的重复使用性及可读性。有些程序设计语言把函数称为方法（method）、过程（procedure）或子程序（subroutine）。

使用函数的好处如下。

- 函数具有重复使用性（reusability），当您编写好一个函数时，可以在程序中的不同地方调用这个函数，而不必重新编写。
- 加上函数后，程序会变得更精简，因为虽然多了调用函数的语句，却少了更多重复的语句。
- 加上函数后，程序的可读性（readability）会提高。
- 把程序拆成几个函数后，写起来会比较轻松，而且程序的逻辑性和正确性都会提高，这样不仅容易理解，也比较好排错、修改与维护。

说了这么多好处，那么函数没有缺点吗？事实上是有的，函数会使程序的执行速度减慢，因为多了一道调用的手续，执行速度自然比直接将语句写进程序里面慢一点。

调用函数流程图见图 4-1。

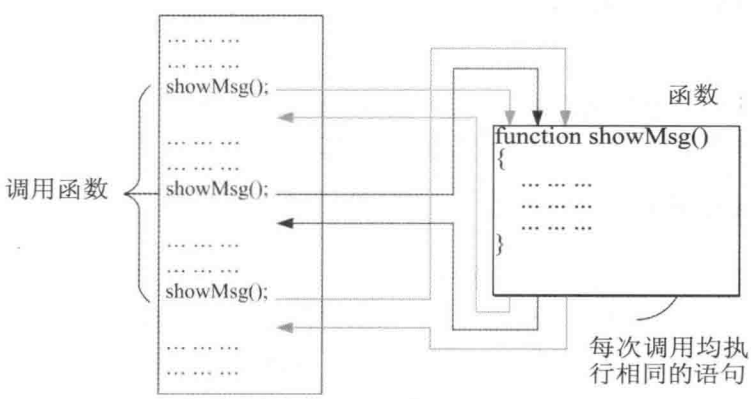


图 4-1

4.2　自定义函数

我们可以使用 function 关键字定义函数，其语法如下：

```
function function_name([argumentlist])
{
  statements;
  [return;|return value;]
  [statements;]
```

```
}
```

- function：这个关键字用来表示要定义函数。
- *function_name*：这是函数的名称，命名规则与变量相同，即第一个字符可以是英文字母或下划线，其他字符可以是英文字母、下划线或阿拉伯数字，不同的是函数的名称没有大小写之分，PHP 会将函数的名称存储为小写。
- {、}：分别标记函数的开头与结尾。
- ([*argumentlist*])：这是函数的参数（argument），我们可以利用参数传递数据给函数，一个以上的参数中间以逗号隔开。
- *statements*：这是函数主要的程序代码部分。
- [return;|return *value*;]：若要将程序的控制权从函数内移转到调用函数处，可以使用 return 语句。当函数没有返回值且不需要提早移转到调用函数处时，return 语句可以省略不写；相反的，当函数有返回值时，return 语句不可以省略不写，而且后面必须加上返回值 *value*。

例如下面的语句用于定义一个名称为 Greeting、没有参数也没有返回值的函数：

```
function Greeting()
{
  echo "欢迎光临 PHP 的世界!";
}
```

请注意，函数必须加以调用才会执行，而且当函数有参数时，参数的个数及顺序不能弄错，即便函数没有参数，小括号（）仍须保留，其语法如下：

```
function_name([argumentlist]);
```

下面是一个例子，程序第 08 ~ 11 行用于定义一个名称为 Greeting、没有参数也没有返回值的函数，第 13 行是调用函数，如此一来，当浏览者载入网页时，就会在网页上显示“欢迎光临 PHP 的世界!”如图 4-2，若第 13 行省略不写，函数将不会执行。

\ch04\func1.php

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:  </head>
06:  <body>
07:    <?php
08:      function Greeting()
09:      {
10:        echo "欢迎光临 PHP 的世界!";
```

```
11:        }
12:
13:        Greeting();
14:     ?>
15:  </body>
16:</html>
```

图 4-2

PHP 3 规定函数定义一定要放在函数调用的前面，但是 PHP 4 以后的版本则无此限制，换句话说，我们也可以将程序第 13 行的函数调用移到第 08 行的函数定义前面，例外的是“条件函数”与“函数中的函数”两种情况。

❖ **条件函数**（conditional function）

若函数定义放在条件中，那么须在条件成立后，才能调用该函数，例如：

```
<?php
  $status = TRUE;
  //此处尚不能调用函数 Greeting()
  if ($status)
  {
     function Greeting()
     {
        echo "欢迎光临 PHP 的世界!";
     }
  }
  //此处之后才能调用函数 Greeting()
  Greeting();
?>
```

❖ **函数中的函数**（function within function）

若函数定义放在其他函数中，那么须在调用其他函数后，才能调用该函数，例如：

```
<?php
  function MyFunction()
  {
    function Greeting()
    {
      echo "欢迎光临 PHP 的世界!";
    }
  }
  //此处尚不能调用函数 Greeting()
  MyFunction();
  //此处之后才能调用函数 Greeting()
  Greeting();
?>
```

4.3 函数的参数

我们可以通过参数传递数据给函数，当参数的个数不只一个时，中间以逗号隔开，而在调用有参数的函数时，要注意参数的个数及顺序不能弄错。PHP 支持的参数传递方式有“传值调用”和“传址调用”两种，以下做进一步的说明。

4.3.1 传值调用

PHP 预设的参数传递方式为“传值调用”（call by value），函数无法改变参数的值，因为 PHP 传递给函数的是参数的值，而不是参数的地址，如此一来，无论函数内的语句如何改变传递进来的参数的值，都不会影响到原来调用函数处的那个参数的值，下面是一个例子，运行结果如图 4-3。

\ch04\func2.php

```
01:<!doctype html>
02:<html>
03:   <head><meta charset="utf-8"></head>
04:   <body>
05:     <?php
06:       $num1 = 1;
07:       $num2 = 100;
08:       function swap($n1, $n2)
09:       {
10:         $temp = $n1;
11:         $n1 = $n2;
12:         $n2 = $temp;
13:         echo '$n1 的值为'.$n1.'<br>';
```

```
14:          echo '$n2 的值为'.$n2.'<br>';
15:      }
16:      swap($num1, $num2);
17:      echo '$num1 的值为'.$num1.'<br>';
18:      echo '$num2 的值为'.$num2.'<br>';
19:   ?>
20: </body>
21:</html>
```

- 08 ~ 15：定义一个名称为 swap、有两个参数、没有返回值的函数，其用途是将两个参数的值交换。
- 16：调用函数，同时将 num1 和 num2 的值（1、100）当成参数传递给函数，于是 n1 和 n2 的值一开始为 1、100，经过交换后，成为 100、1，于是在网页上显示“$n1 的值为 100”、“$n2 的值为 1”。
- 17、18：由于函数采用传值调用，num1、num2 的值并不会受到影响，所以会在网页上显示“$num1 的值为 1”、“$num2 的值为 100”。

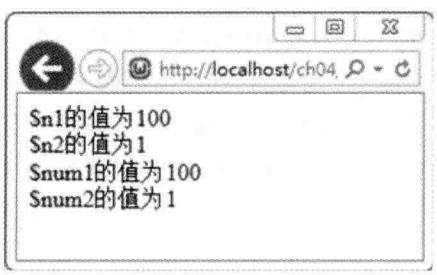


图 4-3

4.3.2　传址调用

PHP 也支持另一种常见的参数传递方式，称为“传址调用”（call by reference），函数能够改变参数的值，因为 PHP 传递给函数的是参数的地址，而不是参数的值，如此一来，只要函数内的语句改变参数的值，原来调用函数处的那个参数的值也会随着改变，因为它们指向同一个地址。

举例来说，假设要将 <\ch04\func2.php> 的函数改为传址调用，那么只要将第 08 行改写成如下代码即可，也就是在所定义的两个参数 n1、n2 前面加上 & 符号，运行结果如图 4-4 所示：

```
function swap(&$n1, &$n2)
```

图 4-4

很明显，网页上所显示的字符串和之前采用传值调用时不同，在更改为传址调用后，num1 和 n1 是指向同一个地址的变量（即同一个变量），而 num2 和 n2 也是指向同一个地址的变量（即同一个变量），因此，一旦 n1 和 n2 的值交换成为 100、1，num1 和 num2 的值也会随着交换成为 100、1。

4.3.3 设置参数的默认值

我们可以在定义函数的同时设置参数的默认值，如此一来，当函数调用没有提供参数的值时，就会自动采用默认值，要注意的是拥有默认值的参数必须放在没有默认值的参数后面。

以下面的程序代码为例，由于程序第 08 行将参数的默认值设置为 '茶'，所以第 12 行虽然没有提供参数的值，但仍会在网页上显示“请给我一杯茶”，至于第 13 行则因为提供了参数的值为 '咖啡'，所以会在网页上显示“请给我一杯咖啡”，运行结果如图 4-5 所示。

\ch04\func3.php

```
01:<!doctype html>
02:<html>
03:   <head>
04:      <meta charset="utf-8">
05:   </head>
06:   <body>
07:      <?php
08:         function drink($kind = '茶')
09:         {
10:            echo '请给我一杯'.$kind.'<br>';
11:         }
12:         drink();
13:         drink('咖啡');
14:      ?>
15:   </body>
16:</html>
```

图 4-5

4.3.4　可变长参数列表

PHP 提供了所谓的“可变长参数列表”（variable-length argument list），也就是函数没有一定的参数个数，为了处理这种函数，我们通常得借助于下列函数：

- func_num_args(): 返回函数的参数个数。
- func_get_arg(*n*): 返回函数的第 *n* + 1 个参数。
- func_get_args(): 返回一个数组，里面包含函数的所有参数，起始键为 0。

以下面的程序代码为例，由于我们并不确定函数的参数个数，所以在程序第 10 行调用 func_num_args() 函数获取参数个数，然后检查看看是否等于 0，若等于 0，就在网页上显示“没有指定地点！”（第 11 行），否则使用 for 循环将参数的值一一显示在网页上（第 13 行），程序运行结果如图 4-6 所示。

\ch04\func4.php

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:  </head>
06:  <body>
07:    <?php
08:      function tour()
09:      {
10:        if（func_num_args() == 0)
11:          echo '没有指定地点！ ';
12:        else
13:          for($i = 0; $i < func_num_args(); $i++) echo func_get_arg($i).'<br>';
14:      }
15:      tour('台北', '台中', '高雄');
16:    ?>
```

```
17:  </body>
18:</html>
```

图 4-6

4.4 函数的返回值

当我们希望从函数返回某个值时，可以使用 return 关键字，以下面的程序代码为例，第 08 行的“return $DegreeC * 1.8 + 32;”语句会将摄氏温度转换为华氏温度并返回，因此，我们可以在第 11 行调用函数，然后将返回值显示出来，运行结果如图 4-7 所示。

\ch04\func5.php

```
01:<!doctype html>
02:<html>
03:  <head><meta charset="utf-8"></head>
04:  <body>
05:    <?php
06:      function Convert2F($DegreeC)
07:      {
08:        return $DegreeC * 1.8 + 32;
09:      }
10:
11:      echo '摄氏 30 度可以转换为华氏'.Convert2F(30).'度';
12:    ?>
13:  </body>
14:</html>
```

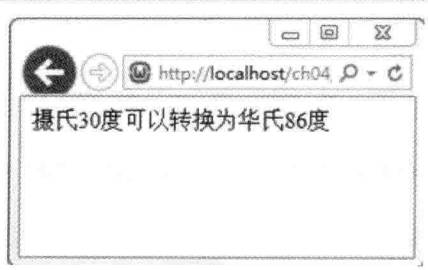


图 4-7

- 函数通常会按顺序执行到大括号(})所标识的结束处，但如有需要，也可以使用 return 关键字提前离开函数，返回原来调用函数的地方。
- 若函数内省略了 return 关键字，将会传回 NULL。
- 使用函数的好处是可以让程序更清楚明了，若您的程序中有很多重复的程序代码，所谓“重复”指的是程序代码之间的差异仅仅是变量有所改变，其他计算过程的程序代码均相同，就可以将这些程序代码编写成函数，调用函数时再将变量当作参数传入。

4.5　局部变量 V.S.全局变量

本节所要讨论的是一个重要的概念，就是变量的有效范围，“有效范围”（scope）指的是程序的哪些片段能够存取变量的值，大部分的 PHP 变量都只有一种有效范围，那就是程序的所有片段均能存取变量的值，这种变量称为“全局变量”（global variable）。例外的是，在函数内定义的变量，被称为“局部变量”（local variable），只有函数内的语句能够存取局部变量的值，下面是一个例子。

\ch04\func8.php

```
01:<!doctype html>
02:<html>
03:   <head><meta charset="utf-8"></head>
04:   <body>
05:      <?php
06:         $Msg = "Hello, This is outside of Func1().";          //设置全局变量 Msg 的值
07:         echo $Msg.'<br>';                                    //显示全局变量 Msg 的值
08:         Func1();                                             //调用 Func1() 函数
09:         echo $Msg.'<br>';                                    //显示全局变量 Msg 的值
10:
11:         function Func1()
12:         {
13:            $Msg = "Hello, This is inside of Func1().";       //设置同名局部变量 Msg 的值
14:            echo $Msg.'<br>';                                 //显示局部变量 Msg 的值
15:         }
16:      ?>
17:   </body>
18:</html>
```

网页上的三行字符串分别是由程序第 07、08、09 行所显示，如图 4-8 所示，其中第 07、09 行会显示全局变量 Msg 的值为 " Hello, This is outside of Func1().", 而第 08 行因为调用

Func1() 函数，故会显示同名局部变量 Msg 的值为 " Hello, This is inside of Func1()."。

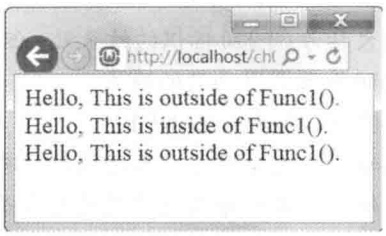


图 4-8

由此可知，即便使用同名局部变量，PHP 一样能够正确区分；相反的，若我们在函数内使用 global 关键字将 Msg 定义为全局变量，结果会变成什么样呢？看看下面的例子就知道了。

\ch04\func9.php

```
01:<!doctype html>
02:<html>
03:   <head><meta charset="utf-8"></head>
04:   <body>
05:     <?php
06:       $Msg = "Hello, This is outside of Func1().";          //设置全局变量 Msg 的值
07:       echo $Msg.'<br>';                                     //显示全局变量 Msg 的值
08:       Func1();                                              //调用 Func1() 函数
09:       echo $Msg.'<br>';                                     //显示全局变量 Msg 的值
10:
11:       function Func1()
12:       {
13:         global $Msg;                                        //使用 global 将 Msg 定义为全局变量
14:         $Msg = "Hello, This is inside of Func1().";         //设置全局变量 Msg 的值
15:         echo $Msg.'<br>';                                   //显示全局变量 Msg 的值
16:       }
17:     ?>
18:   </body>
19:</html>
```

网页上的三行字符串分别是由第 07、08、09 行所显示，如图 4-9 所示，其中第 06 行将全局变量 Msg 的值设置为 "Hello, This is outside of Func1().", 故第 07 行会显示全局变量 Msg 的值为 "Hello, This is outside of Func1()."；接着，第 08 行调用 Func1() 函数，此时，第 13 行使用 global 关键字将 Msg 定义为全局变量，而第 14 行将全局变量 Msg 的值改为 "Hello, This is inside of Func1().", 故第 08、09 行会显示全局变量 Msg 的值为 "Hello, This is inside of Func1()."。

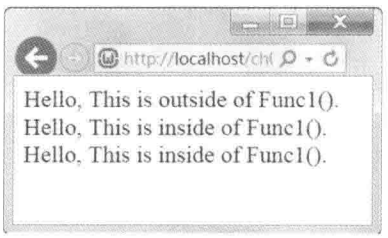


图 4-9

4.6　静态变量

对于函数内的局部变量来说，当我们调用函数时，局部变量会被创建，而在函数执行完毕后，局部变量就会被释放，换句话说，局部变量的值并不会被保留下来。

以下面的程序代码为例，它会在网页上连续显示两个 1，第一个 1 是第一次调用 Add() 函数的结果，而第二个 1 是第二次调用 Add() 函数的结果。在第一次调用 Add() 函数时，局部变量 Result 的初始值为 0，加 1 后变成 1，于是在网页上显示 1，待函数执行完毕后，局部变量 Result 的值会被释放而不会保留下来；接着又再度调用 Add() 函数，此时，局部变量 Result 的初始值仍为 0，加 1 后还是会得到 1，于是在网页上依然会显示 1，如图 4-10 所示。

\ch04\func10.php

```
<!doctype html>
<html>
  <head><meta charset="utf-8"></head>
  <body>
    <?php
      function Add()
      {
        $Result = 0;                    //局部变量 Result 的初始值为 0
        $Result++;                      //将局部变量 Result 的值递增 1
        echo $Result.'<br>';            //在网页上显示局部变量 Result 的值
      }
      Add();                            //调用函数
      Add();                            //调用函数
    ?>
  </body>
</html>
```

图 4-10

若要保留函数内局部变量的值，可以使用 static 关键词将它定义为“静态变量”（static variable），下面是一个例子。

\ch04\func11.php

```
<!doctype html>
<html>
  <head><meta charset="utf-8"></head>
  <body>
    <?php
      function Add()
      {
        static $Result = 0;                    //使用 static 关键词将 Result 定义为静态变量
        $Result++;                             //将静态变量 Result 的值递增 1
        echo $Result.'<br>';                   //在网页上显示静态变量 Result 的值
      }
      Add();                                   //调用函数
      Add();                                   //调用函数
    ?>
  </body>
</html>
```

这段程序代码会在网页上依次显示 1 和 2，如图 4-11 所示。1 是第一次调用 Add() 函数的结果，而 2 是第二次调用 Add() 函数的结果。在第一次调用 Add() 函数时，静态变量 Result 的初始值为 0，加 1 后变成 1，于是在网页上显示 1，由于 Result 是一个静态变量，所以在函数执行完毕后，Result 的值会被保留下来而不会被释放；接着又再度调用 Add() 函数，此时，静态变量 Result 的值为 1，加 1 后会得到 2，于是在网页上显示 2。同理，若我们第三次调用 Add() 函数，网页上会显示多少呢？请您动动脑筋吧！正确的答案是 3。

图 4-11

4.7　匿名函数

PHP 从 5.3.0 版开始支持匿名函数（anonymous function）功能，这项功能允许程序设计人员在没有指定名称的情况下创建函数，下面是一个例子，程序运行结果如图 4-12 所示。

\ch04\func11a.php

```
01:<!doctype html>
02:<html>
03:   <head>
04:      <meta charset="utf-8">
05:   </head>
06:   <body>
07:      <?php
08:         $greet = function($name)
09:         {
10:            printf("Hello %s\r\n", $name);
11:         };
12:
13:         $greet("World!");
14:         $greet("PHP!");
15:      ?>
16:   </body>
17:</html>
```

- 08 ~ 11: 把一个匿名函数指派给变量 greet，该函数会在网页上显示 Hello 和参数 name 的值。
- 13: 调用指派给变量 greet 的匿名函数，且参数 name 的值为 "World!"，所以会在网页上显示 "Hello World!"。
- 14: 调用指派给变量 greet 的匿名函数，且参数 name 的值为 "PHP!"，所以会在网页上显示 "Hello PHP!"。

图 4-12

4.8　可变函数

可变函数（variable function）指的是我们可以动态设置函数的名称，也就是说，当某个

变量的名称后面存在着小括号（），PHP 就会试着找出这个变量所代表的值，然后去执行和该值同名的函数。

以下面的程序代码为例，第 15 行使用了可变函数，由于当时变量 func 的值为'CircleArea'，所以会执行函数调用“CircleArea(10);”；同理，第 17 行使用了可变函数，由于当时变量 func 的值为 'SquareArea'，所以会调用函数“SquareArea(10);”，程序运行结果如图 4-13 所示。

\ch04\func12.php

```
01:<!doctype html>
02:<html>
03:   <head><meta charset="utf-8"></head>
04:   <body>
05:      <?php
06:         function CircleArea($R)
07:         {
08:            echo "半径为 $R 的圆面积为".($R * $R * 3.1416)."<br>";
09:         }
10:         function SquareArea($L)
11:         {
12:            echo "长度为 $L 的正方形面积为".($L * $L)."<br>";
13:         }
14:         $func = 'CircleArea';
15:         $func(10);                          //会调用函数 CircleArea(10);
16:         $func = 'SquareArea';
17:         $func(10);                          //会调用函数 SquareArea(10);
18:      ?>
19:   </body>
20:</html>
```

图 4-13

4.9 实用的 PHP 内部函数

PHP 内置了许多实用的核心函数，包括第 2 章所介绍的类型相关的函数、输出函数、第 3 章所介绍的数组相关的函数，以及本节将介绍的数字函数、日期时间函数、字符串函数等。

除了核心函数之外，还有一些函数会视 PHP 加载哪些 Extensions（扩展部分）而定，例如 PHP 必须加载 MySQL support，才能调用 mysqli_connect() 函数建立数据库连接。至于 PHP 加载哪些 Extensions，只要调用 phpinfo() 函数就知道了。

4.9.1　数字常数

PHP 中的数字常数如下所示。

常数	值	说明
M_PI	3.14159265358979323846	圆周率 π
M_E	2.7182818284590452354	自然对数的底数 e
M_LOG2E	1.4426950408889634074	$\log_2 e$
M_LOG10E	0.43429448190325182765	$\log_{10} e$
M_LN2	0.69314718055994530942	$\log_e 2$
M_LN10	2.30258509299404568402	$\log_e 10$
M_PI_2	1.57079632679489661923	π / 2
M_PI_4	0.78539816339744830962	π / 4
M_1_PI	0.31830988618379067154	1 / π
M_2_PI	0.63661977236758134308	2 / π
M_SQRTPI	1.77245385090551602729	sqrt(π)，即圆周率 π 的平方根
M_2_SQRTPI	1.12837916709551257390	2 / sqrt(π)
M_SQRT2	1.41421356237309504880	sqrt(2)，即 2 的平方根
M_SQRT3	1.73205080756887729352	sqrt(3)，即 3 的平方根
M_SQRT1_2	0.70710678118654752440	1 / sqrt(2)
M_LNPI	1.14472988584940017414	$\log_e(\pi)$
M_EULER	0.57721566490153286061	欧拉常数

4.9.2　数字函数

PHP 中的数字函数如下所示。

函数	说明
取绝对值 abs(x)	返回参数 x 的绝对值，例如 abs(-123.456) 会返回 123.456
三角函数 cos(x)、sin(x)、 tan(x)、acos(x)、 asin(x)、atan(x)	分别返回参数 x 的余弦值（cosine）、正弦值（sine）、正切值（tangent）、反余弦值（arccosine）、反正弦值（arcsine）、反正切值（arctangent）。请注意，参数 x 必须为弧度，而不是角度，换句话说，若要计算 Sin30° 的值，必须先根据公式“弧度 = 角度 × π ÷180”，将角度转换成弧度，故 Sin30° 的值为 Sin(30 * M_PI / 180)，将返回 0.5

（续表）

函数	说明
角度转换成弧度 deg2rad(*x*)	根据公式“弧度 = 角度 × π ÷180”，将参数 *x* 由角度转换成弧度，例如 deg2rad(45) 会返回 0.785398163397(M_PI_4)
弧度转换成角度 rad2deg(*x*)	根据公式“角度 = 弧度 × 180÷π”，将参数 *x* 由弧度转换成角度，例如 rad2deg(M_PI_4) 会返回 45
进制转换 base_convert(*x*, *frombase*, *tobase*)	将参数 *x* 由参数 *frombase* 所指定的进位系统转换成由参数 *tobase* 所指定的进位系统，例如 base_convert(256, 10, 16) 会将十进制数 256 转换成十六进制数 100，然后返回 100
bindec(*x*)	将参数 *x* 由二进制转换为十进制，例如 bindec(111) 会返回 7
decbin(*x*)	将参数 *x* 由十进制转换为二进制，例如 decbin(7) 会返回 111
decoct(*x*)	将参数 *x* 由十进制转换为八进制，例如 decoct(64) 会返回 100
dechex(*x*)	将参数 *x* 由十进制转换为十六进制，例如 dechex(10) 会返回 A
hexedc(*x*)	将参数 *x* 由十六进制转换为十进制，例如 hexdec(F) 会返回 15
octdec(*x*)	将参数 *x* 由八进制转换为十进制，例如 octdec(100) 会返回 64
ceil(*x*)	返回比参数 *x* 大 1 的整数，例如 ceil(4.3) 会返回 5，ceil(9.999) 会返回 10，ceil(-4.3) 会返回 -4，ceil(-9.999) 会返回 -9
floor(*x*)	返回比参数 *x* 小 1 的整数，例如 floor(4.3) 会返回 4，floor(9.999) 会返回 9，floor(-4.3) 会返回 -5
取四舍五入值 round(*x*[, *precision*])	返回与参数 *x* 最接近的整数（即四舍五入），若要指定四舍五入至哪个小数字数，可以加上参数 *precision*，例如 round(3.4) 会返回 3，round(3.5) 会返回 4
取浮点余数 fmod(*x1* , *x2*)	返回参数 *x1* 除以参数 *x2* 的余数，和余数运算符 % 不同的是可以套用于浮点数，例如 fmod(5.7, 1.3) 会返回 0.5，因为 5.7 = 4 * 1.3 + 0.5
取自然对数之底数的某次方值 exp(*x*)	返回自然对数之底数 e 的 *x* 次方值，参数 *x* 代表的是几次方，e 的值约为 2.71828182846，例如 exp(2) 会返回 e 的平方值 7.38905609893
判断是否有限 is_finite(*x*)	若参数 *x* 的值有限，就返回 TRUE，否则返回 FALSE，例如 is_finite(5 / 2) 会返回 TRUE，is_finite(log(0)) 会返回 FALSE
判断是否无限 is_infinite(*x*)	若参数 *x* 的值无限，就返回 TRUE，否则返回 FALSE，例如 is_infinite(5 / 2) 会返回 FALSE，is_infinite(log(0)) 会返回 TRUE
判断是否非数字 is_nan(*x*)	若参数 *x* 的值不是数字（NaN，Not a Number），就返回 TRUE，否则返回 FALSE，例如 is_nan(5 / 2) 会返回 FALSE，is_nan(log(0)) 会返回 TRUE
取对数值 log(*x*)	返回参数 *x* 的自然对数值，例如 log(2) 会返回 0.69314718056，log(exp(2)) 会返回 2。请注意，自然对数是以 e 为底数的对数，e 的值约为 2.71828182846，若要以任意底数 *n* 来计算数值 *x* 的对数值，可以使用公式 $\log_n(x) = \log(x) / \log(n)$，将 *x* 的自然对数值除以 *n* 的自然对数值
以 10 为底数取对数值 $\log_{10}(x)$	返回参数 *x* 的对数值，而且底数为 10，例如 $\log_{10}(10)$ 会返回 1，$\log_{10}(100)$ 会返回 2

（续表）

函数	说明
取次方值 pow(*base*, *exp*)	返回 *base* 的 *exp* 次方值，例如 pow(10, 5) 会返回 10 的 5 次方值为 100000，pow(2, 8) 会返回 2 的 8 次方值为 256
取平方根 sqrt(*x*)	返回参数 *x* 的平方根，例如 sqrt(9) 会返回 9 的平方根为 3
取最大值 max(*x1*, *x2*[, *x3*...]) max(*arr1*[, *arr2*...])	返回参数中的最大值，若参数为字符串，就会被视为 0，例如 max(1, 3, 5, 6, 7) 会返回 7，max(array(2, 4, 5)) 会返回 5，max(0, 'hello') 会返回 0，max(-1, 'hello') 会返回 'hello'
取最小值 min(*x1*, *x2*[, *x3*...]) min(*arr1*[, *arr2*...])	返回参数中的最小值，若参数为字符串，就会被视为 0，例如 min(1, 3, 5, 6, 7) 会返回 1，min(array(2, 4, 5)) 会返回 2，min(0, 'hello') 会返回 0，min(-1, 'hello') 会返回 -1
产生随机数 rand([*min*, *max*])	返回一个大于等于参数 *min*、小于等于参数 *max* 的随机数，若没有提供参数 *min*、*max*，就会返回 0 到 RAND_MAX 之间的随机数，RAND_MAX 的值视平台而定，32 位 Windows 环境下的 RAND_MAX 为 32768
随机数最大值 getrandmax()	返回随机数可能的最大值，例如在 32-bit Windows 环境下调用 getrandmax()，将会返回 32767
圆周率 π　pi()	返回圆周率 π 的值（3.1415926535898）

4.9.3　日期时间函数

PHP 记录日期时间的方式是以 UNIX 时间戳（timestamp）作为运算的依据，所谓“时间戳”是从 1970/1/1 到指定日期或指定时间所经过的秒数，例如 2004/10/3 20:50:32 的时间戳为 1096807832。当我们处理日期时间时，都必须先获取时间戳，再使用格式化函数将它转换成我们熟悉的表示方式。

❖　获取日期时间函数

函数	说明
getdate([int *timestamp*])	将参数 *timestamp* 指定的时间戳转换成日期时间格式，然后返回，若参数 *timestamp* 省略不写，就返回系统目前的日期时间信息，返回值为一个数组，包含下列元素： seconds：以数字格式记录的秒数。 minutes：以数字格式记录的分钟数。 hours：以数字格式记录的小时数。 mday：以数字格式记录的日期。 wday：以数字格式记录的星期几，0 代表星期日，1 代表星期一，依次类推。 mon：以数字格式记录的月份。

（续表）

函数	说明
getdate([int *timestamp*])	year：以数字格式记录的公元年份。 yday：记录该时间戳为一年的第几天。 weekday：以英文名称记录的星期几。 month：以英文名称记录的月份。 0：返回时间戳。 举例来说，假设目前的日期时间为 2015/1/20 15:30，“$Today = getdate();”，则 $Today 为一个数组，记录着日期时间信息，包含上述 11 个元素，$Today["year"]为 2015，$Today["yday"]为 20，$Today["weekday"]为 Tuesday，$Today["month"]为 January，依次类推
time()	获取目前的时间信息，返回值为 UNIX 的时间戳，举例来说，假设目前的时间为 2004/10/3 20:50:32，则 time() 函数会返回时间戳 1096807832
mktime([int *hour*[, int minute[, int *second*[, int *month*[, int *day*[, int *year*[, int *is_dst*]]]]]])	根据参数指定的日期时间建立一个 UNIX 时间戳，其中 *year* 为年，*month* 为月，*day* 为日，*hour* 为小时，*minute* 为分钟，*second* 为秒，*is_dst* 用来设置该日期是否为夏时制（daylight savings），若是就设置为 1，否则设置为 0，若不确定，可以省略或输入 -1。举例来说，mktime(20, 50, 32, 10, 3, 2004, -1) 表示要为 2004/10/3 20:50:32 建立一个 UNIX 时间戳，返回值为 1096807832
checkdate(int *month*, int *day*, int *year*)	判断参数指定的日期是否为有效日期，若是就返回 TRUE，否则返回 FALSE，其中 *year* 为年，*month* 为月，*day* 为日。举例来说，checkdate(10, 3, 2015) 会返回 TRUE，因为它代表 2015/10/3（存在），而 checkdate(13, 3, 2015) 会返回 FALSE，因为它代表 2015/13/3（不存在）

❖ 格式化日期时间函数 date()、gmdate()

```
date(string format [, int timestamp])
gmdate(string format [, int timestamp])
```

这两个函数都可以用来格式化日期时间，差别是 date() 函数返回的是本机计算机的时间，而 gmdate() 返回的是格林威治标准时间（Greenwich Mean Time，GMT）。参数 *format* 用来指定日期时间的显示格式（参阅下页的表格），参数 *timestamp* 代表要格式化的时间戳，若省略不写，表示为系统目前日期时间的时间戳。

请注意，由于 PHP.ini 文件内预设的时区为 UTC（Universal Time Coordinate，世界标准时间），比台湾地区时间晚 8 小时，导致 date() 函数返回的时间可能会晚本机时间 8 小时，若出现这种情况，可以在 PHP.ini 文件内搜索 date.timezone 字符串，删除前面的分号（如有的话），并将等号后面的值修改为 Asia/Taipei，如下所示，然后存盘，再重新启动 Apache 即可：

```
date.timezone = Asia/Taipei
```

至于 gmdate() 返回的是格林威治标准时间（GMT），和 UTC 时间相同，所以就不必修改 PHP.ini 文件。

字符	说明
a；A	a 会显示 am 或 pm；A 会显示 AM 或 PM
c	以 ISO 8601 格式显示日期时间，例如 2015-03-18T12:20:34+00:00。
d	以数字格式显示日期，若日期只有一位数，那么前面要加 0（01～31）
D	以英文缩写显示星期几（Sun、Mon～Sat）
F	以英文全名显示月份（January～December）
g	12 小时制，若小时只有一位数，那么前面不要加 0（1～12）
G	24 小时制，若小时只有一位数，那么前面不要加 0（0～23）
h	12 小时制，若小时只有一位数，那么前面要补 0（01～12）
H	24 小时制，若小时只有一位数，那么前面要补 0（00～23）
i	以数字格式显示分钟，若分钟只有一位数，那么前面要加 0（00～59）
I	判断是否为夏时制，返回 1 表示是，返回 0 表示否
j	以数字格式显示日期，若日期只有一位数，那么前面不要加 0（1～31）
l	以英文全名显示星期几（Sunday～Saturday）
L	判断是否为闰年，返回 1 表示是，返回 0 表示否
m	以数字格式显示月份，若月份只有一位数，那么前面要加 0（00～12）
M	以英文缩写显示月份（Jan～Dec）
n	以数字格式显示月份，若月份只有一位数，那么前面不要加 0（1～12）
O	显示与格林威治标准时间（GMT）的时差，例如 +0800
R	以 RFC 2822 格式显示日期时间，例如 Wed, 18 Mar 2015 12:21:41 +0000
s	以数字格式显示秒数，若秒数只有一位数，那么前面要加 0（00～59）
t	显示该日期指定的月份有几天，例如 31
T	显示本机计算机的时区，例如 Asia/Taipei
U	显示 UNIX 时间戳，例如 1096807832
w	以数字格式显示该日期时间为星期几（0（星期日）～6（星期六））
W	以 ISO-8601 格式显示该日期时间为一年的第几周
Y	以 4 位数显示公元年，例如 2015
y	以 2 位数显示公元年，例如 15
z	显示该日期为一年的第几天

❖ 日期时间运算函数 strtotime()

```
strtotime(string time[, int timestamp])
```

这个函数用来调整时间戳，参数 *time* 是要调整的英文命令，参数 *timestamp* 是要被调整

的时间戳，若省略不写，表示为目前系统日期的时间戳。

下面是几个例子。

- 将日期设置为今天的日期：

```
strtotime("now");
```

- 将日期设置为 2015/1/10 13:00:00：

```
strtotime("10 January 2015 13:00:00");
```

- 将当前日期增加 3 天：

```
strtotime("+3 day");
```

- 将当前日期减少 5 周：

```
strtotime("-5 week");
```

- 将当前日期增加 2 周、减少 2 天、减少 5 个小时、增加 15 秒：

```
strtotime("+2 week -2 days -5 hours +15 seconds");
```

- 将当前日期跳到下个星期日：

```
strtotime("next Sunday");
```

- 将日期跳到 2015/10/5 12:20:10 的上个星期六：

```
strtotime("last Sat", mktime(12, 20, 10, 10, 5, 2015, -1));
```

4.9.4 字符串函数

❖ 字符串转换函数

函数	说明
转换成小写 strtolower(string *str*)	将字符串参数 *str* 转换成小写字母，例如 strtolower("Happy") 会返回 "happy"
转换成大写 strtoupper(string *str*)	将字符串参数 *str* 转换成大写字母，例如 strtoupper("Happy") 会返回 " HAPPY"
字符串的第一个字符大写 ucfirst(string *str*)	将字符串参数 *str* 的第一个字符转换成大写字母，例如 ucfirst("happy birthday") 会返回 "Happy birthday"
字符串的每个单字的第一个字符大写 ucwords(string *str*)	将字符串参数 *str* 的每个单字的第一个字符转换成大写字母，例如 ucwords("happy birthday") 会返回 "Happy Birthday"

（续表）

函数	说明
获取字符的 ASCII 码 ord(string *str*)	返回字符串参数 *str* 第一个字符的 ASCII 码，类型为 int，例如 ord("1") 和 ord("123") 均会返回字符 "1" 的 ASCII 码为 49
获取 ASCII 码代表的字符 chr(int *ascii*)	参数 *ascii* 的类型为 int，代表的是 ASCII 码，返回值为这个 ASCII 码所代表的字符，例如 chr(65) 会返回字符"A"
字符串反转 strrev(string *str*)	将字符串参数 *str* 的字符顺序颠倒过来，例如 strrev("Happy") 会返回字符串 "yppaH"

❖ 字符串比较函数 strcmp()、strcasecmp()、strncmp()、strncasecmp()

```
strcmp(string str1, string str2)
strcasecmp(string str1, string str2)
strncmp(string str1, string str2, int length)
strncasecmp(string str1, string str2, int length)
```

strcmp() 和 strncmp() 两个函数都是用来比较字符串的，前者用来比较两个字符串全部的内容，后者用来比较参数 *length* 指定的字符数，返回值的类型为 int，代表字符串参数 *str1* 与 *str2* 的比较结果，若 *str1* 小于 *str2*，返回值会小于 0；若 *str1* 等于 *str2*，返回值会等于 0；若 *str1* 大于 *str2*，返回值会大于 0。

strcasecmp() 函数和 strcmp() 函数相似，而 strncasecmp() 函数和 strncmp() 函数相似，只是 strcasecmp() 和 strncasecmp() 两个函数在比较字符串时会区分英文字母的大小写，而 strcmp() 和 strncmp() 两个函数则不会。

❖ 取代字符串函数 str_replace()、str_ireplace()

```
str_replace(mixed search, mixed replace, mixed subject [, int count])
str_ireplace(mixed search, mixed replace, mixed subject [, int count])
```

将字符串参数 *subject* 内的 *search* 子字符串替换为 *replace* 子字符串，选择性参数 *count* 用来设置一个变量，用来存放成功取代字符串的次数。举例来说，假设 $str = "Shopping List";，则 str_replace("p", "P", $str, $count) 会返回 "ShoPPing List"，而变量 count 的值为 2，因为有两个 p 被取代为 P。

str_ireplace() 函数和 str_replace() 函数的差别在于 str_replace() 函数会区分英文字母大小写，而 str_ireplace() 函数则不会。

❖ 寻找字符串函数 strchr()、strstr()、stristr()、strrchr()

```
strchr(string haystack, string needle)
strstr(string haystack, string needle)
```

```
stristr(string haystack, string needle)
strrchr(string haystack, string needle)
```

strchr() 函数为 strstr() 函数的别名，两者均是返回第一次找到字符串参数 *needle* 指定之字符串后面的所有字符串。举例来说，假设“$email = "mary@ms17.url.com.tw";”，则 strstr($email, "@") 会返回 "@ms17.url.com.tw"。

此外，您也可以使用 stristr() 函数，其功能和 strchr()、strstr() 函数相似，差别在于 strchr()、strstr() 函数在搜索时会区分英文字母大小写，而 stristr() 函数则不会。

最后一个 strrchr() 函数，其功能也和 strchr()、strstr() 两个函数相似，差别在于它会返回最后一次找到参数 *needle* 指定字符串后面的所有字符串。

❖ 寻找字符串函数 strpos()、stripos()

```
strpos(string haystack, string needle [, int offset])
stripos(string haystack, string needle [, int offset])
```

返回字符串参数 *needle* 在字符串参数 *haystack* 中第一次出现的位置，选择性参数 *offset* 用来设置要开始搜索的位置，例如 strpos("Hello World", "o", 3) 会返回 4。

stripos() 函数和 strpos() 函数相似，差别在于 strpos() 函数在搜索时会区分英文字母大小写，而 stripos() 函数则不会。

❖ 寻找字符串函数 strrpos()、strripos()

```
strrpos(string haystack, string needle [, int offset])
strripos(string haystack, string needle [, int offset])
```

strrpos() 函数和 strpos() 函数相似，但改从字符串的右边开始搜索字符串参数 *needle* 在字符串参数 *haystack* 中第一次出现的位置，选择性参数 *offset* 用来设置要从第几个字符开始搜索，默认值是从最后一个字符开始搜索，例如 strrpos("Hello World", "o") 会返回 7。

strripos() 函数和 strrpos() 函数相似，差别在于 strrpos() 函数在搜索时会区分英文字母大小写，而 strripos() 函数则不会。

❖ 字符串操作函数

函数	说明
获取字符串长度 strlen(string *str*)	返回字符串参数 *str* 的长度，类型为 int，例如 strlen("Hello World") 会返回 11；strlen("程序设计") 会返回 8
由指定字符组成的字符串 str_repeat(string *input*, int *count*)	返回由参数 *input* 指定的字符所组成的字符串，参数 *count* 为参数 *input* 指定的字符所要出现的次数，例如 str_repeat("-", 10) 会返回 "----------"

（续表）

函数	说明
由字符串中指定位置返回指定个数的字符 substr(string *str*, int *start*[, int *length*])	从字符串参数 *string* 的第 *start*+1 个字符开始返回 *length* 个字符，例如 substr("Happy Birthday", 3, 5) 会返回 "py Bi "
删除字符串最左边指定的字符 ltrim(string *str*[, string *charlist*])	删除字符串参数 *str* 指定的字符串最左边符合参数 *charlist* 指定的字符，若参数 *charlist* 省略不写，则会删除字符串参数 *str* 最左边的空格符，然后将结果返回（类型为 string） 举例来说，假设“$mystr = "　小丸子";”，那么 ltrim($mystr) 会返回 "小丸子"；假设“$mystr = "小丸子";”，则 ltrim($mystr, "小") 会返回 "丸子"
删除字符串最右边指定的字符 rtrim(string *str*[, string *charlist*]) chop(string *str*[, string *charlist*])	删除字符串参数 *str* 指定的字符串最右边符合参数 *charlist* 指定的字符，若参数 *charlist* 省略不写，则会删除字符串参数 *str* 最右边的空格符，然后将结果返回（类型为 string）
删除字符串最左边或最右边指定的字符 trim(string *str*[, string *charlist*])	删除字符串参数 *str* 指定的字符串最左边或最右边符合参数 *charlist* 指定的字符，若参数 *charlist* 省略不写，则会删除字符串参数 *str* 最左边或最右边的空格符，然后将结果返回（类型为 string）
将换行符号转换成 HTML 换行元素 nl2br(string *str*)	将字符串参数 *str* 包含的换行符号 \n、\r 或 \r\n 转换成 HTML 换行元素 ，然后返回经过转换的字符串

❖ 将字符串数组组成单一字符串的函数 implode()

implode(string *separator*, array *pieces*)

这个函数用来将参数 *pieces* 指定的字符串数组，以参数 *separator* 指定的分隔符转换成单一字符串。举例来说，假设 $source 为包含 " Jennifer"、"Peter"、"Jean"、"Robert" 4 个元素的字符串数组，则 implode(" ", $source) 会返回 "Jennifer Peter Jean Robert"。

或者，您也可以写成 join(" ", $source)，因为 join() 函数为 implode() 函数的别名，其功能及语法均相同。

❖ 将字符串分解成子字符串数组函数 explode()

explode(string *separator*, string *str*[, int *limit*])

这个函数用来分解字符串，它会以参数 *separator* 作为分隔符，将字符串参数 *str* 分割为多个子字符串，然后返回字符串数组，数组的每个元素各自代表一个子字符串，选择性参数 *limit* 用来设置最多可以分割成几个子字符串，若省略不写，则表示不设置上限，允许任何数量的子字符串。

举例来说，假设“$str = "I Am Happy.";”，则 explode(" ", $str) 会传一个包含 "I"、"Am"、"Happy." 三个元素的字符串数组，而 explode(" ", $str, 2) 则会返回一个包含 "I"、"Am Happy." 两个元素的字符串数组，这是因为我们指定最多只能分割成两个子字符串，所以其余没被分割的字符串均会放在最后一个子字符串中。

事实上，PHP 所提供的函数远多于本节所列出来的函数，但由于篇幅有限，无法一一列举，有需要的读者可以自行参考 PHP 文件（http://www.php.net/ manual/en/）。

编写一个 PHP 网页，使它根据星期日、星期一到星期六按序变换背景图像为 bg0.gif、bg1.gif～bg6.gif，下载资源的 \samples\ch04 文件夹有这些图像文件，执行结果如图 4-14 所示。

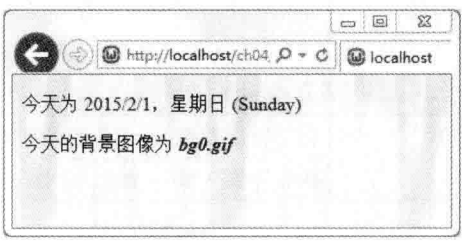


图 4-14

【提示】来自 <\ch04\changeBG.php>

```
<?php
  $Weekday = gmdate("l");                    //调用 gmdate() 函数获取今天是星期几
  switch ($Weekday)                          //根据今天是星期几决定要使用哪个背景图像
  {
    case "Sunday":
      $Bg = "bg0.gif";
      $Weekdayname = "星期日 ($Weekday)";
      break;
    ...
  }
?>
<!doctype html>
<html>
  <head><meta charset="utf-8"></head>
  <body background="<?php echo $Bg; ?>">
    <p>今天为 <?php echo gmdate("Y/n/j"); ?>，<?php echo $Weekdayname; ?></p>
    <p>今天的背景图像为 <b><i><?php echo $Bg; ?></i></b></p>
```

```
  </body>
</html>
```

一、选择题

(　)1.　若要标记函数的开头与结尾，必须使用下列哪个符号？

A. < >　　B. []　　C. ( )　　D. { }

(　) 2.　若要保留函数内局部变量的值，可以使用下列哪个关键词？

A. 不必使用关键词　　B. var

C. static　　D. global

(　) 3.　下列哪个函数可以用来取得四舍五入值？

A. ceil()　　B. round()　　C. floor()　　D. abs()

(　) 4.　下列哪个函数可以用来取得幂方值？

A. sqrt()　　B. pow()　　C. exp()　　D. rand()

(　) 5.　下列哪个函数可以用来获取当前的时间信息？

A. getdate()　　B. gettime()　　C. mktime()　　D. time()

(　) 6.　下列哪个函数可以用来格式化日期时间？

A. gmdate()　　B. checkdate()　　C. getdate()　　D. strtotime()

(　) 7.　下列哪个函数可以用来将字符串转换为小写字母？

A. ucfirst()　　B. ucwords()　　C. strtolower()　　D. strtoupper()

(　) 8.　下列哪个函数可以用来将字符串反转？

A. chr()　　B. ord()　　C. strstr()　　D. strrev()

(　) 9.　下列哪个函数可以用来将换行符号转换成 HTML 换行元素？

A. nl2br()　　B. substr()　　C. strcmp()　　D. strlen()

(　) 10.　strlen("Happy") 的返回值是什么？

A. "happy"　　B. "HAPPY"　　C. "yppaH"　　D. 5

二、实践题

1.　编写一个可以计算整数参数四次方的函数，然后调用这个函数计算 5 的四次方，并将结果显示在网页上。

2.　编写一个可以返回两个数值参数中比较大的参数的函数，然后调用这个函数返回 -5 和 -3 这两个参数中比较大的参数，并将结果显示在网页上。

第5章
文件访问

5.1　访问服务器端的路径

无论您要访问文件夹还是文件，都必须指定路径，PHP 提供了 basename()、pathinfo()、realpath() 等函数用来访问服务器端的路径，以下做进一步的说明。

5.1.1　获取文件名

basename() 函数可以用来获取指定路径的文件名，其语法如下，参数 *path* 为要获取文件名的路径，参数 *suffix* 用来设置返回结果所要排除的字符串，返回值为字符串，若指定的路径没有包含文件名，就返回文件夹名称：

```
basename(string path[, string suffix])
```

下面是一个例子，它会显示当前网页的文件名，程序第 02 行的 $_SERVER 为 PHP 预定义的服务器变量，这是 PHP 内置的超全局数组（superglobal array），而 $_SERVER['PHP_SELF'] 为当前网页的路径，第 03、04 行分别调用 basename() 函数获取指定路径的文件名，但第 04 行多指定了第二个参数，以排除扩展名，程序运行结果如图 5-1 所示。

\ch05\file1.php

```
01:<?php
02:   $path = $_SERVER['PHP_SELF'];
03:   echo basename($path).'<br>';                    //显示当前网页的文件名
04:   echo basename($path, '.php').'<br>';            //显示当前网页的文件名，但排除扩展名
05:?>
```

图 5-1

5.1.2　获取路径信息

pathinfo() 函数可以用来将指定路径分割为路径名称、文件名及扩展名三个部分，其语法如下，参数 *path* 为要进行分割的路径，返回值为包含三个元素的字符串数组，我们可以分别通过 dirname、basename、extension 三个键获取路径名称、文件名（包含扩展名）及扩展名（不

包含小数点）：

```
pathinfo(string path)
```

下面是一个例子，它会显示当前网页的路径及 pathinfo() 函数分割出来的路径名称、文件名及扩展名，程序运行结果如图 5-2 所示。

\ch05\file2.php

```
<?php
   $path = $_SERVER['PHP_SELF'];
   $path_parts = pathinfo($path);
   echo '当前网页的路径：'.$path.'<br>';
   echo '分割出来的路径名称：'.$path_parts['dirname'].'<br>';
   echo '分割出来的文件名：'.$path_parts['basename'].'<br>';
   echo '分割出来的扩展名：'.$path_parts['extension'].'<br>';
?>
```

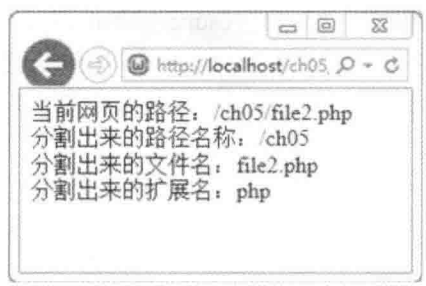


图 5-2

5.1.3 获取绝对路径

realpath() 函数可以用来获取文件的绝对路径，其语法如下，参数 *path* 为要获取绝对路径的文件，若指定的文件存在，就返回文件的绝对路径，否则返回 FALSE：

```
realpath(string path)
```

例如下面的语句会显示当前网页的绝对路径：<\ch05\file3.php>

```
echo '当前网页的绝对路径：'.realpath(basename($_SERVER['PHP_SELF']));
```

5.2 访问服务器端的文件夹

PHP 提供了许多函数用来存取服务器端的文件夹，例如 mkdir() 函数可以用来创建文件夹、rmdir() 函数可以用来删除文件夹、file_exists() 函数可以用来判断文件夹是否存在等，以

下做进一步的说明。

5.2.1　创建文件夹

mkdir() 函数可以用来创建文件夹，其语法如下，若成功创建文件夹，就返回 TRUE，否则返回 FALSE：

```
mkdir(string pathname[, int mode[, bool recursive]])
```

- 参数 *pathname*：用来指定要建立的文件夹的路径。
- 参数 *mode*：用来指定要建立的文件夹的权限模式，Windows 操作系统会忽略此参数，权限模式必须以数字来表示，而且前面要多加一个 0，默认值为 0777，表示最大权限，若不想设置此参数，可以设置为 NULL。
- 参数 *recursive*：用来指定当文件夹路径的其中一个或多个文件夹不存在时，是否一并加以建立，默认值为 FALSE，表示不会一并建立不存在的文件夹。

例如下面的语句是在 C:\myPHP 下建立一个名称为 pictures 的文件夹：

```
mkdir("C:\\myPHP\\pictures");
```

请注意，若 C:\myPHP 不存在，上面的语句将会失败（返回 FALSE），此时得改写成如下的形式，也就是指定第三个参数为 TRUE，一并建立不存在的文件夹：

```
mkdir("C:\\myPHP\\pictures", NULL, TRUE);
```

5.2.2　获取当前的工作文件夹

getcwd() 函数可以用来获取当前的工作文件夹，其语法如下，它没有参数，返回值为字符串，若没有变更过当前的工作文件夹，则返回值为当前网页所在文件夹：

```
getcwd();
```

5.2.3　切换当前的工作文件夹

PHP 当前的工作文件夹预设为当前网页所在的文件夹，chdir() 函数可以用来将当前的工作文件夹切换到其他文件夹，其语法如下，参数 *directory* 用来指定要切换到的文件夹，若成功切换文件夹，就返回 TRUE，否则返回 FALSE：

```
chdir(string directory)
```

例如下面的语句会先切换到 C:\，然后建立一个名称为 pictures 的文件夹：

```
chdir("C:\");
mkdir("pictures");
```

5.2.4 删除文件夹

rmdir() 函数可以用来删除文件夹，其语法如下，参数 *dirname* 为要删除的文件夹路径，若成功删除文件夹，就返回 TRUE，否则返回 FALSE：

```
rmdir(string dirname)
```

请注意，rmdir() 函数只能删除空的文件夹，若要删除的文件夹包含文件或文件夹，就要先删除这些文件或文件夹，否则会失败（返回 FALSE）。

例如下面的语句是在 C:\myPHP 下删除一个名称为 pictures 的文件夹，成功的先决条件是 pictures 必须是空的文件夹：

```
rmdir("C:\\myPHP\\pictures");
```

5.2.5 判断路径是否为文件夹

is_dir() 函数可以用来判断路径是否为文件夹，其语法如下，参数 *filename* 为判断是否为文件夹的路径，若指定的路径存在且为文件夹，就返回 TRUE，否则返回 FALSE，例如 is_dir("C:\") 会返回 TRUE；此外，若只有指定名称，没有包含完整路径，那么会在当前的工作文件夹内寻找是否有指定的文件夹：

```
is_dir(string filename)
```

5.2.6 判断文件夹是否存在

file_exists() 函数可以用来判断文件夹是否存在，其语法如下，参数 *filename* 为判断是否存在的文件夹路径，若存在，就会返回 TRUE，否则返回 FALSE：

```
file_exists(string filename)
```

例如，下面的语句会先判断文件夹是否存在，不存在的话才创建文件夹，否则显示“指定的文件夹已经存在”：

```
$folder_name = "C:\\myPHP\\pictures";
if (!file_exists($folder_name))
   mkdir($folder_name, NULL, TRUE);
else
   echo "指定的文件夹已经存在";
```

而下面的语句会先判断文件夹是否存在，存在的话才删除文件夹，否则显示“指定的文件夹不存在”：

```
$folder_name = "C:\\myPHP\\pictures";
if (file_exists($folder_name))
   rmdir($folder_name);
else
   echo "指定的文件夹不存在";
```

5.2.7　变更文件夹的权限

chmod() 函数可以用来变更文件夹的权限，其语法如下，参数 *filename* 用来指定文件夹名称或文件路径，参数 *mode* 用来指定权限模式，必须以数字来表示，而且前面要多加一个 0，Windows 会忽略此函数：

```
chmod(string filename, int mode)
```

例如下面的语句会将 pictures 文件夹的权限设置为“拥有者”具有读写权限：

```
chmod("pictures", 0600);
```

5.2.8　获取文件夹的父文件夹名称

dirname() 函数可以用来获取文件夹的父文件夹名称，其语法如下，参数 *path* 为要获取父文件夹名称的文件夹，例如 dirname("C:\\Windows\\system32\\drivers") 会返回 "C:\Windows\system32"：

```
dirname(string path)
```

5.2.9　获取文件夹所包含的文件名及子文件夹名称

scandir() 函数可以用来获取文件夹所包含的文件名及子文件夹名称，其语法如下，参数 *directory* 为要获取内容的文件夹，返回值为字符串数组，而且字符串数组存放文件名及子文件夹名称的排序方式取决于参数 *sorting_order* 的设置，预设为升序，若要变更为递减排序，可以将参数 *sorting_order* 设置为 1：

```
scandir(string directory[, int sorting_order])
```

下面是一个例子，它会显示 C:\wamp 文件夹所包含的文件名及子文件夹名称，排序方式为按名称递减排序，并过滤掉 . 和 .. 两个名称，程序运行结果如图 5-3 所示。

\ch05\file4.php

```
<?php
   $file = scandir("C:\\wamp", 1);
   foreach($file as $value)
      if（$value != "." && $value != "..") echo $value . " ";
?>
```

图 5-3

5.3 访问服务器端的文件

PHP 提供了许多函数可以用来访问服务器端的文件，例如 file_exists() 函数可以用来判断文件或文件夹是否存在；is_file() 函数可以用来判断指定的路径是否为文件；copy() 函数可以用来复制文件等，以下做进一步的说明。

5.3.1 判断文件是否存在

file_exists() 函数可以用来判断文件或文件夹是否存在，其语法如下，若参数 *filename* 指定的文件或文件夹存在，就返回 TRUE，否则返回 FALSE：

```
file_exists(string filename)
```

5.3.2 判断指定的路径是否为文件

is_file() 函数可以用来判断指定的路径是否为文件，其语法如下，若参数 *filename* 指定的路径存在且为文件，就返回 TRUE，否则返回 FALSE。当参数 *filename* 没有包含完整路径，会在当前的工作文件夹内寻找是否有指定的文件：

```
is_file(string filename)
```

5.3.3　复制文件

copy() 函数可以用来复制文件，其语法如下，它会将参数 *source* 指定的文件复制到参数 *dest* 指定的位置及文件名，当目标文件已经存在时，旧文件将会被覆盖，若复制文件成功，就返回 TRUE，否则返回 FALSE：

```
copy(string source, string dest)
```

例如下面的语句会将文件 C:\myPHP\license.txt 复制到当前的工作文件夹且文件名为 license(new).txt：

```
copy("C:\\myPHP\\license.txt", "license(new).txt")
```

5.3.4　删除文件

unlink() 函数可以用来删除文件，其语法如下，参数 *filename* 为要删除的文件，若成功删除文件，就返回 TRUE，否则返回 FALSE：

```
unlink(string filename)
```

例如下面的语句会删除当前的工作文件夹内的 myfile.txt 文件，成功的话就显示“删除文件成功！”，否则显示“文件不存在，删除文件失败！”：<\ch05\file5.php>

```
<?php
   $filename = "myfile.txt";
   if (file_exists($filename))                    //先判断文件是否存在
   {
      unlink($filename);                          //存在的话就删除文件并显示成功信息
      echo "删除文件成功！";
   }
   else
   {
      echo "文件不存在，删除文件失败！";              //不存在的话就显示失败信息
   }
?>
```

5.3.5　变更文件名

rename() 函数可以用来变更文件名或文件夹名称，其语法如下，参数 *oldname* 为旧的名称，参数 *newname* 为新的名称，若成功变更名称，就返回 TRUE，否则返回 FALSE：

```
rename(string oldname, string newname)
```

例如，下面的语句会将文件 temp.php 改名为 temp.bak：

```
rename("temp.php", "temp.bak");
```

5.3.6 获取文件属性

PHP 提供了下列几个函数可以用来获取文件属性。

函数	说明
fileatime(string *filename*)	获取文件或文件夹的最后访问时间，返回值为 UNIX 的时间戳（timestamp），您可以使用 gmdate() 函数将之转换为较易懂的格式
filectime(string *filename*)	获取文件或文件夹的创建时间，返回值为 UNIX 的时间戳，您可以使用 gmdate() 函数将之转换为较易懂的格式
filemtime(string *filename*)	获取文件或文件夹的修改时间，返回值为 UNIX 的时间戳，您可以使用 gmdate() 函数将之转换为较易懂的格式
filesize(string *filename*)	获取文件的大小，单位为字节（bytes）
is_readable(string *filename*)	若文件存在且可以读取，就返回 TRUE，否则返回 FALSE
is_writable(string *filename*)	若文件或文件夹存在且可以写入，就返回 TRUE，否则返回 FALSE

例如下面的语句会显示当前网页的文件信息，如图 5-4 所示：<\ch05\file6.php>

```
<?php
  $filename = basename($_SERVER['PHP_SELF']);
  echo '当前网页的建立时间：'.gmdate("Y-m-d H:i:s", filectime($filename)).'<br>';
  echo '当前网页的最后访问时间：'.gmdate("Y-m-d H:i:s", fileatime($filename)).'<br>';
  echo '当前网页的修改时间：'.gmdate("Y-m-d H:i:s", filemtime($filename)).'<br>';
  echo '当前网页的文件大小：'.filesize($filename).'字节';
?>
```

图 5-4

5.4　读取服务器端的文本文件

5.4.1　使用 fread() 函数读取文本文件

使用 fread() 函数读取文本文件的流程为“打开文件”→“读取文件”→“关闭文件”。

❖　一、打开文件

fopen() 函数可以用来打开文件，其语法如下，参数 *filename* 为要打开的文件（文件路径或 URL 网址），参数 *mode* 用来指定要以什么模式打开文件，若成功打开文件，就返回资源变量，否则返回 FALSE：

```
fopen(string filename, string mode)
```

参数 mode 的模式	说明
r	以只读的方式打开文件，并将文件指针置于文件的最前端，若指定的文件不存在，就会打开失败
r+	以可擦写的方式打开文件，并将文件指针置于文件的最前端，若指定的文件不存在，就会打开失败
w	以唯写的方式打开文件且清除文件内容，并将文件指针置于文件的最前端，若文件不存在，就会创建文件
w+	以可擦写的方式打开文件且清除文件内容，并将文件指针置于文件的最前端，若文件不存在，就会创建文件
a	以只写的方式打开文件，并将文件指针置于文件的最后端，若文件不存在，就会创建文件
a+	以可擦写的方式打开文件，并将文件指针置于文件的最后端，若文件不存在，就会创建文件
X	以只写的方式建立且打开文件，并将文件指针置于文件的最前端，若指定的文件存在，就会打开失败，否则会创建且打开文件
x+	以可擦写的方式创建且打开文件，并将文件指针置于文件的最前端，若指定的文件存在，就会打开失败，否则会创建且打开文件

例如，下面的语句会以只读的方式打开当前工作文件夹内的 poetry1.txt 文件：

```
$handle = fopen("poetry1.txt", "r");
```

❖　二、读取文件

打开文件后，我们可以使用 fread() 函数读取文件内容，其语法如下，它会从参数 *handle*

指定的资源变量中读取文件内容，而且是从文件指针处开始读取参数 *length* 指定的字节数，若抵达文件尾端（End Of File，EOF），就停止读取：

```
fread(resource handle, int length)
```

例如，下面的语句会先打开 poetry1.txt 文件，然后读取文件全部内容：

```
$handle = fopen("poetry1.txt", "r");          //以只读的方式开启档案
fread($handle, filesize("poetry1.txt"));      //读取档案全部内容
```

❖ 三、关闭文件

文件内容读取完毕后，我们必须使用 fclose() 函数关闭文件，其语法如下，若成功关闭文件，就返回 TRUE，否则返回 FALSE：

```
fclose(resource handle)
```

下面是一个例子，它会读取 poetry1.txt 文件全部内容，然后显示出来，如图 5-5 所示。由于 fread() 函数读取的文件内容可能包括换行符号 \n、\r、\r\n，但浏览器并不认得它们，故调用 nl2br(string *str*) 函数将参数 *str* 包含的换行符号转换成换行元素
。

\ch05\file7.php

```
<?php
  $filename = "poetry1.txt";
  $handle = fopen($filename, "r");
  if（$handle) //检查文件是否打开成功
  {
    $contents = nl2br(fread($handle, filesize($filename)));
    fclose($handle);
    echo $contents;
  }
  else
    echo "打开文件失败！ ";
?>
<i>凤凰台上凤凰游，凤去台空江自流。</i>
<i>吴宫花草埋幽径，晋代衣冠成古邱。</i>
<i>三山半落青又外，二水中分白鹭洲。</i>
<i>总为浮云能蔽日，长安不见使人愁。</i>
```

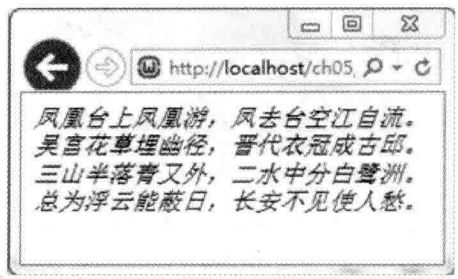


图 5-5

5.4.2　使用 fgets() 函数读取文本文件

fgets() 函数可以从文件指针处读取一行数据，其语法如下：

```
fgets(resource handle)
```

由于 fgets() 函数一次读取一行数据，因此，我们可以使用 while 循环读取文件全部内容，至于循环要执行几次，可以使用 feof() 函数来判断是否已经抵达文件尾端，其语法如下，若已经抵达文件尾端，就返回 TRUE，否则返回 FALSE：

```
feof(resource handle)
```

下面是一个例子，它会读取 poetry1.txt 文件全部内容，然后显示出来，如图 5-6 所示。由于 poetry1.txt 文件内有 HTML 元素 <i>，故网页上所显示的诗句会加上斜体。同样的，由于 fgets() 函数读取的文件内容可能包含换行符号 \n、\r、\r\n，但浏览器并不认得它们，故调用 nl2br(string *str*) 函数将参数 *str* 包含的换行符号转换成换行元素
。

\ch05\file8.php

```
<?php
   $handle = fopen("poetry1.txt", "r");
   //使用 while 循环读取文件全部内容
   while（!feof($handle))
   {
      $line = nl2br(fgets($handle));
      echo $line;
   }
   fclose($handle);
?>
```

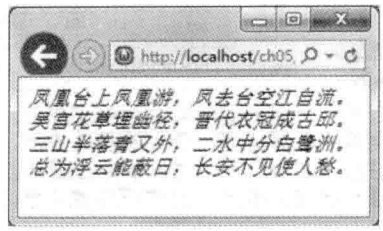


图 5-6

5.4.3 使用 file_get_contents() 函数读取文本文件

file_get_contents() 函数无须经过打开文件及关闭文件的动作即可读取文件全部内容，其语法如下，若成功读取文件全部内容，就返回文件全部内容，否则返回 FALSE：

```
file_get_contents(string filename)
```

下面是一个例子，它会读取 poetry1.txt 文件全部内容，然后显示出来，如图 5-7 所示。由于 poetry1.txt 文件内有 HTML 元素 <i>，故网页上所显示的诗句会加上斜体。

\ch05\file9.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <?php
      echo nl2br(file_get_contents("poetry1.txt"));
    ?>
  </body>
</html>
```

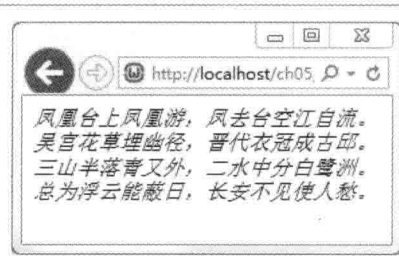


图 5-7

备注

PHP 还提供了一个和 fgets() 函数功能类似的 fgetss(resource *handle*) 函数，两者均是从文件指针处读取一行数据，差别在于 fgetss() 函数会删除文件内的 HTML 元素。

5.5　写入服务器端的文本文件

5.5.1　使用 fwrite()、fputs() 函数写入文本文件

fputs() 函数是 fwrite() 函数的别名，两者功能相同，您可以根据自己的喜好选择使用，其语法如下，若成功写入数据，就返回写入文件的字节数，否则返回 FALSE：

```
fwrite(resource handle, string str [, int length])
fputs(resource handle, string str [, int length])
```

fwrite()、fputs() 函数会将参数 *str* 指定的内容写入参数 *handle* 指定的资源变量，而且是从文件指针处开始写入参数 *length* 指定的字节数，若参数 *str* 指定的内容比参数 *length* 指定的字节数少，就会在写入参数 *str* 指定的内容后停止，若省略参数 *length*，就写入参数 *str* 的全部内容。

使用 fwrite()、fputs() 函数写入文本文件的流程为“打开文件”→“写入文件”→“关闭文件”。请注意，写入文件是从文件指针处开始写入，所以打开文件时指定的文件模式就很重要，若要从文件的最前端写入数据且覆盖旧数据，必须使用 r+；若要从文件的最前端写入数据且先清除旧数据，必须使用 w、w+；若要从文件的最尾端写入数据，必须使用 a、a+。

下面是一个例子，假设 poetry2.txt 文件的原始内容如下：

```
<i>凤凰台上凤凰游，凤去台空江自流。</i>
<i>吴宫花草埋幽径，晋代衣冠成古邱。</i>
```

我们打算在文件的尾端写入如下数据，则文件模式必须使用 a 或 a+，程序运行结果如图 5-8 所示。

```
<i>三山半落青又外，二水中分白鹭洲。</i>
<i>总为浮云能蔽日，长安不见使人愁。</i>
```

\ch05\file10.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <?php
      $contents = "";                                    //此变量用来存放要写入文件的内容
      $handle = fopen("poetry2.txt", "a");               //打开文件
      if（$handle)                                       //若打开文件成功，就写入指定的内容
      {
        $contents .= "\r\n";                             //指定写入文件的内容，包括换行符号
        $contents .= "<i>凤凰台上凤凰游，凤去台空江自流。<\i>\r\n";
```

```
        $contents .= "<i>吴宫花草埋幽径，晋代衣冠成古邱。<\i>\r\n";
        $contents .= "<i>三山半落青又外，二水中分白鹭洲。<\i>\r\n";
        $contents .= "<i>总为浮云能蔽日，长安不见使人愁。<\i>";
        $num = fwrite($handle, $contents);                    //写入文件
        fclose($handle);                                      //关闭文件
        echo "成功写入".$num."个字节";                         //写入完毕后显示成功信息
    }
    else
     echo "打开文件失败";
  ?>
 </body>
</html>
```

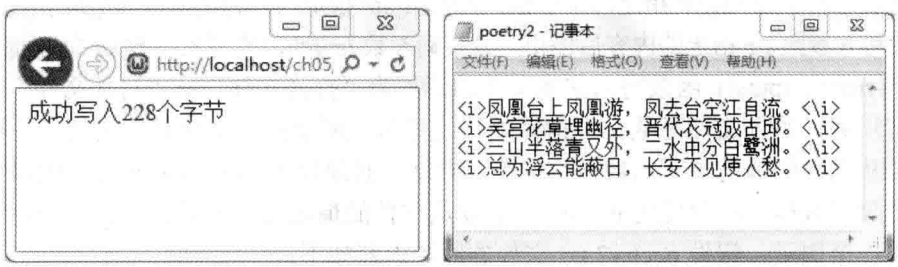


图 5-8

5.5.2 使用 file_put_contents() 函数写入文本文件

file_put_contents() 函数无须经过打开文件及关闭文件的动作即可将指定的内容写入文件，其语法如下，参数 *filename* 为文件名，参数 *data* 为要写入文件的数据，若成功写入文件，就返回写入文件的字节数，否则返回 FALSE：

```
file_put_contents(string filename, string data)
```

请注意，若参数 *filename* 指定的文件不存在，file_put_contents() 函数会自动创建文件，否则会先清除文件内容，再从文件最前端写入数据。从 file_put_contents() 函数的工作模式来看，它等于是按序执行了 fopen()、fwrite() 及 fclose() 函数，而且 fopen() 函数的文件模式为 w。

下面是一个例子，它会将唐诗写入文件 poetry3.txt，程序运行结果如图 5-9 所示。<\ch05\file11.php>

```
<?php
  $contents = "故人具鸡黍，邀我至田家，\r\n";                    //指定要写入文件的内容
  $contents .= "绿树村边合，青山郭外斜，\r\n";
  $contents .= "开轩面场圃，把酒话桑麻，\r\n";
  $contents .= "待到重阳日，还来就菊花。";
  $num = file_put_contents("poetry3.txt", $contents);          //写入文件
```

```
    echo "成功写入".$num."个字节";
?>
```

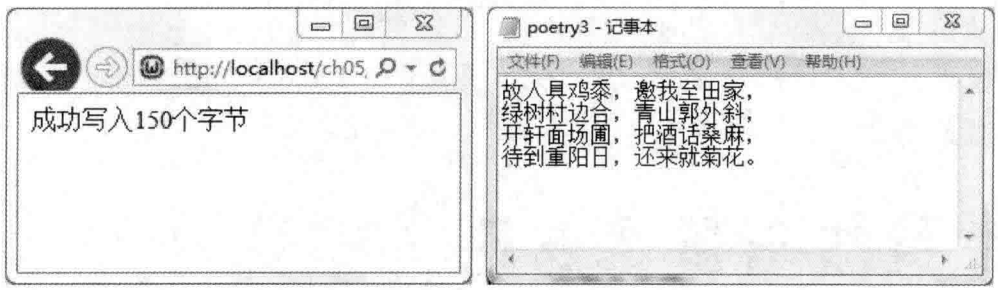


图 5-9

1. 编写一个 PHP 网页，使它打开 sample1.txt 文件（位于下载资源的 \samples\ch05 文件夹），然后一次读取一行，写入另一个新的文本文件 sample2.txt。

\ch05\sample1.txt

```
床前明月光，
疑是地上霜，
举头望明月，
低头思故乡。
```

2. 编写一个 PHP 网页，使它打开 sample3.txt 文件（位于下载资源的 \samples\ch05 文件夹），然后一次读取一个字符，转换成大写字母，再写入另一个新的文本文件 sample4.txt（提示：您可以使用 strtoupper(string *str*) 函数将参数 *str* 转换成大写字母）。

\ch05\sample3.txt

```
Five-character-regular-verse
STOPPING AT A FRIEND'S FARM-HOUSE
Author:Meng Haoran
Preparing me chicken and rice, old friend,
You entertain me at your farm.
We watch the green trees that circle your village
And the pale blue of outlying mountains.
We open your window over garden and field,
To talk mulberry and hemp with our cups in our hands.
...Wait till the Mountain Holiday --
I am coming again in chrysanthemum time.
```

3. 编写一个 PHP 网页，使它显示 C 盘根目录下的所有文件名及子文件夹名称。

第 6 章
GD 绘图与图像处理

6.1　GD 绘图

我们可以使用 PHP 的 GD（Gif Draw）扩充功能进行绘图，其步骤如下：

步骤 01　创建空白图像：使用 imagecreate() 或 imagecreatetruecolor() 函数创建空白图像，背景颜色默认为黑色。

步骤 02　分配颜色：使用 imagecolorallocate() 函数分配图像可以使用的颜色。

步骤 03　绘制线条、图形或文字：使用 PHP 提供的函数绘制线条、图形或文字。

步骤 04　输出图像：将图像传送到客户端浏览器或存储在 Web 服务器中。

步骤 05　释放内存：释放图像占用的内存空间。

6.1.1　创建空白图像

在使用 GD 绘制线条、图形或文字之前，必须先创建空白图像，它就像一块用来作画的画布，我们可以使用 imagecreate() 或 imagecreatetruecolor() 函数创建空白图像，其语法如下：

```
imagecreate(int x_size, int y_size)                    //x_size 为图像宽度，y_size 为图像高度
imagecreatetruecolor(int x_size, int y_size)           //x_size 为图像宽度，y_size 为图像高度
```

若 imagecreate() 或 imagecreatetruecolor() 函数成功创建空白图像，就返回代表该图像的图像标识符（image identifier），这两个函数的差别在于前者只支持 256 色，而后者支持百万色，若要绘制颜色单纯的图形，可以使用 imagecreate() 函数，否则使用 imagecreatetruecolor() 函数。

例如，下面的语句分别创建大小为 800×600 和 1024×768 的空白图像，前者支持 256 色，而后者支持百万色：

```
$im1 = imagecreate(800, 600);                          //单位为像素，背景色彩预设为黑色
$im2 = imagecreatetruecolor(1024, 768);                //单位为像素，背景色彩预设为黑色
```

6.1.2　分配颜色

创建空白图像后，还要使用 imagecolorallocate() 函数分配该图像可以使用的颜色，才能进一步绘制线条、图形或文字，其语法如下：

```
imagecolorallocate(resource image, int red, int green, int blue)
```

参数 *image* 为要分配颜色的图像，参数 *red*、*green*、*blue* 为红、绿、蓝三色，必须介于 0 ~ 255 之间，其中 0 表示缺少该颜色，255 表示该颜色最多，故 imagecolorallocate($im, 0, 0, 0) 表示分配黑色，imagecolorallocate($im, 255, 255, 255) 表示分配白色。

例如，下面的语句创建了一个大小为 200×200、支持百万色的空白图像，并指定该图像

可以使用白、红两种颜色：

```
01:<?php
02:  //创建一个大小为 200×200、支持百万色的空图片
03:  $im = imagecreatetruecolor(200, 200);
04:
05:  //分配白、红两种颜色
06:  $white = imagecolorallocate($im, 255, 255, 255);
07:  $red = imagecolorallocate($im, 255, 0, 0);
08:?>
```

- 03：使用 imagecreatetruecolor() 函数创建一个大小为 200×200、支持百万色的空白图像，变量 $im 即代表该空白图像。
- 06、07：使用 imagecolorallocate() 函数分配白、红两个颜色给第 03 行创建的空白图像使用，变量 $white、$red 分别代表白色和红色。请注意，若是使用 imagecreate() 函数创建空白图像，那么第一个分配的颜色（白色）会自动成为图像的背景颜色，以此例来说，背景颜色会是黑色，因为使用了 imagecreatetruecolor() 函数创建空白图像。

6.1.3 绘制线条、图形与文字

我们可以使用 PHP 提供的函数绘制线条、图形与文字，以下做进一步的说明。

❖ 绘制椭圆形

我们可以使用 imageellipse() 函数绘制椭圆形，其语法如下，它会以颜色 *color* 在画布 *image* 的坐标（*cx*, *cy*）处绘制宽度为 *w*、高度为 *h* 的椭圆形，若宽度与高度相同，表示为圆形：

```
imageellipse(resource image, int cx, int cy, int w, int h, int color)
```

例如下面的语句用于绘制两个椭圆，程序运行结果如图 6-1 所示：

```
01:$im = imagecreate(200, 200);
02:$background = imagecolorallocate($im, 255, 255, 255);
03:$red = imagecolorallocate($im, 255, 0, 0);
04:$green = imagecolorallocate($im, 0, 255, 0);
05:
06://绘制椭圆形
07:imageellipse($im, 100, 100, 199, 199, $red);
08:imageellipse($im, 100, 100, 150, 100, $green);
```

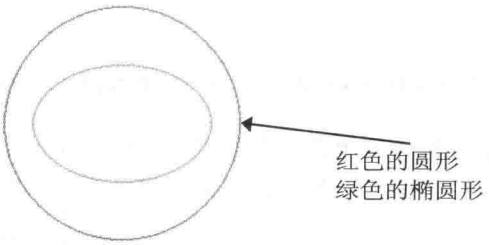


图 6-1

- 01：创建 200 × 200 的空白图像。
- 02：分配图像背景颜色为白色。
- 03、04：分配图像可以使用红、绿两种颜色。
- 07：使用红色从坐标（100, 100）处绘制宽度为 199、高度为 199 的椭圆形，因为宽度与高度相同，所以是圆形。
- 08：使用绿色从坐标（100, 100）处绘制宽度为 150、高度为 100 的椭圆形。

❖ 绘制直线

我们可以使用 imageline() 函数绘制线条，其语法如下，它会以颜色 *color* 在画布 *image* 上绘制从坐标（*x1*, *y1*）到坐标（*x2*, *y2*）的直线：

```
imageline(resource image, int x1, int y1, int x2, int y2, int color)
```

例如，下面的语句是使用黑色绘制从坐标（0, 0）到坐标（100, 100）的直线，如图 6-2 所示：

```
$im = imagecreate(100, 100);
$background = imagecolorallocate($im, 0, 255, 255);
$black = imagecolorallocate($im, 0, 0, 0);
//绘制直线
imageline($im, 0, 0, 100, 100, $black);
```

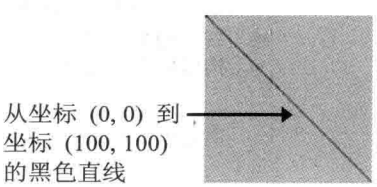


图 6-2

❖ 绘制多边形

我们可以使用 imagepolygon() 函数绘制多边形，其语法如下，它会以颜色 *color* 在画布 *image* 上绘制由 *points* 数组指定的各个点所构成的多边形，*num_points* 则是用来指定多边形的

端点个数：

```
imagepolygon(resource image, array points, int num_points, int color)
```

例如下面的语句是使用黑色线条绘制由坐标（210, 50）、(140, 80）、(110, 20）、(155, 50）所构成的多边形，由于是 4 个坐标，所以是四边形，如图 6-3 所示：

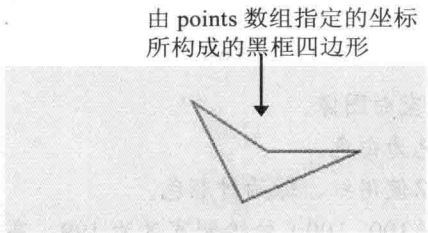


图 6-3

❖ 绘制矩形

我们可以使用 imagerectangle() 函数绘制矩形，其语法如下，它会以颜色 *color* 在画布 *image* 上绘制由坐标（*x1*, *y1*）与（*x2*, *y2*）作为左上角及右下角端点所构成的矩形：

```
imagerectangle(resource image, int x1, int y1, int x2, int y2, int color)
```

例如下面的语句是使用红色绘制到坐标（0, 0）与（99, 99）作为左上角及右下角端点所构成的矩形，如图 6-4 所示：

```
$im = imagecreate(100, 100);
$background = imagecolorallocate($im, 255, 255, 255);
$red = imagecolorallocate($im, 255, 0, 0);
//绘制矩形
imagerectangle($im, 0, 0, 99, 99, $red);
```

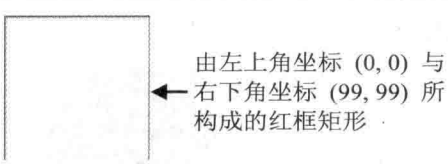


图 6-4

❖ 绘制弧线

我们可以使用 imagearc() 函数绘制弧线，其语法如下，它会以颜色 *color* 在画布 *image* 上从坐标（*cx*, *cy*）处绘制宽度为 *w*、高度为 *h* 的圆弧，而且开始绘制的角度为 *s*（相对于 X 轴），结束的角度为 *e*（顺时针方向）：

imagearc(resource *image*, int *cx*, int *cy*, int *w*, int *h*, int *s*, int *e*, int *color*)

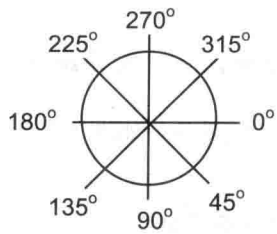


图 6-5

例如，下面的语句是使用绿色从坐标（100, 100）处绘制宽度为 150、高度为 150 的圆弧，而且开始绘制的角度为 90°，结束的角度为 270°，如图 6-6 所示：

```
$im = imagecreate(200, 200);
$background = imagecolorallocate($im, 255, 255, 255);
$green = imagecolorallocate($im, 0, 255, 0);
//绘制弧线
imagearc($im, 100, 100, 150, 150, 90, 270, $green);
```

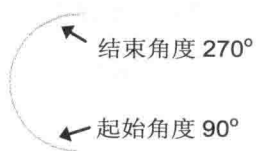


图 6-6

❖ 填充颜色

我们可以使用 imagefill() 函数填充颜色，其语法如下，它会在画布 *image* 上从坐标（*x*, *y*）所在区域填充 *color* 指定的颜色：

imagefill(resource *image*, int *x*, int *y*, int *color*)

例如，下面的语句是从坐标（0, 0）所在区域填充黄色，所以整张图像的背景都会变成黄色：

```
$im = imagecreatetruecolor(100, 100);
$yellow = imagecolorallocate($im, 255, 255, 0);
//填满色彩
imagefill($im, 0, 0, $yellow);
```

❖ 填充椭圆形颜色

我们可以使用 imagefilledellipse() 函数填充椭圆形颜色，其语法如下，它会在画布 *image*

上以颜色 *color* 从坐标（*cx*, *cy*）处填充宽度为 *w*、高度为 *h* 的椭圆形，若宽度与高度相同，表示为填充圆形颜色：

```
imagefilledellipse(resource image, int cx, int cy, int w, int h, int color)
```

例如，下面的语句是使用蓝色填充圆形，如图 6-7 所示：

```
$im = imagecreate(200, 200);
$background = imagecolorallocate($im, 255, 255, 255);
$blue = imagecolorallocate($im, 0, 0, 255);
//填充椭圆形
imagefilledellipse($im, 100, 100, 199, 199, $blue);
```

填充蓝色的圆形
（背景颜色为白色）

图 6-7

❖ 填充多边形颜色

我们可以使用 imagefilledpolygon() 函数填充多边形颜色，其语法如下，它会在画布 *image* 上以颜色 *color* 填充由 *points* 数组指定的各个点所构成的多边形，*num_points* 则用来指定多边形的端点个数：

```
imagefilledpolygon(resource image, array points, int num_points, int color)
```

例如下面的语句是创建黄色背景的空白图像，然后以白色填充至坐标 （210, 50）、(140, 80)、(110, 20) 所构成的三角形，如图 6-8 所示：

```
$points = array(210, 50, 140, 80, 110, 20);
$im = imagecreatetruecolor(250, 100);
$yellow = imagecolorallocate($im, 255, 255, 0);
$white = imagecolorallocate($im, 255, 255, 255);
imagefill($im, 0, 0, $yellow);
//填充多边形，因为只有 3 个边，所以是三角形
imagefilledpolygon($im, $points, 3, $white);
```

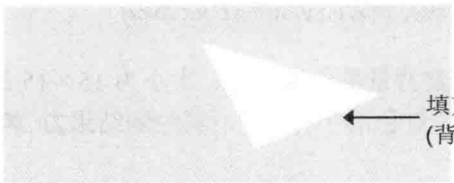


图 6-8

❖ 填充矩形颜色

我们可以使用 imagefilledrectangle() 函数填充矩形颜色，其语法如下，它会在画布 *image* 上以颜色 *color* 填充至坐标（*x1*, *y1*）与（*x2*, *y2*）作为左上角及右下角端点所构成的矩形：

```
imagefilledrectangle(resource image, int x1, int y1, int x2, int y2, int color)
```

例如，下面的语句是以白色填充至坐标（20, 20）、(80, 80）组成的矩形：

```
$im = imagecreatetruecolor(100, 100);
$white = imagecolorallocate($im, 255, 255, 255);
//填充矩形
imagefilledrectangle($im, 20, 20, 80, 80, $white);
```

❖ 绘制字符

我们可以使用 imagechar() 函数绘制字符，其语法如下，它会在画布 *image* 上以颜色 *color* 从坐标（*x*, *y*）处绘制字号为 *size* 的字符 *c*，若参数 *c* 包含多个字符，那么只会绘制出第 1 个字符，字号 *size* 为 1 ~ 5 的数字，数字越大，字体就越大：

```
imagechar(resource image, int size, int x, int y, string c, int color)
```

例如，下面的语句是创建背景颜色为黄色、大小为 15×15 的空白图像，然后从坐标（4, 0）处绘制字号为 5、红色的字符 "8"，其绘制结果为 8：

```
$im = imagecreate(15, 15);
$background = imagecolorallocate($im, 255, 255, 0);
$red = imagecolorallocate($im, 255, 0, 0);
//绘制字符
imagechar($im, 5, 4, 0, "890", $red);
```

❖ 绘制倒伏的字符

我们可以使用 imagecharup() 函数绘制倒伏的字符，其语法如下，它会在画布 *image* 上以颜色 *color* 从坐标（*x*, *y*）处以倒伏方式绘制字号为 *size* 的字符 *c*，若参数 *c* 包含多个字符，那么只会绘出第一个字符，字号 *size* 为 1 ~ 5 的数字，数字越大，字体就越大：

```
imagecharup(resource image, int size, int x, int y, string c, int color)
```

例如，下面的语句是创建背景颜色为黄色、大小为 15×15 的图像，然后从坐标（0, 10）处以倒伏方式绘制字号为 5、红色的字符 "8"，其绘制结果为 ∞：

```
$im = imagecreate(15, 15);
$background = imagecolorallocate($im, 255, 255, 0);
$red = imagecolorallocate($im, 255, 0, 0);
imagecharup($im, 5, 0, 10, "8", $red);
```

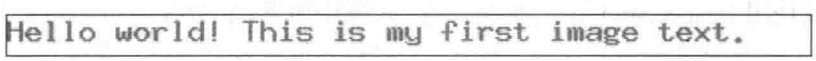
❖ 绘制文字

我们可以使用 imagestring() 函数绘制文字，其语法如下，它会在画布 *image* 上以颜色 *color* 从坐标（*x*, *y*）处绘制字号为 *size* 的文字 *s*，字号 *size* 为 1～5 的数字，数字越大，字体就越大：

```
imagestring(resource image, int size, int x, int y, string s, int color)
```

例如，下面的语句是以粉红色从坐标（0, 0）处绘制 "Hello world! This is my first image text." 文字，字号为 5，请注意，这个函数不支持绘制汉字：

```
$im = imagecreate(400, 20);
$background = imagecolorallocate($im, 255, 255, 255);
$textcolor = imagecolorallocate($im, 255, 0, 255);
imagestring($im, 5, 0, 0, "Hello world! This is my first image text.", $textcolor);
```



❖ 绘制倒伏的文字

我们可以使用 imagestringup() 函数绘制倒伏的文字，其语法如下，它会在画布 *image* 上以颜色 *color* 从坐标（*x*, *y*）处以倒伏方式绘制字号为 *size* 的文字 *s*，字号 *size* 为 1～5 的数字，数字越大，字体就越大：

```
imagestringup(resource image, int size, int x, int y, string s, int color)
```

例如，下面的语句是以蓝色从坐标（2, 395）处绘制 "Hello world! This is my first image text." 的倒伏文字，字号为 5，请注意，这个函数不支持绘制汉字：

```
$im = imagecreate(20, 400);
$background = imagecolorallocate($im, 255, 255, 255);
$textcolor = imagecolorallocate($im, 0, 0, 255);
imagestringup($im, 5, 2, 395, "Hello world! This is my first image text.", $textcolor);
```

❖ 绘制 TrueType 文字

我们可以使用 imagettftext() 函数绘制 TrueType 文字，其语法如下，它会在画布 *image* 上以颜色 *color* 从坐标（*x*, *y*）处绘制文字 *text*，字号为 *size*（单位为点数），文字角度为 *angle*（0 表示由左向右书写的文字），字体文件为 *fontfile*：

```
imagettftext(resource image, float size, float angle, int x, int y, int color, string fontfile, string text)
```

例如，下面的语句是从坐标（0, 20）处绘制一串文字，字号为 16，由左向右书写，字体颜色为蓝色，字体文件为 simhei.ttf：

```
$im = imagecreate(500, 28);
$background = imagecolorallocate($im, 255, 255, 255);
$textcolor = imagecolorallocate($im, 0, 0, 255);
$string = "哇哈哈，imagettftext() 函数可以绘出中文字哦...";
imagettftext($im, 16, 0, 0, 20, $textcolor, "simhei.ttf", $string);
```

哇哈哈，imagettftext() 函数可以绘出中文字哦...

6.1.4　输出图像

前几节所绘制的图像都存储在计算机的内存中，PHP 提供了下列几个函数，可以将图像传送至客户端浏览器或保存在服务器中。

❖ GIF 格式

```
imagegif(resource image[, string filename])
```

这个函数会将图像传送至客户端浏览器或保存在服务器中，参数 *filename* 省略不写的话，表示要将图像传送至浏览器，此时，我们必须指定输出至浏览器的 MIME 类型，例如：

```
header("content-type: image/gif");
imagegif($im);
```

若选择保存在服务器中，可以写成如下的形式：

```
imagegif($im, "images/simple.gif");
```

❖ PNG 格式

```
imagepng(resource image[, string filename])
```

这个函数会将图像传送至客户端浏览器或保存在服务器中，参数 *filename* 省略不写的话，

表示要将图像传送至浏览器，此时，我们必须指定输出至浏览器的 MIME 类型，例如：

```
header("content-type: image/png");
imagepng($im);
```

若选择保存在服务器中，可以写成如下的形式：

```
imagepng($im, "images/simple.png");
```

❖ JPG 格式

```
imagejpeg(resource image[, string filename[, int quality]])
```

这个函数会将图像传送至客户端浏览器或保存在服务器中，参数 *filename* 省略不写的话，表示要将图像传送至浏览器，此时，我们必须指定输出至浏览器的 MIME 类型；参数 *quality* 用来设置图像质量，值为 0 ~ 100 的数字，默认值为 75，数字越大，质量就越好，文件也越大。

例如，下面的语句是将图像传送至浏览器，图像质量为 90：

```
header("content-type: image/jpeg");
imagejpeg($im, null, 90);
```

若选择保存在服务器中，且使用默认的图像质量（75），可以写成如下的形式：

```
imagejpeg($im, "images/simple.jpg");
```

若选择保存在服务器中，且使用最高图像质量，可以写成如下的形式：

```
imagejpeg($im, "images/simple.jpg", 100);
```

在此要告诉您一个小技巧，若是将图像直接输出至浏览器，那么其他网页可以使用 <img> 元素来显示图像，例如 <img src="GeneratePIC.php">，这样便能在别的网页中显示动态生成的图像。

6.1.5 释放内存

图像输出完毕后，我们必须使用 imagedestroy() 函数释放图像占用的内存空间，其语法如下：

```
imagedestroy(resource image)
```

随堂练习

使用 GD 图形处理函数库绘制如图 6-9 所示的图像并将之传送到浏览器，图像格式为 PNG。

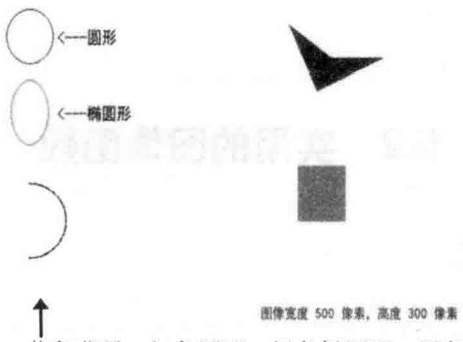


图 6-9

【解答】

\ch06\gd_sample.php

```
<?php
    $points = array(410, 50, 340, 80, 310, 20, 355, 50);
    $im = imagecreatetruecolor(500, 300);
    $background = imagecolorallocate($im, 255, 255, 180);
    $red = imagecolorallocate($im, 255, 0, 0);
    $green = imagecolorallocate($im, 0, 255, 0);
    $black = imagecolorallocate($im, 0, 0, 0);
    //填充背景颜色
    imagefill($im, 0, 0, $background);
    //绘制弧线
    imagearc($im, 30, 200, 80, 70, 270, 90, $black);
    //绘制椭圆形
    imageellipse($im, 30, 30, 50, 50, $red);
    imageellipse($im, 30, 100, 40, 60, $green);
    //填充矩形
    imagefilledrectangle($im, 320, 150, 370, 200, $green);
    //填充多边形颜色
    imagefilledpolygon($im, $points, 4, $red);
    //绘制文字
    $imInfo = "图像宽度 " . imagesx($im) ." 像素，高度 " . imagesy($im) . " 像素";
    imagettftext($im, 12, 0, 60, 37, $black, "simhei.ttf", "<---圆形");
```

```
    imagettftext($im, 12, 0, 60, 107, $black, "simhei.ttf", "<---椭圆形");
    imagettftext($im, 10, 0, 280, 290, $red, "simhei.ttf", $imInfo);
    //输出图像
    header("content-type: image/png");
    imagepng($im);
    //释放图像占用的内存
    imagedestroy($im);
?>
```

6.2 实用的图像函数

6.2.1 获取图像格式

exif_imagetype() 函数会通过文件的第一个位来获取图像格式（不是根据扩展名），其语法如下，参数 *filename* 是文件的相对地址或绝对地址（URL），返回值为 int 类型，对照表 6-1 即可知道图像格式（更多的返回值，可以查询 PHP 在线文件），若指定非图像的文件，则会返回 FALSE 并产生错误信息：

```
exif_imagetype(string filename)
```

表 6-1 返回值及对应的图像格式

返回值	常数	图像格式	MIME 类型
1	IMAGETYPE_GIF	GIF	image/gif
2	IMAGETYPE_JPEG	JPG	image/jpeg
3	IMAGETYPE_PNG	PNG	image/png
4	IMAGETYPE_SWF	SWF	application/x-shockwave-flash
5	IMAGETYPE_PSD	PSD	image/psd
6	IMAGETYPE_BMP	BMP	image/bmp
7	IMAGETYPE_TIFF_II	字节顺序为 IBM PC 的 TIFF	image/tiff
8	IMAGETYPE_TIFF_MM	字节顺序为 Mac 的 TIFF	image/tiff
9	IMAGETYPE_JPC	JPC	application/octet-stream
10	IMAGETYPE_JP2	JP2	image/jp2
11	IMAGETYPE_JPX	JPX	application/octet-stream
12	IMAGETYPE_JB2	JB2	application/octet-stream
13	IMAGETYPE_SWC	SWC	application/x-shockwave-flash
14	IMAGETYPE_IFF	IFF	image/iff
15	IMAGETYPE_WBMP	WBMP	image/vnd.wap.wbmp
16	IMAGETYPE_XBM	XBM	image/xbm

下面是一个例子 <\ch06\exif_imagetype.php>，它会获取文件的图像格式，如图 6-10 所示。

```
<?php
   $imageType = exif_imagetype("images/pic1.jpg");
   if (!$imageType)
      echo "这个文件不是图像";
   elseif ($imageType == IMAGETYPE_JPEG)
      echo "这张图像的格式为 JPG";
?>
```

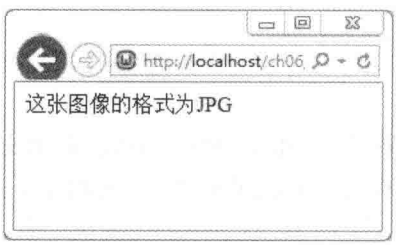


图 6-10

6.2.2　获取图像的大小与格式

getimagesize() 函数可以用来获取图像大小与格式，其语法如下，参数 *filename* 是文件的相对地址或绝对地址（URL），返回值为包含 4 个元素的数组，意义如下，若指定的文件不是图像，则会返回 FALSE 并产生错误信息：

```
getimagesize(string filename)
```

- 第 1 个元素代表图像宽度，单位为像素（pixel）。
- 第 2 个元素代表图像高度，单位为像素（pixel）。
- 第 3 个元素代表图像格式，其返回值的意义与 exif_imagetype() 函数相同。若只是要获取图像格式，则 exif_imagetype() 函数的执行效率较佳。
- 第 4 个元素会以 width="*xxx*" height="*xxx*" 的形式描述图像的宽度和高度，也就是它的返回值可以直接拿来定义 <img> 元素指定的图像的宽度和高度，例如 <img width="*xxx*" height="*xxx*">。

下面是一个例子 <\ch06\getimagesize.php>，它会获取文件的图像大小与格式，如图 6-11 所示。

```
<?php
   $size = getimagesize("images/pic1.jpg");
   echo "图像宽度：$size[0]<br>";
   echo "图像高度：$size[1]<br>";
   echo "图像格式：$size[2]<br>";
   echo "图像长宽字符串：$size[3]";
?>
```

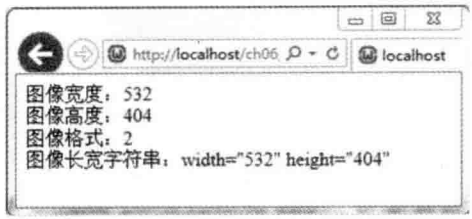


图 6-11

6.2.3 读取外部图像

我们可以分别使用下列函数读取 GIF、JPG 及 PNG 图像文件，参数 *filename* 表示要读取的图像路径及文件名，可以是相对地址或绝对地址（URL），返回值为代表该图像的图像标识符（image identifier）：

```
imagecreatefromgif(string filename)
imagecreatefromjpeg(string filename)
imagecreatefrompng(string filename)
```

下面是一个例子，它会读取图像，然后加入文字，再输出至浏览器，如图 6-12 所示。

图 6-12

\ch06\imagecreatefromjpeg.php

```
<?php
  //读取外部图像
  $im = imagecreatefromjpeg("images/pic1.jpg");
  $textcolor = imagecolorallocate($im, , 0, 255);
  //在图像中加入拍摄者信息
  imagestring($im, 20, 380, 350, "Photo By Jean", $textcolor);
  //输出图像
  header("content-type: image/png");
  imagepng($im);
```

```
    //释放图像占用的内存
    imagedestroy($im);
?>
```

一、选择题

（　）1. 下列哪个函数可以用来创建支持百万色的空白图像？

A. imageallocate()　　B. graphiccreate()

C. imagecreate()　　D. imagecreatetruecolor()

（　）2. 下列哪个函数可以用来分配图像使用的颜色？

A. imagecreate()　　B. imageellipse()

C. imagefill()　　D. imagecolorallocate()

（　）3. 下列哪个函数可以用来绘制圆形？

A. imagecreate()　　B. imageellipse()

C. imagefill()　　D. imagearc()

（　）4. 若要填充矩形颜色，可以使用下列哪个函数？

A. imagefill()　　B. imagearc()

C. imagechar()　　D. imagefilledrectangle()

（　）5. 若要将图像输出成 PNG 格式的图像，可以使用下列哪个函数？

A. exportgif()　　B. exportpng()

C. imagegif()　　D. imagepng()

二、实践实验题

编写一个 PHP 网页，令它绘制如下图形（如图 6-13 所示），包括红色弧线、红色圆形、绿色椭圆形、填充绿色多边形及填充红色矩形。<\ch06\gd_sample2.php>

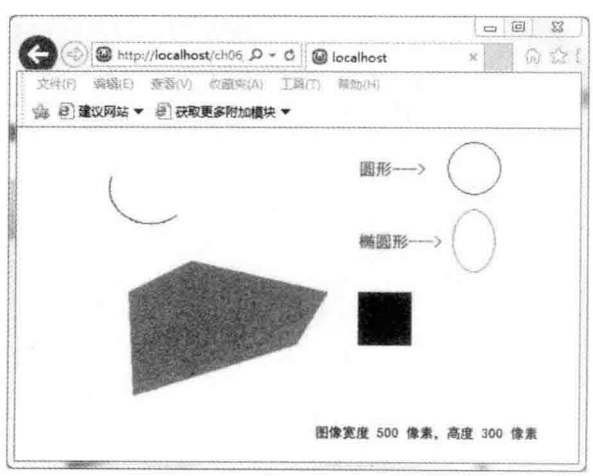


图 6-13

第 7 章
面向对象

7.1　认识面向对象

“面向对象”(Object Oriented，OO）是软件开发过程中极具影响性的突破，越来越多程序设计语言强调其面向对象的特性，PHP 也不例外。

面向对象的优点是对象可以在不同的应用程序中被重复使用，Windows 本身就是一个面向对象的例子，您在 Windows 操作系统中所看到的东西，包括窗口、按钮、对话框、菜单、滚动条、表单、控件、数据库等，均属于对象，您可以将这些对象放进自己编写的程序，然后根据实际情况改变对象的字段或属性（例如标题栏的文字、按钮的大小、对话框的大小与类型等），而不必再为这些对象编写冗长的程序代码。

下面我们来解释几个相关的名词。

- “对象”（object）或“实例”（instance）就像在日常生活中所看到的各种物体，例如房子、计算机、冰箱、汽车、电视等，而对象可能又由许多子对象所组成，比方说，计算机是一种对象，而计算机又是由硬盘、CPU、主板等子对象所组成的；又比方说，Windows 操作系统中的窗口是一种对象，而窗口又是由标题栏、菜单栏、工具栏等子对象所组成的。在 PHP 中，对象是数据与程序代码的组合，它可以是整个应用程序或应用程序的一部分。
- “属性”（property）、“字段”(field）或“成员变量”（member variable）用来描述对象的特征，比方说，计算机是一种对象，而计算机的 CPU 等级、制造厂商等用来描述计算机的特征就是这个对象的属性；又比方说，Windows 操作系统中的窗口是一种对象，而它的大小、位置等用来描述窗口的特征就是这个对象的属性。
- “方法”（method）或“成员函数”（member function）是用来执行对象的动作，比方说，计算机是一种对象，而开机、关机、执行应用程序等动作就是这个对象的方法，如图 7-1 所示。

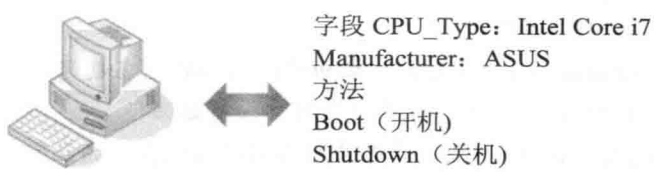


图 7-1

- “事件”（event）是在某些情况下发出特定信号的警告，比方说，假设您有一辆汽车，当您发动汽车却没有关好车门时，汽车会发出哔哔声的警告，这就是一种事件；又比方说，当浏览者单击网页上的按钮时，就会产生一个 onclick 事件，然后我们可以针对这个事件编写处理程序，例如将浏览者输入的数据进行运算、写入数据库或文件、返回 Web 服务器等。

- “类”（class）是对象的分类，就像对象的蓝图，隶属于相同类的对象具有相同的字段、属性、方法及事件，但字段或属性的值则不一定相同。比方说，假设汽车是一种类，它有品牌、颜色、型号等字段及开门、关门、发动等方法，那么一辆白色 BMW 520 汽车就是隶属于汽车类的一个对象或实例，其品牌字段的值为 BMW，颜色字段的值为白色，型号字段的值为 520，而且除了这些字段之外，它还有开门、关门、发动等方法，至于其他车种（例如 BENZ），则为汽车类的其他对象或实例，如图 7-2 所示。

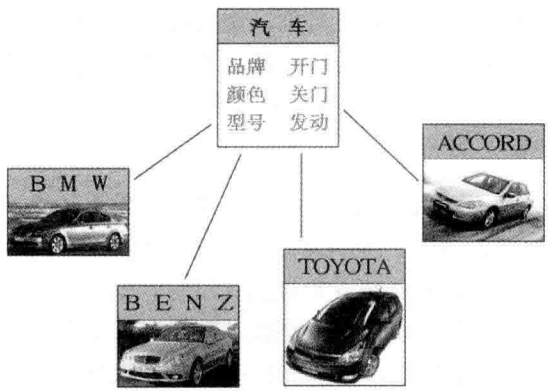


图 7-2

7.2 类与对象

类（class）就像对象（object）的蓝图，PHP 的类可以包含下列成员。

- 属性（又称为字段或成员变量）：用来存放数据的变量。
- 方法（又称为成员函数）：将具有某种功能的语句片段写成独立的程序单元，然后给予特定名称，其实就是类内的函数。
- 常数：用来存放数据的常数。
- 构造函数（constructor）：用来将对象初始化的函数，在创建对象时会自动执行，无须在程序代码内加以调用（有无参数皆可、没有返回值）。
- 析构函数（destructor）：用来释放对象所占用的系统资源的函数，在释放对象时会自动执行，无须在程序代码内加以调用（没有参数、没有返回值）。

7.2.1 定义类

我们可以使用 class 关键字定义类，其语法如下：

```
class class_name [extends parentclass_name]
{
  [public|private|protected|var $property_name [= value];]          //定义属性
```

```
    [[public|private|protected] function method_name(...){...}        //定义方法
    [...]                                                            //定义其他成员
}
```

- class：这个关键字用来表示要定义类。
- *class_name*：这是类的名称，命名规则与变量相同。
- [extends *parentclass_name*]：若类继承自其他类，可以在关键字 extends 的后面加上其他类的名称，否则省略不写。
- {、}：分别标记类的开头与结尾。
- public|private|protected|var $*property_name*[= *value*];：在类内定义属性其实和我们平常定义变量差不多，不同的是前面必须加上下列关键字，以指定属性的“访问级别”（access level）。
  - public：以这个关键字定义的成员能够被任何程序代码访问。
  - private：以这个关键字定义的成员只能被包含其定义的类访问。
  - protected：以这个关键字定义的成员只能被包含其定义的类或其子类访问。
  - var：以这个关键字定义的成员能够被任何程序代码访问（public 的别名）。

至于是否要赋初始值，则视实际情况而定，例如下面的语句是在类内定义一个名称为 Name、初始值为 '小丸子' 的属性：

```
public $Name = '小丸子';                    //定义名称为 Name、初始值为 '小丸子' 的属性
```

- [public|private|protected]function *method_name*(...){...}：在类内定义方法就和我们平常定义函数一样，若 public、private、protected 省略不写，表示为 public，例如下面的语句是在类内定义一个名称为 ShowName、没有参数、没有返回值的方法：

```
class Employee                              //定义名称为 Employee 的类
{
   public $Name = '小丸子';                 //定义名称为 Name、初始值为 '小丸子' 的属性
   public function ShowName()               //定义名称为 ShowName、没有参数、没有返回值的方法
   {
      echo '这名员工的名字为'.$this->Name;
   }
}
```

注意 Name 的前面没有$符号

在上面的程序代码中，由于我们想在方法内访问相同类所定义的属性 Name，故使用了特殊的变量 $this 和 -> 运算符，变量 $this 指的是对象本身，而 -> 运算符可以用来访问对象的成员。

7.2.2 创建对象

原则上，类属于“引用类型”（reference type），无法直接访问，必须先使用 new 关键字创建类的对象，才能访问对象的成员，例如：

```
$Obj = new Employee();                    //创建名称为 Obj、隶属于 Employee 类的对象
```

成功创建类的对象后，就可以使用运算符 -> 访问对象的成员了，例如：

```
$Obj->Name = '花轮';                      //将对象的属性 Name 的值变更为 '花轮'
$Obj->ShowName();                         //调用对象的方法 ShowName()
```

我们可以将这些语句整合到下面的例子中，其中变量 $this 指的是对象本身，程序运行结果如图 7-3 所示。

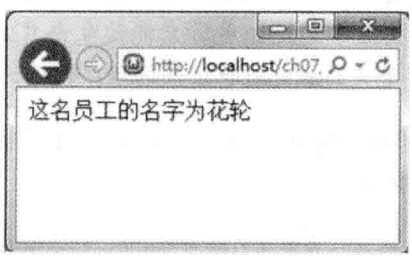


图 7-3

\ch07\oop1.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <?php
      class Employee
      {
        public $Name = '小丸子';
        public function ShowName()
        {
          echo '这名员工的名字为'.$this->Name;
        }
      }
      $Obj = new Employee();
      $Obj->Name = '花轮';
      $Obj->ShowName();
```

```
    ?>
  </body>
</html>
```

7.2.3　static 关键字

在前一节中，我们先创建类的对象，然后通过对象和 -> 运算符访问属性或方法等成员，但事实上，有些类可能纯粹是要提供一般用途的成员让用户访问。以下面的程序代码为例，MyMath 类提供了 Cubic() 方法让用户计算三次方，为了方便使用，于是使用 static 关键字将该方法定义为静态方法，如此一来，用户就可以通过类的名称和 :: 运算符进行调用，而不必创建类的对象，程序运行结果如图 7-4 所示。

\ch07\oop2.php

```
01:<!doctype html>
02:<html>
03:   <head>
04:     <meta charset="utf-8">
05:   </head>
06:   <body>
07:     <?php
08:       class MyMath
09:       {
10:         public static function Cubic($X)
11:         {
12:           return $X * $X * $X;
13:         }
14:       }
15:       echo '5 的三次方为'.MyMath::Cubic('5');
16:     ?>
17:   </body>
18:</html>
```

由于没有创建对象，故不能使用 -> 运算符，而是改用 :: 运算符。

- 08 ~ 14：在 MyMath 类内使用 static 关键字定义一个名称为 Cubic 的静态方法，其返回值为参数的三次方。
- 15：通过类的名称和 :: 运算符调用 MyMath 类内的静态方法 Cubic()，以计算 5 的三次方，然后将结果显示在网页上。请注意，:: 运算符（Scope Resolution Operator）不仅能用来访问类内的静态成员，还能用来访问类内的常数（constant）或被覆盖的成员（overridden member）。

图 7-4

7.2.4 类常数

我们也可以在类内定义“常数”（constant），但和定义一般常数不同的是必须改用 const 关键字，而且当我们要访问类内的常数时，只能通过类的名称和 ::运算符，不能通过对象。

下面是一个例子，要特别提醒，常数的名称前面没有 $ 符号，而且常数的值必须是一个常数表达式，不能是变量、属性、数学运算的结果或函数调用，下面程序的运行结果如图 7-5 所示。

\ch07\oop3.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <?php
      class Circle
      {                    定义常数 PI (前面没有$符号)
        const PI = 3.14;
        public $Radius;
        public function ShowArea()
        {                                          self 关键词代表目前类
          echo '圆面积为'.($this->Radius * $this->Radius * self::PI);
        }
      }
            通过类的名称和::运算符访问常数
      echo '圆周率为'.Circle::PI.'<br>';
      $Obj = new Circle();
      $Obj->Radius = 10;
      $Obj->ShowArea();
    ?>
  </body>
```

```
</html>
```

图 7-5

7.2.5 构造函数

构造函数（constructor）是用来将对象初始化的函数，在创建对象时会自动执行，常见的初始化动作有设置初始值、打开文件、建立数据库连接、建立网络连接等。PHP 支持的构造函数名称为__construct（注意前面为两个下划线），有无参数皆可，没有返回值，下面是一个例子，其运行结果如图 7-6 所示。

\ch07\oop4.php

```
01:<!doctype html>
02:<html>
03:   <head><meta charset="utf-8"></head>
04:   <body>
05:      <?php
06:         class Employee
07:         {
08:            public $Name;                          //定义名称为 Name 的属性以存放员工的名字
09:            function __construct($Str)             //通过构造函数为员工的名字赋值并显示说明信息
10:            {
11:               $this->Name = $Str;
12:               echo '已经创建名字为'.$this->Name.'的对象！ ';
13:            }
14:         }
15:         $Obj = new Employee('小红豆');
16:      ?>                    此处须传入参数给建构函数
17:   </body>
18:</html>
```

- 09 ~ 13：定义一个构造函数，该函数有一个参数，没有返回值，它会在创建对象时自动执行，将参数的值赋值给属性 Name（即设置属性 Name 的初始值），然后显示说明信息。
- 15：创建一个隶属于 Employee 类的对象，此时会自动执行构造函数。

图 7-6

7.2.6 析构函数

析构函数（destructor）是用来释放对象所占用的系统资源的函数，在释放对象时会自动执行，无须在程序代码内加以调用，常见的释放动作有清除设置值、关闭文件、结束数据库连接、中断网络连接等。PHP 支持的析构函数名称为__destruct（注意前面为两个下划线），没有参数，也没有返回值，下面是一个例子。

\ch07\oop5.php

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:  </head>
06:  <body>
07:    <?php
08:      class Employee
09:      {
10:        public $Name;                          //定义名称为 Name 的属性以存放员工的名字
11:        function __construct($Str)             //通过构造函数赋值员工的名字并显示说明信息
12:        {
13:          $this->Name = $Str;
14:          echo '已经创建名字为'.$this->Name.'的对象！ ';
15:        }
16:
17:        function __destruct()                  //通过析构函数清除员工的名字并显示说明信息
18:        {
19:          $this->Name = NULL;
20:          echo '这个对象已经被释放！ ';
21:        }
22       }
23:
24:      $Obj = new Employee('小红豆'); //创建对象（会自动执行构造函数)
25:      $Obj = NULL;                             //释放对象（会自动执行析构函数)
```

```
26:     ?>
27:   </body>
28:</html>
```

这个网页的执行结果如图 7-7 所示：

图 7-7

- 17 ~ 21：定义一个析构函数，该函数没有参数，也没有返回值，它会在释放对象时自动执行，将属性 Name 的值设置为 NULL（即清除属性 Name 的值），然后显示“这个对象已经被释放!”。
- 24：创建一个名称为 Obj、隶属于 Employee 类的对象，此时会自动执行构造函数，将属性 Name 的值设置为参数 '小红豆'，然后显示“已经创建名字为小红豆的对象!”。
- 25：将对象 Obj 设置为 NULL 的意义就是释放对象，此时会自动执行析构函数，清除属性 Name 的值，然后显示“这个对象已经被释放!”。

7.2.7　比较对象

我们可以使用下列两个运算符比较对象。

- ==：当两个对象隶属于相同类且有相同属性与值时，会返回 TRUE。
- ===：当两个对象指向相同类的相同实例时，会返回 TRUE。

下面是一个例子，其中 $Obj1 和 $Obj2 隶属于相同类且具有相同的属性与值，故第 19 行使用 == 运算符比较的结果会返回 TRUE，而显示“$Obj2 的成员与值均和 $Obj1 相同”，不过，由于 $Obj1 和 $Obj2 指向不同的实例，故第 21 行使用 === 运算符比较的结果会返回 FALSE，而显示“$Obj2 和 $Obj1 指向不同的实例”；相反的，由于第 18 行使 $Obj3 指向 $Obj1 所指向的实例，故第 23 行使用 === 运算符比较的结果会返回 TRUE，而显示“$Obj3 和 $Obj1 指向相同的实例”，程序的运行结果如图 7-8 所示。

\ch07\oop6.php

```
01:<!doctype html>
02:<html>
03:   <head>
04:     <meta charset="utf-8">
```

```
05:  </head>
06:  <body>
07:    <?php
08:      class Employee
09:      {
10:        public $Name;                    //定义名称为 Name 的属性以存放员工的名字
11:        function __construct($Str)       //通过构造函数赋值员工的名字
12:        {
13:          $this->Name = $Str;
14:        }
15:      }
16:      $Obj1 = new Employee('小红豆');     //令$Obj1 指向 Name 属性为 '小红豆' 的实例
17:      $Obj2 = new Employee('小红豆');     //令$Obj2 指向 Name 属性为 '小红豆' 的另一实例
18:      $Obj3 = $Obj1;                     //令$Obj3 指向$Obj1 所指向的实例
19:      if（$Obj2 == $Obj1) echo '$Obj2 的成员与值均和$Obj1 相同'.'<br>';
20:      else echo '$Obj2 的成员或值和$Obj1 不同'.'<br>';
21:      if（$Obj2 === $Obj1) echo '$Obj2 和$Obj1 指向相同的实例'.'<br>';
22:      else echo '$Obj2 和$Obj1 指向不同的实例'.'<br>';
23:      if（$Obj3 === $Obj1) echo '$Obj3 和$Obj1 指向相同的实例'.'<br>';
24:      else echo '$Obj3 和$Obj1 指向不同的实例'.'<br>';
25:    ?>
26:  </body>
27:</html>
```

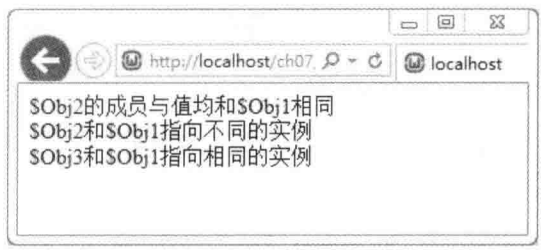


图 7-8

7.3 继承

“继承”（inheritance）是面向对象程序设计中非常重要的一环，所谓继承就是从现有的类创建新的类，这个现有的类叫做“基类”（base class），由于是用来作为基础的类，故又称为“父类”（parent class、superclass），而这个新的类则叫做“派生类”（derived class），由于是继承自基类，故又称为“子类”（child class、subclass）或“扩充类”（extended class），类的继承关系示意图如图 7-9 所示。

子类不仅继承了父类的非私有成员，还可以加入新的成员或“覆盖”（override）继承自父类的方法，也就是将继承自父类的方法重新定义，而且在这个过程中，父类的方法并不会

受到影响。继承的优点是父类的程序代码只要编写与排错一次，就可以在其子类重复使用，如此一来，不仅节省时间与开发成本，也提高了程序的可靠性，有助于原始问题的概念化。

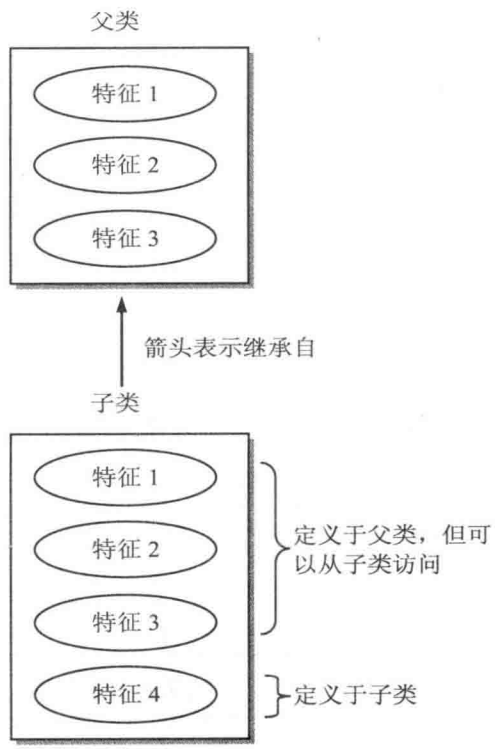


图 7-9

7.3.1　定义子类

定义子类其实和定义一般类差不多，不同的是要在子类的名称后面加上 extends 关键字并指定父类的名称，其语法如下：

```
class childclass_name extends parentclass_name
{
  [...]
}
```

子类的特色是继承了父类的非私有成员，同时还可以加入新的成员，或“覆盖”（override）继承自父类的方法。下面是一个例子，它所呈现的是如图 7-10 所示的继承关系，也就是链状继承。

\ch07\oop7.php

```
<!doctype html>
```

```
<html>
  <head><meta charset="utf-8"></head>
  <body>
    <?php
      class A
      {
        //...
      }
      class B extends A
      {
        //...
      }
      class C extends B
      {
        //...
      }
    ?>
  </body>
</html>
```

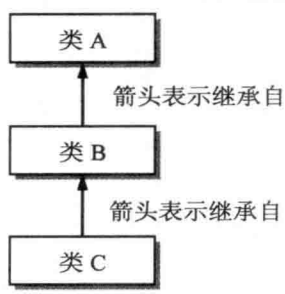


图 7-10

下面是另一个例子，它所呈现的是如图 7-11 所示的继承关系，也就是一个父类可以有多个子类。

\ch07\oop8.php

```
<!doctype html>
<html>
  <head><meta charset="utf-8"></head>
  <body>
    <?php
      class W
      {
        //...
      }
```

```
    class X extends W
    {
      //...
    }
    class Y extends W
    {
      //...
    }
    class Z extends W
    {
      //...
    }
  ?>
 </body>
</html>
```

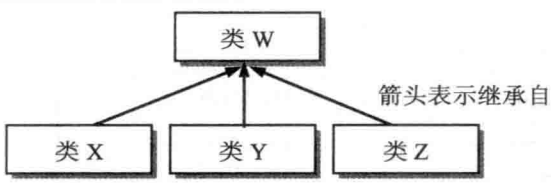


图 7-11

7.3.2　设置成员的访问级别

我们可以使用下列访问修饰关键字设置成员的访问级别，以及能够被继承与否，若在定义成员时省略了访问修饰关键字或使用 var 关键字，表示为 public。

- public：以这个关键字定义的成员能够被任何程序代码访问，包括被继承。
- private：以这个关键字定义的成员只能被包含其定义的类访问，不能被继承。
- protected：以这个关键字定义的成员只能被包含其定义的类或其子类访问，包括被继承。

现在，我们将使用 PHP 语法表示如图 7-12 所示的继承关系。

\ch07\oop9.php

```
01:<?php
02:   class ParentClass                          //定义父类
03:   {
04:      public $Field1;                        //定义 public 属性（能够被继承）
05:      private $Field2;                       //定义 private 属性（不能被继承）
06:      protected $Field3;                     //定义 protected 属性（能够被继承）
```

```
07:     public function Method1(){}                    //定义 public 方法（能够被继承）
08:     private function Method2(){}                   //定义 private 方法（不能被继承）
09:     protected function Method3(){}                 //定义 protected 方法（能够被继承）
10:  }
11:
12:  class ChildClass extends ParentClass              //定义子类
13:  {
14:     public $Field4;                                 //定义 public 属性（能够被继承）
15:     private $Field5;                                //定义 private 属性（不能被继承）
16:     protected $Field6;                              //定义 protected 属性（能够被继承）
17:     public function Method4(){}                    //定义 public 方法（能够被继承）
18:     private function Method5(){}                   //定义 private 方法（不能被继承）
19:     protected function Method6(){}                 //定义 protected 方法（能够被继承）
20:  }
21:?>
```

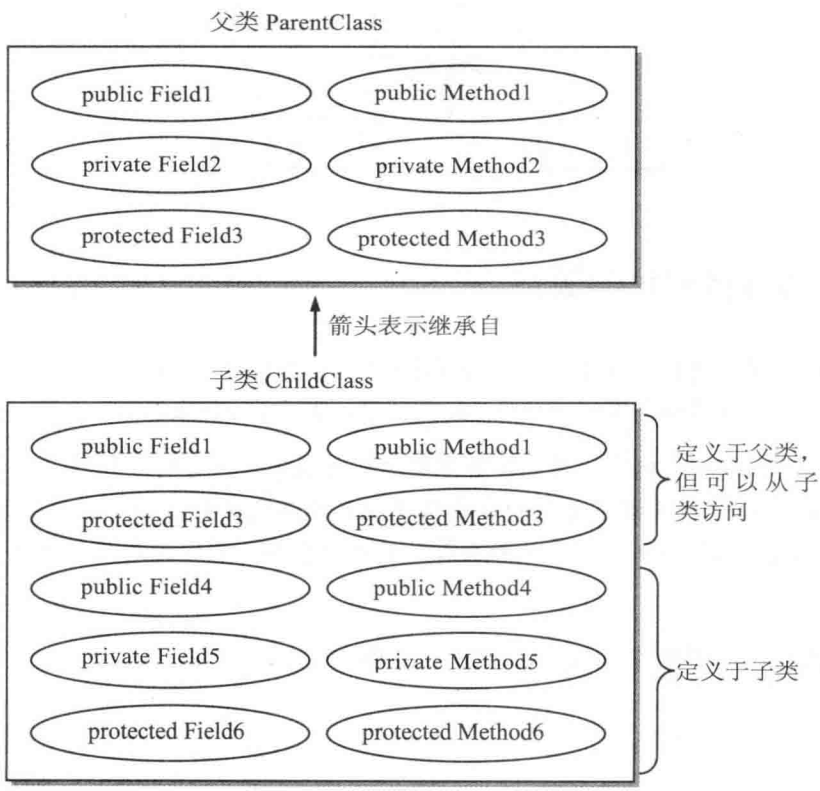


图 7-12

- 02 ~ 10：定义一个名称为 ParentClass 的类，里面有三个属性 Field1、Field2、Field3 和三个方法 Method1()、Method2()、Method3()，访问级别分别为 public、private、

protected、public、private、protected，由于只有非私有成员才能被继承，因此，只有 Field1、Field3、Method1()、Method3() 能够被其子类继承，同时 Field1 和 Method1() 能够被所有程序代码访问，Field2、Method2() 只能被相同类内的程序代码访问，而 Field3、Method3() 能够被相同类和其子类内的程序代码访问。

- 12 ~ 20：定义一个名称为 ChildClass 的类，而且这个类继承自 ParentClass 类，因此，除了第 14 ~ 19 行所定义的三个属性 Field4、Field5、Field6 和三个方法 Method4()、Method5()、Method6() 之外，它还继承了来自 ParentClass 类的非私有成员 Field1、Field3、Method1()、Method3()。

最后要说明的是 ParentClass 类外的程序代码和 ChildClass 类外的程序代码能够访问哪些成员，基本上，它们只能访问 public 成员，也就是 Field1、Method1()、Field4、Method4()，若访问了其他成员，就会产生错误信息，例如 ChildClass::Method2(); 将会产生类似 "Fatal error: Call to private method ParentClass::method2() from context" 的错误信息。

父类内定义为 private 的成员只能被父类内的程序代码访问，其他在父类外的程序代码 (包括子类) 均不得访问，对于安全性较高、不允许父类外的程序代码访问的成员就必须定义为 private；相反的，父类内定义为 protected 的成员则能够被父类及其子类内的程序代码访问，所以在使用继承的同时，您必须考虑清楚是否允许使用者通过继承的方式访问某些成员，如果是的话，才要将这些成员定义为 protected。

虽然 protected 赋予了子类访问某些成员的弹性，同时也适度保护了这些成员，毕竟子类外的程序代码无法加以访问，但这中间其实还是存在着潜伏的危险，因为有心人士可能会通过继承的方式随意篡改父类的 protected 成员，影响程序的运行，所以在定义父类的成员访问级别时应该要仔细思考。

父类别与子类别构成了所谓“类层次结构”(class hierarchy)，类层次结构的实现相当简单，重点在于如何设计类层次结构，原则上，类层次结构由上到下的定义应该是由广义进入狭义，比方说，父类别 Employee 泛指公司员工，而其子类别 Managers、Asistants 则分别表示经理级的员工和助理性质的员工。

7.3.3　覆盖继承自父类的方法

“覆盖”(override）指的是子类将继承自父类的方法重新定义，而且在这个过程中，父类的方法并不会受到影响。我们通常通过覆盖的技巧来实现面向对象程序设计的“多态”

（polymorphism），有关多态的进一步讨论请参阅相关书籍。

以下面的程序代码为例，父类 Payroll 的 Payment() 方法会根据小时数及钟点费算出工资（第 05 ~ 08 行），而子类 BonusPayroll 覆盖继承自父类的 Payment() 方法，令它除了根据小时数及钟点费算出工资之外，还会加上奖金 5000（第 12 ~ 15 行），该程序的运行结果如图 7-13 所示。

\ch07\oop10.php

```
01:<?php
02:   class Payroll                                          //定义父类
03:   {
04:      public $Name;                                      //定义属性（能够被继承）
05:      public function Payment($Hours, $PayRate)          //定义方法（能够被继承）
06:      {
07:         return $Hours * $PayRate;
08:      }
09:   }
10:   class BonusPayroll extends Payroll                    //定义子类
11:   {
12:      public function Payment($Hours, $PayRate)          //覆盖父类的方法
13:      {
14:         return $Hours * $PayRate + 5000;
15:      }
16:   }
17:   $Obj1 = new Payroll();
18:   $Obj2 = new BonusPayroll();
19:   echo '尚未加上奖金的工资为'.$Obj1->Payment(100, 80).'<br>';
20:   echo '加上奖金之后的工资为'.$Obj2->Payment(100, 80).'<br>';
21:?>
```

图 7-13

7.3.4 调用父类内被覆盖的方法

在本节中，我们将告诉您一个实用的小技巧，也就是子类如何调用父类内被覆盖的方法，以前一节的 <\ch07\oop10.php> 为例，由于子类在重新定义 Payment() 方法时有部分语句和

父类的 Payment() 方法相同，因此，我们可以调用父类的 Payment() 方法来取代，如下所示：

```
public function Payment($Hours, $PayRate)
{
   return $Hours * $PayRate + 5000;
}
```

这些语句和父类的Payment()方法相同

```
public function Payment($Hours, $PayRate)
{
   return parent::Payment($Hours, $PayRate) + 5000;
}
```

parent 关键词代表目前所在的子类的父类，通过这个关键字，就可以调用父类内被覆盖的方法，当然您也可以直接指定父类的名称，如下所示：

```
public function Payment($Hours, $PayRate)
{
   return Payroll::Payment($Hours, $PayRate) + 5000;
}
```

原则上，子类可以覆盖继承自父类的任何方法，但有时我们可能希望父类的某个方法不要被子类所覆盖，此时可以在父类定义该方法时加上 final 关键字，例如下面的语句将禁止子类覆盖父类的 Payment() 方法：

```
final public function Payment($Hours, $PayRate)
```

7.3.5　抽象方法

“抽象方法”（abstract method）是一种特殊的方法，它必须放在“抽象类”（abstract class）内，只有定义的部分，没有实现的部分，而且实现的部分必须由子类提供。至于抽象类则是一种特殊的类，只有类的定义和部分实现，必须通过子类来实现或扩充其功能，同时程序设计人员不可以创建其对象，换句话说，抽象类只能被继承，不能被实例化（instantiation）。

以下面的程序代码为例，父类 Payroll 是一个抽象类，里面有一个属性 Name 和一个抽象方法 Payment()，它的实现部分由子类 BonusPayroll 提供，该程序的运行结果如图 7-14 所示。

\ch07\oop11.php

```
<?php
   abstract class Payroll
   {
```

此语句的开头要有 abstract，因为抽象方法必须放在抽象类内

```
    public $Name;
    abstract public function Payment($Hours, $PayRate);    ←定义抽象方法
}
class BonusPayroll extends Payroll
{
    public function Payment($Hours, $PayRate)
    {                                                      覆盖抽象方法
        return $Hours * $PayRate + 5000;                   (参数个数必须相同)
    }
}
$Obj = new BonusPayroll();
echo '加上奖金之后的工资为'.$Obj->Payment(100, 80).'<br>';
?>
```

图 7-14

7.3.6 子类的构造函数与析构函数

原则上，子类会继承父类的构造函数与析构函数，若子类没有定义自己的构造函数与析构函数，一旦创建隶属于子类的对象或释放隶属于子类的对象，就会分别自动执行父类的构造函数与析构函数，否则会分别自动执行子类的构造函数与析构函数。

下面是一个例子，父类 ParentClass 定义了一个属性 Field1（第 10 行）、构造函数（第 11 ~ 15 行）与析构函数（第 17 ~ 21 行），而子类 ChildClass 只定义了一个属性 Field2（第 26 行）。

由于子类没有定义自己的构造函数与析构函数，因此，在第 29 行创建隶属于子类的对象时，会自动执行父类的构造函数，也就是赋给属性 Field1 的值为传入的参数，然后显示“创建对象时成功将 Field1 的值设置为 100”，而在第 30 行释放隶属于子类的对象时，会自动执行父类的析构函数，也就是赋值属性 Field1 的值为 0，然后显示“释放对象时成功将 Field1 的值设置为 0”，该程序的运行结果如图 7-15 所示。

\ch07\oop12.php

```
01:<!doctype html>
02:<html>
03:   <head>
```

```
04:    <meta charset="utf-8">
05:  </head>
06:  <body>
07:    <?php
08:      class ParentClass                                    //定义父类
09:      {
10:        protected $Field1;                                 //定义父类的属性
11:        function __construct($Value)                      //定义父类的构造函数
12:        {
13:          $this->Field1 = $Value;
14:          echo '创建对象时成功将 Field1 的值设置为'.$this->Field1.'<br>';
15:        }
16:
17:        function __destruct()                             //定义父类的析构函数
18:        {
19:          $this->Field1 = 0;
20:          echo '释放对象时成功将 Field1 的值设置为'.$this->Field1.'<br>';
21:        }
22:      }
23:
24:      class ChildClass extends ParentClass                //定义子类
25:      {
26:        protected $Field2;                                 //定义子类的属性
27:      }
28:
29:      $MyObject = new ChildClass(100);                    //会自动执行父类的构造函数
30:      $MyObject = NULL;                                   //会自动执行父类的析构函数
31:    ?>
32:  </body>
33:</html>
```

图 7-15

下面是另一个例子，同样，父类 ParentClass 定义了一个属性 Field1（第 10 行）、构造函数（第 11 ~ 15 行）与析构函数（第 16 ~ 20 行），而这次子类 ChildClass 不仅定义了一个属性 Field2（第 24 行），还定义了自己的构造函数（第 25 ~ 29 行）与析构函数（第 30~ 34 行）。

由于子类已经定义自己的构造函数与析构函数，因此，在第 37 行建立隶属于子类的对象时，会自动执行子类的构造函数，也就是赋给属性 Field2 的值为传入的参数，然后显示“创建对象时成功将 Field2 的值设置为 100”，注意是 Field2，不是 Field1。

而在第 38 行释放隶属于子类的对象时，会自动执行子类的析构函数，也就是赋给属性 Field2 的值为 0，然后显示“释放对象时成功将 Field2 的值设置为 0”，注意是 Field2，不是 Field1，该程序的运行结果如图 7-16 所示。

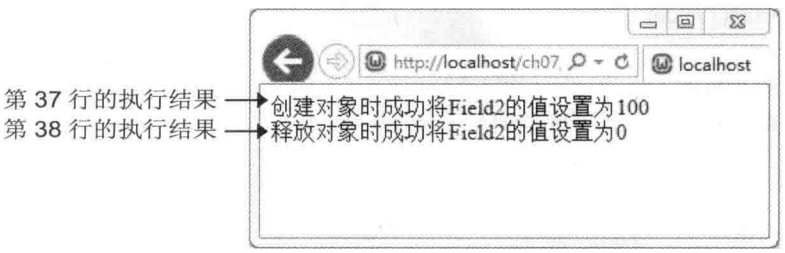


图 7-16

\ch07\oop13.php

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:  </head>
06:  <body>
07:    <?php
08:      class ParentClass                                    //定义父类
09:      {
10:        protected $Field1;                                 //定义父类的属性
11:        function __construct($Value)                       //定义父类的构造函数
12:        {
13:          $this->Field1 = $Value;
14:          echo '创建对象时成功将 Field1 的值设置为'.$this->Field1.'<br>';
15:        }
16:        function __destruct()                              //定义父类的析构函数
17:        {
18:          $this->Field1 = 0;
19:          echo '释放对象时成功将 Field1 的值设置为'.$this->Field1.'<br>';
20:        }
21:      }
22:      class ChildClass extends ParentClass                 //定义子类
23:      {
24:        protected $Field2;                                 //定义子类的属性
25:        function __construct($Value)                       //定义子类的构造函数
26:        {
```

```
27:         $this->Field2 = $Value;
28:         echo '创建对象时成功将 Field2 的值设置为'.$this->Field2.'<br>';
29:       }
30:       function __destruct()                          //定义子类的析构函数
31:       {
32:         $this->Field2 = 0;
33:         echo '释放对象时成功将 Field2 的值设置为'.$this->Field2.'<br>';
34:       }
35:     }
36:
37:     $MyObject = new ChildClass(100);                 //会自动执行子类的构造函数
38:     $MyObject = NULL;                                //会自动执行子类的构造函数
39:   ?>
40: </body>
41:</html>
```

最后要告诉您一个技巧，若要在子类调用父类的构造函数或析构函数，可以使用如下语句，其中构造函数的参数取决于实际情况，而析构函数则没有参数：

```
parent::__construct(参数);
parent::__destruct();
```

7.4　命名空间

“命名空间”（namespace）是一种命名方式，用来组织各个类、函数、常数等，它和这些元素的关系就像文件系统中目录与文件的关系一样，举例来说，假设MyClass隶属于 \A\B\C 命名空间，那么若要建立一个名称为 Obj、隶属于 MyClass 的对象，可以写成如下的形式：

```
$Obj = new \A\B\C\MyClass;
```

其中第一个反斜杠（\）表示“全局空间”（global space），就像文件系统中的根目录一样，在 PHP 开始支持命名空间之前，或当 PHP 程序没有定义任何命名空间时，所有类、函数、常数等都是放在全局空间的，至于其他反斜杠（\）则用来连接命名空间内所包含的子命名空间、类、函数、常数等，例如此处的 A、B、C 均为子命名空间。

PHP 之所以支持命名空间，目的如下。

- 解决名称冲突的问题：当您自己编写的 PHP 程序和 PHP 内置或其他人编写的类、函数、常数发生名称冲突时，可以利用命名空间来解决。
- 提供设置别名的功能：当 PHP 程序里面的类、函数或常数的名称太长或不易理解时，可以利用命名空间来设置简短易读的别名（alias）。

事实上，若您的程序并没有遇到上述的问题，那么您可以不用理会命名空间的概念，因

为程序依然能够正常执行，以下面两段程序代码为例，其功能是相同的，只是第二段程序代码加入了命名空间的概念。

```
<?php
$Obj = new Class1;
?>

<?php
$Obj = new \Class1;
?>
```

原则上，命名空间的命名方式及分类按照类、函数、常数的性质而定，同时 PHP 程序均放在全局空间 \ 内。若要在 PHP 程序中自定义命名空间，可以使用 namespace 关键字，下面是一个例子，它会在网页上显示变量 Y 的值（1），要注意的是 namespace 语句必须放在文件的最前端。下面程序的运行结果如图 7-17 所示。

\ch07\namespace.php

```
01:<?php
02:   namespace my\name;                  //在全局空间内定义 my\name 命名空间
03:   class MyClass {}                    //在\my\name 命名空间内定义 MyClass 类
04:   function Myfunction() {}            //在\my\name 命名空间内定义 Myfunction 函数
05:   const MYCONST = 1;                  //在\my\name 命名空间内定义 MYCONST 常数
06:   $X = new \my\name\MyClass;          //创建 MyClass 类的对象，写成$X = new MyClass;亦可
07:   $Y = \my\name\MYCONST;              //定义 Y 为常数 MYCONST，写成$Y = MYCONST;亦可
08:   echo $Y;                           //显示 Y 的值
09:?>
```

图 7-17

最后我们来说明如何利用命名空间设置别名，以 <\ch07\namespace.php> 为例，假设我们在第 02 行的下一行插入如下语句，表示使用 use 关键字将 my\name 命名空间的别名设置为 A：

```
use my\name as A;
```

这么一来，第 06、07 行可以改写成如下的形式，使用别名 A 取代 my\name 命名空间：

```
$X = new A\MyClass;
$Y = A\MYCONST;
```

随堂练习

（　）1. 在没有自行定义命名空间的情况下，PHP 程序是放在下列哪个命名空间的？

A. \　　B. .　　C. ~　　D. \root

（　）2. 下列哪一项可以访问定义为 private 的变量？

A.　整个程序　B. 子类　　C. 接口　　D. 包含其定义的类

（　）3. 以下列哪个关键词定义的方法表示不可以被子类覆盖？

A. abstract　　B. virtual　　C. override　　D. final

（　）4. 在类内定义常数可以使用下列哪个关键字？

A. var　　B. const　　C. function　　D. public

（　）5. 下列关于构造函数的叙述哪一个是错误的？

A. 名称为 __construct　　B. 使用 new 创建对象时会自动执行

C. 不可以有参数　　D. 子类会继承父类的构造函数

（　）6. 下列哪个运算符可以用来访问对象的成员？

A. ::　　B. =>　　C. ->　　D. .

（　）7. 下列哪个运算符可以直接访问类内的方法或常数，而无须创建对象？

A. ::　　B. =>　　C. ->　　D. .

（　）8. 若创建三个隶属于相同类的对象，那么内存中有几份成员数据？

A. 1　　B. 2　　C. 3　　D. 4

（　）9. 创建子类时要使用下列哪个关键字指定父类的名称？

A. extends　　B. interface　　C. class　　D. implements

（　）10. 子类无法访问父类内的下列哪个关键字定义的属性？

A. public　　B. protected　　C. var　　D. private

（　）11. 下列哪个关键字代表目前所在的子类的父类？

A. this　　B. mybase　　C. parent　　D. base

（　）12. PHP 允许父类有多个子类，亦允许子类有多个父类，对不对？

A. 对　　B. 不对

第 8 章

在网页之间传递信息

8.1　搜集网页上的数据

说起如何搜集网页上的数据，您可能会马上联想到“表单”（form），没错，我们的确可以通过表单请求浏览者输入数据，但只有表单是不够的，若没有搭配其他技术，当用户从表单所在的网页切换到另一个网页时，他在表单中输入的数据霎时会烟消云散，而诸如 PHP、ASP、JSP、CGI 等动态网页技术，正是搜集数据最重要的关键所在，它们能够从网页上抓取数据，以备未来使用。

8.1.1　建立表单

表单（form）可以提供输入接口，让用户输入数据，然后将数据返回 Web 服务器，以做进一步的处理，常见的应用有 Web 搜索、网络购票、在线问卷、会员登录、在线订购等。表单的建立包含下列两个部分。

（1）使用 <form> 和 <input> 元素编写表单的接口，例如单行文本框、单选按钮、复选框等。

（2）编写表单的处理程序，也就是表单的后端处理，例如将表单数据传送到电子邮件地址、写入文件、写入数据库或进行查询等。

此处会列出几个与表单有关的 HTML 元素供您参考，包括 <form>、<input>、<textarea>、<select>、<option> 等，若您已经相当熟悉，可以跳过这些 HTML 元素的介绍，直接翻到后面的范例。

❖ <form> 元素

<form> 元素用来在 HTML 文件中插入表单，其属性如下，标记星号（※）者为 HTML 5 新增的属性。

- accept="...": 指定 MIME 类型（超过一个的话，中间以逗号隔开），作为 Web 服务器处理表单数据的根据，例如 accept="image/gif, image/jpeg"。
- accept-charset="...": 指定表单数据的字符编码方式，Web 服务器必须根据指定的字符编码方式处理表单数据。字符编码方式定义于 RFC2045（超过一个的话，中间以逗号隔开），例如 accept-charset="ISO-8859-1"。
- action="*uri*": 指定表单处理程序的相对或绝对地址，若要将表单数据传送到电子邮件地址，可以指定电子邮件地址的 *uri*；若没有指定 action 属性的值，表示使用默认的表单处理程序，例如：

```
<form method="post" action="handler.php">
<form method="post" action="mailto:jean@hotmail.com">
```

- enctype="...": 指定将表单数据返回 Web 服务器所采用的编码方式，默认值为 "application/x-www-form-urlencoded"，若允许上传文件给 Web 服务器，则 enctype 属性的值要指定为 "multipart/form-data"；若要将表单数据传送到电子邮件地址，则 enctype 属性的值要指定为 "text/plain"。
- method="{get,post}": 指定表单数据传送给表单处理程序的方式，当 method="get" 时，表单数据会被存放在 HTTP GET 变量（$_GET），表单处理程序可以通过这个变量获取表单数据；当 method="post" 时，表单数据会被存放在 HTTP POST 变量（$_POST）中，表单处理程序可以通过这个变量获取表单数据；若没有指定 method 属性的值，表示为默认值 get。
- name="...": 指定表单的名称（限英文且唯一），此名称不会显示出来，但可以作为后端处理之用，供 Script 或表单处理程序使用。
- target="...": 指定用来显示表单处理程序之结果的目标框架。
- autocomplete="{on,off,default}"（※）: 指定是否启用自动完成功能，on 表示启用，off 表示关闭，default 表示继承所属之 <form> 元素的 autocomplete 属性，而 <form> 元素的 autocomplete 属性默认为 on，例如下面的语句是让用来输入用户名称的单行文本框具有自动完成功能:

```
<p>Username:<input type="text" autocomplete="on"></p>
```

- novalidate（※）: 指定在提交表单时不进行验证。
- HTML 元素的全局属性和事件属性，其中比较重要的有 onsubmit="..." 用来指定当用户传送表单时所要执行的 Script，以及 onreset="..." 用来指定当用户清除表单时所要执行的 Script。

❖ <input> 元素

<input> 元素用来在表单中插入输入字段或按钮，其属性如下，标记星号（*）者为 HTML 5 新增的属性，该元素没有结束标签。

- align="{left,center,right}"（Deprecated）: 指定图像提交按钮的对齐方式（当 type="image" 时）。
- accept="..." :指定提交文件时的 MIME 类型(以逗号隔开),例如 <input type="file" accept="image/gif,image/jpeg">。
- checked: 将单选按钮或复选框预设为已选取的状态。
- disabled: 取消表单字段，使表单数据无法被接受或提交。
- maxlength="*n*": 指定单行文本框、密码字段、搜索字段等表单字段的最多字符数。
- name="...": 指定表单字段的名称（限英文且唯一），此名称不会显示出来，但可以作为后端处理之用。
- notab: 不允许用户以按[Tab]键的方式移至表单字段。

- readonly：不允许用户变更表单字段的数据。
- size="*n*"：指定单行文本框、密码字段、搜索字段等表单字段的宽度（*n* 为字符数），size 属性和 maxlength 属性的差别在于它并不是指定用户可以输入的字符数，而是指定用户在界面上可以看到的字符数。
- src="*uri*"：指定图像提交按钮的地址（当 type="image" 时）。
- type="*state*"：指定表单字段的输入类型，稍后有进一步的说明。
- usemap：指定浏览器端影像地图所在的文件地址及名称。
- value="..."：指定表单字段的初始值。
- form="*formid*"（*）：指定表单字段隶属于 ID 为 *formid* 的表单。
- min="*n*"、max="*n*"、step="*n*"（*）：指定数字输入类型或日期输入类型的最小值、最大值和间隔值。
- required　（*）：指定用户必须在表单字段中输入数据，例如 <input type="search" required> 是指定用户必须在搜索字段中输入数据，否则浏览器会出现提示文字请求输入。
- multiple（*）：允许用户提交多个文件，例如 <input type="file" multiple>，或允许用户输入以逗号分隔的多个电子邮件地址，例如 <input type="email" multiple>。
- pattern="..."（*）：针对表单字段指定进一步的输入格式，例如 <input type="tel" pattern="[0-9]{4}(\-[0-9]{6})"> 是指定输入值须符合 xxxx-xxxxxx 的格式，而 x 为 0~9 的数字。
- autocomplete="{on,off,default}"（*）：指定是否启用自动完成功能，on 表示启用，off 表示关闭，default 表示继承所属之 <form> 元素的 autocomplete 属性，而 <form> 元素的 autocomplete 属性默认为 on。
- autofocus（*）：指定在加载网页的当下，使焦点自动移至表单字段。
- placeholder="..."　（*）：指定在表单字段内显示提示文字，待用户将焦点移至表单字段，该提示文字会自动消失。
- list（*）：list 属性可以和 HTML 5 新增的 <datalist> 元素搭配，让用户从预先输入的列表中选择数据或自行输入其他数据。
- HTML 元素的全局属性和事件属性，其中比较重要的有 onfocus="..." 用来指定当用户将焦点移至表单字段时所要执行的 Script，onblur="..." 用来指定当用户将焦点从表单字段移开时所要执行的 Script，onchange="..." 用来指定当用户修改表单字段时所要执行的 Script，onselect="..." 用来指定当用户在表单字段选取文字时所要执行的 Script。

要特别说明的是 type="*state*" 属性，HTML 4.01 提供了如表 8-1 所示的输入类型。

表 8-1　HTML 4.01 已有的 type 属性值

HTML 4.01 已有的 type 属性值	输入类型	HTML 4.01 已有的 type 属性值	输入类型
type="text"	单行文本框	type="reset"	重新输入按钮
type="password"	密码字段	type="file"	上传文件
type="radio"	单选按钮	type="image"	图像提交按钮
type="checkbox"	复选框	type="hidden"	隐藏字段
type="submit"	提交按钮	type="button"	一般按钮

HTML 5 则针对 type="*state*" 属性新增了如表 8-2 所示的输入类型。

表 8-2　HTML5 新增的 type 属性值

HTML 5 新增的 type 属性值	输入类型	HTML 5 新增的 type 属性值	输入类型
type="email"	电子邮件地址	type="date"	日期
type="url"	网址	type="time"	时间
type="search"	搜索字段	type="datetime"	UTC 世界标准时间
type="tel"	电话号码	type="month"	月份
type="number"	数字	type="week"	一年的第几周
type="range"	指定范围内的数字	type="datetime-local"	本地日期时间
type="color"	颜色		
注意：为了保持和旧版浏览器的向下兼容性，type 属性的默认值为 "text"，当浏览器不支持 HTML 5 新增的 type 属性值时，就会显示默认的单行文本框。			

事实上，HTML 5 除了提供更多的输入类型，更重要的是它会进行数据验证，举例来说，假设我们将 type 属性指定为 "email"，那么浏览器会自动验证用户输入的数据是否符合正确的电子邮件地址格式，若不符合，就提示用户重新输入。在浏览器内建数据验证的功能后，我们就不必再处处编写 JavaScript 程序代码验证用户输入的数据，不仅省时省力，网页的操作也会更顺畅。

❖ <textarea> 元素

<textarea> 元素用来在表单中插入多行文本框，其属性如下，标记星号（*） 者为 HTML 5 新增的属性。

- cols="*n*"：指定多行文本框的宽度（*n* 为字符数）。
- disabled：取消多行文本框，使之无法访问。
- name="..."：指定多行文本框的名称（限英文且唯一），此名称不会显示出来，但可以作为后端处理之用。
- readonly：不允许用户修改多行文本框的数据。
- rows="*n*"：指定多行文本框的高度（*n* 为行数）。

- form="*formid*"（*）：指定多行文本框隶属于 ID 为 *formid* 的表单。
- required　（*）：指定用户必须在多行文本框中输入数据。
- autofocus（*）：指定在加载网页的当下，令焦点自动移至多行文本框。
- placeholder="..."　（*）：指定在多行文本框内显示提示文字，待用户将焦点移至多行文本框，该提示文字会自动消失。
- HTML 元素的全局属性和事件属性，其中比较重要的有 onfocus="..." 用来指定当用户将焦点移至表单字段时所要执行的 Script，onblur="..." 用来指定当用户将焦点从表单字段移开时所要执行的 Script，onchange="..." 用来指定当用户修改表单字段时所要执行的 Script，onselect="..." 用来指定当用户在表单字段选择文字时所要执行的 Script。

在预设的情况下，多行文本框是呈现空白不显示任何数据，若要在多行文本框中显示默认的数据，可以将数据放在 <textarea> 元素里面。

❖ <select> 元素

<select> 元素用来搭配 <option> 元素在表单中插入下拉式菜单，其属性如下，标记星号（*）者为 HTML 5 新增的属性。

- multiple：指定用户可以在下拉式菜单中选择多个项目。
- name="..."：指定下拉式菜单的名称（限英文且唯一），此名称不会显示出来，但可以作为后端处理之用。
- readonly：不允许用户变更下拉式菜单的项目。
- size="*n*"：指定下拉式菜单的高度。
- form="*formid*"（*）：指定下拉式菜单隶属于 ID 为 *formid* 的表单。
- required　（*）：指定用户必须在下拉式菜单中选择项目。
- autofocus（*）：指定在加载网页的当下，使焦点自动移至多行文本框。
- HTML 元素的全局属性和事件属性，其中比较重要的有 onfocus="..."、onblur="..."、onchange="..."、onselect="..."。

❖ <option> 元素

<option> 元素是放在 <select> 元素里面的，用来指定下拉式菜单的项目，其属性如下，该元素没有结束标签。

- disabled：取消下拉式菜单的项目，使之无法访问。
- selected：指定预先选取的项目。
- value = "..."：指定下拉式菜单项目的值（中英文皆可），在用户单击“提交”按钮后，被选择的下拉式菜单项目的值会返回 Web 服务器，若没有指定 value 属性，那么下拉式菜单项目的数据会返回 Web 服务器。

- HTML 元素的全局属性和事件属性。

下面是一个例子，它通过表单询问消费者对于手机使用的意见。

\ch08\phone.html

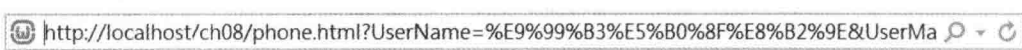
```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <form>
      姓   名：<input type="text" name="UserName" size="40"><br>
      E-Mail：<input type="text" name="UserMail" size="40" value="username@mailserver"><br>
      年  龄：
      <input type="radio" name="UserAge" value="Age1">未满 20 岁
      <input type="radio" name="UserAge" value="Age2" checked>20~29
      <input type="radio" name="UserAge" value="Age3">30~39
      <input type="radio" name="UserAge" value="Age4">40~49
      <input type="radio" name="UserAge" value="Age5">50 岁以上<br>
      您使用过哪些品牌的手机？
      <input type="checkbox" name="UserPhone[]" value="hTC" checked>hTC
      <input type="checkbox" name="UserPhone[]" value="Apple">Apple
      <input type="checkbox" name="UserPhone[]" value="ASUS">ASUS
      <input type="checkbox" name="UserPhone[]" value="acer">acer<br>
      您使用手机时最常碰到哪些问题？<br>
      <textarea name="UserTrouble" cols="45" rows="4">上网速度不够快</textarea><br>
      您使用过哪些电信业服务商？(可复选)
      <select name="UserNumber[]" size="4" multiple>
        <option value="中华电信">中华电信
        <option value="台湾大哥大" selected>台湾大哥大
        <option value="远传">远传
        <option value="亚太电信">亚太电信
      </select><br>
      <input type="submit" value="提交">
      <input type="reset" value="重新输入">
    </form>
  </body>
</html>
```

截至目前为止，这个程序只是一个单纯的 HTML 文件（扩展名为 .html），由于我们尚未自定义表单处理程序，因此，若在浏览器的地址栏输入 http://localhost/ch08/ phone.html，并按 Enter 键，然后填完表单，再单击“提交”按钮，表单数据将会被返回 Web 服务器，如图

8-1 所示。

图 8-1

那么表单数据是以何种形式返回 Web 服务器呢？在单击“提交”按钮后，地址栏会出现如下信息，从 http://localhost/ch08/phone.html 后面的问号（?）开始就是表单数据，第一个字段的名称为 UserName，虽然我们输入“陈小贞”，但由于将表单数据返回 Web 服务器所采用的编码方式默认为 "application/x-www-form-urlencoded"，故“陈小贞”会变成 %E9%99%B3%E5%B0%8F%E8%B2%9E；接下来是“&”符号，这表示下一个表单字段的开始；同理，下一个“&”符号的后面又是另一个表单字段的开始。



http://localhost/ch08/phone.html?UserName=%E9%99%B3%E5%B0%8F%E8%B2%9E&UserMail=jean@hotmail.com&UserAge=Age2&UserPhone%5B%5D=htc&UserPhone%5B%5D=Apple&UserTrouble=%E6%89%8B%E6%A9%9F%E9%9B%BB%E6%B1%A0%E5%BE%85%E6%A9%9F%E6%99%82%E9%96%93%E4%B8%8D%E5%A4%A0%E4%B9%85&UserNumber%5B%5D=%E5%8F%B0%E7%81%A3%E5%A4%A7%E5%93%A5%E5%A4%A7&UserNumber%5B%5D=%E9%81%A0%E5%82%B3

8.1.2　表单的后端处理

我们知道，在浏览者输入表单数据并单击“提交”按钮后，表单数据将会被返回 Web 服务器，至于表单数据的返回方式则取决于 <form> 元素的 method 属性，当 method="get" 时，表单数据会被存放在 HTTP GET 变量（$_GET）中，表单处理程序可以通过此变量获取表单数据；当 method="post" 时，表单数据会被存放在 HTTP POST 变量（$_POST）中，表单处理程序可以通过此变量获取表单数据；若没有指定 method 属性的值，表示为默认值 get。

get 和 post 最大的差别在于 get 所能传送的字符长度不得超过 255，而且在传送密码字段时，post 会将浏览者输入的密码加以编码，而 get 不会将浏览者输入的密码加以编码，从这种

角度来看，post 的安全性显然比 get 高。

此外，若我们使用了 <form> 元素的 action 属性指定表单处理程序，那么表单数据不仅会被返回 Web 服务器，也会被传送给表单处理程序，以做进一步的处理，例如将表单数据以 E-mail 的形式传送给指定的收件人、将表单数据写入或查询数据库、将表单数据张贴在留言板或聊天室等。

事实上，建立表单的接口并不难，难的在于编写表单处理程序，目前表单处理程序可以使用 PHP、ASP/ASP.NET、JSP、CGI 等服务器端 Scripts 来编写。

在本节中，我们会先示范如何将表单数据以 E-mail 形式传送给指定的收件人，之后再示范如何通过简单的 PHP 程序读取表单数据并制作成确认网页。

❖ 将表单数据以 E-mail 形式传送给指定的收件人

若要将表单数据以 E-mail 形式传送给指定的收件人，可以使用 <form> 元素的 action 属性指定收件人的电子邮件地址，举例来说，我们可以将 <\ch08\phone.html> 第 7 行的 <form> 元素改写成如下的形式，那么在浏览者填完表单并单击“提交”按钮后，就会将表单数据传送到 jean@hotmail.com，之后只要启动电子邮件程序接收新邮件，便能获取表单数据：

```
<form method="post" action="mailto:jean@hotmail.com">
```

❖ 读取表单数据并制作成确认网页

为了让浏览者知道其所输入的表单数据已经成功返回 Web 服务器，我们通常会在浏览者单击“提交”按钮后显示确认网页，如图 8-2 所示。举例来说，我们可以将 <\ch08\phone.html> 第 7 行的 <form> 元素改写成如下的形式，指定确认网页为 <\ch08\confirm.php>，如图 8-3 所示。然后另存新文件为 <\ch08\phone2.html>，就会得到如下的执行结果：

```
<form method="post" action="confirm.php">
```

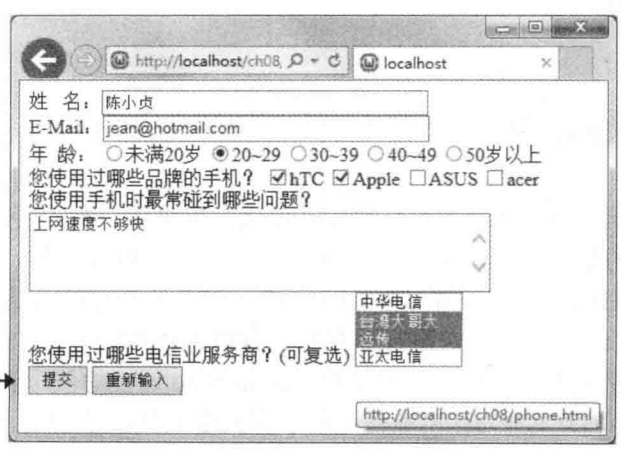


图 8-2

图 8-3

我们先把确认网页 <\ch08\confirm.php> 的程序代码列出来，再解释其中的意义，请您务必理解这个程序，因为里面使用了 PHP 读取表单字段的技巧。

\ch08\confirm.php

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:  </head>
06:  <body>
07:    <?php
08:      $Name = $_POST["UserName"];                    //读取第一个单行文本框的数据
09:      $Mail = $_POST["UserMail"];                    //读取第二个单行文本框的数据
10:      switch($_POST["UserAge"])                      //读取在单选按钮中选取的选项
11:      {
12:        case "Age1":
13:          $Age = "未满 20 岁";
14:          break;
15:        case "Age2":
16:          $Age = "20~29";
17:          break;
18:        case "Age3":
19:          $Age = "30~39";
20:          break;
21:        case "Age4":
22:          $Age = "40~49";
23:          break;
24:        case "Age5":
25:          $Age = "50 岁以上";
```

```
26:     }
27:     $Phone = $_POST["UserPhone"];          //读取在复选框中选择的选项
28:     $Trouble = $_POST["UserTrouble"];      //读取多行文本框的数据
29:     $Number = $_POST["UserNumber"];        //读取在下拉式菜单中选择的选项
30:   ?>
31:   <p><i><?php echo $Name; ?></i>，您好！您输入的资料如下：</p>
32:   电子邮件地址：<?php echo $Mail; ?><br>
33:   年龄：<?php echo $Age; ?><br>
34:   曾经使用过的手机品牌：<?php foreach($Phone as $Value) echo $Value.' '; ?><br>
35:   使用手机时最常碰到的问题：<?php echo $Trouble; ?><br>
36:   使用过哪些电信业服务商：<?php foreach($Number as $Value) echo $Value.' '; ?>
37: </body>
38:</html>
```

- 08：通过 HTTP POST 变量（$_POST），获取浏览者在名称为 "UserName" 的单行文本框内所输入的数据，然后赋值给变量 Name。
- 09：通过 HTTP POST 变量（$_POST），获取浏览者在名称为 "UserMail" 的单行文本框内所输入的数据，然后赋值给变量 Mail。
- 10 ~ 26：通过 HTTP POST 变量（$_POST）和 switch 判断结构，获取浏览者在名称为 "UserAge" 的单选按钮中所选择的选项（单选），然后赋值给变量 Age。
- 27：通过 HTTP POST 变量（$_POST），获取浏览者在名称为 "UserPhone" 的复选框中所选择的选项（可复选），然后赋值给变量 Phone。请注意，由于 UserPhone 是一个数组，所以变量 Phone 的值也会是一个数组。
- 28：通过 HTTP POST 变量（$_POST），获取浏览者在名称为 "UserTrouble" 的多行文本框内所输入的数据，然后赋值给变量 Trouble。
- 29：通过 HTTP POST 变量（$_POST），获取浏览者在名称为 "UserNumber" 的下拉式菜单中所选取的选项（可复选），然后赋值给变量 Number。请注意，由于 UserNumber 是一个数组，所以变量 Number 的值也会是一个数组。
- 31：这行语句里面穿插了 PHP 程序代码 <?php echo $Name; ?>，目的是显示变量 Name 的值，也就是浏览者在名称为 "UserName" 的单行文本框内所输入的数据（例如 "陈小贞"）。
- 32：这行语句里面穿插了 PHP 程序代码 <?php echo $Mail; ?>，目的是显示变量 Mail 的值，也就是浏览者在名称为 "UserMail" 的单行文本框内所输入的数据（例如 "jean@hotmail.com"）。
- 33：这行语句里面穿插了 PHP 程序代码 <?php echo $Age; ?>，目的是显示变量 Age 的值，也就是浏览者在名称为 "UserAge" 的单选按钮中所选取的选项（单选）（例如 "20~29"）。
- 34：这行语句里面穿插了 PHP 程序代码 <?php foreach($Phone as $Value) echo $Value.' '; ?>，目的是显示变量 Phone 的值，也就是浏览者在名称为 "UserPhone"

的复选框中所选择的选项（可复选）。请注意，由于变量 Phone 的值是一个数组，所以我们使用 foreach 循环来显示数组的每个元素（例如 "hTC"、"Apple"）。

- 35：这行语句里面穿插了 PHP 程序代码 <?php echo $Trouble; ?>，目的是显示变量 Trouble 的值，也就是浏览者在名称为 "UserTrouble" 的多行文本框内所输入的数据（例如 "手机电池待机时间不够久"）。
- 36：这行语句里面穿插了 PHP 程序代码 <?php foreach($Number as $Value) echo $Value.' '; ?>，目的是显示变量 Number 的值，也就是浏览者在名称为 "UserNumber" 的下拉菜单中所选取的选项（可复选）。请注意，由于变量 Number 的值是一个数组，所以我们使用 foreach 循环来显示数组的每个元素（例如 "台湾大哥大"、"远传"）。

- HTTP GET 变量（$_GET）、HTTP POST 变量（$_POST）和第 5 章所使用的服务器变量（$_SERVER）一样，都是 PHP 内置的超全局数组（superglobal array）。
- 除了确认网页之外，表单数据其实有更广泛的用途，例如写入文件、写入数据库或进行查询，这些内容我们会在“MySQL 数据库”篇与“应用实例”篇做示范。

编写如下表单网页 <bank.html>（如图 8-4）与确认网页 <calculate.php>（如图 8-5），其中本利和 = 本金 + 本金×(年利率×月数 / 12）。

图 8-4

图 8-5

【提示】

来自 \ch08\bank.html

```
<form method="post" action="calculate.php">
  <fieldset>
    <legend>个人资料</legend>
    请输入姓名  ：<input type="text" name="UserName" size="30"><br>
    请输入 E-Mail：<input type="text" name="UserMail" size="30"><br>
  </fieldset><br>
  <fieldset>
    <legend>计算存款本利和</legend>
    请输入本金（例如 500000）：<input type="text" name="UserCache" size="20"><br>
    请输入年利率（例如 0.05）：<input type="text" name="UserRate" size="20"><br>
    请输入月数（例如 11）       ：<input
      type="text" name="UserMonth" size="20"><br>
  </fieldset><br>
  <input type="submit" value="开始计算">
  <input type="reset" value="重新输入">
</form>
```

来自 \ch08\calcnlate.php

```
<?php
  $Name = $_POST["UserName"];
  $Rate = $_POST["UserRate"];
  $Cache = $_POST["UserCache"];
  $Month = $_POST["UserMonth"];
  $Total = $Cache + $Cache * $Rate * $Month / 12;
```

```
?>
  ...
  <body>
    <p><i><b><?php echo $Name; ?></b></i>，您好！</p>
    当本金为<?php echo $Cache; ?>、年利率为<?php echo $Rate; ?>、
    月数为<?php echo $Month; ?>时，本利和将为<i><b><?php echo $Total; ?></b></i>。
    <p><a href="bank.html">回上页</a></p>
  </body>
</html>
```

随堂练习

将前一个随堂练习的表单网页 <bank.html>（如图 8-6）与确认网页 <calculate.php>（如图 8-7）合并成同一个网页，如下面的 <calculate2.php>所示。

图 8-6

图 8-7

【提示】

首先，将表单的 action 属性指定为网页本身，即 "calculate2.php"（第 03 行）；接着，在表单中插入一个名称为 "Send" 的隐藏字段（第 04 行），用来判断浏览者有没有单击“开始计算”按钮，如果有的话，"Send" 字段的值会被指定为 TRUE，而 "Send" 字段的值一旦为 TRUE，在网页执行到第 02 行的 if 时，将因条件不成立，而跳到第 09 行的 else，换句话说，在浏览者单击“开始计算”按钮后，"Send" 字段的值就会被指定为 TRUE，然后调用表单处理程序 "calculate2.php"，而表单处理程序在执行到第 02 行的 if 时，会因为 "Send" 字段的值为 TRUE，而跳到第 09 行的 else，进行本利之和的计算并将结果显示在网页上。

取自 \ch08\calculate2.php

```
01:<body>
02:   <?php if (!isset($_POST["Send"])) { ?>
03:   <form method="post" action="calculate2.php">
04:     <input type="hidden" name="Send" value="TRUE">
05:     ...
06:   </form>
07:   <?php
08:     }
09:     else
10:     {
11:         $Name = $_POST["UserName"];
12:         $Rate = $_POST["UserRate"];
13:         $Cache = $_POST["UserCache"];
14:         $Month = $_POST["UserMonth"];
15:         $Total = $Cache + $Cache * $Rate * $Month / 12;
16:   ?>
17:   <p><i><b><?php echo $Name; ?></b></i>，您好！</p>
18:   当本金为<?php echo $Cache; ?>、
19:   年利率为<?php echo $Rate; ?>、
20:   月数为<?php echo $Month; ?>时，
21:   本利和将为<i><b><?php echo $Total; ?></b></i>。
22:   <?php } ?>
23:</body>
```

此属性亦可写成 action="<?php echo $_SERVER['PHP_SELF']; ?>"，超全局数组变量 $_SERVER['PHP_SELF'] 代表目前网页

8.2 HTTP Header

我们在第 1 章曾介绍过，Web 采用的是客户机-服务器架构（client-server model），其中客户机端（client）的计算机可以通过网络连接获取另一部计算机的资源或服务，而提供资源或服务的计算机就称为服务器端（server）。

当浏览器向 Web 服务器送出请求时，它并不只是将要打开网页的网址传送给 Web 服务

器，还会连同自己的浏览器类型、版本等信息一并传送过去，这些信息称为 Request Header（请求标头）。

相反的，当 Web 服务器响应浏览器的请求时，它并不只是将要打开的网页传送给浏览器，还会连同该网页的文件大小、日期等信息一并传送过去，这些信息称为 Response Header（响应标头），而 Request Header 和 Response Header 则统称为 HTTP Header（HTTP 标头）。

我们可以使用 PHP 内置的 header() 函数传送自定义的 HTTP Header，常见的应用有网页重定向、用户与密码认证等，其语法如下：

```
header(string string[, bool replace[, int http_response_code]])
```

- 参数 *string*：用来设置所要传送的 HTTP Header。
- 参数 *replace*：用来设置当有相同类型的 HTTP Header 存在时，是否加以取代，默认值为 TRUE，表示会加以取代。
- 参数 *http_response_code*：用来设置 HTTP Response Code。
- 下面是一个例子，它会传送两个相同类型的 HTTP Header，因为第二个语句指定了第二个参数为 FALSE，表示不会加以取代：

```
<?php
  header('WWW-Authenticate: Negotiate');
  header('WWW-Authenticate: NTLM', FALSE);
?>
```

8.2.1　网页重定向

我们直接以下面的例子来示范网页重定向，当浏览者在 <\ch08\choose.html> 的下拉菜单中选择所要连接的网站并单击“GO!”按钮时，表单处理程序 <\ch08\redirect.php> 就会调用 header() 函数，将网页重定向到浏览者所选择的网站，如图 8-8 所示。

\ch08\choose.html

```
01:<!doctype html>
02:<html>
03:   <head>
04:     <meta charset="utf-8">
05:   </head>
06:   <body>
07:     <form method="post" action="redirect.php">
08:       <select name="mySelect" size="1">
09:         <option value="http://tw.yahoo.com/">雅虎奇摩
10:         <option value="http://www.yam.com/">番薯藤
11:         <option value="http://www.google.com.tw/">Google
12:       </select>
```

```
13:         <input type="submit" value="GO!">
14:     </form>
15:  </body>
16:</html>
01:<?php
02:  $URL = $_POST["mySelect"];                    //通过 $_POST["mySelect"]变量读取所选择的网站 URL
03:  header("Location: $URL");                     //调用 header() 函数将网页重定向到所选择的网站
04:  exit();                                       //调用 exit() 函数确保不再执行后面的程序代码
05:?>
```

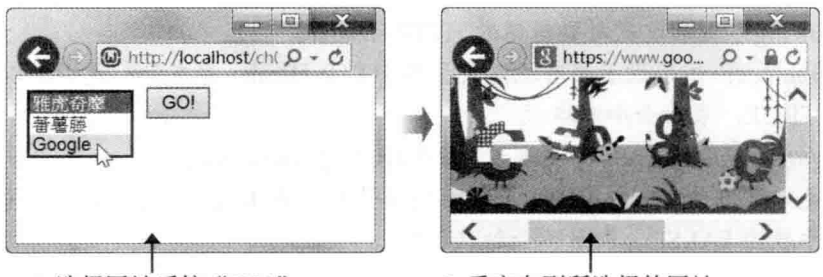


图 8-8

这个例子的重点在于 <\ch08\choose.html> 的第 07 行使用 action 属性指定表单处理程序为相同路径下的 redirect.php，以及 <\ch08\redirect.php> 的第 02 行通过 $_POST["mySelect"] 变量读取浏览者所选择的网站 URL，然后在第 03 行调用 header() 函数，将网页重定向到浏览者所选择的网站。

请注意，使用 header() 函数传送自定义的 HTTP Header 必须放在任何输出的前面，否则会导致 header() 函数执行失败。以下面的程序代码为例，它会得到类似 Warning: Cannot modify header information - headers already sent by...的警告，原因是在执行到 header("Location: $URL"); 语句之前，<html>、<body> 等元素已经被输出到客户端了。

```
<html>
  <body>
    <?php
      $URL = $_POST["mySelect"];
      header("Location: $URL");
      exit();
    ?>
  </body>
</html>
```

提醒您，请记得以 UTF-8 无 BOM 的编码方式存盘，也就是要确认 NotePad++ 的“编码”子菜单中核取了“编译成 UTF-8 码(文件头部无 BOM)”，才不会因为在自定义的 HTTP Header

前面先输出了 BOM，而导致 header() 函数执行失败。

8.2.2　用户与密码认证

在下面的例子 <\ch08\authen.php> 中，我们使用 header() 函数进行用户与密码认证，也就是在浏览者每次进入网站前，先显示如图 8-9 所示的对话框请求浏览者输入用户名称与密码，若浏览者有输入用户名称与密码，然后单击“确定”按钮，就会显示如图 8-10 所示的执行界面；相反的，若浏览者没有输入用户名称与密码，而是直接单击“取消”按钮，那么会显示如图 8-11 所示的执行界面。

\ch08\authen.php

```
01:<?php
02:   header("Content-type: text/html; charset=utf-8");                    //指定网页编码方式为 UTF-8
03:   if（!isset($_SERVER['PHP_AUTH_USER']))
04:   {
05:      header('WWW-Authenticate: Basic realm="快乐网站"');
06:      echo "抱歉！您没有输入密码！";
07:      exit();
08:   }
09:   else
10:   {
11:      echo "{$_SERVER['PHP_AUTH_USER']}您好！<br>";
12:      echo "您输入的密码为{$_SERVER['PHP_AUTH_PW']}！";
13:   }
14:?>
```

- 02: 调用 header() 函数传送自定义的 HTTP Header，以指定网页的编码方式为 UTF-8，若缺少这行语句，一旦程序里面要输出中文，可能会变成乱码。
- 03 ~ 08：由于浏览者所输入的用户名称与密码会分别存放在变量 $_SERVER['PHP_AUTH_USER']和 $_SERVER['PHP_AUTH_PW']中，因此，第 03 行用来检查浏览者是否已经在如图 8-9 的对话框输入用户名称与密码，若是第一次执行这个网页或之前直接单击“取消”按钮，没有输入用户名称与密码，就会执行第 05 ~ 07 行，其中第 05 行是显示如图 8-9 的对话框，第 06 行是显示字符串 "抱歉！您没有输入密码！"，第 07 行是终止程序，不再执行后面的程序代码。
- 09 ~ 13：若第 03 行检查到浏览者已经在如图 8-10 的对话框中输入用户名称与密码，那么会执行第 11 和第 12 行，其中第 11 行通过变量 $_SERVER['PHP_AUTH_USER'] 读取浏览者所输入的用户名称，然后显示出来，第 12 行通过变量 $_SERVER['PHP_AUTH_PW']读取浏览者所输入的密码，然后显示出来。

图 8-9

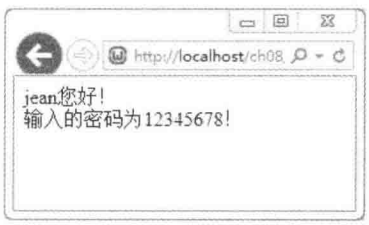


图 8-10

图 8-11

8.2.3 自动导向到 PC 版或移动版网页

目前诸如智能手机、平板电脑等移动设备非常普遍，由于不同设备的屏幕大小不同，若所有设备都浏览和 PC 一样的网页，势必会给用户带来困扰，而且不同设备的浏览器所支持的功能也不尽相同，因此，我们有必要替不同设备设计不同版本的网页，以下就为您示范如何将用户导向至 PC 版或移动版网页。

\ch08\detect.php

```
01:<?php
02:   if（detect_mobile())                    //若 detect_mobile() 返回 true，表示为移动设备
03:      $url = "mobile.php";                 //将变量 url 指定为移动版网页
04:   else
05:      $url = "PC.php";                     //将变量 url 指定为 PC 版网页
06:   header("Location: $url");               //重定向到变量 url 指定的网页
07:   exit();
08:   function detect_mobile()                //这个函数用来检测用户端是否为移动设备
09:   {
10:      $mobile_list="/(alcatel|amoi|android|avantgo|blackberry|benq|blazer|cell|docomo|
            dopod|ericsson|foma|htc|helio|hosin|huawei|iemobile|iphone|ipad|ipod|j2me|
            java|midp|mini|mmp|mobi|motorola|nokia|padfone|palm|panasonic|philips|
            phone|sagem|samsung|sharp|smartphone|sony|softbank|symbian|t-mobile|
            telus|vodafone|wap|webos|windows ce|wireless|xda|xoom|zte|
            opera\s*mobi|opera\*mini|320x320|240x320|176x220)/i";
11:      return preg_match($mobile_list, strtolower($_SERVER['HTTP_USER_AGENT']));
```

```
12: }
13:?>
```

- 02 ~ 05：调用 detect_mobile() 函数检测用户端是否为移动设备，如果是的话，就将变量 url 指定为移动版网页，否则将变量 url 指定为 PC 版网页。
- 08 ~ 12：定义 detect_mobile() 函数来检测用户端是否为移动设备，如果是就返回 TRUE，否则返回，其中第 10 行以正则表示式定义哪些 User Agent String 属于移动设备，而第 11 行是调用 PHP 内置的 preg_match() 函数进行正则表示式的字符串对比，而全局变量 $_SERVER['HTTP_USER_AGENT']可以用来获取操作系统及浏览器类型。

当用户通过 PC 版浏览器执行这个网页 <detect.php> 时，会被自动导向至 <PC.php> 网页，如图 8-12 所示。

相反的，当用户通过移动版浏览器执行这个网页 <detect.php>时，会被自动导向至 <mobile.php> 网页，如图 8-13 所示。

图 8-12

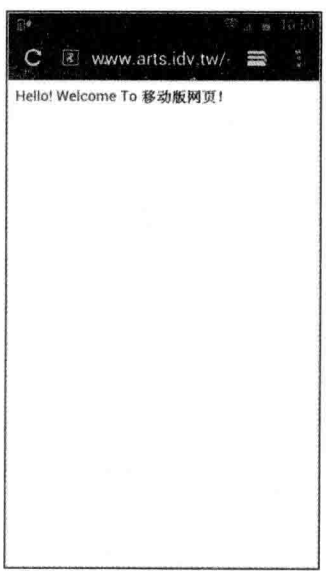


图 8-13

8.3　Cookie

Cookie 是浏览者访问某些网站时，Web 服务器在客户端写入的一些小文件，换句话说，Cookie 存放在客户端的内存或磁盘中，可以记录浏览者的各个信息，例如何时造访该网站、从事过哪些活动、购物车内有哪些商品等，这么一来，等浏览者下次再度造访该网站，只要查询 Cookie 的记录就会认得浏览者了。

一般来说，Cookie 有下列几项优点。

- Cookie 预设的生命周期起始于浏览器开始执行时，结束于浏览器终止执行时，此时的 Cookie 存放在客户端的内存中，但您可以自行设置 Cookie 的生命周期（通常以秒数为单位），将它写入客户端的磁盘，这样就不必担心 Cookie 自动消失而遗漏某些信息了。
- Cookie 存放在客户端的内存或磁盘中，不会占用 Web 服务器的资源。
- Cookie 可以记录浏览者的个人信息，于是网站的制作者便能根据 Cookie 记录的信息，设计出独具个人风格的网页或信息。
- 相对的，Cookie 也有下列几项缺点。
- 若遇到不支持 Cookie 的浏览器，或浏览者禁止 Web 服务器在客户端写入 Cookie，那么 Cookie 就英雄无用武之地了，此时得设法改用其他方式。
- Cookie 存放在客户端，可能会被浏览者删除或拒绝写入。
- Cookie 可能会造成安全上的威胁，导致个人信息被窃取。

8.3.1 写入 Cookie

我们可以使用 PHP 内置的 setcookie() 函数写入 Cookie，其语法如下：

```
setcookie(string name[, string value[, int expire[, string path[, string domain[, bool secure]]]]])
```

- 参数 *name*：用来设置 Cookie 的名称，只有这个参数不能省略，其他参数均能省略。
- 参数 *value*：用来设置 Cookie 的值，若这个参数的值为 ""，表示删除 Cookie。
- 参数 *expire*：用来设置 Cookie 的生命周期，例如 time() + 60 × 60 × 24 × 30 表示 Cookie 的生命周期为从现在起 60 × 60 × 24 × 30 秒（30 天）内，若没有设置这个参数，那么 Cookie 是存放在客户端的内存中的，当浏览器终止执行时，Cookie 也会随着消失，不会写入客户端的磁盘。
- 参数 *path*：用来设置 Cookie 在客户端的存放路径，例如 "/" 表示根目录，若没有设置这个参数，那么 Windows 7、Windows Vista 的用户可以在类似 C:\用户\Jean\AppData\Roaming\Microsoft\Windows\Cookies 的文件夹中找到 Cookie，而 Windows XP 的用户可以在类似 C:\Windows\System\Internet Temporary Files 的文件夹中找到 Cookie。
- 参数 *domain*：用来设置能够访问 Cookie 的网域，举例来说，假设 Web 服务器同时有 www.lucky.com.tw 和 forum.lucky.com.tw 两个网域，为了不让其中一个网域访问另一个网域的 Cookie，就必须将这个参数设置为 "www.lucky.com.tw" 或 "forum.lucky.com.tw"。
- 参数 *secure*：用来设置是否通过安全连接（SSL、HTTPS）传送 Cookie，默认值为 FALSE，表示不通过安全连接传送 Cookie。

若 setcookie() 函数执行成功，就返回 TRUE，否则返回 FALSE，要注意的是无论返回值是什么，都不意味着浏览者是否接受 Cookie。

下面是一个例子，它会在客户端写入，名称分别为 "UserName"、"UserAge"，值分别为 "小丸子"，10，生命周期为从现在起 60×60×24 秒（1 天）内的 Cookie：<\ch08\cookie1.php>

```
<?php
   header("Content-type: text/html; charset=utf-8");                    //指定网页编码方式为 UTF-8
   setcookie("UserName", "小丸子", time() + 60 * 60 * 24);
   setcookie("UserAge", 10, time() + 60 * 60 * 24);
?>
```

我们可以打开浏览器执行这个 PHP 程序，然后在客户端的磁盘上找到刚才写入的 Cookie，并打开 Cookie 来看看，上面果然记录了 Cookie 的名称、值及路径（注：Windows 7 的用户可以在类似 C:\用户\Jean\AppData\Roaming\Microsoft\ Windows\Cookies 的文件夹中找到 Cookie，这是一个被隐藏起来的操作系统文件）。

图 8-14

从图 8-14 可以看到，setcookie() 函数会将 Cookie 的值加以编码，例如 "小丸子" 被编码为%E5%B0%8F%E4%B8%B8%E5%AD%90，若不要将 Cookie 的值加以编码，可以改用 setrawcookie() 函数，其语法如下，参数的意义和 setcookie() 函数相同：

```
setrawcookie(string name[, string value[, int expire[, string path[, string domain[,bool secure]]]]])
```

我们可以使用 setrawcookie() 函数将程序改写成如下的形式：<\ch08\cookie2.php>

```
<?php
   header("Content-type: text/html; charset=utf-8");                    //指定网页编码方式为 UTF-8
   setrawcookie("UserName", "小丸子", time() + 60 * 60 * 24);
   setrawcookie("UserAge", 10, time() + 60 * 60 * 24);
?>
```

打开浏览器执行这个 PHP 程序，然后打开 Cookie 来看看，Cookie 的值果然没有被编码，如图 8-15 所示。

图 8-15

请注意，当我们写入多个相同名称的 Cookie 时，写入的操作会按序执行，例如下面的程序代码会写入一个名称为 "UserName"、值为 "Mary" 的 Cookie，因为第二个语句写入的值会覆盖第一个语句写入的值：

```
setcookie("UserName", "George");
setcookie("UserName", "Mary");
```

此外，写入 Cookie 的操作必须放在任何输出的前面，否则会导致 setcookie() 函数执行失败。为了避免这个问题，我们可以在输出的前面调用 ob_start() 函数，将输出放进缓冲区，待写入 Cookie 的操作完成后，再调用 ob_end_flush() 函数，取出缓冲区的输出。下面是一个例子，由于写入 Cookie 的操作前面有输出（第 04 行），故须调用 ob_start() 和 ob_end_flush() 函数（第 03、07 行），以免出现警告，该程序的运行结果如图 8-16 所示。

\ch08\cookie3.php

```
01:<?php
02:   header("Content-type: text/html; charset=utf-8");      //指定网页编码方式为 UTF-8
03:   ob_start();                                           //将输出放进缓冲区
04:   echo "Hello World!";                                  //字符串会暂时被放进缓冲区
05:   setcookie("UserName", "小丸子", time() + 60 * 60 * 24);
06:   setcookie("UserAge", 10, time() + 60 * 60 * 24);
07:   ob_end_flush();                                       //取出缓冲区的输出，于是显示 "Hello World!"
08:?>
```

图 8-16

最后要告诉您，setcookie() 函数也可以用来删除 Cookie，只是第二个参数必须设置为 ""，例如下面的语句会删除名称为 "UserName" 的 Cookie：

```
setcookie("UserName", "");
```

8.3.2 读取 Cookie

我们可以通过 $_COOKIE 变量读取 Cookie，这是 PHP 内置的超全局数组，例如下面的第一个语句是建立一个名称为 "UserName"、值为 "Mary" 的 Cookie，而第二个语句则是读取名称为 "UserName" 的 Cookie 的值并显示出来：

```
setcookie("UserName", "Mary");
echo $_COOKIE["UserName"];
```

此处要传授您一个小技巧，就是 Cookie 也可以用来存放数组，以下面的程序代码为例，第 03 ~ 05 行是在 Cookie 存放一个名称为 "Words" 的数组，里面有 "垦丁"、"春吶"、"真好玩" 三个元素，然后第 06 行使用 if 检查 $_COOKIE["Words"]是否已设置，如已设置的话，就执行第 07 ~ 08 行，使用 foreach 循环显示各个元素，该程序的运行结果如图 8-17 所示。此外，第 21 章有使用 Cookie 制作购物车的例子，相当实用，建议您仔细阅读。

\ch08\cookie4.php

```
01:<?php
02:   header("Content-type: text/html; charset=utf-8");        //指定网页编码方式为 UTF-8
03:   setcookie("Words[0]", "垦丁");
04:   setcookie("Words[1]", "春吶");
05:   setcookie("Words[2]", "真好玩");
06:   if（isset($_COOKIE["Words"]))
07:      foreach（$_COOKIE["Words"]as $key => $value)
08:         echo "$key : $value <br>";
09:?>
```

图 8-17

8.4　Session

在说明什么是 Session 之前，我们先来复习一下 Web 服务器是如何处理客户端请求的。基本上，当 Web 服务器收到客户端的请求时，它会找出相关的文件或程序，然后加以执行，将结果转换成 HTML 文件，再传送给客户端并中断连接。由于 Web 服务器在处理完客户端的请求后便会中断连接，所以 Web 服务器并没有记录客户端的信息，若要记录客户端的信息，必须使用一些特殊的技巧，常见的有文件访问、表单处理程序、Cookie 及本节所要介绍的 Session。

正如前面所说，Session 的用途是记录客户端的信息，而且每个客户端拥有各自的 Session，如图 8-18。举例来说，假设目前有 5 位浏览者连接到网站，那么这 5 位浏览者均拥有各自的 Session，我们可以通过 Session 记录每位浏览者的各个信息，例如姓名、访问次数、订购的商品、送货地址、持卡信息、累计的游戏点数、持有的宝物等。

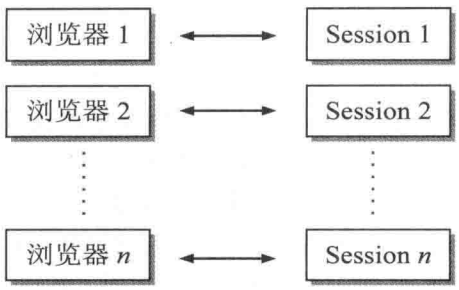


图 8-18

Session 预设的生命周期起始于浏览器开始执行时，结束于浏览器终止执行时，此时的 Session 存放在服务器端的内存中，但您可以自行设置 Session 的生命周期（通常是以秒数为单位），将它写入服务器端的磁盘中，这样就不必担心 Session 自动消失而遗漏了某些信息。

这样的描述听起来很熟悉，似乎跟 Cookie 很类似，对，事实上，Session 就相当于是服务器端的 Cookie，之所以有 Session，主要是考虑到不支持 Cookie 的浏览器及浏览者禁止 Web 服务器在客户端写入 Cookie 等情况。不过，为了减轻服务器端的负担，一般还是建议以 Cookie 取代 Session 记录客户端的信息。

8.4.1 访问 Session

PHP 支持的 Session 包含下列两个部分。

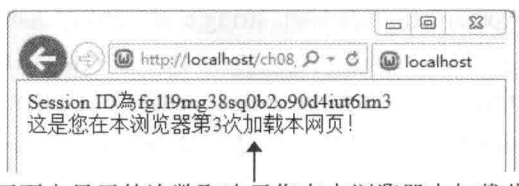
- Session ID（SID）：这是浏览器访问网页时被指派的一个唯一的标识符，以和其他浏览器进行区分，而且是存放在客户端的 Cookie 或嵌入 URL 一起传送。
- Session 变量：这是 Session 内所记录的变量，存放于服务器端的特殊文件，而且一个 Session ID 有一个文件。

我们可以通过变量 $_SESSION 访问 Session 内所记录的变量，这是 PHP 内置的超全局数组（superglobal array），下面是一个例子，该程序的运行结果如图 8-19 所示。

\ch08\session1.php

```
01:<?php
02:   header("Content-type: text/html; charset=utf-8");    //指定网页编码方式为 UTF-8
03:   session_start();                                     //启用 Session 功能
04:   echo "Session ID 为" . session_id() . "<br>";       //显示 Session ID
05:   if（!isset($_SESSION['Count']))                     //检查 Session 内是否尚未记录变量 Count
06:      $_SESSION['Count']= 1;                           //是就将变量 Count 设置为 1
07:   else
08:      $_SESSION['Count']++;                            //否则将变量 Count 的值递增 1
09:   echo "这是您在本浏览器第{$_SESSION['Count']}次加载本网页！ ";
10:?>
```

- 03：调用 session_start() 函数，通知 PHP 要启用 Session 功能，任何要访问 Session 的网页前面都要调用这个函数。
- 04：调用 session_id() 函数获取当前浏览器的 Session ID，然后显示出来。
- 05 ~ 08：使用 if 检查当前的 Session 内是否尚未记录一个名称为 Count 的 Session 变量（$_SESSION['Count']），是就执行第 06 行的代码，设置一个名称为 Count、值为 1 的 Session 变量，否则执行第 08 行的代码，将名称为 Count 的 Session 变量的值递增 1。



网页上显示的次数取决于您在本浏览器中加载此网页的次数，比方说，您只要重复单击 [重新刷新] 按钮，次数就会逐一递增。

图 8-19

由于 Session 预设的生命周期起始于浏览器开始执行时，结束于浏览器终止执行时，因此，只要浏览器没有关闭，Session 内记录的变量 Count 的值就会被保留下来，即使浏览器之后连接到其他网页，然后又回到这个网页，变量 Count 的值仍不会消失。

8.4.2　Session 相关的函数

PHP 内置了几个 Session 相关的函数，下面为您介绍比较重要的几个。

- session_start()：通知 PHP 要启用 Session 功能，返回值恒为 TRUE。由于访问网页的浏览器都会被指派一个唯一的 Session ID（SID），以和其他浏览器进行区分，而且 Session ID 存放在客户端的 Cookie 或嵌入 URL 一起传送，因此，session_start() 函数会在客户端的 Cookie 或 HTTP 参数里面检查是否有 Session ID，没有的话，就创建一个新的 Session ID，否则根据找到的 Session ID 恢复所有 Session 变量的值。
- session_unset()：释放所有 Session 变量。
- session_destroy()：清除所有 Session 变量的值。
- session_id([string *id*])：若指定了参数 *id*，表示以参数 *id* 取代当前的 Session ID，否则返回当前的 Session ID。
- session_name([string *name*])：若指定了参数 *name*，表示以参数 *name* 取代当前的 Session 名称，否则返回当前的 Session 名称。
- session_regenerate_id()：替当前的 Session 重新生成一个 Session ID，成功的话，就返回 TRUE，否则返回 FALSE。
- session_encode()：将当前的 Session 内容加以编码，然后返回编码后的字符串。

- session_decode(string *data*): 将参数 *data* 加以解码，还原成当前的 Session 内容，成功的话，就返回 TRUE，否则返回 FALSE。
- session_write_close(): 保存 Session 数据并终止当前的 Session，通常在 Script 结束时，Session 数据都会被保存，无须另外调用这个函数，除非是您做了一些 PHP 并不知道 Script 已经结束的动作，例如网页重定向。这个函数有一个别名称为 session_commit()。
- session_save_path([string *path*]): 将当前的 Session 存放路径设置为参数 *path* 所指定的路径。
- session_set_cookie_params(int *lifetime*[, string *path*[, string *domain*[, bool *secure*]]]): 这个函数必须放在 session_start() 函数的前面，用来设置 php.ini 配置文件内 session.cookie_lifetime 、 session.cookie_path 、 session.cookie_ domain 、 session.cookie_secure 4 个参数的值，这些参数的意义如表 8-3 所示。

表 8-3 php-ini 配置文件内的参数

参数	说明
session.cookie_lifetime	设置 Session Cookie 的生命周期，默认值为 0，表示持续到浏览器关闭时，若变更为其他数字，则表示秒数
session.cookie_path	设置 Session Cookie 的有效路径，默认值为 /
session.cookie_domain	设置 Session Cookie 的有效网域，默认值为没有设置
session.cookie_secure	设置 Session Cookie 是否通过安全连接（SSL、HTTPS）传送，默认值为 Off，表示否

- session_get_cookie_params(): 返回 php.ini 配置文件内 session.cookie_lifetime、session.cookie_path、session.cookie_domain、session.cookie_secure 4 个参数的值，返回值为数组，有 "lifetime"、"path"、"domain"、"secure" 4 个键，分别表示 Session Cookie 的生命周期、有效路径、有效网域、是否通过安全连接（SSL、HTTPS）传送。

一、选择题

（ ）1. 我们可以使用下列哪个变量获取表单字段的值？

A. $_SERVER　　B.$_COOKIE　　C. $_POST　　D.$_SESSION

（ ）2. 下列哪个函数可以删除 Cookie？

A. clearcookie()　　B.setcookie()　　C. ob_start()　　D.ob_end_flush()

（ ）3. 下列关于变量、Cookie、Session 的比较哪一项是错误的？

A.变量的生命周期起始于网页开始执行时，结束于网页终止执行时

B. Cookie 和 Session 预设的生命周期都等于浏览器的执行期间

C.我们可以自行设置 Cookie 和 Session 的生命周期

D. Cookie 和 Session 均存放在客户端的内存或磁盘中

二、实践实验题

编写如下所示的表单网页 <profile.html> 与确认网页 <reply.php>，如图 8-20 所示。

图 8-20

第 9 章
使用 Ajax

9.1　认识 Ajax

Ajax 是 Asynchronous JavaScript And XML 的简写，代表 Ajax 具有异步、使用 JavaScript 与 XML 等技术的特性。虽然 Ajax 的概念早在 Microsoft 公司于 1999 年推出 Internet Explorer 5 时，就已经存在，但并不是很受重视，直到近年来被大量应用于 Google Maps、Gmail 等 Google 网页，才迅速蹿红。例如用户可以在导入 Ajax 技术的 Google Maps 中平顺地拖曳整张地图，而不会有延迟，也不会因为单击地图的操作按钮，导致网页重新刷新，而必须浪费时间等待；又例如导入 Ajax 技术的在线游戏与社群网站可以让用户拥有更实时的响应，减少操作延迟或界面重新刷新的情况。

为了让您了解导入 Ajax 技术的动态网页和传统的动态网页有何不同，我们先来说明传统的动态网页如何工作，其工作方式如图 9-1 所示，当浏览者改变下拉菜单中选取的项目、单击按钮或做出任何与 Web 服务器互动的操作时，就会产生 Http Request，将整个网页内容返回 Web 服务器，即使这次的操作只需要一个字段的数据，浏览器仍会将所有字段的数据都返回 Web 服务器，Web 服务器在收到数据后，就会执行指定的操作，然后以 Http Response 的方式，将执行结果全部送回浏览器（包括完全没有变动过的数据、图像、JavaScript 等）。

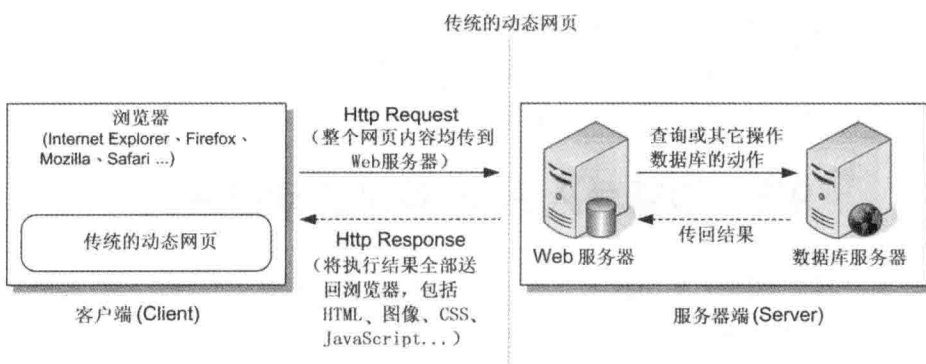


图 9-1

浏览器在收到数据时，便会将整个网页内容重新显示，所以浏览者通常都会看到网页闪一下，当网络太慢或网页内容太大时，浏览者看到的可能不是一闪，而是界面停格，完全无法与网页互动，相当浪费时间。

相反，导入 Ajax 技术的动态网页工作方式则如图 9-2 所示，当浏览者改变下拉菜单中选取的项目、单击按钮或做出任何与 Web 服务器互动的操作时，浏览器端会使用 JavaScript 通过 XMLHttpRequest 对象送出异步的 Http Request，此时只会将需要的字段数据返回 Web 服务器（不是全部数据），然后执行指定的操作，并以 Http Response 的方式，将执行结果送回浏览器（不包括完全没有变动过的数据、图像、JavaScript 等），浏览器在收到数据后，可以使用 JavaScript 通过 DHTML 或 DOM（Document Object Model）模式来更新特定字段。

图 9-2

由于整个过程均使用异步技术，无论是将数据返回服务器或接收服务器送回的执行结果并更新特定字段等操作，都在后台运行，因此，浏览者不会看到网页闪一下，界面也不会停止，浏览者在这个过程中仍能进行其他操作。

由前面的讨论可知，Ajax 是客户端的技术，它让浏览器能够与 Web 服务器进行异步沟通，服务器端的程序写法不会因为导入 Ajax 技术而有太大差异。事实上，Ajax 功能已经被实现为 JavaScript 解释器（interpreter）原生的部分，导入 Ajax 技术的动态网页将享有下列效益。

- 异步沟通无须将整个网页内容返回 Web 服务器，能够节省网络带宽。
- 由于只返回部分数据，所以能够减轻 Web 服务器的负担。
- 不会像传统的动态网页那样产生短暂空白或闪动的情况。

9.2 编写导入 Ajax 技术的动态网页

为了让您对网页如何导入 Ajax 技术有初步的认识，我们将其工作过程描绘如图 9-3 所示，首先，使用 JavaScript 创建 XMLHttpRequest 对象；接着，通过 XMLHttpRequest 对象传送异步 Http Request，Web 服务器一收到 Http Request，就会执行预先写好的程序代码（后端程序可以是 JSP、PHP、ASP/ASP.NET 等），再将结果以纯文本或 XML 格式返回浏览器；最后，仍使用 JavaScript 根据返回来的结果更新网页内容，整个过程都是异步并在后台运行，而且浏览器的所有操作均是通过 JavsScript 来完成的。

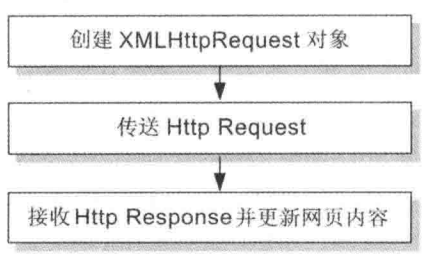


图 9-3

❖ 一、创建 XMLHttpRequest 对象

在不同浏览器中创建 XMLHttpRequest 对象的 JavaScript 语法不尽相同，主要分为下列三种。

- Internet Explorer 5 浏览器

```
var XHR = new ActiveXObject("Microsoft.XMLHTTP");
```

- Internet Explorer 6+浏览器

```
var XHR = new ActiveXObject("Msxml2.XMLHTTP");
```

- 其他非 Internet Explorer 浏览器

```
var XHR = new XMLHttpRequest();
```

由于我们无法事先得知浏览器的种类，于是针对前述的 JavaScript 语法编写如下跨浏览器的 Ajax 函数，它可以在目前主流的浏览器中创建 XMLHttpRequest 对象。

\ch09\utility.js

```
function createXMLHttpRequest()
{
  try                                  //其他非 IE 浏览器
  {
    var XHR = new XMLHttpRequest();
  }
  catch(e1)                            //若捕捉到错误，表示客户端不是非 IE 浏览器
  {
    try                                //IE6+浏览器
    {
      var XHR = new ActiveXObject("Msxml2.XMLHTTP");
    }
    catch(e2)                  //若捕捉到错误，表示客户端不是 IE6+浏览器
    {
      try                              //IE5 浏览器
      {
        var XHR = new ActiveXObject("Microsoft.XMLHTTP");
      }
      catch(e3)                //若捕捉到错误，表示客户端不支持 Ajax
      {
        XHR = false;
      }
    }
```

```
    }
    return XHR;
}
```

日后若网页需要创建 XMLHttpRequest 对象，就将 utility.js 文件 include 进来，然后调用 createXMLHttpRequest() 函数即可，如下所示：

```
var XHR = createXMLHttpRequest();
```

❖ 二、传送 Http Request

成功创建 XMLHttpRequest 对象后，我们必须做下列设置才能传送异步 Http Request。

步骤 01 首先，调用 XMLHttpRequest 对象的 open() 方法，来设置要向 Web 服务器请求什么资源（文本文件、网页等），其语法如下，参数 *method* 用来指定建立 Http 连接的方式，例如 GET、POST、HEAD，参数 *URL* 为要请求的文件地址，参数 *async* 用来指定是否使用异步调用，默认值为 true:

```
open(string method, string URL, bool async)
```

例如，下面的程序代码会向 Web 服务器以 GET 方式异步请求 poetry.txt 文件：

```
var XHR = createXMLHttpRequest();
XHR.open("GET", "poetry.txt", true);
```

步骤 02 接着，在 Web 服务器接收到数据，进行处理并返回结果后，XMLHttpRequest 对象的 readyState 属性会改变，进而触发 onreadystatechange 事件，因此，我们可以通过 onreadystatechange 事件处理程序接收 Http Response，例如下面的语句表示当发生 onreadystatechange 事件时，就执行 handleStateChange() 函数，来获取 Web 服务器返回的结果：

```
XHR.onreadystatechange = handleStateChange;
```

步骤 03 最后，调用 XMLHttpRequest 对象的 send() 方法，来送出 Http Request，其语法如下，参数 *content* 是要传送给 Web 服务器的参数，例如 "UserName=Jerry &PageNo=1"，当您以 GET 方式传送 Request 时，由于不需要传送参数，故参数 *content* 为 null，而当您以 POST 方式传送 Request 时，则可以指定要传送的参数：

```
send(string content)
```

综合前面的讨论，可以整理成如下的形式：

```
var XHR = createXMLHttpRequest();
XHR.open("GET", "poetry.txt", true);
XHR.onreadystatechange = handleStateChange;
```

```
XHR.send(null);
function handleStateChange()
{
  //此处用来获取 Web 服务器传回的结果
}
```

❖　三、接收 Http Response 并更新网页内容

由于我们只能通过 XMLHttpRequest 对象的 onreadystatechange 事件了解 Http Request 的执行状态，因此，接收 Http Response 的程序代码是写在 onreadystatechange 事件处理程序中的，也就是前面例子所指定的 handleStateChange() 函数。

XMLHttpRequest 对象的 readyState 属性会记录目前处于哪个阶段，返回值为 0～4 的数字，其中 4 代表 Http Request 执行完毕。不过，Http Request 执行完毕并不等于执行成功，因为有可能发生指定的资源不存在或执行错误的情况，所以我们还得判断 XMLHttpRequest 对象的 status 属性，只有当 status 属性返回 200 时，才代表执行成功，此时 statusText 属性会返回 OK，若指定的资源不存在，则 status 属性会返回 404，而 statusText 属性会返回 Object Not Found。

当 Web 服务器返回的数据为文字时，我们可以通过 XMLHttpRequest 对象的 responseText 属性获取执行结果；当 Web 服务器返回的数据为 XML 文件时，我们可以通过 XMLHttpRequest 对象的 responseXML 属性获取执行结果。

此外，XMLHttpRequest 对象还提供了下列方法。

- abort()：停止 HTTP Reqeust。
- getAllResponseHeaders()：获取所有 HTTP 标头信息。
- getResponseHeader(string *Name*)：获取参数 *Name* 指定的标头信息。

现在，我们就来看个实际应用，这个例子的执行结果如图 9-4 所示，当浏览者单击[显示诗句]按钮时，就会读取服务器端的 poetry.txt 文本文件，然后将文件内容显示在按钮下面。

图 9-4

为了让您做比较，我们先采用传统的 PHP 写法，如下所示，由于这个网页尚未导入 Ajax 技术，所以在单击[显示诗句]按钮时，整个网页会重载而快速闪一下，然后在按钮下面显示诗句。

\ch09\program1.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body>
    <form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
      <input type="submit" value="显示诗句"><br><br>
      <?php if（!isset($_POST["Send"])) { ?>
      <input type="hidden" name="Send" value="TRUE">
      <?php }
        else echo file_get_contents("poetry.txt");
      ?>
    </form>
  </body>
</html>
```

下面的程序则是改成导入 Ajax 技术，注意网页的扩展名为 .html。

\ch09\program2.html

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:    <script src="utility.js" type="text/javascript"></script>
06:    <script type="text/javascript">
07:      var XHR = null;
08:
09:      function startRequest()
10:      {
11:        XHR = createXMLHttpRequest();
12:        XHR.open("GET", "poetry.txt", true);
13:        XHR.onreadystatechange = handleStateChange;
14:        XHR.send(null);
15:      }
16:
17:      function handleStateChange()
18:      {
19:        if（XHR.readyState == 4)
20:        {
21:          if（XHR.status == 200)
```

```
22:                 document.getElementById("span1").innerHTML = XHR.responseText;
23:             else
24:                 window.alert("文件打开错误!");
25:         }
26:     }
27:   </script>
28: </head>
29: <body>
30:   <form id="form1">
31:     <input id="button1" type="button" value="显示诗句" onclick="startRequest()">
32:     <br><br> <span id="span1"></span>
33:   </form>
34: </body>
35:</html>
```

这个网页的执行结果如图 9-5 所示，和前面的 <\ch09\program1.php> 相同，但扩展名为 .html，表示它只是一个在客户端执行的网页。

图 9-5

- 05：将 utility.js 文件 include 进来，方便创建 XMLHttpRequest 对象。
- 06 ~ 27：这是客户端的 JavaScript 脚本，用来进行异步传输。
- 07：定义一个名称为 XHR 的全局变量，用来代表即将创建的 XMLHttpRequest 对象。
- 09 ~ 15：定义 startRequest() 函数，这个函数会在浏览者单击[显示诗句]按钮时执行，第 11 行是创建 XMLHttpRequest 对象，第 12 行是指定以 GET 方式向服务器请求 poetry.txt 文本文件，第 13 行是指定在 XMLHttpRequest 对象的 readyState 属性改变时执行 handleStateChange() 函数，第 14 行是送出异步请求。
- 17 ~ 26：定义 handleStateChange() 函数，它会在 XMLHttpRequest 对象的 readyState 属性改变时执行，故会重复触发多次，第 19 行的 if 条件用来判断 XMLHttpRequest 对象的 readyState 属性是否返回 4，是的话，表示异步传输完成，就执行第 20 ~ 25 行，而第 21 行的 if 条件用来判断 XMLHttpRequest 对象的 status 属性是否返回 200，是的话，就执行第 22 行，将返回值显示在 元素的内容，否则，就执行第 24

行，显示错误信息。

事实上，JavaScript 也可以在客户端直接调用服务器端的 PHP 程序，下面是一个例子，它会调用服务器端的 PHP 程序 <GetServerTime.php> 显示格林威治标准时间（GMT），执行结果如图 9-6 所示。

\ch09\program3.html

```
01:<!doctype html>
02:<html>
03:   <head>
04:      <meta charset="utf-8">
05:      <script type="text/javascript" src="utility.js"></script>
06:      <script type="text/javascript">
07:         var XHR = null;
08:         function startRequest()
09:         {
10:            XHR = createXMLHttpRequest();          在此调用服务器端的 PHP 程序
11:            XHR.open("GET", "GetServerTime.php", true);
12:            XHR.onreadystatechange = handleStateChange;
13:            XHR.send(null);
14:         }
15:
16:         function handleStateChange()
17:         {
18:            if（XHR.readyState == 4)
19:            {
20:               if（XHR.status == 200)
21:                  document.getElementById("span1").innerHTML = XHR.responseText;
22:               else
23:                  window.alert("无法显示时间!");
24:            }
25:         }
26:      </script>
27:   </head>
28:   <body>
29:      <form id="form1">
30:         <input id="button1" type="button" value="显示时间" onclick="startRequest()">
31:         <br><br><span id="span1"></span>
32:      </form>
33:   </body>
34:</html>
```

图 9-6

这个网页的扩展名为 .html，表示它只是一个在客户端执行的网页，而且它和前面的 <\ch09\program2.html> 几乎相同，最大的差别在于第 11 行改以 GET 方式向服务器请求 PHP 程序 <GetServerTime.php>：

```
11:          XHR.open("GET", "GetServerTime.php", true);
```

至于 PHP 程序 <GetServerTime.php> 的内容则相当简单，就是调用 gmdate() 函数显示格林威治标准时间（GMT）。

\ch09\GetServerTime.php

```
<?php
  echo gmdate("Y-m-d H:i:s");
?>
```

备注

要想在网页上导入 Ajax 技术，除了自行编写前面的 JavaScript 之外，还有一些现成的套件可以使用，例如 xajax、SAJAX、JPSPAN (ScriptServer)、AJASON、flxAJAX、AjaxAC 等，您可以在 Sourceforge (http://sourceforge.net/) 进行下载。

第 10 章

jQuery Mobile 移动版网页

10.1　认识 jQuery Mobile

随着无线网络与移动通信的蓬勃发展，以及智能手机、平板电脑等移动设备的快速普及，用户可以随时随地上网回复、打卡、浏览网页。虽然移动设备的浏览器大多能够顺利读取并显示传统的 PC 版网页，但受限于较小的屏幕，用户往往得通过频繁地拉近、拉远、滚动网页来阅读网页的信息，相当不方便。

为此，开始有越来越多的网站推出所谓的“移动版”，也就是移动版网站（mobile website）。举例来说，图 10-1 是 Yahoo!奇摩的 PC 版网站（http://tw.yahoo.com/），而图 10-2 是 Yahoo!奇摩的移动版网站（http://tw.yahoo.com/mobile/），加以浏览后，我们发现，对于 Yahoo!奇摩这种信息浏览类型的网站来说，其移动版除了着重执行效能之外，信息的分类与动态的设计也不能忽视，才能带给移动设备的用户直观流畅的操作体验。

图 10-1

图 10-2

事实上，网站推出移动版已经成为趋势，但问题来了，市面上有多种不同的移动平台，例如 Android、iOS、Windows 8、BlackBerry OS 等，即便同样是 Android 平台，也有许多不同品牌、不同尺寸的智能手机、平板电脑等设备，总不能针对每个设备都推出专用的移动版网站吧？显然网页设计人员需要一个能够跨平台、跨设备的架构，让他们设计的移动版网站可以在不同的移动设备上显示相同的结果，而本章所要介绍的 jQuery Mobile 就是一个能够达成此目标的架构。

jQuery Mobile 官方网站（http://jquerymobile.com/）的说明指出，jQuery Mobile 是一个基于 jQuery 与 jQuery UI（用户界面），以 HTML 为基础的统一用户界面系统，横跨所有受欢迎的移动设备平台，其轻量级的程序代码具有弹性且容易更换主题设计。

上文提及的 jQuery 是一个快速、轻巧、功能强大且跨浏览器的 JavaScript 函数库，属于开放源码，网页设计人员可以利用这个函数库所提供的 API 简化 HTML 与 JavaScript 之间的操作，例如选择 HTML 元素、操作 HTML 文件、处理事件、创建动画效果、导入 Ajax 技术

等。至于 jQuery UI 则是一个基于 jQuery 的 JavaScript 函数库，用来建立交互式用户界面，例如按钮、对话框、列表视图（ListView）、工具栏、导航条、表单、拨动式切换开关等控件，拖曳、拖放或放大缩小等操作，设置 CSS 样式、颜色动画、淡入、淡出等动画效果。

使用 jQuery Mobile 开发的移动版网页具有下列特点：

- 能够在不同的移动设备上显示相同的结果，达到跨平台、跨设备、跨浏览器的目的。
- 为触控设备提供优化的用户界面。
- 程序代码轻巧简单。
- 可更换或自定义主题。
- 网页主要使用 HTML 5 语法来编写，除非是一些高级的功能，否则就算不懂 JavaScript，也能使用 jQuery Mobile。

移动版网页和 PC 版网页一样是扩展名为 .html 或 .htm 的纯文本文件，任何能够用来输入纯文本的编辑工具，都可以用来编写移动版网页，例如 Notepad++。要注意的是 PC 版浏览器不太适合用来测试移动版网页，最好是在移动设备上做实际的测试，或在 PC 机上安装移动版浏览器的模拟器来进行测试，例如 Opera Mobile Emulator，您可以连接到 http://www.opera.com/zh-tw/developer/mobile-emulator 下载并安装 Windows、Mac 或 Linux 平台的模拟器。

10.2 编写 jQuery Mobile 移动版网页

使用 jQuery Mobile 开发移动版网页需要一些相关文件，包括：

- jQuery Mobile 核心 CSS 文件（例如 jquery.mobile-*XX*.min.css，*XX* 为版本）
- jQuery 核心 JavaScript 文件（例如 jquery-*XX*.min.js）
- jQuery Mobile 核心 JavaScript 文件（例如 jquery.mobile-*XX*.min.js）
- 对于这些文件，我们可以通过下列两种方式来获取。
- 下载 jQuery 与 jQuery Mobile 套件：到 http://jquery.com/download/ 和 http://jquerymobile.com/download/ 下载 jQuery 与 jQuery Mobile 套件，例如 jquery-1.9.1.min.js 和 jquery.mobile-1.3.1.ZIP，然后将解压缩得到的文件和文件夹复制到网站项目的根目录。
- 使用 CDN（Content Delivery Networks）：在网页中引用 jQuery 与 jQuery Mobile 官方网站提供的文件，而不是将相关文件复制到网站项目的根目录。我们可以在 http://jquerymobile.com/download/ 找到类似如下的程序代码，将它复制到网页的 <head> 区块即可：

```
<link rel="stylesheet" href="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.css" />
<script src="http://code.jquery.com/jquery-1.9.1.min.js"></script>
```

```
<script src="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.js"></script>
```

jquery.mobile-1.3.1.min.css、jquery-1.9.1.min.js、jquery.mobile-1.4.2.min.js 为 jQuery Mobile 核心 CSS 文件、jQuery 核心 JavaScript 文件、jQuery Mobile 核心 JavaScript 文件，文件名中的 1.3.1 或 1.9.1 为版本号，而 .min 为最小化的文件，也就是不含空白、换行、注释并经过压缩，推荐给正式版使用。

除了这些文件，我们可能还会使用到 jquery.mobile.structure-1.3.1.min.css 、jquery.mobile.theme-1.3.1.css 等文件，以使用主题。jQuery 与 jQuery Mobile 属于开放源码，能够免费使用，注意不要删除文件开头的版权信息。现在，我们就以实际的例子示范如何开发 jQuery Mobile 移动版网页。

\ch10\jQM1.html

```
01:<!doctype html>
02:<html>
03:   <head>
04:      <meta charset="utf-8">
05:      <title>我的 jQuery Mobile 程序</title>
06:      <link rel="stylesheet" href="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.css" />
07:      <script src="http://code.jquery.com/jquery-1.9.1.min.js"></script>
08:      <script src="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.js"></script>
09:      <meta name="viewport" content="width=device-width, initial-scale=1">
10:   </head>
11:   <body>
12:   </body>
13:</html>
```

目前这份文件还只是一个空白网页，图 10-3 是在智能手机上的浏览结果，而图 10-4 是在 Opera Mobile Emulator 上的模拟界面。

图 10-3

图 10-4

虽然jQuery Mobile文件不一定要上传到Web服务器，例如此文件的位置为C:\jQM1.html，那么只要在模拟器的地址栏输入 file://C:/jQM1.html 即可进行浏览，但我们的建议是将文件上传到支持 PHP 的 Web 服务器，然后在移动版浏览器或模拟器的网址栏输入以 http:// 开头的网址，例如 http://www.arts.idv. tw/~jean/jQM1.html，这样才能执行使用 PHP 技术的移动版网页。

这份文件的重点如下。

- 06：使用 CDN 引用 jQuery Mobile 核心 CSS 文件。
- 07：使用 CDN 引用 jQuery 核心 JavaScript 文件。
- 08：使用 CDN 引用 jQuery Mobile 核心 JavaScript 文件。
- 09：定义一个 metadata，名称为 "viewport"，内容为 "width=device-width, initial-scale=1"，表示将网页宽度指定为移动设备的屏幕宽度、缩放比为 1:1。

在加入网页内容之前，我们先解释一个概念，一份 jQuery Mobile 文件可以包含一个或多个“页面”（page），而一个页面又可以包含一个或多个“角色”（role），“页面”就像“角色”的容器。举例来说，我们在 <jQM1.html> 的 <body> 区块加入第 12～24 行的程序代码，分别使用 4 个 元素和 data-role 属性定义页面、页首、内容和页尾等角色，一个典型的页面通常包含页首、内容和页尾，其中内容是一定要存在的。至于这些角色如何区分，则取决于 data-role 属性的值，表 10-1 为一些常见的角色。

表 10-1　常见的角色

角色	说明	角色	说明
page	页面	listview	列表视图
header	页首	dialog	对话框
content	内容	fieldcontain	表单字段容器
footer	页尾	controlgroup	控件组
navbar	导航条	collapsible	可折叠面板
button	可视化按钮	slider	拨动式切换开关

\ch10\jQM1.html

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:    <title>我的 jQuery Mobile 程序</title>
06:    <link rel="stylesheet" href="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.css" />
07:    <script src="http://code.jquery.com/jquery-1.9.1.min.js"></script>
08:    <script src="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.js"></script>
09:    <meta name="viewport" content="width=device-width, initial-scale=1">
```

```
10:  </head>
11:  <body>
12:    <div data-role="page">
13:      <div data-role="header">
14:        <h1>航海王</h1>
15:      </div>
16:      <div data-role="content">
17:        <p>海贼王黄金．罗杰遗留下一个被称为 ONEPIECE 的神秘宝藏，
18:        而主角“鲁夫”找了海盗克星“索隆”、女贼“娜美”、
19:        可爱驯鹿“乔巴”等几位伙伴要一起寻找传说中的宝藏。</p>
20:      </div>
21:      <div data-role="footer">
22:        <h4>&copy;快乐影视</h4>
23:      </div>
24:    </div>
25:  </body>
26:</html>
```

页首：第 13～15 行　内容：第 16～20 行　页尾：第 21～23 行　页面：第 12～24 行

目前这份文件包含一个页面，图 10-5 是在智能手机上的浏览结果，而图 10-6 是在 Opera Mobile Emulator 上的模拟界面。

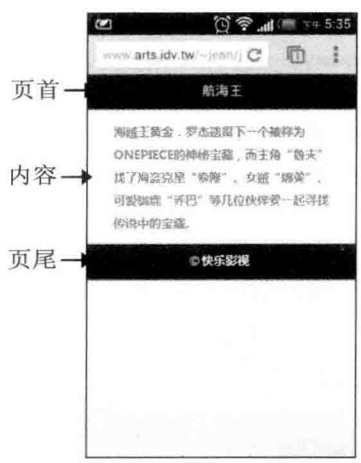


图 10-5

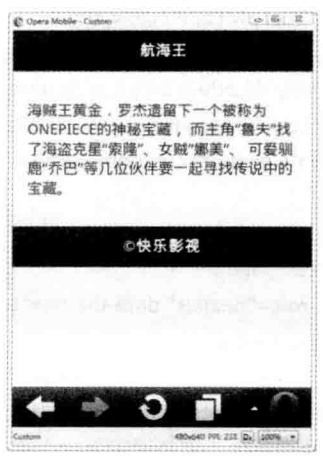


图 10-6

10.3　主题

为了让移动版网页显得更美观，jQuery Mobile 提供了 data-theme="a-z" 属性用来指定用户界面组件的主题（theme），这是一组关于文档版式配置、样式与颜色的定义，而且 jQuery Mobile 内置了如表 10-2 所示的 5 种主题。

表 10-2　内置的主题

主题	颜色
data-theme="a"	黑色
data-theme="b"	蓝色
data-theme="c"	银色
data-theme="d"	灰色
data-theme="e"	黄色

页面预设采用主题 c，但我们可以针对页首、页尾、按钮、列表视图、工具栏等用户界面组件指定主题，若某个元素没有指定主题，就会沿用其父元素的主题。图 10-7 是一个例子，页首和页尾分别指定套用主题 b、e，而内容没有指定主题，故会沿用其父元素（页面）的主题 c。

\ch10\jQM1b.html

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>我的 jQuery Mobile 程序</title>
    <link rel="stylesheet" href="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.css" />
    <script src="http://code.jquery.com/jquery-1.9.1.min.js"></script>
    <script src="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.js"></script>
    <meta name="viewport" content="width=device-width, initial-scale=1">
  </head>
  <body>
    <div data-role="page">
      <div data-role="header" data-theme="b">
        <h1>航海王</h1>
      </div>
      <div data-role="content">
        <p>海贼王黄金．罗杰遗留下一个被称为 ONEPIECE 的神秘宝藏，
        而主角“鲁夫”找了海盗克星“索隆”、女贼“娜美”、
        可爱驯鹿“乔巴”等几位伙伴要一起寻找传说中的宝藏。</p>
      </div>
      <div data-role="footer" data-position="fixed" data-theme="e">
        <h4>&copy;快乐影视</h4>
      </div>
    </div>
  </body>
</html>
```

页首指定套用主题 b (蓝色)

内容沿用其父元素 (页面) 的主题 c (灰色)

加上此属性将页尾固定显示在页面下方

页尾指定套用主题 e (黄色)

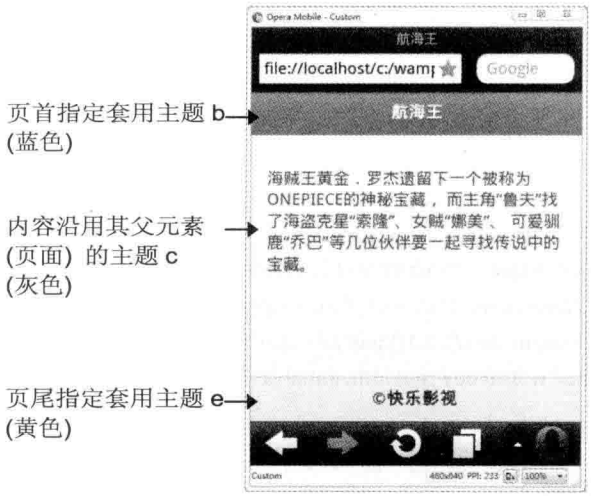


图 10-7

data-theme 属性的合法值为 "a-z"，也就是 "a"、"b"、"c"、...、"z"，除了 "a-e" 等 jQuery Mobile 内建的主题之外，您也可以连接到 jQuery Mobile 官方网站提供的 ThemeRoller（http://jquerymobile.com/themeroller/），自定义专用的主题，进一步的说明可以到 ThemeRoller 网站查看。

10.4　超链接

jQuery Mobile 文件的超链接可以分为下列几种类型。

- 内部链接（internal link）：链接同一份 jQuery Mobile 文件的页面。
- 外部链接（external link）：链接不同的 jQuery Mobile 文件，而且两份文件必须放在相同网络中。
- 绝对外部链接（absolute external link）：链接非 jQuery Mobile 文件或其他网络中的网页。

10.4.1　内部链接

我们直接以下面的例子说明如何建立内部链接，也就是链接同一份 jQuery Mobile 文件的页面，当用户单击第一个页面的第二页超链接时，就会打开第二个页面；相反的，当用户单击第二个页面的第一页超链接或移动版浏览器的上一页按钮时，就会返回第一个页面，如图 10-8 所示。

\ch10\jQM2.html

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:    <title>我的 jQuery Mobile 程序</title>
06:    <link rel="stylesheet" href="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.css" />
07:    <script src="http://code.jquery.com/jquery-1.9.1.min.js"></script>
08:    <script src="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.js"></script>
09:    <meta name="viewport" content="width=device-width, initial-scale=1">
10:  </head>
11:  <body>
12:    <div data-role="page" id="firstPage">
13:      <div data-role="header">
14:        <h1>航海王</h1>
15:      </div>
16:      <div data-role="content">
17:        <p>这是第一页，前往<a href="#secondPage">第二页</a>。</p>
18:      </div>
19:      <div data-role="footer">
20:        <h4>&copy;快乐影视</h4>
21:      </div>
22:    </div>
23:    <div data-role="page" id="secondPage">
24:      <div data-role="header">
25:        <h1>航海王</h1>
26:      </div>
27:      <div data-role="content">
28:        <p>这是第二页，返回<a href="#firstPage">第一页</a>。</p>
29:      </div>
30:      <div data-role="footer">
31:        <h4>&copy;快乐影视</h4>
32:      </div>
33:    </div>
34:  </body>
35:</html>
```

- 12～22：定义第一个页面并将其 id 属性指定为 "firstPage"。
- 23～33：定义第二个页面并将其 id 属性指定为 "secondPage"。
- 17：使用 <a> 元素将字符串“第二页”指定为链接到第二个页面的超链接，识别名称前面要加上“#”符号。
- 28：使用 <a> 元素将字符串“第一页”指定为链接到第一个页面的超链接，识别名

称前面要加上“#符号。

图 10-8

请注意，虽然我们在 <head> 区块内使用 <title> 元素将文件的标题指定为“我的 jQuery Mobile 程序”，但在浏览结果中，两个页面的标题都是页首中的“航海王”，若要指定各个页面的标题，可以使用 jQuery Mobile 提供的 data-title 属性，举例来说，假设在第 12、23 行加上 data-title 属性，如下图 10-9 所示，就可以将两个页面的标题分别指定为“第一页”和“第二页”：

```
12:     <div data-role="page" id="firstPage" data-title="第一页">
...
23:     <div data-role="page" id="secondPage" data-title="第二页">
```

图 10-9

10.4.2 外部链接

我们直接以下面的例子说明如何建立外部链接，也就是链接不同的 jQuery Mobile 文件，此例有两份 jQuery Mobile 文件 <jQM3a.html> 和 <jQM3b.html> 放在相同网络中，当用户单击 <jQM3a.html> 的“第二份文件”超链接时，就会打开 <jQM3b.html>；相反的，当用户单击 <jQM3b.html> 的“第一份文件”超链接或移动版浏览器的上一页按钮时，就会返回 <jQM3a.html>，如图 10-10 所示。

\ch10\jQM3a.html

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:    <title>我的 jQuery Mobile 程序</title>
06:    <link rel="stylesheet" href="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.css" />
07:    <script src="http://code.jquery.com/jquery-1.9.1.min.js"></script>
08:    <script src="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.js"></script>
09:    <meta name="viewport" content="width=device-width, initial-scale=1">
10:  </head>
11:  <body>
12:    <div data-role="page">
13:      <div data-role="header">
14:        <h1>航海王</h1>
15:      </div>
16:      <div data-role="content">
17:        <p>这是第一份文件，前往<a href="jQM3b.html">第二份文件</a>。</p>
18:      </div>
19:      <div data-role="footer">
20:        <h4>&copy;快乐影视</h4>
21:      </div>
22:    </div>
23:  </body>
24:</html>
```

指定链接到第二份 jQuery Mobile 文件

\ch10\jQM3b.html

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:    <title>我的 jQuery Mobile 程序</title>
06:    <link rel="stylesheet" href="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.css" />
07:    <script src="http://code.jquery.com/jquery-1.9.1.min.js"></script>
08:    <script src="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.js"></script>
```

```
09:     <meta name="viewport" content="width=device-width, initial-scale=1">
10:   </head>
11:   <body>
12:     <div data-role="page">
13:       <div data-role="header">
14:         <h1>航海王</h1>
15:       </div>
16:       <div data-role="content">
17:         <p>这是第二份文件，返回<a href="jQM3a.html">第一份文件</a>。</p>
18:       </div>
19:       <div data-role="footer">
20:         <h4>&copy;快乐影视</h4>
21:       </div>
22:     </div>
23:   </body>
24:</html>
```

指定链接到第一份 jQuery Mobile 文件

1. 按“第二份文件”　2. 开启第二份文件，然后按“第一份文件”　3. 返回第一份文件

图 10-10

10.4.3　绝对外部链接

我们直接以下面的例子说明如何建立绝对外部链接，也就是链接非 jQuery Mobile 文件或其他网域的网页，当用户单击 <jQM4.html>中的“百度”超链接时，就会打开百度网站，要注意的是第 17 行必须加上 data-rel="external" 属性指定要建立绝对外部链接，该程序的运行结果如图 10-11。

\ch10\jQM4.html

```
01:<!doctype html>
02:<html>
03:   <head>
04:      <meta charset="utf-8">
05:      <title>我的 jQuery Mobile 程序</title>
06:      <link rel="stylesheet" href="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.css" />
07:      <script src="http://code.jquery.com/jquery-1.9.1.min.js"></script>
08:      <script src="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.js"></script>
09:      <meta name="viewport" content="width=device-width, initial-scale=1">
10:   </head>
11:   <body>
12:      <div data-role="page">
13:         <div data-role="header">
14:            <h1>航海王</h1>
15:         </div>
16:         <div data-role="content">
17:            <p>前往<a href="http://www.baidu.com/" data-rel="external">百度</a></p>
18:         </div>
19:         <div data-role="footer">
20:            <h4>&copy;快乐影视</h4>
21:         </div>
22:      </div>
23:   </body>
24:</html>
```

图 10-11

第 17 行之所以必须加上 data-rel="external" 属性，主要是因为 jQuery Mobile 为了减少加载时间，预设会以 Ajax 方式加载页面或网页，也就是只加载页面或网页有更新的部分，但是对于其他网域的网页，我们必须取消 jQuery Mobile 的 Ajax 功能，才能在每次打开其他网域的网页时才会重载。除了使用 data-rel="external" 属性，我们也可以改用 data-ajax="false" 属性或 target="_blank" 属性取消 Ajax 功能。

外部链接虽然能够链接相同网络中的不同的 jQuery Mobile 文件，但被链接的文件只能包含单一页面，否则会加载错误，若要链接不同份 jQuery Mobile 文件的页面，就必须改用绝对外部链接。举例来说，假设 <jQM2.html> 文件包含两个页面，id 分别为 "firstPage" 和 "secondPage"，而另一份 <jQM1.html> 文件要链接 <jQM2.html> 文件的 "secondPage" 页面，那么此超链接可以写成如下的形式，文件名称与页面名称中间以 # 符号连接：

```
<a href="jQM2.html#secondPage" data-rel="external"></a>
```

10.5　对话框

jQuery Mobile 文件的对话框就像另一种页面（page），只是周围多了边框，左上角多了关闭按钮，适合用来显示确认信息、提示信息或没有层级关系的信息。下面是一个例子，当用户单击 <jQM5.html> 的“鲁夫简介”超链接时，就以对话框的形式显示 <introduction.html> 的介绍文字，单击左上角的“关闭”按钮即可返回 <jQM5.html>，值得注意的是 <jQM5.html> 的第 18 行在 <a> 元素内加上 data-rel="dialog" 属性，指定以对话框的形式显示超链接的内容，如图 10-12 所示。

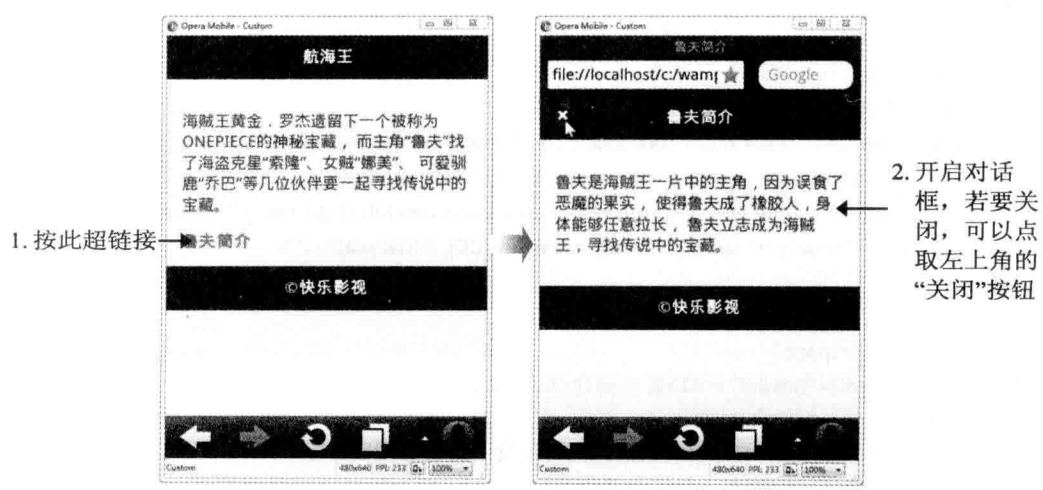


图 10-12

\ch10\jQM5.html

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:    <title>我的 jQuery Mobile 程序</title>
06:    <link rel="stylesheet" href="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.css" />
07:    <script src="http://code.jquery.com/jquery-1.9.1.min.js"></script>
08:    <script src="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.js"></script>
09:    <meta name="viewport" content="width=device-width, initial-scale=1">
10:  </head>
11:  <body>
12:    <div data-role="page">
13:      <div data-role="header"> <h1>航海王</h1> </div>
14:      <div data-role="content">
15:        <p>海贼王黄金·罗杰遗留下一个被称为 ONEPIECE 的神秘宝藏，
16:        而主角“鲁夫”找了海盗克星“索隆”、女贼“娜美”、
17:        可爱驯鹿“乔巴”等几位伙伴要一起寻找传说中的宝藏。</p>
18:        <a href="introduction.html" data-rel="dialog">鲁夫简介</a>
19:      </div>
20:      <div data-role="footer"> <h4>&copy;快乐影视</h4> </div>
21:    </div>
22: </body>
23:/html>
```

指定以对话框的形式显示超链接的内容

\ch10\introduction.html

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:    <title>我的 jQuery Mobile 程序</title>
06:    <link rel="stylesheet" href="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.css" />
07:    <script src="http://code.jquery.com/jquery-1.9.1.min.js"></script>
08:    <script src="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.js"></script>
09:    <meta name="viewport" content="width=device-width, initial-scale=1">
10:  </head>
11:  <body>
12:    <div data-role="page">
13:      <div data-role="header"><h1>鲁夫简介</h1></div>
14:      <div data-role="content">
15:        <p>鲁夫是海贼王一片中的主角，因为误食了恶魔的果实，
16:        使得鲁夫成了橡胶人，身体能够任意拉长，
```

```
17:          鲁夫立志成为海贼王，寻找传说中的宝藏。</p>
18:        </div>
19:        <div data-role="footer"> <h4>&copy;快乐影视</h4></div>
22:     </div>
23:  </body>
24:</html>
```

10.6　按钮

jQuery Mobile 内置了丰富的用户界面组件（UI component），例如按钮、工具栏、折叠面板、导航条、列表视图、表单等。在本节中，我们要介绍按钮（button），jQuery Mobile 针对 HTML 超链接元素（<a>）和表单输入元素（type="button"、type="submit"、type="reset"）提供了更美观、更容易单击的按钮，之所以有此设计，主要是因为超链接元素的文字经常出现在字里行间，单击区域也局限于文字本身，用户往往无法精确单击到超链接，而按钮的单击区域则相对较大。

10.6.1　建立按钮

若要将超链接元素或表单输入元素描绘成按钮，可以加上 data-role="button" 属性，例如：

```
<a href="story.html" data-role="button">故事介绍</a>
<a href="role.html" data-role="button">角色介绍</a>
<a href="images.html" data-role="button">精彩图片</a>
```

浏览结果如图 10-13 所示，每个按钮默认会占用一行。jQuery Mobile 还提供了迷你按钮，只要加上 data-mini="true" 属性，就会得到如图 10-14 所示的浏览结果，若再加上 data-inline="true" 属性，则会将按钮并排在同一行，如图 10-15 所示。

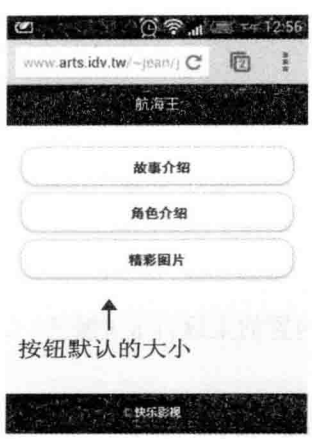


图 10-13

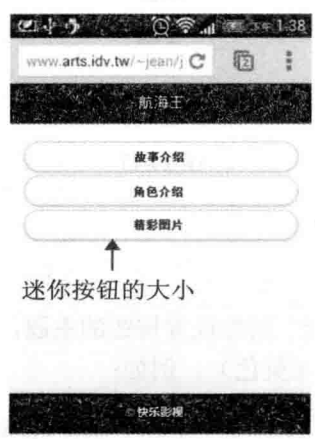


图 10-14

图 10-15

10.6.2 设置按钮的图标

若要设置按钮的图标，可以加上 data-icon="*icon-name*" 属性，其中 *icon-name* 就是图标的名称，对照如图 10-16 所示。

图 10-16

按钮的图标默认会显示在左侧，若要变更图标的位置，可以加上 data-iconpos="left | right | top | bottom | notext" 属性（默认值为 left），例如：

```
<a href="story.html" data-role="button" data-icon="arrow-r" data-iconpos="left">故事介绍</a>
<a href="role.html" data-role="button" data-icon="arrow-r" data-iconpos="right">角色介绍</a>
<a href="images.html" data-role="button" data-icon="arrow-r" data-iconpos="top">精彩图片</a>
<a href="theater.html" data-role="button" data-icon="arrow-r" data-iconpos="bottom">剧场版</a>
<a href="discuss.html" data-role="button" data-icon="arrow-r" data-iconpos="notext">讨论区</a>
```

浏览结果如图 10-17 所示。

图 10-17

10.6.3 设置按钮的主题

我们可以使用 data-theme="a-z" 属性设置按钮的主题，内置的主题有 a（黑色）、b（蓝色）、c（银色）、d（灰色）、e（黄色），例如：

```
<a href="story.html" data-role="button" data-icon="arrow-r" data-theme="e">故事介绍</a>
```

10.6.4　设置按钮的特殊效果

按钮默认为圆角加阴影，若要取消加阴影，可以加上 data-shadow="false" 属性；若要取消圆角，可以加上 data-corners="false" 属性，例如：

```
<a href="story.html" data-role="button" data-icon="arrow-r">故事介绍</a>
<a href="role.html" data-role="button" data-icon="arrow-r" data-shadow="false">角色介绍</a>
<a href="images.html" data-role="button" data-icon="arrow-r" data-corners="false">精彩图片</a>
```

浏览结果如图 10-18 所示。

图 10-18

10.6.5　设置控件组

有时我们可能需要将几个有关联的按钮组合在一起，此时可以利用 元素加上 data-role="controlgroup" 属性定义一个控件组容器，然后将按钮放在这个容器里面，例如：

```
<div data-role="controlgroup">
   <a href="plus.html" data-role="button" data-icon="plus">新增</a>
   <a href="delete.html" data-role="button" data-icon="delete">删除</a>
   <a href="edit.html" data-role="button" data-icon="edit">编辑</a>
</div>
```

浏览结果如图 10-19 所示，控件组里面的按钮默认会垂直排列，若要变更为水平排列，可以加上 data-type="horizontal" 属性，浏览结果如图 10-20 所示。

图 10-19

图 10-20

10.7　工具栏

jQuery Mobile 提供了页首行和页尾行两个标准的工具栏（toolbar），虽然不是必需的区

域，但多数页面通常会包含这两者。

10.7.1 页首行

页首行（header）位于页面上方，通常用来放置标题或“返回”等按钮，标题文字建议以 <h1> 元素来标记，但使用 <h2>～<h6> 元素亦可，例如：

```
<div data-role="header">
  <h1>航海王</h1>
</div>
```

当内容超过页面的长度时，页首行可能会被卷出页面，如图 10-21 左图所示，此时可以加上 data-position="fixed" 属性，将它固定显示在页面顶端，如图 10-21 右图所示：

```
<div data-role="header" data-position="fixed">
  <h1>航海王</h1>
</div>
```

图 10-21

我们可以在页首行加入按钮，例如下面的语句会在页首行加入一个“回首页”按钮，而且按钮默认的位置在左侧，浏览结果如图 10-22 所示（注：若要将按钮改放在右侧，可以加上 class="ui-btn-right" 属性）。

```
<div data-role="header">
  <a href="home.html" data-role="button" data-icon="home">回首页</a>
  <h1>航海王</h1>
</div>
```

图 10-22

此外，页首行预设的主题为 a（黑色），若要设置页首行的主题，可以加上 data-theme="a-z" 属性，内置的主题有 a（黑色）、b（蓝色）、c（银色）、d（灰色）、e（黄色）。

10.7.2　页尾行

页尾行（footer）位于页面下方，通常用来放置版权声明、联络信息、具有预览功能或相关联的一组按钮，例如下面的语句是在页尾行加入 3 个按钮，结果如图 10-23 所示：

```
<div data-role="footer">
   <a href="add.html" data-role="button" data-icon="plus">新增</a>
   <a href="up.html" data-role="button" data-icon="arrow-u">上一笔</a>
   <a href="down.html" data-role="button" data-icon="arrow-d">下一笔</a>
</div>
```

图 10-23

同样的，页尾行预设的主题为 a（黑色），若要设置页尾行的主题，可以加上 data-theme="a-z" 属性，内置的主题有 a（黑色）、b（蓝色）、c（银色）、d（灰色）、e（黄色）。

此外，页尾行虽然位于页面下方，但实际的高度会随着内容自动调整，当内容超过页面的长度时，页尾行可能会被卷出页面，此时可以加上 data-position="fixed" 属性，将它固定显示在页面底部。

10.8　导航条

jQuery Mobile 提供了一个基本的 navbar widget，可以用来创建最多包含 5 个按钮的导航条（navigation bar）。导航条可以放在页首行、页尾行或内容区等区域，下面是一个例子<\ch10\jQMUI3.html>，它将导航条放在页首行，如图 10-24 所示。

```
01:<div data-role="page">
02:   <div data-role="header">
03:      <div data-role="navbar">
04:         <ul>
05:            <li><a href="story.html" class="ui-btn-active ui-state-persist">故事介绍</a></li>
06:            <li><a href="role.html">角色介绍</a></li>
07:            <li><a href="images.html">精彩图片</a></li>
08:         </ul>
09:      </div>
10:   </div>
11:   <div data-role="content">
```

```
12:     <img src="piece.jpg" width="100%">
13:   </div>
14:   <div data-role="footer" data-position="fixed">
15:     <h4>&copy;快乐影视</h4>
16:   </div>
17:</div>
```

- 03、09: 在页首行里面加上 元素和 data-role="navbar" 属性表示要建立导航条。
- 04 ~08: 使用 <ul> 和 <li> 元素定义项目列表，这些项目将成为导航条的按钮，请注意第 05 行的 ui-btn-active 表示将此项目设置为默认的按钮，ui-state-persist 表示每次加载页面都将此项目恢复到预先选取的状态。

图 10-24

❖ 设置导航条按钮的图标与位置

若要设置导航条按钮的图标，可以在 <li> 元素加上 data-icon="*icon-name*" 属性，其中 *icon-name* 是图标的名称，第 10.6.2 小节列出了名称对照表。此外，图标默认会显示在导航条按钮的上方，若要改变图标的位置，可以在 元素加上 data-iconpos="left | right | top | bottom | notext" 属性，结果如图 10-25 所示：

```
<div data-role="header">
  <div data-role="navbar" data-iconpos="left">
    <ul>
      <li><a href="add.html" class="ui-btn-active ui-state-persist" data-icon="plus">新增</a></li>
      <li><a href="delete.html" data-icon="delete">删除</a></li>
      <li><a href="edit.html" data-icon="edit">编辑</a></li>
    </ul>
  </div>
</div>
```

图 10-25

❖ 设置导航条的主题

导航条预设会沿用其父容器的主题，若要变更主题，可以在其父容器加上 data-theme="a-z" 属性，例如图 10-26 是在页首行加上 data-theme="e" 属性的浏览结果。

图 10-26

10.9　可折叠区块

由于移动设备的屏幕较小，因此，我们可以善用 jQuery Mobile 提供的可折叠区块（collapsible block）将某个项目或某个主题的内容折叠起来，待用户单击该项目再展开内容，例如下面的语句是在内容里面加上 元素和 data-role="collapsible" 属性表示要创建可折叠区块，结果如图 10-27 所示：

```
<div data-role="content">
  <div data-role="collapsible">
    <h3>乔巴</h3>
    <p>身份船医，梦想成为能治百病的神医。</p>
  </div>
</div>
```

图 10-27

下面是几个小技巧。

- 可折叠区块预设是折叠起来的状态，若要一加载页面就展开内容，可以加上 data-collapsed="false" 属性。
- 可折叠区块的折叠图标默认是一个加号，若要改变折叠图标，可以加上 data-collapsed-icon 属性，例如 data-collapsed-icon="arrow-r" 是将折叠图标设置为向右箭头；相反的，展开图标默认是一个减号，若要变更展开图标，可以加上 data-expanded-icon 属性，例如 data-expanded-icon="arrow-d" 是将展开图标设置为向下箭头。

- 折叠图标和展开图标默认的位置在左侧，若要改变位置，可以加上 data-iconpos="left | right | top | bottom | notext" 属性，例如 data-iconpos="right" 是将图标放在右侧。

10.10 可折叠区块群组

我们可以利用 jQuery Mobile 提供的 Collapsible set Widget 将多个可折叠区块放在一起成为一个群组，称为可折叠区块群组（collapsible set of collapsible blocks），下面是一个例子<\ch10\jQMUI4.html>。

```
01:<div data-role="content">
02:    <div data-role="collapsible-set">
03:       <div data-role="collapsible">
04:          <h3>乔巴</h3>
05:          <p>身份船医，梦想成为能治百病的神医。</p>
06:       </div>
07:       <div data-role="collapsible">
08:          <h3>索隆</h3>
09:          <p>主角鲁夫的伙伴，梦想成为世界第一的剑士。</p>
10:       </div>
11:       <div data-role="collapsible">
12:          <h3>佛朗基</h3>
13:          <p>传说中的船匠－汤姆的弟子，打造了千阳号</p>
14:       </div>
15:    </div>
16:</div>
```

- 02、15：在内容里面加上 元素和 data-role="collapsible-set" 属性，表示要创建可折叠区块群组。
- 03 ~ 14：加上 3 个 元素和 data-role="collapsible" 属性表示要创建 3 个可折叠区块。

结果如图 10-28 所示。

图 10-28

10.11　列表视图

对于一些条列式的项目、超链接或内容，我们可以使用 jQuery Mobile 提供的 Listview Widget 建立列表视图（listview），让这些数据排列得井然有序。

10.11.1　创建列表视图

创建列表视图最常见的方式是使用 <ul> 元素和 data-role="listview" 属性，然后使用 <li> 元素指定各个项目，下面是一个例子 <\ch10\jQMUI5.html>，浏览结果如图 10-29 所示。

```
<div data-role="content">
  <ul data-role="listview">
    <li>故事介绍</li>
    <li>角色介绍</li>
    <li>精彩图片</li>
  </ul>
</div>
```

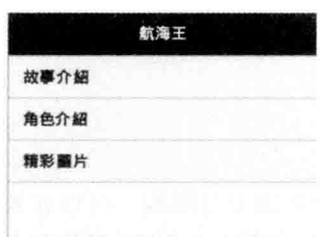


图 10-29

当然这些项目可以具有超链接功能，例如：

```
<div data-role="content">
  <ul data-role="listview">
    <li><a href="story.html">故事介绍</a></li>
    <li><a href="role.html">角色介绍</a></li>
    <li><a href="images.html">精彩图片</a></li>
  </ul>
</div>
```

浏览结果如图 10-30 所示，每个项目的右侧会出现一个向右箭头，表示为超链接。

若要令超链接项目内缩且为圆角，可以在 <ul data-role="listview"> 元素加上 data-inset="true" 属性，浏览结果如图 10-31 所示。

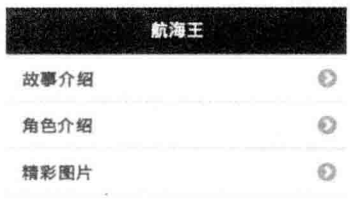


图 10-30

图 10-31

或者，我们也可以将 <ul> 元素换成 <ol> 元素，此时会在项目前面加上编号，浏览结果如图 10-32 所示。

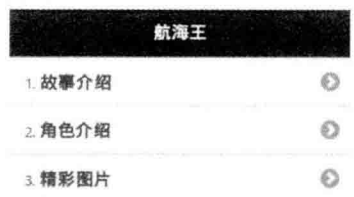


图 10-32

10.11.2 设置分隔线

若要将列表视图中的项目分组并加上分隔线，可以在要作为分隔线的项目（ 元素）中加入 data-role="list-divider" 属性，例如 <\ch10\jQMUI6.html>：

```
01:<div data-role="content">
02:   <ul data-role="listview">
03:      <li data-role="list-divider">个性化</li>
04:      <li>面板</li>
05:      <li>桌布</li>
06:      <li>主画面</li>
07:      <li data-role="list-divider">账户管理</li>
08:      <li>Facebook</li>
09:      <li>Google</li>
10:      <li>气象</li>
11:      <li>邮件</li>
12:   </ul>
13:</div>
```

浏览结果如图 10-33 所示，分隔线预设的主题为 b（蓝色），若要加以变更，可以在第 02 行的 <ul> 元素加上 data-divider-theme="a-z" 属性。

图 10-33

10.11.3　设置计数气泡与侧边内容

计数气泡（count bubble）是用圆圈圈起来的数字，显示在项目名称右侧，通常用来表示还有多少尚未完成的工作或尚未读取的项目，下面是一个例子 <\ch10\jQMUI9.html>，重点是在要设置计数气泡的元素（例如 元素）中加上 class="ui-li-count" 属性，浏览结果如图 10-34 所示。

```
<div data-role="content">
  <ul data-role="listview">
    <li><a href="inbox.html">收件箱<span class="ui-li-count">20</span></a></li>
    <li><a href="outbox.html">发件箱<span class="ui-li-count">0</span></a></li>
    <li><a href="drafts.html">草稿<span class="ui-li-count">1</span></a></li>
    <li><a href="drafts.html">寄件备份<span class="ui-li-count">5</span></a></li>
  </ul>
</div>
```

图 10-34

若是将上面 5 个 元素的 class="ui-li-count" 属性换成 class="ui-li-aside" 属性，那么数字就不会呈现气泡外观，而是以文字显示在项目名称右侧，如图 10-35 所示。

图 10-35 所示

10.11.4 设置搜索功能

我们可以在列表视图中设置搜索功能，以搜索包含用户输入之文字的项目，举例来说，只要在 <\ch10\jQMUI6.html> 的 <ul> 元素中加上 data-filter="true" 属性，就会出现如图 10-36 所示的搜索方块，此时若输入“o”，就会搜索包含“o”的项目。

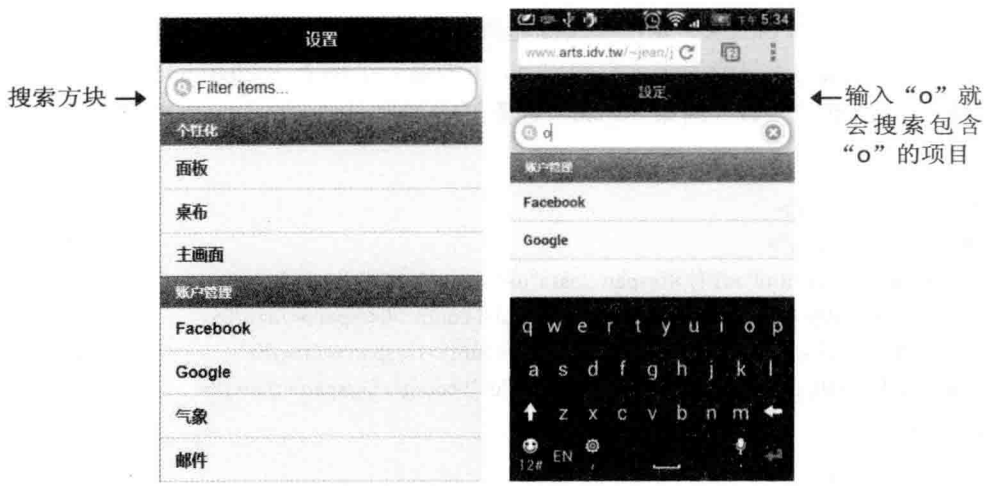


图 10-36

```
01:<div data-role="content">
02:   <ul data-role="listview" data-filter="true">
03:      <li data-role="list-divider">个人化</li>
04:      <li>面板</li>
05:      <li>桌布</li>
06:      <li>主画面</li>
07:      <li data-role="list-divider">账户管理</li>
08:      <li>Facebook</li>
09:      <li>Google</li>
10:      <li>气象</li>
```

```
11:     <li>邮件</li>
12:   </ul>
13:</div>
```

搜索方块预设的主题为 c（银色），若要加以变更，可以加上 data-filter-theme ="a-z" 属性；此外，属性搜索方块默认的文字为“Filter items...”，若要自定义文字，可以加上 data-filter-placeholder="*text*" 属性，其中 *text* 为自定义文字。

10.11.5　设置图标与缩略图

我们可以针对列表视图的项目设置图标或缩略图，图标（icon）为 16×16 像素，而且必须套用 ui-li-icon 类，缩略图（thumbnail）为 80×80 像素，无须套用 ui-li-icon 类。图标会显示在项目的左侧，它和用来表示超链接的向右箭头图标是不同的，下面是一个例子 <\ch10\jQMUI10.html>，浏览结果如图 10-37 所示。

```
<div data-role="page">
  <div data-role="header">
    <h1>账户管理</h1>
  </div>
  <div data-role="content">
    <ul data-role="listview">
      <li><img src="fb2.png"><h3>Facebook</h3><p>查看脸书最新动态</p></li>
      <li><img src="google2.png"><h3>Google</h3><p>查看 Google 账户</p></li>
      <li><img src="weather.png" class="ui-li-icon">气象</li>
      <li><img src="email.png" class="ui-li-icon">邮件</a></li>
    </ul>
  </div>
</div>
```

设定缩略图(80×80像素)

设定图标(图像文件为 16×16 像素，必须套用 ui-li-icon 类)

图 10-37

10.12 表单

jQuery Mobile 支持标准的 HTML 表单元素，若您对于 HTML 有哪些表单元素感到陌生，可以参阅第 8 章，此处不再重复说明。jQuery Mobile 表单其实和标准的 HTML 表单差不多，都是类似如下的形式：

```
<form method="post" action="confirm.php">
  ...
</form>
```

在预设的情况下，jQuery Mobile 会以 Ajax 方式提交表单，若要取消 Ajax 功能，改用标准的 HTTP Request，可以在 <form> 元素中加上 data-ajax="false" 属性。

10.12.1 字段容器

jQuery Mobile 提供的字段容器（field container）虽然不是一种必要性的容器，但是将表单字段放入此种容器，却可以让表单更美观，因为字段容器可以根据移动设备当前是横向或纵向，自动将表单字段排列整齐，而且还会加上分隔线，下面是一个例子 <\ch10\jQMform1.html>，其中第 03 行、第 08 行和第 09 行、第 11 行是各自利用一个 元素并加上 data-role="fieldcontain" 属性来设置字段容器。

```
01:<div data-role="content">
02:   <form>
03:     <div data-role="fieldcontain">
04:       <label for="username">输入姓名：</label>
05:       <input type="text" name="username">
06:       <label for="userpwd">输入密码：</label>
07:       <input type="password" name="userpwd">
08:     </div>
09:     <div data-role="fieldcontain">
10:       <input type="submit" value="确定">
11:     </div>
12:   </form>
13:</div>
```

第一个字段容器里面有一个单行文本框和一个密码字段

第二个字段容器里面有一个提交按钮

图 10-38 为移动设备呈纵向时的浏览结果，此时标签和字段各排一列，两个字段容器之间则有一条分隔线，而图 10-39 为移动设备呈横向时的浏览结果，此时标签和字段会排成一行。

图 10-38

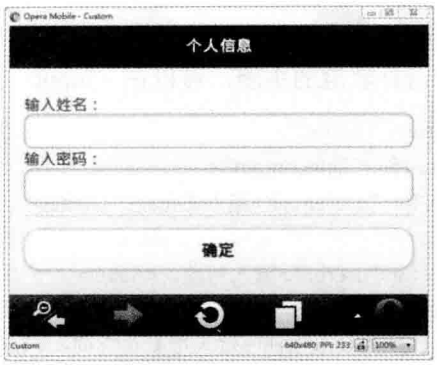


图 10-39

您可以试着在表单字段中输入数据，由于“输入姓名：”字段的输入类型是单行文本框（type="text"），故会将数据显示出来，而“输入密码：”字段的输入类型是密码字段（type="password"），故会将数据显示为圆点，如图 10-40 所示。

图 10-40

10.12.2　文字输入字段

除了前一节示范的单行文本框（<input type="text">）和密码字段（<input type="password">），HTML 还提供了其他文字输入字段，例如：

文字输入字段	输入类型
<input type="email">	电子邮件地址
<input type="url">	网址
<input type="search">	搜索字段
<input type="tel">	电话号码
<input type="number">	数字
<input type="range">	以滑杆输入指定范围内的数字

我们可以直接将这些文字输入字段放入 jQuery Mobile 的字段容器，下面是一个例子 <\ch10\jQMform2.html>，其中滑杆字段可以让用户通过移动滑杆的方式输入指定范围内的数字，若要设置整个滑杆字段的主题，可以在 <input> 元素中加上 data-theme="a-z" 属性，若只要设置滑杆轨道的主题，可以在 <input> 元素中加上 data-track-theme="a-z" 属性。

```
<div data-role="content">
  <div data-role="fieldcontain">
    <label for="useremail">输入 E-mail：</label>
    <input type="email" name="useremail">
    <label for="userurl">输入网址：</label>
    <input type="url" name="userurl">
    <label for="usersearch">输入搜索关键词：</label>
    <input type="search" name="usersearch">
    <label for="usertel">输入电话：</label>
    <input type="tel" name="usertel">
    <label for="usernumber">输入数字 1-10：</label>
    <input type="number" name="usernumber" max="10" min="1">
    <label for="userrange">选择数字 1-10：</label>
    <input type="range" name="userrange" max="10" min="1">
  </div>
</div>
```

浏览结果如图 10-41 所示，若要显示迷你版的文字输入字段，可以在 <input> 元素中加上 data-mini="true" 属性。

图 10-41

10.12.3 日期时间输入字段

HTML 5 还新增了如下的日期时间输入字段，我们可以直接将这些字段放入 jQuery Mobile 的字段容器，下面是一个例子 <\ch10\jQMform3.html>。

日期时间输入字段	说明
<input type="date">	日期
<input type="time">	时间
<input type="datetime">	UTC 世界标准时间
<input type="month">	月份
<input type="week">	一年的第几周
<input type="datetime-local">	本地日期时间

```
<div data-role="fieldcontain">
   <label for="userdate">输入日期：</label>
   <input type="date" name="userdate">
</div>
```

浏览结果如图 10-42 所示，不同的浏览器可能提供不同的实现方式。

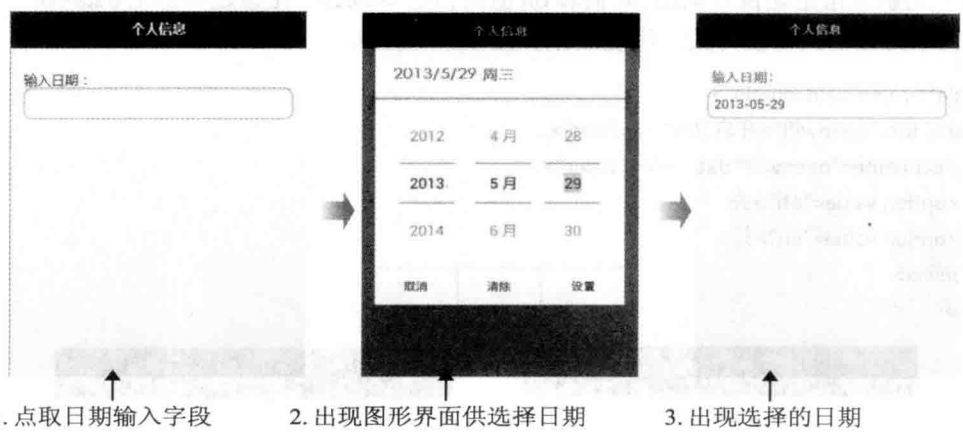


图 10-42

10.12.4　多行文本框

我们可以将多行文本框放入 jQuery Mobile 的字段容器，下面是一个例子<\ch10\jQMform4.html>。

```
<div data-role="fieldcontain">
   <label for="userintro">输入自我介绍：</label>
   <textarea name="userintro"></textarea>
</div>
```

浏览结果如图 10-43 所示，多行文本框会随着输入的文字变多而自动变大，若要显示迷你版的多行文本框，可以在 <textarea> 元素中加上 data-mini="true" 属性。

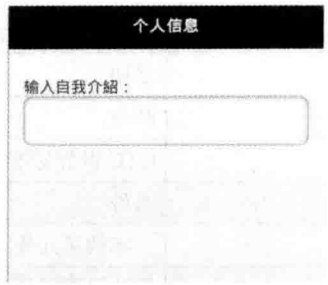


图 10-43

10.12.5 拨动式切换开关

我们可以利用 jQuery Mobile 提供的 Slider Widget 制作拨动式切换开关（flip toggle switch），下面是一个例子 <\ch10\jQMform6.html>，重点在于使用 <select> 元素搭配两个 <option> 元素来指定切换开关的 off 值和 on 值，而且 <select> 元素还要加上 data-role="slider" 属性，表示为拨动式切换开关。浏览结果如图 10-44 所示。

```
01:<div data-role="fieldcontain">
02:   <label for="openwifi">开启 Wi-Fi：</label>
03:   <select name="openwifi" data-role="slider">
04:      <option value="off">关
05:      <option value="on">开
06:   </select>
07:</div>
```

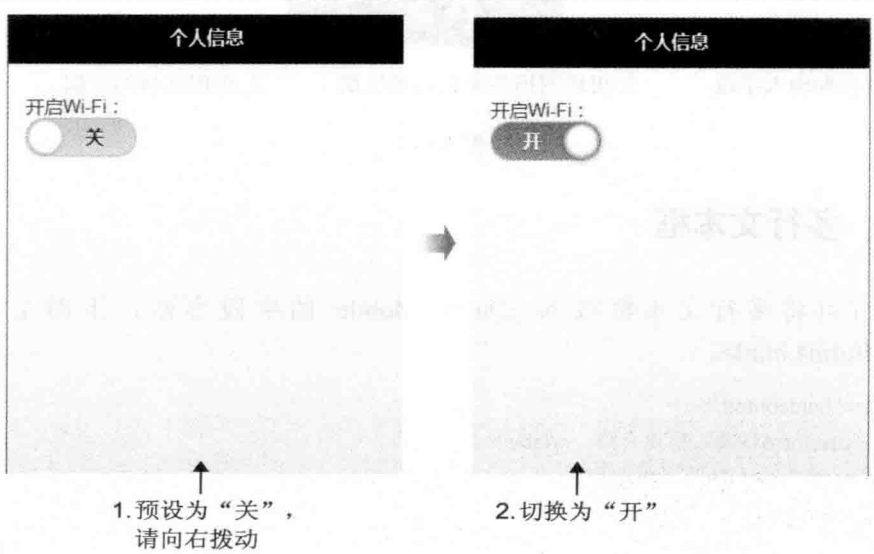


图 10-44

若要显示迷你版的切换开关，可以在 <select> 元素中加上 data-mini="true" 属性。若要

设置整个切换开关的主题，可以在 <select> 元素加上 data-theme="a-z" 属性，若只要设置开关轨道的主题，可以在 <select> 元素中加上 data-track-theme="a-z" 属性。

10.12.6　下拉式菜单

和传统网页一样，我们可以使用 <select> 元素搭配 <option> 元素在移动版网页中插入下拉式菜单，不同的是 jQuery Mobile 会以可触控的按钮形式来显示下拉式菜单，并会针对不允许复选和允许复选的下拉式菜单提供不同的接口。

下面是一个例子 <\ch10\jQMform7a.html>，这个下拉式菜单不允许复选，而且“台湾大哥大”为预先选取的项目。浏览结果如图 10-45 所示。

```
<div data-role="fieldcontain">
  <label for="userphone">您使用哪家业者的服务？ </label>
  <select name="userphone">
    <option value="中华电信">中华电信
    <option value="台湾大哥大" selected>台湾大哥大
    <option value="远传">远传
    <option value="威宝">威宝
  </select>
</div>
```

图 10-45

接着，我们将这个例子改写为允许复选的下拉式菜单 <\ch10\jQMform7b.html>，也就是在 <select> 元素加上 multiple 属性。

```
<div data-role="fieldcontain">
```

```
<label for="userphone">您使用哪家业者的服务？ </label>
<select name="userphone[]" multiple>
  <option value="中华电信">中华电信
  <option value="台湾大哥大" selected>台湾大哥大
  <option value="远传">远传
  <option value="威宝">威宝
</select>
</div>
```

浏览结果如图 10-46 所示，请注意第三个界面，jQuery Mobile 会以逗号隔开复选的项目，并加上气泡数字标记选取几个项目。

1. 一开始会显示预先选取的项目（例如“台湾大哥大”），请点取此按钮

2. 出现列表供选择，请点取多个项目（例如“台湾大哥大”和“远传”）

3. 选取的项目背景色显示为蓝色

图 10-46

由于 jQuery Mobile 会以可触控的按钮形式来显示下拉式菜单，所以适用于按钮的 data-* 属性都可以套用到 <select> 元素，例如加上 data-mini="true" 属性可以显示迷你按钮，加上 data-theme="a-z" 可以设置主题。

10.12.7 复选框

我们可以使用 <input> 元素加上 type="checkbox" 属性在移动版网页中插入复选框，下面是一个例子 <\ch10\jQMform8.html>，重点如下。

- 02、10：将 <input> 元素放在 <fieldset> 元素里面，并在 <fieldset> 元素加上 data-role="controlgroup" 属性，表示要创建为群组。
- 05、07、09：使用 <label> 元素作用于第 04、第 06 和第 08 行的 <input> 元素提供项目的文字，这么做的目的是让项目容易被单击。请注意，每个 <input> 元素的 id

属性必须是唯一的，以供 <label> 元素的 for 属性参照，而隶属于相同群组的复选框必须有相同的 name 属性。

- 03：由于 <label> 元素被用来提供项目的文字，所以在第 03 行使用 <legend> 元素加上说明文字。

```
01:<div data-role="fieldcontain">
02:   <fieldset data-role="controlgroup">
03:      <legend>您使用过哪些品牌的手机？</legend>
04:      <input type="checkbox" name="userphone[]" id="brand1" value="htc">
05:      <label for="brand1">hTC</label>
06:      <input type="checkbox" name="userphone[]" id="brand2" value="Apple">
07:      <label for="brand2">Apple</label>
08:      <input type="checkbox" name="userphone[]" id="brand3" value="SONY">
09:      <label for="brand3">SONY</label>
10:   </fieldset>
11:</div>
```

浏览结果如图 10-47 所示。

图 10-47

若要显示迷你版的项目，可以在 <input> 元素加上 data-mini="true" 属性。此外，jQuery Mobile 默认会以垂直排列的方式显示复选框，若要变更为水平排列，可以在 <fieldset> 元素加上 data-type="horizontal" 属性，图 10-48 是在第 02 行加上 data-type="horizontal" 属性，改变为水平排列的浏览结果。

图 10-48

10.12.8 单选按钮

我们可以使用 <input> 元素加上 type="radio" 属性在移动版网页中插入单选按钮，下面是一个例子 <\ch10\jQMform9.html>，重点如下。

- 02、10：将 <input> 元素放在 <fieldset> 元素里面，并在 <fieldset> 元素中加上 data-role="controlgroup" 属性，表示要创建为群组。
- 05、07、09：使用 <label> 元素作用于第 04、06、08 行的 <input> 元素提供项目的文字，这么做的目的是让项目容易被单击。请注意，每个 <input> 元素的 id 属性必须是唯一的，以供 <label> 元素的 for 属性参照，而隶属于相同群组的单选按钮必须有相同的 name 属性。
- 03：由于 <label> 元素被作用于提供项目的文字，所以在第 03 行使用 <legend> 元素加上说明文字。

仔细观察可以发现，这段程序代码和前一节的例子 <\ch10\jQMform8.html> 颇为类似，主要的差异在于单选按钮不允许复选，而复选框允许复选。

```
01:<div data-role="fieldcontain">
02:   <fieldset data-role="controlgroup">
03:      <legend>您最喜欢哪种水果？ </legend>
04:      <input type="radio" name="fruit" id="kind1" value="peach">
05:      <label for="kind1">水蜜桃</label>
06:      <input type="radio" name="fruit" id="kind2" value="apple">
07:      <label for="kind2">苹果</label>
08:      <input type="radio" name="fruit" id="kind3" value="banana">
```

```
09:     <label for="kind3">香蕉</label>
10:  </fieldset>
11:</div>
```

浏览结果如图 10-49 所示。

图 10-49

若要显示迷你版的项目，可以在 <input> 元素中加上 data-mini="true" 属性。此外，jQuery Mobile 默认会以垂直排列的方式显示单选按钮，若要变更为水平排列，可以在 <fieldset> 元素中加上 data-type="horizontal" 属性，图 10-50 是在第 02 行加上 data-type="horizontal" 属性，变更为水平排列的浏览结果。

图 10-50

10.12.9 读取表单字段的数据

在本章的最后，我们直接以下面的例子示范如何在jQuery Mobile移动版网页中使用PHP，来读取用户在表单字段输入的数据，见图 10-51。我们已经在第 8.1.2 小节讲解过这些技巧，如有疑问，可以参阅第 8.1.2 小节的说明。

\ch10\jQMform10.html

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>我的 jQuery Mobile 程序</title>
    <link rel="stylesheet" href="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1. min.css" />
    <script src="http://code.jquery.com/jquery-1.9.1.min.js"></script>
    <script src="http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.js"> </script>
    <meta name="viewport" content="width=device-width, initial-scale=1">
  </head>
  <body>
    <div data-role="page">
      <div data-role="header">
        <h1>个人信息</h1>
      </div>
      <div data-role="content">
        <form method="post" action="confirm.php">      ← 指定表单的确认网页
          <div data-role="fieldcontain">
            <label for="username">输入姓名：</label>
            <input type="text" name="username">
            <fieldset data-role="controlgroup" data-type="horizontal">
              <legend>您最喜欢哪种水果？</legend>
              <input type="radio" name="fruit" id="kind1" value="peach">
              <label for="kind1">水蜜桃</label>
              <input type="radio" name="fruit" id="kind2" value="apple">
              <label for="kind2">苹果</label>
              <input type="radio" name="fruit" id="kind3" value="banana">
              <label for="kind3">香蕉</label>
            </fieldset>
          </div>
          <div data-role="fieldcontain">
            <input type="submit" value="确定">
          </div>
        </form>
      </div>
```

```
    </div>
  </body>
</html>
```

\ch10\confirm.php

```
<?php
  $username = $_POST["username"];                    //读取单行文本框的数据
  switch($_POST["fruit"])                            //读取在单选按钮中选取的选项
  {
    case "peach":
      $fruit = "水蜜桃";
      break;
    case "apple":
      $fruit = "苹果";
      break;
    case "banana":
      $fruit = "香蕉";
      break;
  }
?>
<p><?php echo $username; ?>，您好！您最喜欢的水果是<?php echo $fruit; ?></p>
```

图 10-51

第 11 章
管理 MySQL 数据库

11.1　认识数据库

“数据库”(database）是一组相关数据的集合，这些数据之间可能具有某些关联，允许用户从不同的观点来加以访问，例如学校的选课系统、公司的进销存系统、图书馆的图书目录、医疗院所的病历系统、银行的存款帐户等。

我们将用来操作与管理数据库的软件称为“数据库管理系统”(DataBase Management System，DBMS），例如 MySQL、Microsoft Access、SQL Server、Oracle、Sybase、Informix 等。通过 DBMS，用户可以对数据进行定义、建立、处理与共享，其中“定义”(defining）是指明数据类型、结构及相关限制，“建立”(constructing）是输入并存储数据，“处理”(manipulating）是包括查询、新增、更新、删除等动作，而“共享”(sharing）是让多个用户同时访问数据库。

我们在前几章介绍过服务器端文件的访问、Cookie、Session 等存储数据的方式，这些方式与数据库的比较如下。

- 数据库：适合记录大量数据，可以查询、插入、删除与更新数据，虽然查询速度较快，但打开数据库连接需花费较多时间。
- Cookie：适合记录浏览者的各个信息，存放在客户端的内存或磁盘中，默认的生命周期起始于浏览器开始执行，结束于浏览器终止执行，但也可以自行设置 Cookie 的生命周期，缺点是 Cookie 可能会被浏览者删除或拒绝写入，且容易造成安全上的威胁。
- Session：适合记录浏览者的各个信息，存放在服务器端的内存中，默认的生命周期起始于浏览器开始执行，结束于浏览器终止执行，但也可以自行设置 Session 的生命周期，缺点是占用服务器端的内存，为了减轻服务器端的负担，一般还是建议以 Cookie 取代 Session 记录客户端的信息。
- 服务器端文件的访问：适合记录少量数据，可以写入或读取数据，而且没有生命周期的问题，但是当数据量较大时，文件访问将变得没有效率。

现在，我们来看个例子，这个例子属于“关系数据库”(relational database），也就是数据库里面包含几个数据表，而且数据表之间会有共同的字段，使数据表之间产生关联。

假设关系数据库里面有下列 4 个数据表，名称分别为“学生资料”、“语文成绩”、“数学成绩”、“英文成绩”，其中“座号”字段为共同的字段：

座号	姓名	生日	通信地址
1	小丸子	1994/01/01	台北市罗斯福路一段 9 号 9 楼
2	花轮	1995/05/06	台北市师大路 20 号 3 楼
3	藤木	1994/12/20	台北市温州街 42 巷 7 号之 1
4	小玉	1995/03/17	台北市龙泉街 3 巷 12 弄 28 号
5	丸尾	1994/08/11	台北市金门街 100 号 5 楼
6	永泽	1994/10/22	台北市和平东路二段 85 巷 109 号 15 楼之 3

座号	语文分数
1	80
2	95
3	88
4	98
5	93
6	81

座号	数学分数
1	75
2	100
3	90
4	92
5	97
6	92

座号	英文分数
1	82
2	97
3	85
4	88
5	100
6	94

有了这些数据表，我们就可以使用 DBMS 进行各项查询，例如座号为 5 的学生叫什么、住在哪里、语文考多少分、英文分数高于 90 的有哪几位学生、将数学分数由高至低排列等。

此外，通过共同字段可以产生如下数据，即结合“学生资料”、“语文成绩”、“数学成绩”、“英文成绩”4 个数据表产生“总分”数据。

座号	姓名	总分	通信地址
1	小丸子	237	台北市罗斯福路一段 9 号 9 楼
2	花轮	292	台北市师大路 20 号 3 楼
3	藤木	263	台北市温州街 42 巷 7 号之 1
4	小玉	278	台北市龙泉街 3 巷 12 弄 28 号
5	丸尾	290	台北市金门街 100 号 5 楼
6	永泽	267	台北市和平东路二段 85 巷 109 号 15 楼之 3

最后，我们来介绍几个数据库的术语，以下面的网页为例，这是 MySQL 数据库服务器的管理接口，其中总共有 10 笔数据，每笔数据都称为“记录”(record)，而在同一笔记录（同一行）中，又包含了 no（座号）、name（姓名）、chinese（语文）、math（数学）、english（英文）5 个“字段”（field），这些记录的组合称为“数据表”（table），而几个性质类似的数据表集合起来则称为“数据库”(database)，如图 11-1 所示。

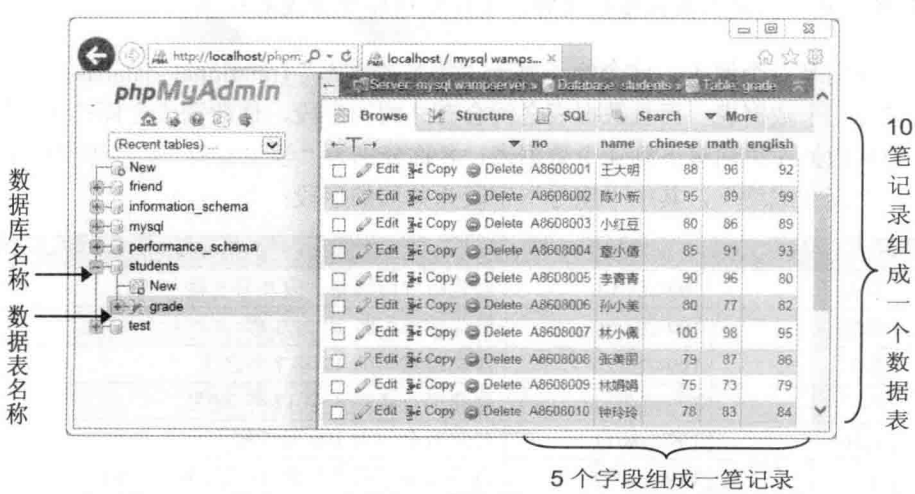


图 11-1

11.2　PHP 与数据库

PHP 提供许多连接数据库所需的函数库，表 11-1 是 PHP 支持的数据库。

表 11-1　PHP 支持的数据库

数据库	操作系统	授权方式
Adabas D	Unix、Windows	商业用途（commercial）
DBA/DBM	Unix	商业用途、开放源码（open source）
dBase	Windows	商业用途
Empress	Unix、Windows	商业用途
filepro	Unix、Windows	商业用途
IBM DB2	Unix、Windows	商业用途
Informix	Unix、Windows	商业用途
Interbase	Unix、Windows	商业用途
Microsoft Access	Windows	商业用途
Microsoft SQL Server	Windows	商业用途
mSQL	Unix	共享软件（shareware）
MySQL	Unix、Windows	商业用途、开放源码
Oracle	Unix、Windows	商业用途
PostgreSQL	Unix	开放源码
Solid	Unix、Windows	商业用途
SQLite	Unix、Windows	开放源码

如表 11-1 所示，PHP 支持许多数据库，而本书则选择使用 MySQL 数据库服务器，理由是 MySQL 为开放源码软件，可以免费获取，再加上它具有快速、简单、可靠、功能齐全、跨平台等特性，因此，MySQL 数据库服务器可以说是搭配 PHP 的最佳选择。

11.3　使用 phpMyAdmin 管理 MySQL 数据库

phpMyAdmin 是一个免费且受欢迎的 MySQL 数据库管理工具，支持多国语言，当然也包括我们所熟悉的中文。由于 phpMyAdmin 是使用 PHP 程序代码编写而成的工具，所以只要 Web 服务器能够执行 PHP，就可以使用 phpMyAdmin。

步骤 01　打开浏览器，在地址栏输入 http://localhost/phpmyadmin 并按 Enter 键（注意 phpmyadmin 都是小写），或在 Windows 任务栏的 图标上单击鼠标左键，然后选取 phpMyAdmin，如图 11-2 所示。

步骤 02　待出现如下界面，表示 phpMyAdmin 正常运行，请将[语系 - Language]字段设

置为 English，因为 phpMyAdmin 4.1.14 在中文界面执行某些功能会发生 JavaScript 错误，必须使用英文界面才能正常运行，如图 11-3 所示。

图 11-2

图 11-3

11.3.1 添加、删除、修改登录账号与密码

步骤 01 用浏览器打开 http://localhost/phpmyadmin，然后单击 Users 超链接，如图 11-4 所示。

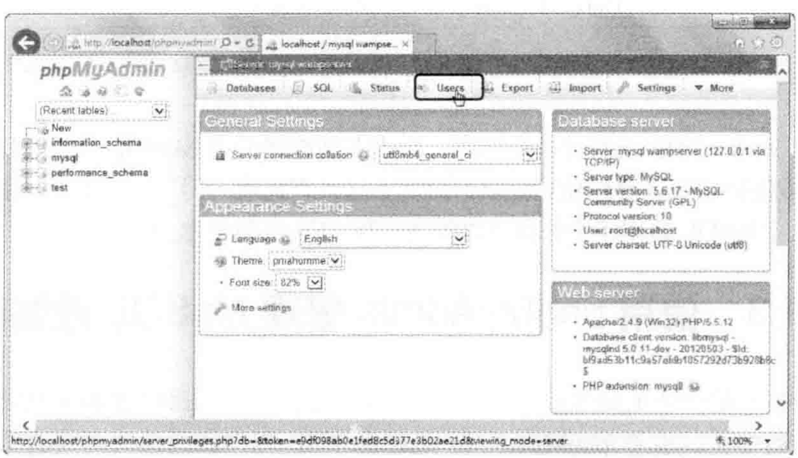


图 11-4

步骤 02 单击 User 为 root，[Host]为 localhost 账号最右边的 Edit Privileges 超链接，如图 11-5 所示。

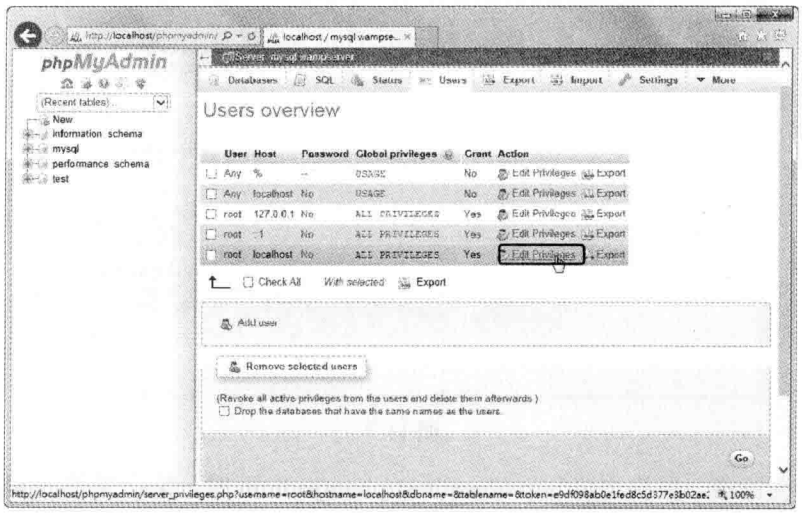


图 11-5

步骤 03　按照图 11-6 所示的操作来变更 root 账号的登录密码。

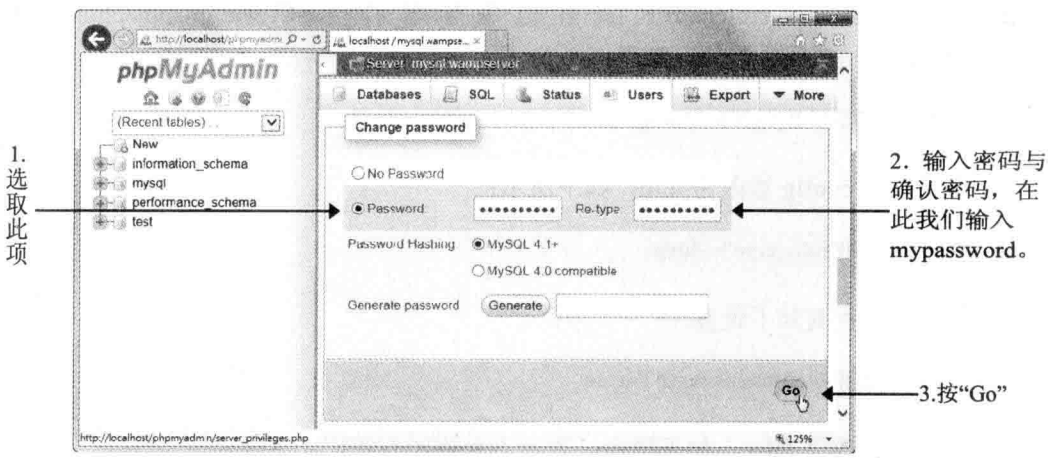


图 11-6

步骤 04　待出现如图 11-7 所示的界面，表示密码变更成功。事实上，在[Users]功能里，除了变更密码之外，还可以为 MySQL 数据库创建新的用户、变更用户权限、删除用户等，请您自己试试看。

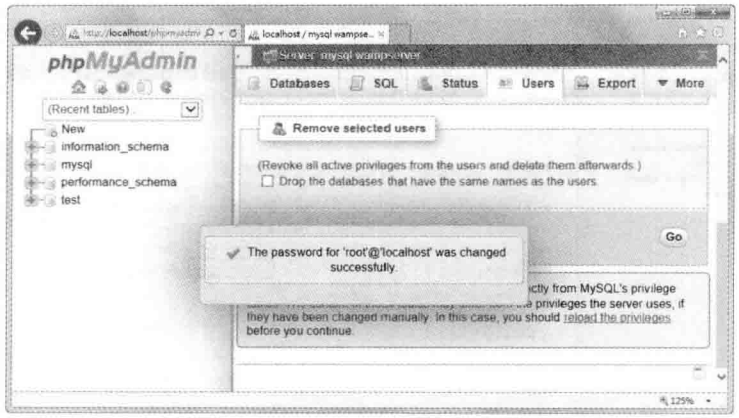


图 11-7

当您关闭浏览器，再次浏览 http://localhost/phpmyadmin 时，您会发现 phpMyAdmin 发生错误，无法连接至 MySQL 数据库，这是因为我们变更了 root 账号的密码，我们必须手动变更 phpMyAdmin 的配置设置，才能正常使用。

步骤 01 使用 Notepad++ 或其他编辑器打开 C:\wamp\apps\phpmyadmin4.1.14\config.inc.php，然后找到如下设置：

```
$cfg['Servers'][$i]['auth_type']= 'config';
```

将字符串 config 修改为 http，如下所示：

```
$cfg['Servers'][$i]['auth_type']= 'http';
```

步骤 02 找到如下设置：

```
$cfg['Servers'][$i]['AllowNoPassword']= true;
```

将 true 修改为 false，如下所示，表示不允许没有密码：

```
$cfg['Servers'][$i]['AllowNoPassword']= false;
```

步骤 03 找到如下设置：

```
$cfg['PmaNoRelation_DisableWarning']= true;
```

在它的下面新增以下设置，这样才能使用 SQL 来删除数据库：

```
$cfg['AllowUserDropDatabase']= true;
```

步骤 04 保存 config.inc.php。

步骤 05 再次用浏览器打开 http://localhost/phpmyadmin，随即会出现如图 11-8 所示的对话框，要求您输入 MySQL 的账号与密码，在此，我们输入账号 root，输入密码 mypassword，

然后单击“确定”按钮。

图 11-8

11.3.2　创建数据库

phpMyAdmin 是一个功能强大的管理工具，除了可以使用它管理权限，还可以用它来创建数据库，其步骤如下。

步骤 01　用浏览器打开 http://localhost/phpmyadmin 并登录，您可以看见如图 11-9 所示的网页，请单击界面上方的 Databases 超链接，或左窗格的 New 超链接。

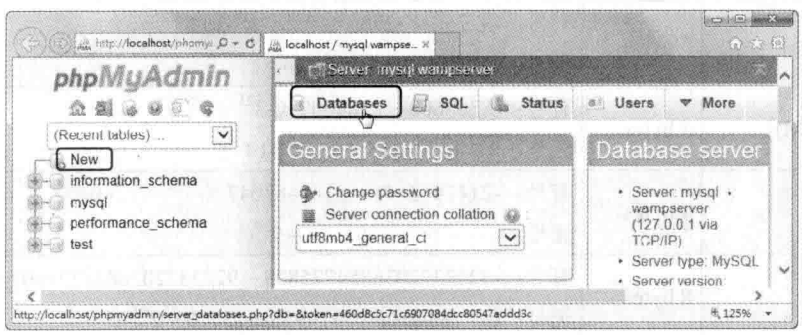


图 11-9

步骤 02　请在 Create database 字段中输入数据库名称，例如 friend，然后将数据库的编码与排序方式设为 utf8_general_ci，再单击 Create 键。请注意，MySQL 数据库服务器虽然支持中文数据库名称，但数据库名称还是请尽量使用英文字母和阿拉伯数字来命名，避免产生无法预期的错误，如图 11-10 所示。

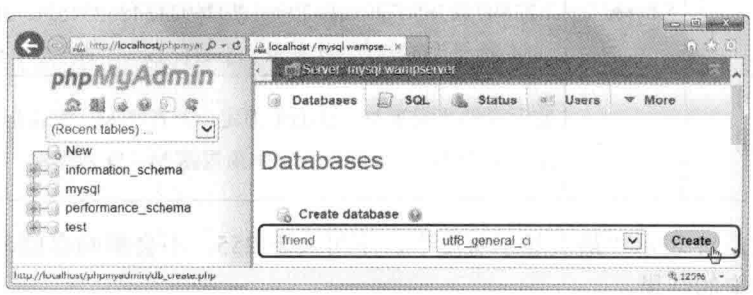
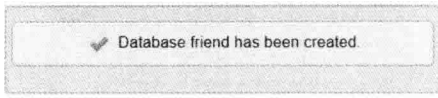


图 11-10

步骤 03 成功创建数据库，您可以在界面上看到如下信息。



11.3.3 创建数据表

在创建数据表之前，我们来介绍 MySQL 数据库服务器提供了哪些数据类型。

❖ **数值类型（numeric types）**

数据类型	空间	范围
TINYINT[(M)]	1 byte	有号：-128～127（-2^7～2^7 - 1） 无号：0～255（0～2^8 - 1） MySQL 无 BOOLEAN 类型，若要存放布尔值，可以使用 TINYINT(1)，值为 0，表示 FALSE，若值不为 0，表示 TRUE（M 表示“最大显示宽度”）
SMALLINT[(M)]	2 bytes	有号：-32768～32767（-2^{15}～2^{15}-1） 无号：0～65535（0～2^{16}-1）
MEDIUNINT[(M)]	3 bytes	有号：-8388608～8388607（-2^{23}～2^{23}-1） 无号：0～16777215（0～2^{24}-1）
INT[(M)] INTEGER[(M)]	4 bytes	有号：-2147483648～2147483647（-2^{31}～2^{31}-1） 无号：0～4294967295（0～2^{32}-1）
BIGINT[(M)]	8 bytes	有号：-9223372036854775808～9223372036854775807（-2^{63}～2^{63}-1） 无号：0～18446744073709551615（0～2^{64}-1）
FLOAT(p)	4 bytes 8 bytes	若 p <= 24，则视为 FLOAT 若 25 <= p <= 53，则视为 DOUBLE
FLOAT[(M,D)] (单精度浮点数)	4 bytes	-3.402823466E+38～-1.175494351E-38、0、 1.175494351E-38～3.402823466E+38
DOUBLE[(M,D)] REAL[(M,D)] (双精度浮点数)	8 bytes	-1.7976931348623157E+308~-2.2250738585072014E-308、0、 2.2250738585072014E-308~1.7976931348623157E+308 （M 表示“精度”，D 表示“小数位数”）
DECIMAL[(M[,D])] NUMERIC[(M[,D])] DEC[(M[,D])]	?	实际存储范围视 M、D 的值而定，若省略 M，默认值为 10，若省略 D，默认值为 0，占用的内存空间须根据 M、D 来设置

对整数来说，M 表示“最大显示宽度”，不可大于 255，不会影响存储范围，M 的大小只会影响显示出来的结果。

对浮点数来说，M 是“精度”(即总位数），不可大于 65，D 是“小数位数”，不可大于 30，也不能大于 M - 2，即小数位数最多不可超过 30。

举例来说，假设将两个字段均设置为 INT(4)，然后分别在两个字段中存入数字 12 和 123456，那么显示出来的结果是什么呢？答案是 12 和 123456，原因是虽然将两个字段均设置为 INT(4)，表示只显示 4 个位数，但事实上，若实际存储的数据位数比 M 大，仍会显示实际位数。

我们再来看一个例子，假设将两个字段分别设置为 INT(4) 和 INT(4) ZEROFILL（ZEROFILL 会在前面的空位补 0），然后两个字段均存入数字 12，那么显示结果分别是 12 和 0012。

❖ 日期与时间类型（date and time types）

数据类型	空间	范围
DATE	3bytes	日期类型，存储范围为 '1000-01-01'～'9999-12-31'，格式为 'YYYY-MM-DD'
DATETIME	8bytes	日期时间类型，存储范围为 '1000-01-01 00:00:00'～'9999-12-31 23:59:59'，格式为 'YYYY-MM-DD HH:MM:SS'
TIMESTAMP	4bytes	时间戳，存储范围为 '1970-01-01 00:00:00'～'2038-01-09 03:14:07'
TIME	3bytes	时间类型，存储范围为 '-838:59:59'～'838:59:59'，格式为 'HH:MM:SS'
YEAR[(2|4)]	1byte	以 2 或 4 位数字格式来存储年份，默认值为 4 4 位数：1901～2155 及 0000 2 位数：70～69，表示 1970～2069

❖ 字符串类型（string types）

数据类型	空间	范围
CHAR(M)	M * v bytes	固定长度字符串，M 表示字符长度，必须介于 0～255，若省略 M，则默认值为 1。v 为字段中单个字符或汉字最多占用的字节数，例如 CHAR(9) 存储字符串 "金牛座 Taurus"，每个英文字母占 1 个 byte，每个汉字各占 2 个 bytes，故 v 为 2，而此字符串共占用 9 * 2 = 18 bytes
VARCHAR(M)	L+1byte 或 L+2bytes	可变长度字符串，M 表示允许存储的最大字符数，必须介于 0～65,535。L 表示字段字符或者汉字实际占用的位数，若 L<= 255 bytes，则需要的存储空间为 L+1bytes，若 L>255bytes，则为 L+2bytes
TINYTEXT	L+1byte	可变长度字符串，最多可以存储 255（2^8 -1）字符，若包含多位文字，则可存储的字符数会减少
TEXT	L+2bytes	可变长度字符串，最多可以存储 65,535（2^{16} -1）字符，若包含多字节文字（如汉字），则可存储的字符数会减少

（续表）

数据类型	空间	范围
MEDIUMTEXT	L+3bytes	可变长度字符串，最多可以存储 16,777,215（2^{24} - 1）字符，若包含多字节文字，则可存储的字符数会减少
LONGTEXT	L+4bytes	可变长度字符串，最多可以存储 4,294,967,295（2^{32} - 1）字符，若包含多字节文字，则可存储的字符数会减少
TINYBLOB	L+1byte	可变长度字符串，最多可以存储 255 bytes（2^8 -1）
BLOB	L+2bytes	可变长度字符串，最多可以存储 65535 bytes（2^{16} -1），即 65KB
MEDIUMBLOB	L+3bytes	可变长度字符串，最多可以存储 16,777,215 bytes（2^{24} - 1），即 16MB
LONGBLOB	L+4bytes	可变长度字符串，最多可以存储 4,294,967,295 bytes(2^{32} - 1)，即 4GB

BLOB 的全名为 binary large object，它与 TEXT 的主要差别在于，BLOB 有大小写之分，而 TEXT 没有大小写之分。

当您使用 CHAR 类型存储数据时，若将字段设为定 CHAR(10），表示字符串长度固定为 10 个字符，但您若只存入“测试”两个汉字，那么 MySQL 会自动以空格符填补至最大存储长度，也就是变成“测试　　”，因为测试两个字占用 4 个字节（注：一个汉字占 2 个字节），所以测试后面会自动补上空格符，因此，无论存入的内容为何，所存储的字符数都是相同的。

在了解 MySQL 提供的数据类型后，我们将为您示范如何建立 friend_club 数据表，这个数据表包含下列 8 个字段。

字段名	数据类型	长度	主键	说明
no	SMALLINT	-	☑	编号字段
name	VARCHAR	5	☐	姓名字段
sex	CHAR	1	☐	性别字段
age	VARCHAR	10	☐	年龄字段
star_signs	VARCHAR	3	☐	星座字段
height	VARCHAR	10	☐	身高字段
weight	VARCHAR	10	☐	体重字段
career	VARCHAR	10	☐	职业字段

您可以按照如下步骤创建数据表，同样的，数据表名称请尽量使用英文字母和阿拉伯数字来命名，MySQL 数据库服务器虽然支持中文数据表名称，但在某些情况下，中文数据表名称却可能会产生无法预期的错误。

步骤 01 用浏览器打开 http://localhost/phpmyadmin 并登录，您可以看见如图 11-11 所示的网页，请在左窗格用鼠标单击要创建数据表的数据库，例如 friend，在右窗格的 Name 字段

输入数据表的名称，例如 friend_club，在[Number of columns]字段输入要建立的字段数目，例如 8，然后按 Go。

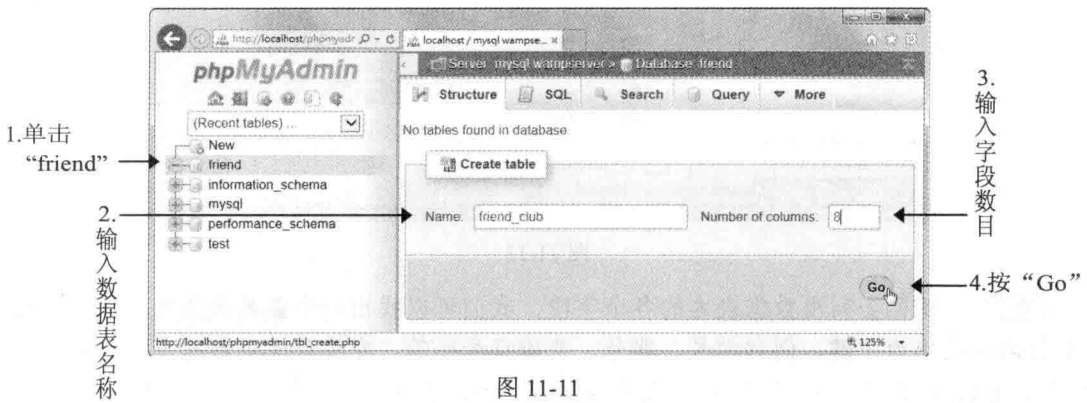


图 11-11

步骤 02　出现如图 11-12 所示的网页让我们定义数据表的字段名、数据类型和数据长度，您也可以输入数据表注释文字，同样的，字段名请尽量使用英文字母和阿拉伯数字来命名，输入完毕后按 Save。

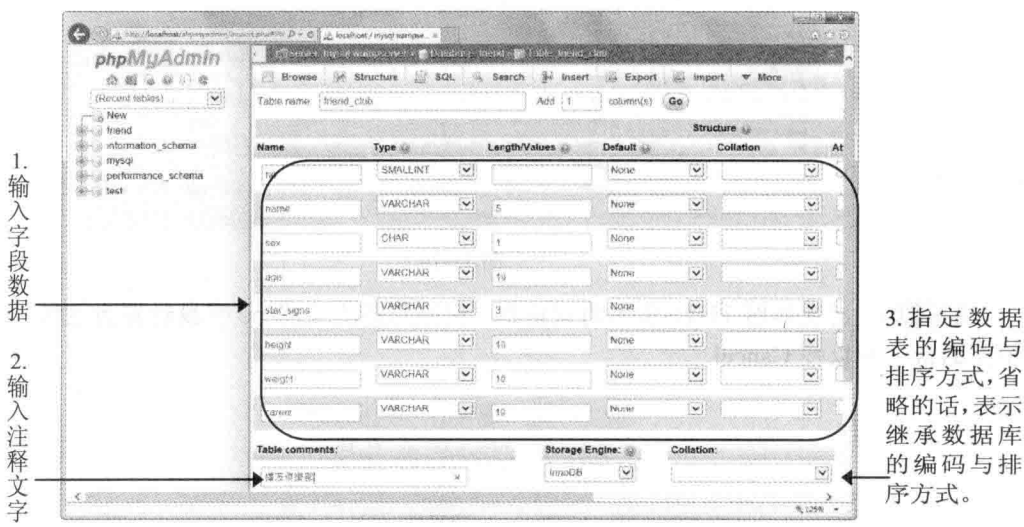


图 11-12

步骤 03　建立完成后会看到如图 11-13 所示的数据表，请单击 Structure 超链接。

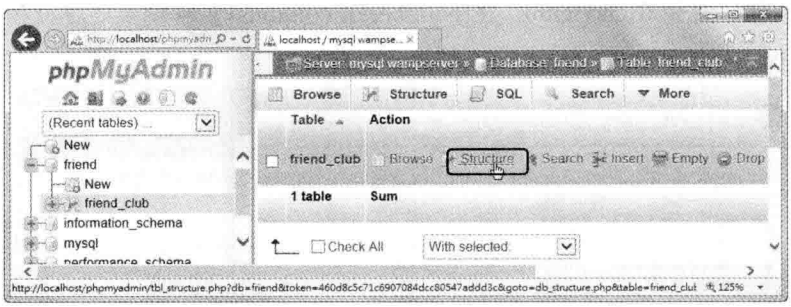


图 11-13

步骤 04 随即会列出数据表内的各个字段，我们可以找出一个最具代表性，而且数据不会重复的字段作为主键，例如学号、账号、身份证号码等。在设计关系数据库时，每个数据表都应该设置主键，以作为不同数据表之间的关联字段。现在，请单击 no 字段的[More]\[Primary]超链接来将 no 字段设置为主键，如图 11-14 所示。

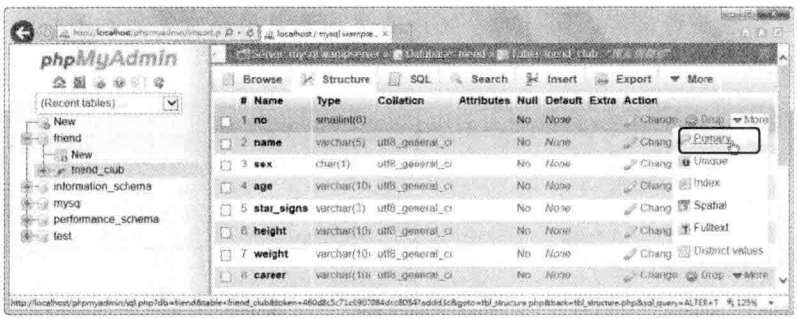


图 11-14

步骤 05 出现如图 11-15 所示的对话框，询问您是否要将 no 字段设置为主键，请按 OK，不想的话，可以按 Cancel。

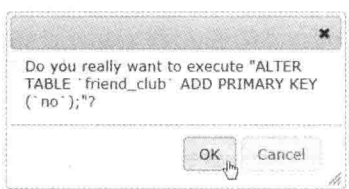


图 11-15

步骤 06 若要修改或删除字段，可以核选字段，然后按 Change 或 Drop，如图 11-16 所示。

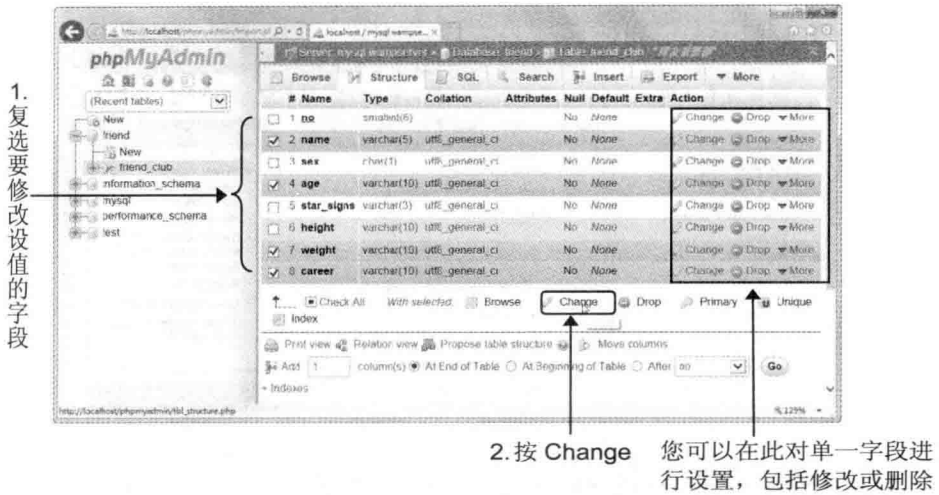


图 11-16

步骤 07　如图 11-17 所示是我们核选 name、age、weight、career 4 个字段，并按 Change 之后的结果，此处能看到这 4 个字段的字段名、类型、长度等，您可以修改字段定义，然后按 Save 完成修改。

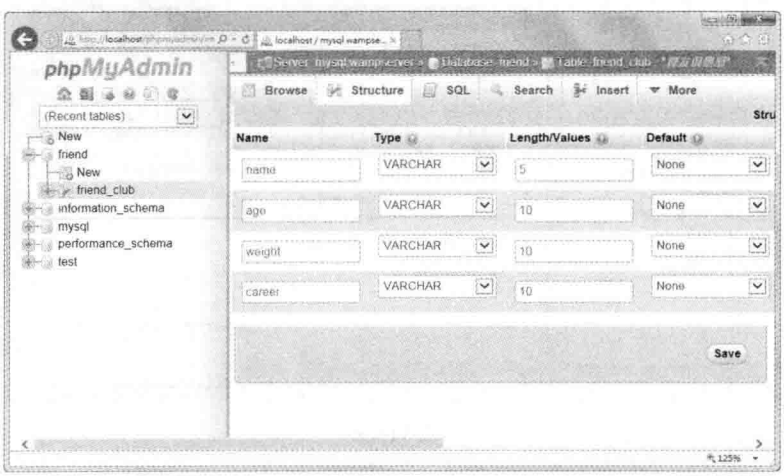


图 11-17

11.3.4　新增记录

步骤 01　用浏览器打开 http://localhost/phpmyadmin 并登录，您可以看见如图 11-18 所示的网页，请在左窗格中单击 friend 超链接。

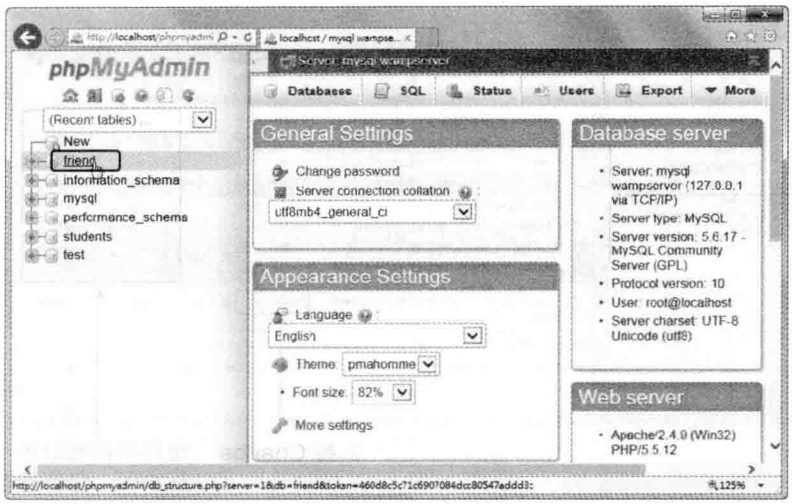


图 11-18

步骤 02 在右窗格中单击 friend_club 数据表右边的 Insert 超链接，表示要在 friend_club 数据表中新增数据，如图 11-19 所示。

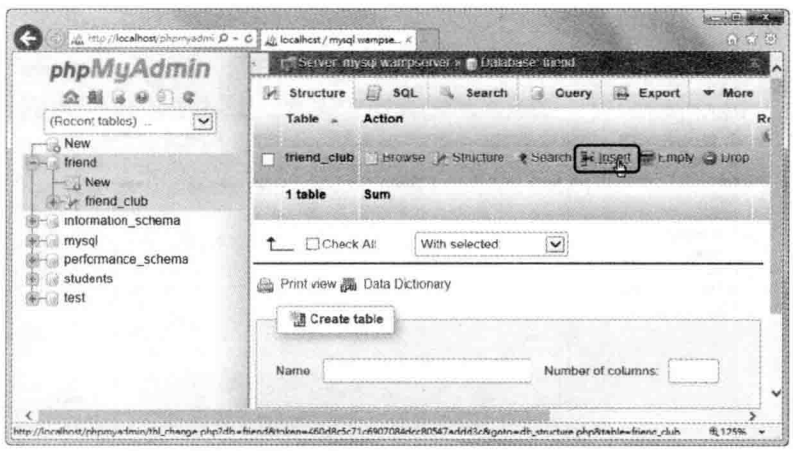


图 11-19

步骤 03 在如图 11-20 所示的网页中输入各个字段的内容，按 Go。

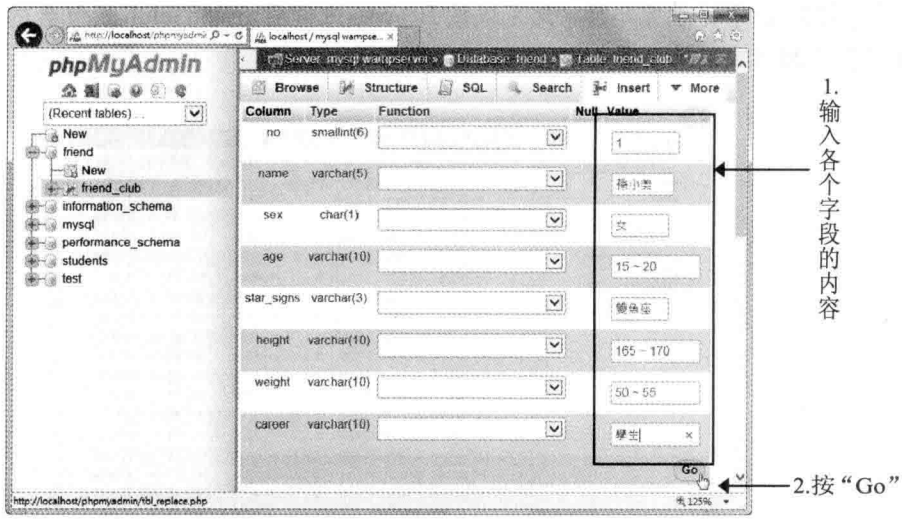


图 11-20

步骤 04　成功加入记录。

步骤 05　重复步骤 2～3，在 friend_club 数据表中加入下面所示的 19 个记录。

no	name	sex	age	star_signs	height	weight	career
2	小燕子	女	20～25	牡羊座	155～160	45～50	上班族
3	柏原崇	男	20～25	天蝎座	175～180	65～70	SOHO 族
4	莫召奴	男	25～30	天秤座	175～180	65～70	上班族
5	叶小钗	男	30～35	魔羯座	165～170	60～65	老师
6	流川枫	男	15～20	射手座	180～185	65～70	上班族
7	林阿土	男	25～30	牡羊座	170～175	65～70	老师
8	赵冰冰	女	20～25	处女座	155～160	45～50	学生
9	嘟嘟	男	15～20	狮子座	165～170	55～60	学生
10	晴子	女	15～20	双子座	160～165	45～50	学生
11	小兰	女	25～30	巨蟹座	165～170	50～55	上班族
12	凯蒂	女	20～25	双鱼座	160～165	45～50	公务员
13	樱桃子	女	25～30	天秤座	155～160	55～60	SOHO 族
14	亮亮	女	25～30	射手座	165～170	50～55	公务员
15	小齐	男	25～30	水瓶座	170～175	55～60	上班族
16	安琪	女	15～20	狮子座	165～170	50～55	学生
17	林达	女	20～25	双鱼座	165～170	50～55	公务员
18	陈小东	男	25～30	魔羯座	175～180	65～70	上班族
19	CoCo	女	20～25	狮子座	170～175	55～60	上班族
20	安室	女	30～35	处女座	155～160	45～50	老师

步骤 06 新增记录完毕后，您可以按 Browse 来查看 friend_club 所包含的记录，当然，您也可以在此编辑或删除记录，如图 11-21 所示。

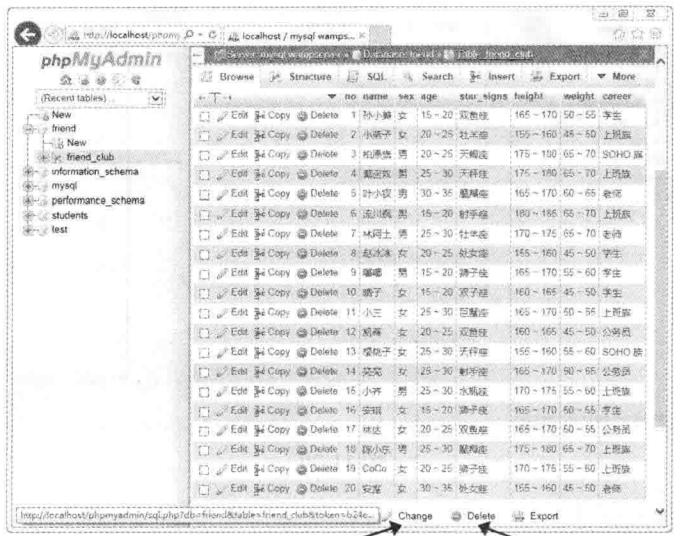


图 11-21

11.3.5 导出数据库

当您要备份数据库或将数据库移到其他计算机时，可以先导出数据库，然后再把数据库导入到目标计算机。导出数据库的步骤如下。

步骤 01 用浏览器打开 http://localhost/phpmyadmin 并登录，您可以看见如图 11-22 所示的网页，请在左窗格中单击 friend 超链接，然后单击图中的 Export 超链接。

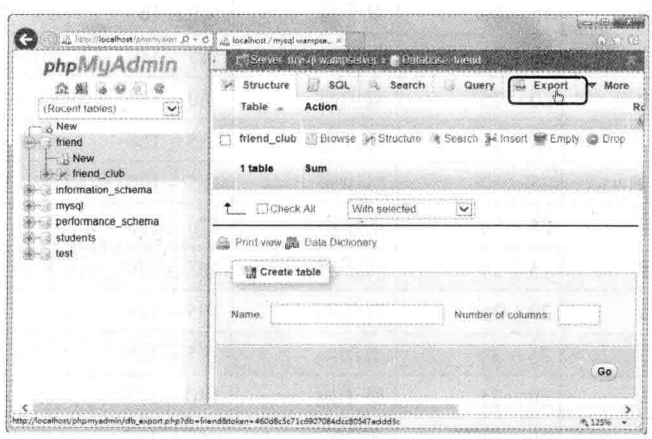


图 11-22

步骤 02　出现[Exporting tables from "friend" database]网页，请按照下面的提示操作：

- 在[Export Method]字段选择[Custom - display all possible options]。
- 选择[Add CREATE DATABASE / USE statement]选项。
- 其他字段保留默认值，用鼠标单击页面最下方的 Go。

步骤 03　出现如图 11-23 所示的对话框，请按[保存]右边的倒三角形按钮，选择[另存为]。

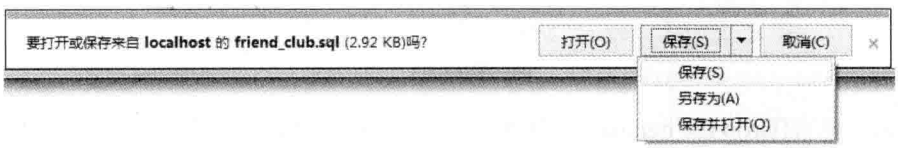


图 11-23

步骤 04　在[另存为]对话框中选择文件的存储位置，然后输入文件名，再按[保存]，如图 11-24 所示。

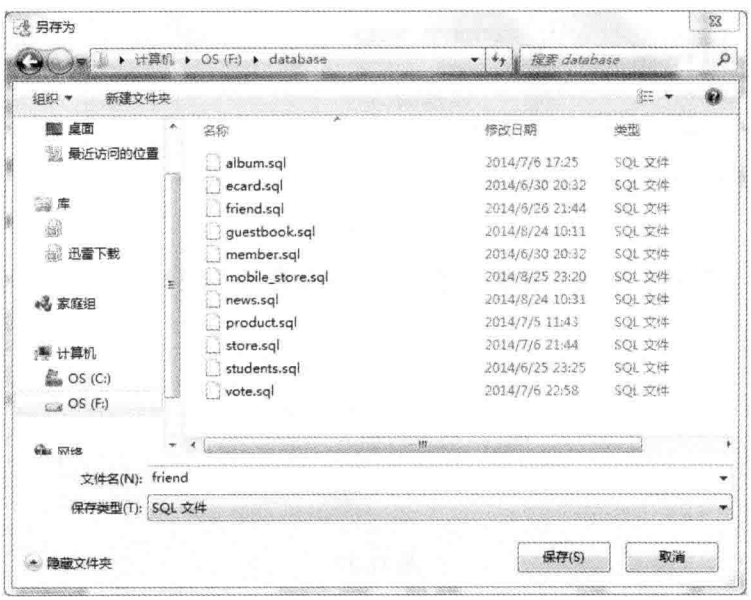


图 11-24

步骤 05　事实上，我们所导出的 friend.sql 只是一个文本文件，里面包含创建数据库、数据表及新增记录的 SQL 语法，您可以使用记事本打开这个文件，学习其中的语法。

11.3.6　删除数据库或数据表

对于错误或不再使用的数据库或数据表，您可以按照如下步骤将它删除。

步骤 01　用浏览器打开 http://localhost/phpmyadmin 并登录，您可以看见如下网页，请在左窗格中单击 friend 超链接，然后单击[Operations]，如图 11-25 所示。

图 11-25

步骤 02 单击[Drop the database（DROP）]，如图 11-26 所示。

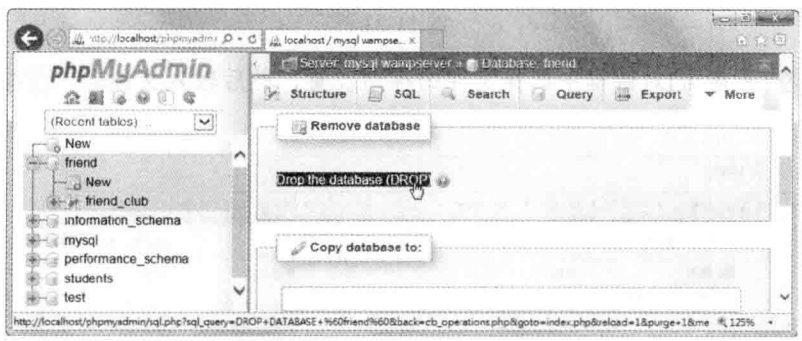


图 11-26

步骤 03 出现对话框询问是否真的要删除整个数据库，是的话，请按[OK]，如图 11-27 所示。

图 11-27

11.3.7 导入数据库

导入数据库是重建数据库最佳的方式，它可以帮您节省许多时间，当然这有一个前提，就是您之前必须备份过该数据库。

步骤 01 用浏览器打开 http://localhost/phpmyadmin 并登录，然后单击图 11-28 所示的 Import 超链接。

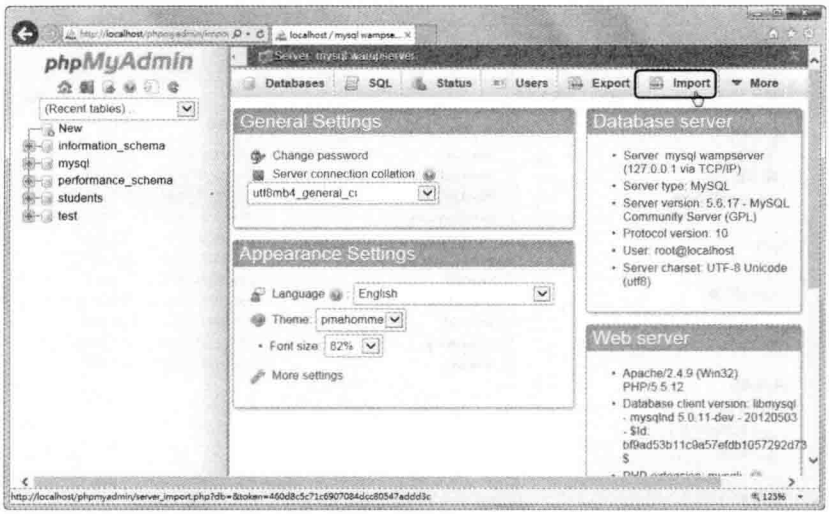


图 11-28

本书所使用的数据库均已导出存放在可供网络下载的资源 \database 文件夹，您可以按照本节的步骤，导入所有 .sql 备份文件，以节省重建数据库的时间。

步骤 02　出现如图 11-29 所示的网页，请单击[浏览]按钮来选择数据库备份文件（在此为下载资源的 \database\friend.sql）。

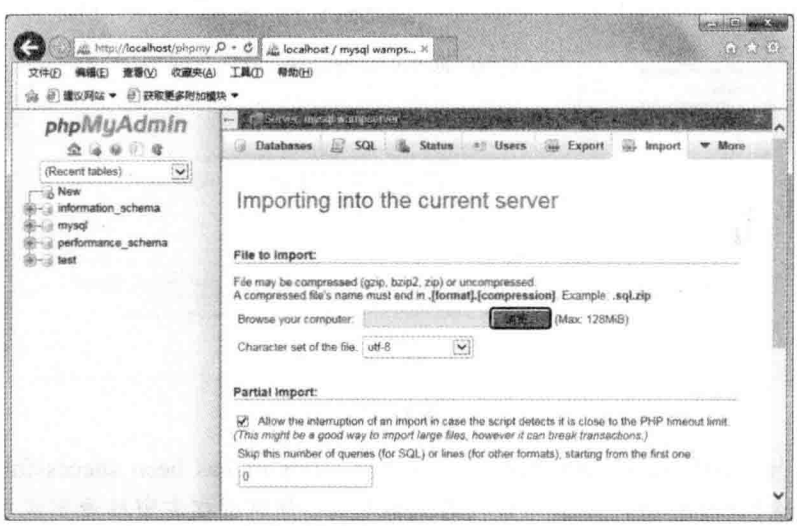


图 11-29

步骤 03　选取下载资源的 \database\friend.sql，然后按[打开]，如图 11-30 所示。

图 11-30

步骤 04 出现如图 11-31 所示的网页，[Browse your computer]字段出现选择的文件路径，请按[Go]。

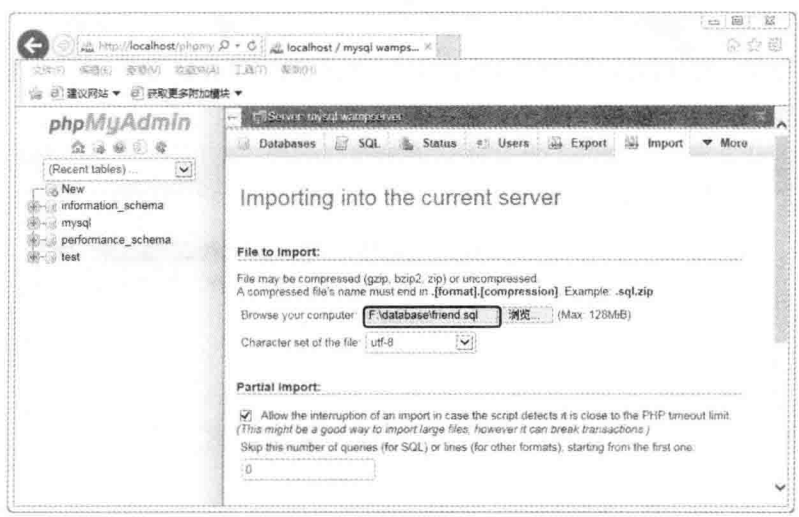


图 11-31

步骤 05 出现如图 11-32 所示的网页，告诉您 “Import has been successfully finished, 13 queries executed. (friend.sql)”，表示成功导入数据库，您可以在左窗格看到成功导入的 friend 数据库。

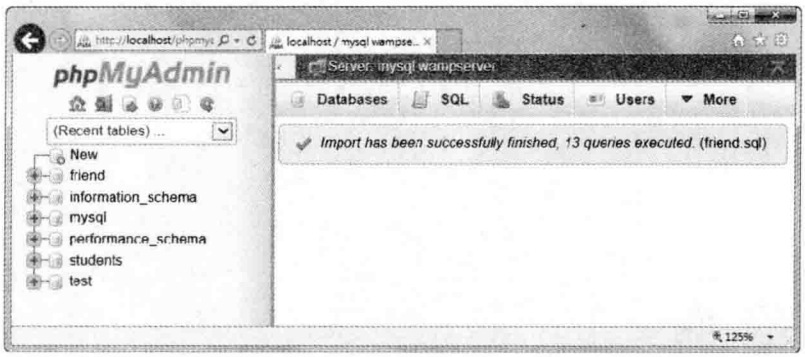

http://localhost/phpmyadmin
localhost / mysql wampse...
phpMyAdmin
Server: mysql (wampserver)
Databases
SQL
Status
Users
More
(Recent tables) ...
New
friend
information_schema
mysql
performance_schema
students
test
Import has been successfully finished, 13 queries executed. (friend.sql)
125%

图 11-32

12 章

SQL 查询

12.1　认识 SQL 查询

SQL 是 Structured Query Language 的简写，诸如 MySQL、Access、Oracle、SQL Server 等关系数据库均采用这个标准语言来进行数据库查询。SQL 不仅可以用来进行数据库查询，还可以用来添加、更新与删除记录。

在开始介绍 SQL 指令的语法之前，请您先按照第 11.3.7 小节的方法导入下载资源的 \samples\database\students.sql，这个数据库的名称为 students，包含一个名称为 grade 的数据表，里面共有 10 个记录。

使用 phpMyAdmin 执行 SQL 指令的步骤如下。

步骤 01　用浏览器打开 http://localhost/phpmyadmin 并登录，在左窗格中选取要查询的数据库，在此我们选择 students，接着单击 Query window 按钮，如图 12-1 所示。

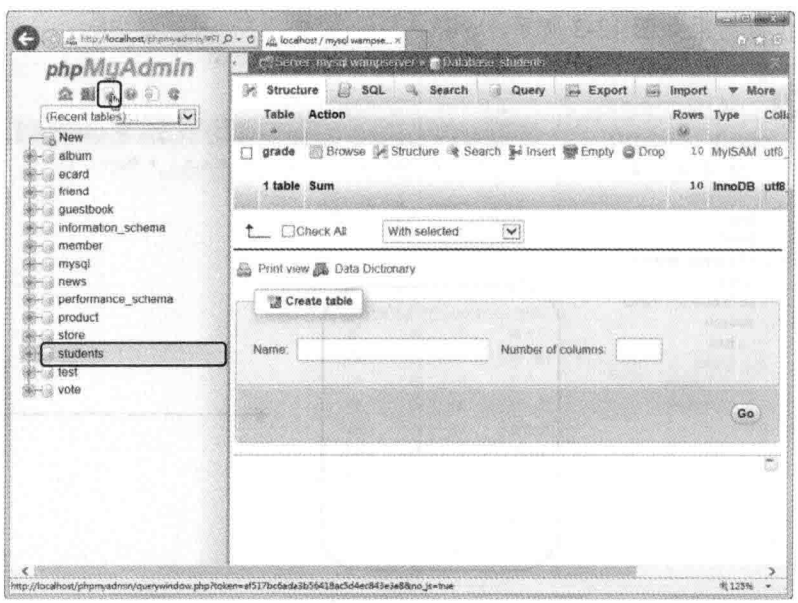


图 12-1

步骤 02　在[Run SQL query/queries on database students:]字段输入“SELECT name, english, math, chinese FROM grade”，然后按 Go，本章所有 SQL 指令均在如图 12-2 所示的网页中执行。这个 SQL 指令可以从 grade 数据表筛选出 name、english、math 和 chinese4 个字段，虽然 grade 数据表的 chinese 字段放在 english 字段前面，但在使用 SQL 指令时不必按照字段的顺序进行筛选，唯一要注意的是筛选出来的字段顺序会按照 SQL 指令所指定的字段顺序，而不是数据来源的字段顺序。

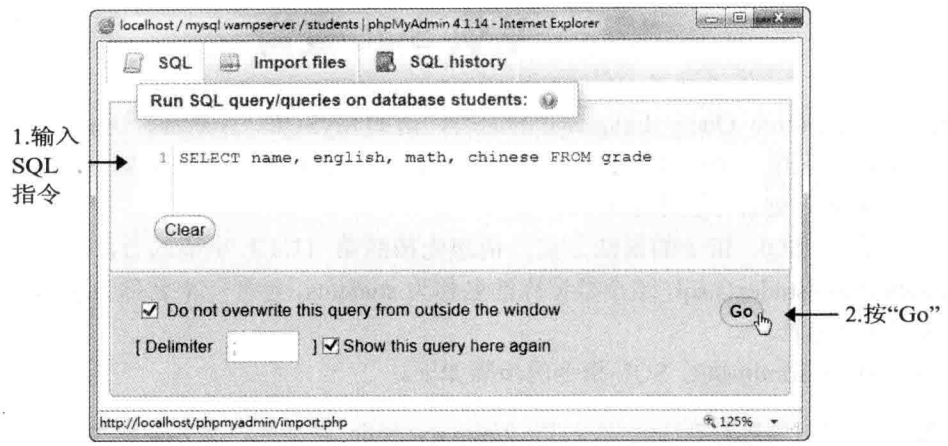


图 12-2

步骤 03 执行结果如图 12-3 所示。

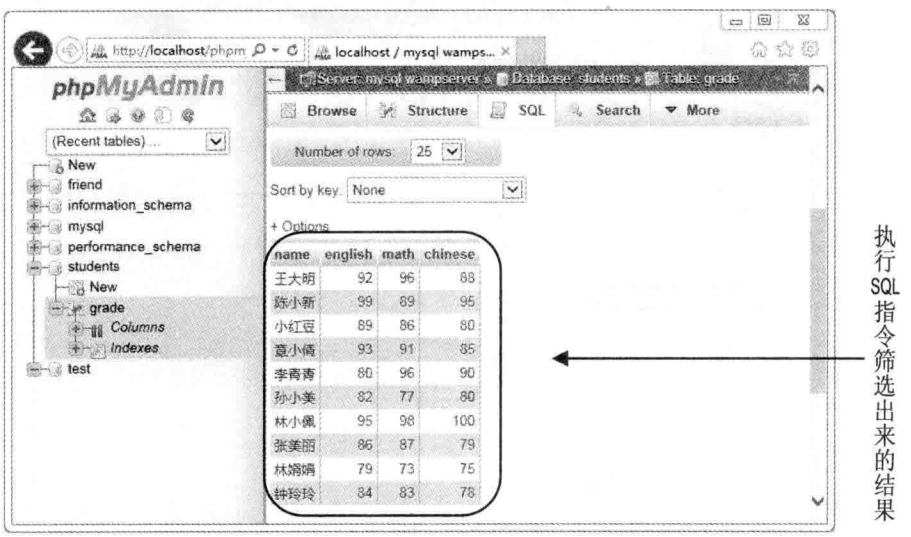


图 12-3

12.2 筛选记录

当我们想从数据表筛选记录时，可以使用 SQL 指令中的 SELECT 语句，其语法如下：

```
SELECT 字段名
FROM 数据表名称
[WHERE 搜寻子句]
[ORDER BY 排序子句 {ASC|DESC}]
```

以前一节的 grade 数据表为例，我们可以列举出一些 SQL 指令。

- 从 grade 数据表筛选出所有记录的 name、english 和 chinese 3 个字段:

```
SELECT name, english, chinese FROM grade
```

- 从 grade 数据表筛选出所有记录的所有字段:

```
SELECT * FROM grade
```

- 从 grade 数据表筛选出所有记录的 name 和 english 两个字段，然后分别将这两个字段更改为“姓名”与“英文”:

```
SELECT name AS 姓名, english AS 英文 FROM grade
```

- 从 grade 数据表筛选出所有记录的 name 字段，然后将 chinese、math、english3 个字段相加后的分数产生为新的 total 字段:

```
SELECT name, chinese+math+english AS total FROM grade
```

随堂练习

从 grade 数据表筛选出所有记录的 name、chinese、math 和 english 4 个字段，然后将 chinese、math 和 english 3 个字段相加后的分数生成为新的 total 字段，执行结果如图 12-4 所示。

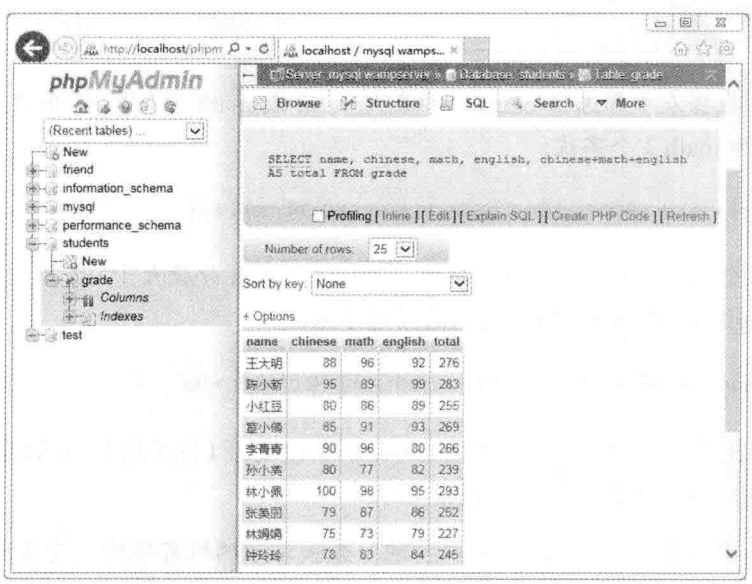


图 12-4

【提示】

这个 SQL 指令的语法如下：

```
SELECT name, chinese, math, english, chinese+math+english AS total FROM grade
```

12.2.1 SELECT ... FROM ... WHERE ... 语法（筛选）

SELECT ... FROM ... 语法会返回数据表的所有记录，但有时我们需要将筛选范围限制在符合某些条件的记录，例如语文（chinese）成绩在 90 分以上之所有记录的“姓名”(name）和“数学”(math）两个字段，此时，我们得加上 WHERE 子句来设置筛选范围，例如：

```
SELECT name, math FROM grade WHERE chinese > 90
```

WHERE 子句可以包含任何逻辑运算，只要返回值为 TRUE 或 FALSE 即可，表 12-1 为 SQL 语法所支持的比较运算符和逻辑运算符：

表 12-1 SQL 语法支持的运算符

比较运算符	说明	逻辑运算符	说明
=	等于	AND	若操作数均为 TRUE，就返回 TRUE，否则返回 FALSE
<	小于		
>	大于	OR	若任一操作数为 TRUE，就返回 TRUE，否则返回 FALSE
<=	小于等于		
>=	大于等于	NOT	若操作数为 TRUE，就返回 FALSE，否则返回 TRUE
!=	不等于		
<>			

- 从 grade 数据表中筛选出 chinese 分数大于 90 或 math 分数大于 90 之记录的 name、chinese 和 math 3 个字段：

```
SELECT name, chinese, math FROM grade WHERE chinese > 90 OR math > 90
```

- 从 grade 数据表筛选出 chinese 分数小于 90 且 math 分数大于 90，或 chinese 分数小于 90 且 english 分数大于 90 之记录的所有字段：

```
SELECT * FROM grade WHERE chinese < 90 AND（math > 90 OR english > 90)
```

除了前述的比较运算符和逻辑运算符，SQL 语法也支持 LIKE 运算符，这个运算符接受 % 通配符，% 表示任何长度的字符串（包括 0）。

- 从 grade 数据表筛选出 name 是以“陈”开头记录的所有字段，请注意，字符串的前后要记得加上单引号（'）：

```
SELECT * FROM grade WHERE name LIKE '陈%'
```

- 从 grade 数据表筛选出 name 是以“陈”开头记录的 name 和 math 字段，请注意，字

符串的前后要记得加上单引号（'）：

```
SELECT name, math FROM grade WHERE name LIKE '陈%'
```

WHERE 条件子句也接受如下句型：

- 我们可以在 WHERE 条件子句中加入 IN 判断字段数据的范围，以下面的 SQL 指令为例，它会筛选出 chinese 字段为 80、85 或 88 记录的所有字段：

```
SELECT * FROM grade WHERE chinese IN（80, 85, 88)
```

- 如果数据的范围为文字字符串，那么千万别忘了在字符串的前后加上单引号（'），例如：

```
SELECT * FROM grade WHERE name IN（'陈小新', '林小佩', '孙小美')
```

- 我们可以在 WHERE 条件子句中加入 BETWEEN 限制筛选范围，以下面的 SQL 指令为例，它会筛选出 math 字段在 80～90（包含 80 和 90）记录的所有字段：

```
SELECT * FROM grade WHERE math BETWEEN 80 AND 90
```

12.2.2　SELECT ... FROM ... ORDER BY ... 语法（排序）

有时我们会需要将筛选出来的记录按照递增或递减顺序来进行排序，举例来说，假设要按照语文成绩由低到高的递增顺序来进行排序，那么，得加上 ORDER BY 排序子句：

```
SELECT * FROM grade ORDER BY chinese ASC
```

由于 ORDER BY 排序子句预设的排序方式为递增，因此，ASC 可以省略不写，若要改为由高到低的递减顺序，则要改写为如下的形式：

```
SELECT * FROM grade ORDER BY chinese DESC
```

事实上，我们也可以根据不只一个字段来进行排序，举例来说，假设要先按照 chinese 成绩的高低进行递减排序，再按照 math 成绩的高低进行递减排序，那么可以写成如下的形式：

```
SELECT * FROM grade ORDER BY chinese DESC, math DESC
```

从 grade 数据表筛选出所有字段，然后将 chinese、math 和 english 3 个字段相加后的分数生成为新的 total 字段，再按照 total 字段由高到低进行排序，如图 12-5 所示。

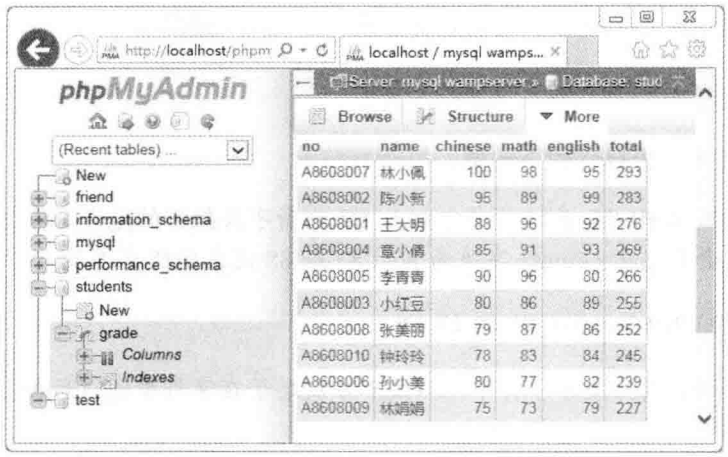


图 12-5

【提示】

这个 SQL 指令的写法有下列两种，ORDER BY 子句可以使用字段别名：

```
SELECT no, name, chinese, math, english, chinese+math+english AS total
  FROM grade ORDER BY chinese+math+english DESC

SELECT no, name, chinese, math, english, chinese+math+english AS total
  FROM grade ORDER BY total DESC
```

从 grade 数据表筛选出 name、math 两个字段，然后将 chinese、english 两个字段相加后的分数生成为新的 language 字段，再将 language 字段在 165 分以上者按照 math 分数由高到低进行排序，如图 12-6 所示。

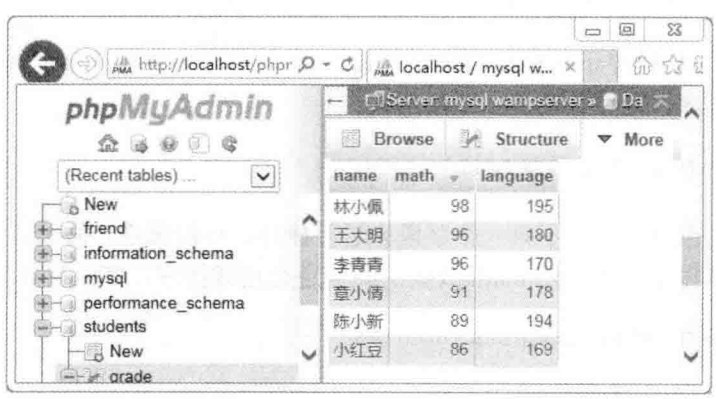


图 12-6

【提示】

这个 SQL 指令的写法如下：

```
SELECT name, math, chinese+english AS language
```

```
FROM grade WHERE chinese+english > 165 ORDER BY math DESC
```

12.2.3　SELECT ... LIMIT 语法（设置最多返回的记录数）

有时符合查询条件的记录可能有很多个，但我们并不需要看到所有记录，只是想看看前几个记录。举例来说，假设我们希望 grade 数据表的记录按照语文分数由高到低进行排序，但只要看看前 5 个记录，那么可以加上 LIMIT 语法来限制最多返回的个数：

```
SELECT * FROM grade ORDER BY chinese DESC LIMIT 5
```

LIMIT 语法还支持另一种写法，例如：

```
SELECT * FROM grade ORDER BY chinese DESC LIMIT 3, 5
```

这表示按照 chinese 分数由高到低进行排序，返回的记录从第 4 个开始，共返回 5 个，所以是得到第 4 ~ 8 个记录。

从 grade 数据表筛选出所有字段，然后将 chinese、math 和 english 3 个字段相加后的分数生成为新的 total 字段，再按照 total 字段由高到低将前 3 名显示在浏览器中，如图 12-7 所示。

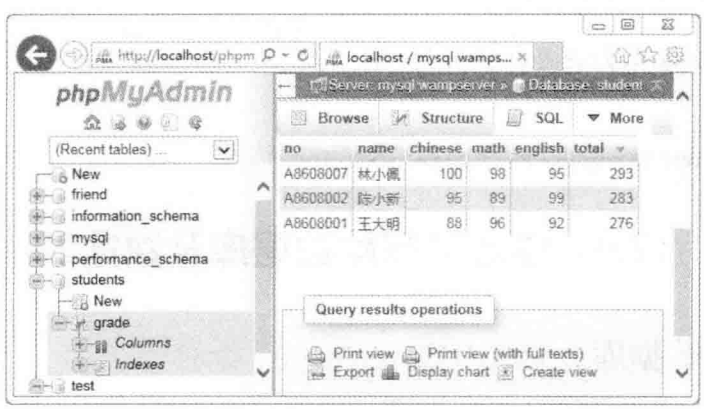


图 12-7

12.3　添加、更新与删除记录

12.3.1　使用 INSERT 语句新增记录

SQL 指令的 INSERT 语句可以在数据表内插入新的记录，其语法如下：

```
INSERT INTO 数据表名称（字段1, 字段2, 字段3...) Values（数据1, 数据2, 数据3...)
```

假设要在 grade 数据表内插入一个新的记录，其字段数据分别为“A8608011”、“小丸子”、“88”、“95”、“92”，那么可以写成如下的形式：

```
INSERT INTO grade（no, name, chinese, math, english) VALUES（'A8608011', '小丸子', 88, 95, 92)
```

12.3.2 使用 UPDATE 语句更新记录

SQL 指令的 UPDATE 语句可以更新数据表内现有的记录，其语法如下：

```
UPDATE 数据表名称 SET 字段1 = 数据1, 字段2 = 数据2... WHERE 条件
```

假设要将 grade 数据表内 no 字段为“A8608011”记录的 name 字段更新为“张小毛”、english 字段更新为“100”，那么可以写成如下的形式：

```
UPDATE grade SET name = '张小毛', english = 100 WHERE no = 'A8608011'
```

12.3.3 使用 DELETE 语句删除记录

SQL 指令的 DELETE 语句可以删除数据表内已有的记录，其语法如下：

```
DELETE FROM 数据表名称 WHERE 条件
```

假设要删除 grade 数据表内 english 分数低于 85 且 math 分数低于 85 的记录，那么可以写成如下的形式：

```
DELETE FROM grade WHERE english < 85 AND math < 85
```

12.4 创建与删除数据库及数据表

12.4.1 创建数据库

SQL 指令的 CREATE DATABASE 语句可以创建数据库，其语法如下，DEFAULT CHARACTER SET utf8 COLLATE utf8_general_ci 是让数据库支持多国语言：

```
CREATE DATABASE 数据库名称 DEFAULT CHARACTER SET utf8 COLLATE utf8_general_ci
```

假设要创建名称为 web_database 的数据库，可以写成如下的形式：

```
CREATE DATABASE web_database DEFAULT CHARACTER SET utf8 COLLATE utf8_general_ci
```

12.4.2　删除数据库

SQL 指令的 DROP DATABASE 语句可以删除数据库，其语法如下：

```
DROP DATABASE 数据库名称
```

假设要删除名称为 web_database 的数据库，可以写成如下的形式：

```
DROP DATABASE web_database
```

12.4.3　创建数据表

SQL 指令的 CREATE TABLE 语句可以创建新的数据表，其语法如下：

```
CREATE TABLE 数据表名称 (字段名 1 数据类型 [字段选项][, 字段名 2 数据类型
[字段选项] …… [, 字段名 n 数据类型 [字段选项]]][, PRIMARY KEY (字段名)])
```

假设要创建一个名称为 email 的数据表，里面只有一个名称为 address 的字段，数据类型为 TEXT，字段选项为 NOT NULL，可以写成如下的形式：

```
CREATE TABLE email（address TEXT NOT NULL)
```

我们再看另一个比较复杂的例子，假设要创建一个名称为 guestbook 的数据表，其字段结构如下：

字段名	数据类型	字段选项
id	INT	UNSIGNED、NOT NULL、AUTO_INCREMENT、PRIMARY KEY。
name	VARCHAR(8)	NOT NULL
content	MEDIUMTEXT	NOT NULL
date	DATETIME	NOT NULL

可以写成如下的形式：

```
CREATE TABLE guestbook（id INT UNSIGNED NOT NULL AUTO_INCREMENT,
                        name VARCHAR(8) NOT NULL,
                        content MEDIUMTEXT NOT NULL,
                        date DATETIME NOT NULL,
                        PRIMARY KEY（id ))
```

12.4.4　删除数据表

SQL 指令的 DROP TABLE 语句可以删除数据表，其语法如下：

```
DROP TABLE 数据表名称
```

假设要删除名称为 guestbook 的数据表，可以写成如下：

```
DROP TABLE guestbook
```

第 13 章

访问 MySQL 数据库

13.1 PHP 与 MySQL 数据库

从 PHP 5.5 开始，过去用来访问 MySQL 数据库的 mysql_connect()、mysql_close() 等数十个函数都标记为过时（deprecated），在新版 PHP 推出时，这些函数可能会被直接删除，建议勿再使用，以免程序出错。

数据库访问有两种语法：面向对象和函数，前者可以充分展示 PHP 的面向对象功能，而后者则容易学习，因此，本书将采用函数语法。不过，在说明函数语法之前，我们先来简单介绍面向对象语法，好让您对此语法有一个基本概念，若您想进一步学习，可以参考 PHP 文件（http://ca2.php.net/manual/en/book.mysqli.php）。

下面是一个例子，它会打开 product 数据库，从中读取 category 为主板的数据，并显示出来，浏览结果如图 13-1 所示。

\ch13\mysqli_oo.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>category 为主板的数据</title>
  </head>
  <body>
    <h1 align="CENTER">零部件报价表</h1>
    <?php
      $mysqli = new mysqli("localhost", "root", "mypassword", "product");

      if（$mysqli->connect_errno)
        die("无法建立数据连接: " . $mysqli->connect_error);

      $mysqli->query("SET NAMES utf8");
      $result = $mysqli->query("SELECT * FROM PRICE WHERE category = '主板'");

      echo "<table border='1' align='center'><tr align='center'>";

      while（$field = $result->fetch_field())    // 显示字段名
        echo "<td>" . $field->name . "</td>";
      echo "</tr>";

      while（$row = $result->fetch_row())
      {
        echo "<tr>";

        for（$i = 0; $i < $result->field_count; $i++)
```

```
            echo "<td>" . $row[$i]. "</td>";

        echo "</tr>";
    }

    echo "</table>";

    $result->free();
    $mysqli->close();
?>
</body>
</html>
```

图 13-1

PHP 提供了数十个函数让用户访问 MySQL 数据库，我们将在相关章节中为您介绍如下的函数。

函数	说明	页数
mysqli_affected_rows()	获取最近一次执行 INSERT、UPDATE、REPLACE 或 DELETE 命令时，被影响的记录数	13-18
mysqli_close()	关闭数据连接	13-7
mysqli_connect()	建立数据连接	13-5
mysqli_data_seek()	移动记录指针	13-34
mysqli_connect_errno()	返回最近一次调用 mysqli_connect() 函数所产生的错误代码	13-12
mysqli_connect_error()	返回最近一次调用 mysqli_connect() 函数所产生的错误信息	13-12
mysqli_errno()	返回最近一次访问 MySQL 所产生的错误代码	13-12
mysqli_error()	返回最近一次访问 MySQL 所产生的错误信息	13-12
mysqli_fetch_array()	将查询结果存入关联数组或数值数组	13-29

（续表）

函数	说明	页数
mysqli_fetch_assoc()	将查询结果存入关联数组（associative array）	13-33
mysqli_fetch_field()	从查询结果中获取字段信息，返回值为 object 类型	13-24
mysqli_fetch_object()	从查询结果中获取记录信息，返回值为 object 类型	13-33
mysqli_fetch_row()	从查询结果中获取记录信息，返回值为数组类型	13-26
mysqli_field_seek()	移动字段指针	13-25
mysqli_free_result()	释放查询结果所占用的内存	13-27
mysqli_get_client_info()	获取 MySQL 客户端函数库的版本信息	13-8
mysqli_get_host_info()	获取 MySQL 主机（host）的相关信息	13-9
mysqli_get_proto_info()	获取 MySQL 数据库协议（protocol）的版本信息	13-10
mysqli_get_server_info()	获取 MySQL 数据库的版本信息	13-11
mysqli_num_fields()	获取执行 SELECT 语句时，结果所包含的字段数	13-18
mysqli_num_rows()	获取执行 SELECT 语句时，结果所包含的记录数	13-18
mysqli_query()	执行 SQL 指令	13-15
mysqli_select_db()	打开数据库	13-13

13.2 建立与关闭数据连接

13.2.1 建立数据连接

在使用 PHP 访问 MySQL 数据库之前，必须先建立“数据连接”，也就是使用指定的账号与密码登录 MySQL 数据库服务器。

我们可以使用 mysqli_connect() 函数建立数据连接，其语法如下，若建立数据连接成功，就会返回连接标识符（link identifier），否则返回 FALSE：

```
mysqli_connect([string host[, string username[, string password[,string dbname]]]])
```

- *host*：MySQL 数据库服务器的计算机名称、DNS 名称或 IP 地址，例如 localhost，参数可以包含 port 信息，例如 localhost:1000，若省略此参数，默认值为 localhost:3306。
- *username*：登录 MySQL 数据库服务器的账号。
- *password*：登录 MySQL 数据库服务器的密码。
- *dbname*：默认的数据库名称。

下面是一个例子，它会试着建立数据连接，其中程序第 09 行是调用 mysqli_connect() 函数建立数据连接，此处的主机名、登录账号与密码请根据您的实际情况进行设置。若建立数据连接失败，就会执行 or 后面的 die("无法建立连接")，终止程序并显示“无法建立连接”，否则会执行第 10 行，显示“成功建立连接”，结果如图 13-2 所示。

\ch13\mysqli_connect.php

```
01:<!doctype html>
02:<html>
03:   <head>
04:      <meta charset="utf-8">
05:      <title>建立数据连接</title>
06:   </head>
07:   <body>
08:      <?php
09:         $link = mysqli_connect("localhost", "root", "mypassword") or die("无法建立连接");
10:         echo "成功建立连接";
11:      ?>
12:   </body>
13:</html>
```

图 13-2

- 当您使用 mysqli_connect() 函数建立数据连接发生错误时，会出现额外的错误信息，若不要在网页上显示这些错误信息，可以在调用 mysqli_connect() 函数的前面加上错误控制运算符 @。
- 为了方便起见，本书所有的 PHP 程序在建立数据连接时，均是以 root 账号登录 MySQL 数据库服务器，但站在安全性角度来看，这样的做法并不妥当，因为 root 账号的权限太大，通常只有在管理 MySQL 数据库服务器时，才会使用 root 账号登入，以免密码外泄。

13.2.2　关闭数据连接

虽然使用 mysqli_connect() 函数所建立的数据连接在所有程序代码执行完毕后会自动关

闭，但我们建议，在不需要访问数据库时，就使用 mysqli_close() 函数关闭数据连接，无须等到所有程序代码执行完毕，其语法如下：

```
mysqli_close(resource link_identifier)
```

下面是一个例子，它和 <\ch13\mysqli_connect.php> 的差别在于第 11 行是以手动的方式关闭数据连接。

\ch13\mysqli_close.php

```
01:<!doctype html>
02:<html>
03:   <head>
04:      <meta charset="utf-8">
05:      <title>关闭数据连接</title>
06:   </head>
07:   <body>
08:      <?php
09:         $link = mysqli_connect("localhost", "root", "mypassword") or die("无法建立连接");
10:         echo "成功建立连接";
11:         mysqli_close($link);
12:      ?>
13:   </body>
14:</html>
```

13.3 访问 MySQL 数据库服务器

13.3.1 获取 MySQL 客户端函数库的版本信息

我们可以使用 mysqli_get_client_info() 函数获取 MySQL 客户端函数库的版本信息，其语法如下，它没有参数，直接使用即可，返回值为 string 类型：

```
mysqli_get_client_info()
```

下面是一个例子，它会显示 MySQL 客户端程序库的版本信息，浏览结果如图 13-3 所示。

\ch13\mysqli_get_client_info.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>获取 MySQL 客户端程序库的版本信息</title>
```

```
  </head>
  <body>
    <?php
      echo "MySQL 客户端程序库的版本：  " . mysqli_get_client_info();
    ?>
  </body>
</html>
```

图 13-3

13.3.2　获取 MySQL 主机的相关信息

我们可以使用 mysqli_get_host_info() 函数获取 MySQL 主机的相关信息，其语法如下，*link_identifier* 为连接标识符：

```
mysqli_get_host_info(resource link_identifier)
```

下面是一个例子，它会显示 MySQL 主机的相关信息，其中连接标识符 $link 通过 TCP/IP 通信协议连接到 localhost 这台 MySQL 数据库服务器。浏览结果如图 13-4 所示。

\ch13\mysqli_get_host_info.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>获取 MySQL 主机的信息</title>
  </head>
  <body>
    <?php
      $link = mysqli_connect("localhost", "root", "mypassword");
      echo '$link  连接的主机为：' . mysqli_get_host_info($link);
      mysqli_close($link);
    ?>
```

```
  </body>
</html>
```

图 13-4

13.3.3 获取 MySQL 数据库协议的版本信息

我们可以使用 mysqli_get_proto_info() 函数获取 MySQL 数据库协议的版本信息，其语法如下，*link_identifier* 为连接标识符：

```
mysqli_get_proto_info(resource link_identifier)
```

下面是一个例子，它会显示 MySQL 数据库协议的版本信息。结果如图 13-5 所示。

\ch13\mysqli_get_proto_info.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>获取 MySQL 数据库协议的版本信息</title>
  </head>
  <body>
    <?php
      $link = mysqli_connect("localhost", "root", "mypassword");
      echo '$link 资源变量的协议版本为： ' . mysqli_get_proto_info($link);
      mysqli_close($link);
    ?>
  </body>
</html>
```

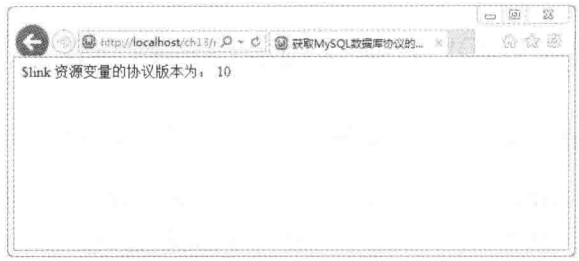


图 13-5

13.3.4　获取 MySQL 数据库服务器的版本信息

我们可以使用 mysqli_get_server_info() 函数获取 MySQL 数据库服务器的版本信息，其语法如下，*link_identifier* 为连接标识符：

```
mysqli_get_server_info(resource link_identifier)
```

下面是一个例子，它会显示 MySQL 数据库服务器的版本信息。结果如图 13-6 所示。

\ch13\mysqli_get_server_info.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>获取 MySQL 数据库的版本信息</title>
  </head>
  <body>
    <?php
      $link = mysqli_connect("localhost", "root", "mypassword");
      echo '$link 连接主机的数据库版本为：  ' . mysqli_get_server_info($link);
      mysqli_close($link);
    ?>
  </body>
</html>
```

图 13-6

13.3.5 获取访问 MySQL 数据库服务器的错误信息

我们可以通过下列函数获取在访问数据库时，所产生的错误代码与错误信息。

- mysqli_connect_errno(): 返回最近一次调用 mysqli_connect() 函数所产生的错误代码。
- mysqli_connect_error(): 返回最近一次调用 mysqli_connect() 函数所产生的错误信息。
- mysqli_errno(resource *link_identifier*): 返回最近一次访问 MySQL 数据库所产生的错误代码。
- mysqli_error(resource *link_identifier*): 返回最近一次访问 MySQL 数据库所产生的错误信息。

下面是一个例子，它改写自 <\ch13\mysqli_connect.php>。

\ch13\mysqli_error.php

```
01:<!doctype html>
02:<html>
03:   <head>
04:      <meta charset="utf-8">
05:      <title>显示错误代码及信息</title>
06:   </head>
07:   <body>
08:      <?php
09:         $link = @mysqli_connect("localhost", "root", "mypassword1")
10:            or die("无法建立连接: " . mysqli_connect_errno() . " " . mysqli_connect_error());
11:         echo "成功建立连接";
12:         mysqli_close($link);
13:      ?>
14:   </body>
15:</html>
```

调用 mysqli_connect() 函数建立数据连接，此处故意将登录 MySQL 数据库服务器的密码误设为"mypassword1"，以产生错误

一旦数据连接建立失败，就会执行此行，显示错误代码与错误信息

13.4 执行 SQL 指令

在执行 SQL 指令之前，除了要与 MySQL 数据库服务器建立数据连接，还要打开指定的数据库，才能对该数据库执行 SQL 指令。

13.4.1 打开数据库

我们可以使用 mysqli_select_db() 函数打开数据库，其语法如下，若打开数据库成功，就

返回 TRUE，否则返回 FALSE：

```
mysqli_select_db(resource link_identifier, string database_name)
```

- link_identifier：连接标识符。
- database_name：要打开的数据库名称。

下面是一个例子，它会调用 mysqli_select_db() 函数打开 students 数据库。

\ch13\mysqli_select_db.php

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:    <title>打开数据库</title>
06:  </head>
07:  <body>
08:    <?php
09:      $link = mysqli_connect("localhost", "root", "mypassword")
10:        or die("无法建立连接: " . mysqli_connect_error());
11:      mysqli_select_db($link, "students")
12:        or die（"无法打开 students 数据库: " . mysqli_error($link));
13:      mysqli_close($link);
14:    ?>
15:  </body>
16:</html>
```

还记得 mysqli_connect() 函数的语法吗？其语法如下，其中 *dbname* 参数可以用来指定默认的数据库名称。

```
mysqli_connect([string host[, string username[, string password[,string dbname]]]])
```

因此，mysqli_select_db.php 也可以写成如下的形式，在使用 mysqli_connect() 函数建立数据连接时，直接使用第 4 个参数来指定要使用哪个数据库。

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>打开数据库</title>
  </head>
  <body>
    <?php
      $link = mysqli_connect("localhost", "root", "mypassword", "students")
```

```
        or die("无法建立连接: " . mysqli_connect_error());

    mysqli_close($link);
  ?>
 </body>
</html>
```

13.4.2 执行 SQL 指令

打开数据库后，我们可以使用 mysqli_query() 函数执行 SQL 指令，其语法如下：

```
mysqli_query(resource link_identifier, string query)
```

- link_identifier: 连接标识符。
- query: 要执行的 SQL 指令。
- mysqli_query() 函数的执行结果有下列两种:
- 失败: 一律返回 FALSE。
- 成功: 返回 TRUE，当 mysqli_query() 函数执行 SELECT、SHOW、EXPLAIN 或 DESCRIBE 语句时，则会返回资源标识符（mysqli_result 对象），指向查询结果，您可以将它想象成位于内存中的数据库。

下面是一个例子，它会使用 mysqli_query() 函数执行 SELECT 语句。

\ch13\mysqli_query.php

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:    <title>执行 SELECT 语句</title>
06:  </head>
07:  <body>
08:    <?php
09:      $link = mysqli_connect("localhost", "root", "mypassword")
10:        or die("无法建立连接: " . mysqli_connect_error());
11:
12:      mysqli_select_db($link, "product")
13:        or die（"无法打开 prodcut 数据库: " . mysqli_error($link));
14:
15:      $sql = "SELECT * FROM price WHERE category = '主板'";
16:      $result = mysqli_query($link, $sql);
17:
18:      mysqli_close($link);
```

```
19:     ?>
20:   </body>
21:</html>
```

- 09：调用 mysqli_connect() 函数建立数据连接，此处的主机名、登录账号与密码请根据您的实际情况进行设置。
- 10：当建立数据连接失败时，会执行此行，显示“无法建立连接”并调用 mysqli_connect_error() 函数显示错误信息，然后终止程序。
- 12：打开 product 数据库。
- 13：当打开 product 数据库失败时，会执行此行，显示“无法打开 product 数据库”并调用 mysqli_error() 函数显示错误信息，然后终止程序。
- 15：指定要执行的 SQL 指令。
- 16：使用 mysqli_query() 函数执行 SQL 指令，$result 的返回值为资源标识符，指向查询结果。

请注意，强烈建议您在第 12 行之前使用 mysqli_query() 函数执行“SET NAMES utf8”指令，指定查询所要使用的字符集名称，因为我们通常会将 MySQL 数据库的编码方式设置为 UTF-8，才能支持多国字符，故须将查询所要使用的字符集名称设置为 utf8，否则一旦查询结果包含非 ASCII 字符，将会出现乱码。

```
mysqli_query($link, "SET NAMES utf8");
```

由于每次访问数据库都必须建立数据连接并打开数据库，因此，我们编写下列函数，在需要建立数据连接或执行 SQL 指令时，就可以直接调用，减少重复编写程序代码。

\ch13\dbtools.inc.php

```
01:<?php
02:   function create_connection()
03:   {
04:      $link = mysqli_connect("localhost", "root", "mypassword")
05:         or die("无法建立数据连接: " . mysqli_connect_error());
06:      mysqli_query($link, "SET NAMES utf8");
07:      return $link;
08:   }
09:
10:   function execute_sql($link, $database, $sql)
11:   {
12:      mysqli_select_db($link, $database)
13:         or die("打开数据库失败: " . mysqli_error($link));
14:      $result = mysqli_query($link, $sql);
15:      return $result;
16:   }
```

```
17:?>
```

- 02 ~ 08: 定义 create_connection() 函数，用来建立数据连接，第 06 行可以解决数据库中文乱码问题。
- 10 ~ 16: 定义 execute_sql() 函数，用来执行指定的 SQL 指令，此函数包含 3 个参数，*link* 用来指定要使用的数据连接，*database* 用来指定数据库名称，*sql* 用来指定要执行的 SQL 指令，举例来说，假设您要对 product 数据库执行 "SELECT * FROM PRICE WHERE category = '主板'" 指令，可以写成如下的形式:

```
execute_sql($link, "prodcut", " SELECT * FROM PRICE WHERE category = '主板'");
```

13.4.3 获取执行 SQL 指令被影响的记录数或字段数

PHP 提供下列三个函数，让我们能够得知在执行 SQL 指令后，有多少个记录或多少个字段受到影响。

- mysqli_num_rows() 函数: 适用于执行 SELECT 语句，可以返回被筛选出来的记录数，其语法如下，参数 *result* 为资源标识符（resource identifier）:

```
mysqli_num_rows(resource result)
```

- mysqli_num_fields() 函数: 适用于执行 SELECT 语句，可以返回被筛选出来的字段数目，其语法如下，参数 *result* 为资源标识符:

```
mysqli_num_fields(resource result)
```

- mysqli_affected_rows() 函数: 适用于执行 INSERT、UPDATE、REPLACE、DELETE 语句，可以返回有多少个记录受到该语句的影响，其语法如下，参数 *link_identifier* 为连接标识符（link identifier）:

```
mysqli_affected_rows(resource link_identifier)
```

若在执行 DELETE 语句时没有指定 WHERE 子句，将导致数据表内所有记录被删除，此时，mysqli_affected_rows() 函数会返回 0，而不是实际被删除的记录数。

若最近一次执行 SQL 指令的结果为失败，那么 mysqli_affected_rows() 函数会返回 -1。

此外，在执行 UPDATE 语句时，mysqli_affected_rows() 函数返回的是实际被更新的记录数，而不是符合 WHERE 子句的记录数，因为当指定的新值与旧值相同时，并不会有更新的动作。

下面是一个例子，它会执行 SELECT 语句，然后显示筛选出几个记录及字段数目。浏览结果如图 13-7 所示。

\ch13\mysqli_num_rows.php

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:    <title>执行 SELECT 语句时，被影响的记录及字段数目。</title>
06:  </head>
07:  <body>
08:    <?php
09:      require_once("dbtools.inc.php");
10:      $link = create_connection();
11:      $sql = "SELECT * FROM price WHERE category = '主板'";
12:      $result = execute_sql($link, "product", $sql);
13:      echo "category = “主板” 的记录有 " . mysqli_num_rows($result) . " 笔";
14:      echo "，包含 " . mysqli_num_fields($result) . " 个字段。";
15:      mysqli_close($link);
16:    ?>
17:  </body>
18:</html>
```

- 09：使用 require_once() 函数将 dbtools.inc.php 引用进来，我们才能调用自行定义的 create_connection() 与 execute_sql() 两个函数。
- 10：调用自行定义的 create_connection() 函数建立数据连接。
- 11 ~ 12：调用自行定义的 execute_sql() 函数来对 product 数据库执行 “SELECT * FROM price WHERE category = '主板'” 指令。

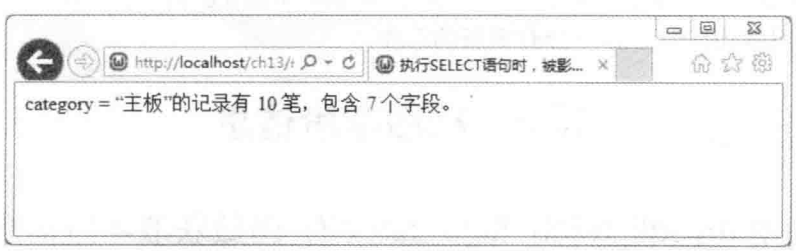


图 13-7

下面是一个例子，它会执行 UPDATE 语句更新记录，然后显示被更新的记录数。浏览结果如图 13-8 所示。

\ch13\mysqli_affected_rows.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
```

```
    <title>执行 UPDATE 语句时，被影响的记录数目。</title>
  </head>
  <body>
    <?php
      require_once("dbtools.inc.php");

      $link = create_connection();
      $sql = "UPDATE price SET url = 'www.asus.com/tw/' WHERE url = 'tw.asus.com'";
      $result = execute_sql($link, "product", $sql);

      echo "执行 UPDATE 语句时，共有 " . mysqli_affected_rows($link) . " 笔记录受影响";

      mysqli_close($link);
    ?>
  </body>
</html>
```

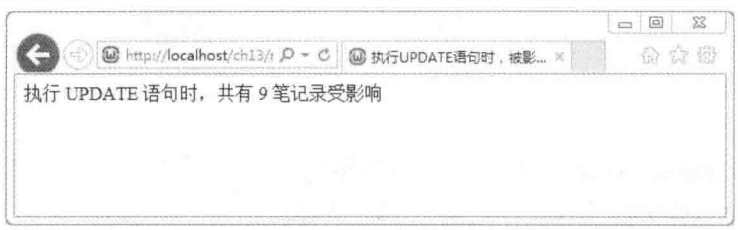


图 13-8

请注意，若您将上述范例的 SQL 指令改成“UPDATE price SET url = 'tw.asus.com' WHERE url = 'tw.asus.com'”，即新值与旧值相同，那么 mysqli_affected_rows() 函数返回来的值为 0，因为当新值与旧值相同时，并不会有更新的动作。

13.5 获取字段信息

13.5.1 使用 mysqli_fetch_field_direct() 函数获取字段信息

成功使用 mysqli_query() 函数执行 SELECT 语句后，该函数会返回资源标识符，此时，我们可以使用 mysqli_fetch_field_direct() 函数获取字段信息，其语法如下：

```
mysqli_fetch_field_direct(resource result, int field_offset)
```

- *result*: 资源标识符。
- *field_offset*: 字段的序号，0 表示第一个字段，1 表示第二个字段，2 表示第三个字段，以次类推。

mysqli_fetch_field_direct() 函数的返回值为 object 类型，常用的属性如下。

属性	说明
name	字段名
orgname	字段的原始名称
table	字段所属的数据表名称
orgtable	字段所属的原始数据表名称
db	字段所属的数据库名称
max_length	字段内容实际存放的最大长度，不是数据库内设置的数据长度
length	字段在数据库内设置的数据长度
type	字段类型，3 代表 Integer、10 代表 DATE、253 代表 VARCHAR

举例来说，假设要获取第 2 个字段信息，可以写成如下的形式：

```
$meta = mysqli_fetch_field_direct($result, 1);
```

假设要获取第 2 个字段的信息并显示其字段名及数据类型，可以写成如下的形式：

```
$meta = mysqli_fetch_field_direct($result, 1);
echo "字段名：$meta->name";
echo "数据类型：$meta->type";
```

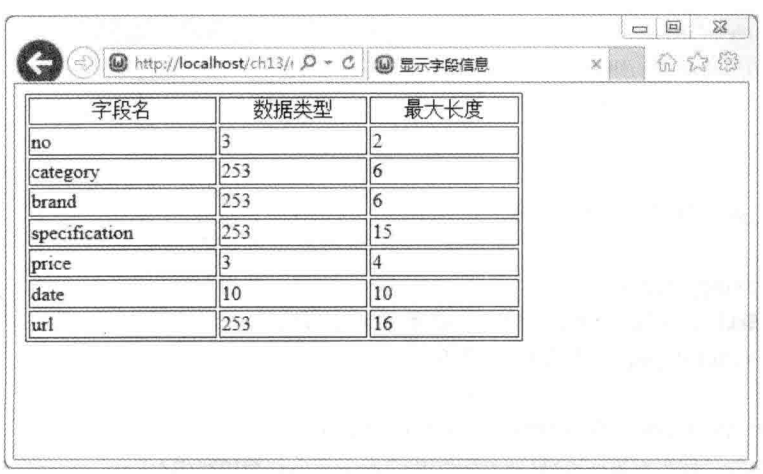
下面是一个例子。浏览结果如图 13-9 所示。

\ch13\mysqli_fetch_field_direct.php

```
01:<!doctype html>
02:<html>
03:   <head>
04:      <meta charset="utf-8">
05:      <title>显示字段信息</title>
06:   </head>
07:   <body>
08:      <?php
09:         require_once("dbtools.inc.php");
10:
11:         $link = create_connection();
12:         $sql = "SELECT * FROM PRICE WHERE category = '主板'";
13:         $result = execute_sql($link, "product", $sql);
14:
15:         echo "<table width='400' border='1'><tr align='center'>";
16:         echo "<td>字段名</td><td>数据类型</td><td>最大长度</td></tr>";
17:
18:         $i = 0;
```

```
19:       while（$i < mysqli_num_fields($result))
20:       {
21:          $meta = mysqli_fetch_field_direct($result, $i);
22:          echo "<tr>";
23:          echo "<td>$meta->name</td>";
24:          echo "<td>$meta->type</td>";
25:          echo "<td>$meta->max_length</td>";
26:          echo "</tr>";
27:          $i++;
28:       }
29:       echo "</table>";
30:
31:       mysqli_close($link);
32:    ?>
33: </body>
34:</html>
```

- 09：使用 require_once() 函数将 dbtools.inc.php 引用进来，我们才能调用自行定义的 create_connection() 与 execute_sql() 两个函数。
- 11：调用自行定义的 create_connection() 函数建立数据连接。
- 12 ~ 13：让自行定义的 execute_sql() 函数来对“product”数据库执行“SELECT * FROM PRICE WHERE category = '主板'”指令。
- 18 ~ 28：显示数据表内所有字段的字段名、数据类型及最大长度，其中数据类型的返回值为数值，不同数字代表不同数据类型，例如 3 代表 Integer、10 代表 DATE、253 代表 VARCHAR，详细信息可以参考 PHP 文件（http://www.php.net/manual/en/mysqli-result.fetch-field-direct.php）。
- 31：关闭数据连接。



字段名	数据类型	最大长度
no	3	2
category	253	6
brand	253	6
specification	253	15
price	3	4
date	10	10
url	253	16

图 13-9

13.5.2　使用 mysqli_fetch_field() 函数获取字段信息

我们也可以使用 mysqli_fetch_field() 函数获取字段信息，其语法如下：

```
mysqli_fetch_field(resource result)
```

mysqli_fetch_field() 函数会返回与 mysqli_fetch_field_direct() 函数相同的 object，下面这个例子的执行结果与第 13.5.1 小节一样。

\ch13\mysqli_fetch_field.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>显示字段信息</title>
  </head>
  <body>
    <?php
      require_once("dbtools.inc.php");
      $link = create_connection();
      $sql = "SELECT * FROM PRICE WHERE category = '主板'";
      $result = execute_sql($link, "product", $sql);
      echo "<table width='400' border='1'><tr align='center'>";
      echo "<td>字段名</td><td>数据类型</td><td>最大长度</td></tr>";
      while（$meta = mysqli_fetch_field($result))
      {
        echo "<tr>";
        echo "<td>$meta->name</td>";
        echo "<td>$meta->type</td>";
        echo "<td>$meta->max_length</td>";
        echo "</tr>";
      }
      echo "</table>" ;
      mysqli_close($link);
    ?>
  </body>
</html>
```

13.5.3　移动字段指针

请您回想一下 mysqli_fetch_field() 函数的语法：

```
mysqli_fetch_field(resource result)
```

此函数可以用来获取当前字段的信息，那么 mysqli_fetch_field() 函数如何知道当前位于哪个字段呢？是这样的，事实上有一个字段指针记录着当前位于哪个字段。

mysqli_field_seek() 函数可以让我们轻松地移动字段指针，其语法如下，若移动字段指针成功，就返回 TRUE，否则返回 FALSE：

```
mysqli_field_seek(resource result, int field_offset)
```

- *result*：资源标识符。
- *field_offset*：字段的序号，0 表示第一个字段，1 表示第二个字段，2 表示第三个字段，以次类推。

举例来说，在下面的程序代码执行完毕后，变量 $meta 到底是存储第几个字段的信息呢？答案是第 5 个字段，因为我们在第一个语句中已经将字段指针移到第 5 个字段了。

```
$seek_result = mysqli_field_seek($result, 4);
$meta = mysqli_fetch_field($result);
```

13.6 获取记录内容

在获取字段信息后，接下来就要想办法获取记录的内容，请您仔细阅读本节。

13.6.1 使用 mysqli_fetch_row() 函数获取记录内容

SELECT 语句执行完毕后所返回来的资源标识符，其实就是筛选的结果，里面可能包含多个记录，其中有一个记录指针，用来标记当前的记录是在第几笔，记录指针的默认值为 0，表示在第一个记录。

PHP 提供的 mysqli_fetch_row() 函数可以用来读取一个记录，然后将记录指针移到下一个，若读不到记录，就返回 FALSE，其语法如下：

```
mysqli_fetch_row(resource result)
```

举例来说，下面的代码表示要读取 5 个记录，然后将读取的记录分别存放在数组 row1、row2、row3、row4、row5 中，由于记录指针的默认值为 0，因此，数组 row1 存放的是第 1 个记录，数组 row2 存放的是第 2 个记录，数组 row3 存放的是第 3 个记录，数组 row4 存放的是第 4 个记录，数组 row5 存放的是第 5 个记录：

```
$row1 = mysqli_fetch_row($result);
$row2 = mysqli_fetch_row($result);
$row3 = mysqli_fetch_row($result);
$row4 = mysqli_fetch_row($result);
```

```
$row5 = mysqli_fetch_row($result);
```

在存放记录的数组中（例如 row1、rows2、row3、rows4、row5），键代表的是字段序号，换句话说，若要显示第 2 个记录的第 3 个字段，可以写成如下的形式：

```
echo $row2[2];
```

同理，若要显示第 3 个记录的第 1 个字段，可以写成如下的形式：

```
echo $row3[0];
```

查询结果所包含的记录会占用服务器的内存，虽然在所有程序代码执行完毕后会自动释放占用的内存，但我们建议，可以在适当的时候使用 mysqli_free_result() 函数释放内存，不要等到所有程序代码执行完毕才自动释放，这个函数的语法如下，参数 *result* 为资源标识符：

```
mysqli_free_result(resource result)
```

我们可以使用嵌套循环来显示所有记录内容，外部循环用来读取每个记录，内部循环用来读取每个字段，下面是一个例子。浏览结果如图 13-10 所示。

\ch13\mysqli_fetch_row.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>显示所有记录</title>
  </head>
  <body>
    <h1 align="CENTER">零部件报价表</h1>
    <?php
      require_once("dbtools.inc.php");
      $link = create_connection();
      $sql = "SELECT * FROM PRICE WHERE category = '主板'";
      $result = execute_sql($link, "product", $sql);

      echo "<table border='1' align='center'><tr align='center'>";
      for（$i = 0; $i < mysqli_num_fields($result); $i++)     // 显示字段名
        echo "<td>" . mysqli_fetch_field_direct($result, $i)->name. "</td>";
      echo "</tr>";

      while（$row = mysqli_fetch_row($result))
      {
        echo "<tr>";
        for($i = 0; $i < mysqli_num_fields($result); $i++)
```

```
            echo "<td>$row[$i]</td>";
        echo "</tr>";
    }

    echo "</table>" ;

    mysqli_free_result($result);
    mysqli_close($link);
?>
</body>
</html>
```

图 13-10

13.6.2 使用 mysqli_fetch_array() 函数获取记录内容

mysqli_fetch_array() 函数也用来读取记录并存放在数组中，然后将记录指针移到下一个，若读不到记录，就返回 FALSE，不同之处在于获取字段内容时，mysqli_fetch_row() 函数以字段序号获取字段内容，而 mysqli_fetch_array() 函数则可以使用字段序号或字段名获取字段内容。

mysqli_fetch_array() 函数的语法如下：

```
mysqli_fetch_array(resource result[, int result_type])
```

- *result*：资源标识符。
- *result_type*：指定获取字段内容的方式，参数值有 MYSQLI_NUM、MYSQLI_ASSOC、MYSQLI_BOTH。mysqli_fetch_array() 和 mysqli_fetch_row() 函数的差别就在于这个参数，若将参数值设置为 MYSQLI_NUM，表示只能使用字段序号获取字段内容，这种方式和 mysqli_fetch_row() 函数一样；若将参数值设置为 MYSQLI_ASSOC，表

示只能使用字段名获取字段内容；若将参数值设置为 MYSQLI_BOTH，表示可以使用字段序号及字段名获取字段内容。

下面的程序代码改写自前一节的 <\ch13\mysqli_fetch_row.php>，但这次换用 mysqli_fetch_array() 函数，参数 *result_type* 设置为 MYSQLI_NUM，表示只能使用字段序号获取字段内容，您会发现写法和使用 mysqli_fetch_row() 函数时一样。

\ch13\mysqli_fetch_array_01.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>显示所有记录</title>
  </head>
  <body>
    <h1 align="center">零部件报价表</h1>
    <?php
      require_once("dbtools.inc.php");
      $link = create_connection();
      $sql = "SELECT * FROM PRICE WHERE category = '主板'";
      $result = execute_sql($link, "product", $sql);

      echo "<table border='1' align='center'><tr align='center'>";
      for（$i = 0; $i < mysqli_num_fields($result); $i++)
        echo "<td>" . mysqli_fetch_field_direct($result, $i)->name. "</td>";
      echo "</tr>";

      while（$row = mysqli_fetch_array($result, MYSQL_NUM))
              {
        echo "<tr>";
        for($i = 0; $i < mysqli_num_fields($result); $i++)
          echo "<td>$row[$i]</td>";

        echo "</tr>";
      }
      echo "</table>" ;
      mysqli_free_result($result);
      mysqli_close($link);
    ?>
  </body>
</html>
```

下面的程序代码也改写自前一节的 <\ch13\mysqli_fetch_row.php>，但这次换用 mysqli_fetch_array() 函数，且参数 *result_type* 设置为 MYSQLI_ASSOC，表示只能使用字段名获取字段内容，我们将不同的地方反白显示以进行比较。

\ch13\mysqli_fetch_array_02.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>显示所有记录</title>
  </head>
  <body>
    <h1 align="center">零部件报价表</h1>
        <?php
      require_once("dbtools.inc.php");
      $link = create_connection();
      $sql = "SELECT * FROM PRICE WHERE category = '主板'";
      $result = execute_sql($link, "product", $sql);

      echo "<table border='1' align='center'><tr align='center'>";
      for（$i = 0; $i < mysqli_num_fields($result); $i++)
        echo "<td>" . mysqli_fetch_field_direct($result, $i)->name. "</td>";
      echo "</tr>";

      while（$row = mysqli_fetch_array($result, MYSQLI_ASSOC))
      {
        echo "<tr>";
        echo "<td>" . $row["no"]. "</td>";
        echo "<td>" . $row["category"]. "</td>";
        echo "<td>" . $row["brand"]. "</td>";
        echo "<td>" . $row["specification"]. "</td>";
        echo "<td>" . $row["price"]. "</td>";
        echo "<td>" . $row["date"]. "</td>";
        echo "<td>" . $row["url"]. "</td>";
        echo "</tr>";
      }
      echo "</table>" ;
      mysqli_free_result($result);
      mysqli_close($link);
    ?>
  </body>
</html>
```

下面的程序代码也是改写自前一节的 <\ch13\mysqli_fetch_row.php>，但这次换用 mysqli_fetch_array() 函数，且参数 *result_type* 设置为 MYSQLI_BOTH，表示可以使用字段序号或字段名获取字段内容，我们将不同的地方反白显示以进行比较。

\ch13\mysqli_fetch_array_03.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>显示所有记录</title>
  </head>
  <body>
    <h1 align="CENTER">零部件报价表</h1>
    <?php
      require_once("dbtools.inc.php");
      $link = create_connection();
      $sql = "SELECT * FROM PRICE WHERE category = '主板'";
      $result = execute_sql($link, "product", $sql);
      echo "<table border='1' align='center'><tr align='center'>";
      for（$i = 0; $i < mysqli_num_fields($result); $i++)
        echo "<td>" . mysqli_fetch_field_direct($result, $i)->name. "</td>";
      echo "</tr>";
      while（$row = mysqli_fetch_array($result, MYSQLI_BOTH))
      {
        echo "<tr>";
        echo "<td>" . $row["no"]. "</td>";
        echo "<td>" . $row[1]. "</td>";
        echo "<td>" . $row["brand"]. "</td>";
        echo "<td>" . $row["specification"]. "</td>";
        echo "<td>" . $row[4]. "</td>";
        echo "<td>" . $row[5]. "</td>";
        echo "<td>" . $row["url"]. "</td>";
        echo "</tr>";
      }
      echo "</table>" ;
      mysqli_free_result($result);
      mysqli_close($link);
    ?>
  </body>
</html>
```

13.6.3 使用 mysqli_fetch_assoc() 函数获取记录内容

mysqli_fetch_ assoc() 也用来读取记录内容并存放在数组中，然后将记录指针移到下一个，其语法如下，参数 *result* 为资源标识符：

```
mysqli_fetch_assoc(resource result)
```

mysqli_fetch_assoc() 函数的功能就相当于 mysqli_fetch_array() 函数搭配 MYSQLI_ASSOC 参数值，所以在读取字段内容时，只能使用字段名。

13.6.4 使用 mysqli_fetch_object() 函数获取记录内容

mysqli_fetch_object() 函数也用来读取记录内容，然后将记录指针移到下一个，若读不到记录，就返回 FALSE，其语法如下，参数 *result* 为资源标识符：

```
mysqli_fetch_object(resource result)
```

由于 mysqli_fetch_object() 函数的返回值为 object 类型，记录的每个字段都会变成该对象的属性，所以在读取字段内容时，只能使用字段名。

若要改写前一节的 <\ch13\mysqli_fetch_array_02.php>，只要改写 while 循环即可，如下所示：

```
while ($row = mysqli_fetch_object($result))
{
   echo "<tr>";
   echo "<td>$row->no</td>";
   echo "<td>$row->category</td>";
   echo "<td>$row->brand</td>";
   echo "<td>$row->specification</td>";
   echo "<td>$row->price</td>";
   echo "<td>$row->date</td>";
   echo "<td>$row->url</td>";
   echo "</tr>";
}
```

13.6.5 移动记录指针

还记得我们说过，资源标识符里面有一个记录指针，用来标记当前的记录在第几个，而 mysqli_fetch_row()、mysqli_fetch_array()、mysqli_fetch_assoc()、mysqli_fetch_object() 等函数均用来读取当前的记录，然后将记录指针移至下一个，可是问题来了，您有没有想过，若只要读取第 10 个记录，难道要执行 10 次这些函数吗？

当然不是！PHP 提供了 mysqli_data_seek() 函数可以让我们轻松地移动记录指针，其语法

如下，若移动记录指针成功，就返回 TRUE，否则返回 FALSE：

```
mysqli_data_seek(resource result, int row_number)
```

- *result*：资源标识符。
- *row_number*：这是记录的序号，0 表示第一个记录，1 表示第二个记录，以次类推。

举例来说，下面的程序代码执行完毕后，变量 $row 到底是存放了第几个记录的内容呢？答案是第 10 个记录，因为我们在第一行语句中已经将记录指针移至第 10 个了。

```
$seek_result = mysqli_data_seek($result, 9);
$row = mysqli_fetch_row($result);
```

13.7　分页浏览

“分页浏览”是网页数据库经常使用的功能，当筛选出来的记录太多时，我们通常不会一次全部显示，而是以分页的方式来显示，才不会造成浏览单一网页的速度过慢。

图 13-11 是我们在本节中所要制作的分页网页 <\ch13\show_record.php>，每页显示 5 个记录，网页最下方有一个导航条，可以让您快速浏览其他页的记录。

图 13-11

\ch13\show_record.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>分页浏览</title>
  </head>
  <body>
    <h1 align="center">零部件报价表</h1>
    <?php
      require_once("dbtools.inc.php");

      //指定每页显示几个记录
      $records_per_page = 5;

      //获取要显示第几页的记录
      if（isset($_GET["page"]))
        $page = $_GET["page"];
      else
        $page = 1;

      //建立数据连接
      $link = create_connection();

      //执行 SQL 命令
      $sql = "SELECT category AS '零部件种类', brand AS '品牌', specification AS
                '规格', price AS '价格', date AS '报价日期' FROM Price";
      $result = execute_sql($link, "product", $sql);

      //获取字段数
      $total_fields = mysqli_num_fields($result);

      //获取记录数
      $total_records = mysqli_num_rows($result);

      //计算总页数
      $total_pages = ceil($total_records / $records_per_page);

      //计算本页第一个记录的序号
      $started_record = $records_per_page *（$page - 1);

      //将记录指针移至本页第一个记录的序号
```

```
42:         mysqli_data_seek($result, $started_record);
43:
44:         //显示字段名
45:         echo "<table border='1' align='center' width='800'>";
46:         echo "<tr align='center'>";
47:         for（$i = 0; $i < $total_fields; $i++)
48:            echo "<td>" . mysqli_fetch_field_direct($result, $i)->name . "</td>";
49:         echo "</tr>";
50:
51:         //显示记录
52:         $j = 1;
53:         while（$row = mysqli_fetch_row($result) and $j <= $records_per_page)
54:         {
55:            echo "<tr>";
56:            for($i = 0; $i < $total_fields; $i++)
57:               echo "<td>$row[$i]</td>";
58:            $j++;
59:            echo "</tr>";
60:         }
61:         echo "</table>" ;
62:
63:         //产生导航条
64:         echo "<p align='center'>";
65:         if（$page > 1)
66:            echo "<a href='show_record.php?page=".（$page - 1) . "'>上一页</a> ";
67:         for（$i = 1; $i <= $total_pages; $i++)
68:         {
69:            if（$i == $page)
70:               echo "$i ";
71:            else
72:               echo "<a href='show_record.php?page=$i'>$i</a> ";
73:         }
74:         if（$page < $total_pages)
75:            echo "<a href='show_record.php?page=".（$page + 1) . "'>下一页</a> ";
76:         echo "</p>";
77:
78:         //释放内存空间
79:         mysqli_free_result($result);
80:         mysqli_close($link);
81:      ?>
82:   </body>
83:</html>
```

- 13：指定每页显示几个记录，此处是 5，您可以根据实际情况进行设置。
- 16～19：设置要显示第几页的记录，一开始会先获取网址参数 page，我们使用 isset() 函数判断变量 $_GET["page"]是否获取了数值，若获取了数值，表示有浏览者指定要查看第 page 页的记录，就将变量 page 设置为获取的数值，否则将变量 page 设置为 1，让网页显示第一页的记录。
- 25～26：指定 SQL 指令，用来获取 category、brand、specification、price、date 5 个字段的所有记录并指定其中文别名。
- 36：计算总页数，此处使用了 ceil() 函数，一旦总页数出现小数点，就无条件进位。
- 39：计算在当前要显示的页数中，第一个记录是位于查询结果的第几行。
- 52 ～ 60：显示某一区间范围内的记录，第 53 行的意思是当读取到记录且 $j <= $records_per_page 时，才会执行 while 区块内的程序代码来显示记录，其中 $j <= $records_per_page 用来控制每页显示的记录数，此处为 5。
- 65～75：制作导航条，让浏览者快速换页。第 65～66 行的意思是当目前页数大于第一页时，就插入“上一页”超链接；第 74～75 行的意思是当目前页数小于最后一页时，就插入“下一页”超链接；第 67～73 行用来产生所有页码，当前页数的页码为纯文本，若非当前页数的页码则为超链接。

第 14 章

Google 地图应用网站

14.1 认识 Google API

顾名思义，Google API 就是 Google 提供给第三方或程序设计人员使用的函数或子程序，其种类繁多，大部分的 Google 服务都有对应的 API 可以调用，如下所示：

- Google Maps API：通过一系列的 Google Maps API，用户可以将 Google 的交互式地图集成到自己的网站或博客。
- Android：Android 是移动设备的软件套件，包括操作系统、中间件及主要的应用程序，而 Android SDK 则包含所有必要的工具和 API，可以协助用户开发适用于 Android 移动设备的应用程序。
- Google Calendar API：通过这组日历工具，用户可以编辑自己的日程安排，并集成到自己的网站或博客。
- Google Toolbar API：通过这组 API，用户可以建立 Google 工具栏的自定义按钮。
- YouTube API：通过这组 API，用户可以将 YouTube 的视频内容与功能集成到自己的应用程序、网站或设备，例如搜索视频、上传视频、建立播放列表、自定义播放程序等。
- Google Earth API：通过 Google Earth 插件及其 JavaScript API，可以让用户将 Google Earth（3D 数字地球）加入自己的网站，并进一步绘制标记与线条、在地表上加入图像、增加 3D 模型或加载 KML 文件，以建立复杂的 3D 地图应用程序。
- Google AJAX Search API：通过这组 API，用户可以将 Google 的搜索功能集成在自己的网站，而且搜索项目包括地图、视频、新闻、博客、图像、书籍等。

不同的 Google API 有不同的用途与使用方式，在本章中，我们会以实际的例子为您示范如何使用 Google Maps API 将 Google Maps 加入网页。至于其他 Google API 的相关信息，有兴趣的读者请自行参考 Google Developers 网站（https://developers.google.com/）。

14.2 在网页中加入 Google Maps

在本节中，我们会以实际的例子说明如何使用 Google Maps API 将 Google Maps 加入网页，其执行结果如图 14-1 所示，这个网页不仅能够在 PC 版浏览器上运行，也能够在移动版浏览器上运行。若您有需要，还可以结合第 12 章介绍的 jQuery Mobile，将这个网页改写成移动版的界面。

图 14-1

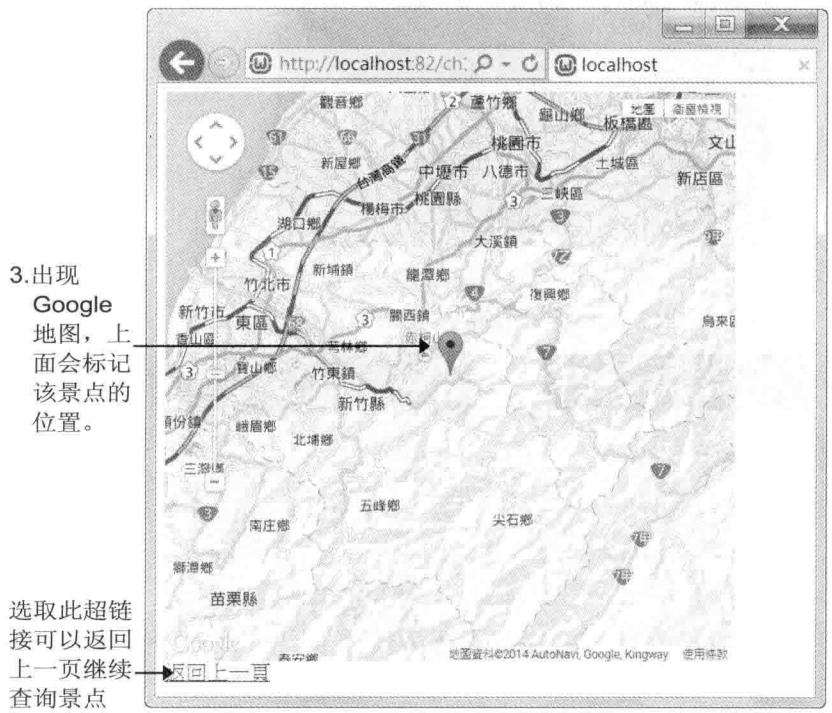


图 14-2

您可以往上下左右的方向拖曳地图，也可以拖曳图 14-2 左侧的调整杆，以放大或缩小地图的显示比例。此外，您还可以单击右上角的[卫星查看]，改以卫星图的方式显示地图，例如图 14-3 所示，至于图 14-4 则是移动版浏览器的执行结果。

图 14-3　点取右上角的“卫星检视”，改以卫星图显示地图

图 14-4　移动版浏览器的执行结果

看到如此犀利的地图功能，您是否以为程序代码会非常复杂？其实不会！而这主要就是因为有 Google Maps API 的帮助，我们赶快来看一下相关的程序代码吧。

这个例子是由下列两个程序所组成：

- <\ch14\farm.php>：这是主程序，用来显示复选列表，用户只要核取景点，然后按[查看地图]，就会调用 <showmap.php>。
- <\ch14\showmap.php>：这个程序负责通过 Google Maps API 显示地图，并在地图上标记景点的位置。

\ch14\farm.php

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:  </head>
06:  <body>
07:    <?php
08:      $farm_address=array('桃园县观音乡大堀村大湖路二段 131-1 号',
09:                          '台中县新社乡中和村中兴路 233 号',
10:                          '新竹县尖石乡嘉乐 130 号',
11:                          '彰化县彰化市卦山路 23 号');
12:    ?>
13:    <form method="post" action="showmap.php">
14:      <input type="radio" name="farm" value="<?php echo $farm_address[0]; ?>" checked>
         青林农场<br>
15:      <input type="radio" name="farm" value="<?php echo $farm_address[1]; ?>">
         安妮公主的花园<br>
16:      <input type="radio" name="farm" value="<?php echo $farm_address[2]; ?>">
         熏衣草森林<br>
17:      <input type="radio" name="farm" value="<?php echo $farm_address[3]; ?>">
         卦山月圆<br>
18:      <input type="submit" value="查看地图">
19:    </form>
20:  </body>
21:</html>
```

- 08 ~ 11: 定义一个名称为 farm_address 的数组来存放各个景点的地址，若景点个数很多，可以改用 MySQL 数据库来存放相关数据。
- 13 ~ 19: 在网页上插入一个表单，里面有一个包含四个项目的复选列表和一个[查看地图]按钮，其中第 13 行是利用 action 属性指定表单处理程序为 showmap.php，当用户单击[查看地图]按钮时，就会执行 showmap.php，而第 14 ~17 行则是插入包含四个项目的复选列表，要注意的是这些项目的 value 属性被指定为各个景点的地址。

\ch14\showmap.php

```
01:<!doctype html>
02: <html>
03:  <head>
04:    <meta charset="utf-8">
05:    <script type="text/javascript" src="http://maps.google.com/maps/api/js?sensor=false">
```

```
06:     </script>
07:     <script type="text/javascript">
08:       var geocoder;
09:       var map;
10:
11:       function initialize()
12:       {
13:         geocoder = new google.maps.Geocoder();
14:         var latlng = new google.maps.LatLng(0, 0);
15:         var myOptions = {zoom:10, center:latlng, mapTypeId:google.maps.MapTypeId.ROADMAP};
16:         map = new google.maps.Map(document.getElementById("map_canvas"), myOptions);
17:         codeAddress();
18:       }
19:
20:       function codeAddress()
21:       {
22:         var address = document.getElementById("address").value;
23:         if（geocoder)
24:         {
25:             geocoder.geocode({'address': address}, function(results, status)
26:           {
27:             if（status == google.maps.GeocoderStatus.OK)
28:             {
29:               map.setCenter(results[0].geometry.location);
30:               var marker = new google.maps.Marker({map:map,position:results[0].geometry.location});
31:             }
32:             else
33:             {
34:               alert("查看地图失败，原因在于：" + status);
35:             }
36:           });
37:         }
38:       }
39:     </script>
40:   </head>
41:   <body onload="initialize()">
42:     <input type="hidden" id="address" value="<?php echo $_POST["farm"]; ?>">
43:     <div id="map_canvas" style="width:500px; height:500px;"></div>
44:     <a href="farm.php">返回上一页</a>
45:   </body>
46:</html>
```

- 05、06: 载入 Google Maps API，这些 API 是以 JavaScript 所编写。当您使用 Google Maps

API 时，必须通过 sensor 参数指明应用程序是否使用了传感器（例如 GPS 定位器）来判断用户的位置，此处因为没有使用传感器，所以传送 sensor=false。

- 11 ~ 18：定义 initialize() 函数，负责地图的初始化工作并调用 codeAddress()函数。
- 13：建立一个隶属于 google.maps.Geocoder 类的对象，然后赋值给变量 geocoder，用来存放地理编码的结果，所谓“地理编码”就是将地址转换为地理坐标，例如台湾高雄市可以转换为纬度 22.638 095 和经度 120.325 699。
- 14：建立一个隶属于 google.maps.LatLng 类的对象，然后赋值给变量 latlng，用来存放纬度和经度，此处是将预设的纬度和经度赋值为（0, 0）。
- 15：定义一个变量 myOptiopn，用来指定地图选项，包括缩放比例为 10、中心点为变量 latlng 所指定的纬度和经度、地图类型为预设的 2D 地图。
- 16：建立一个隶属于 google.maps.Map 类的对象，然后赋值给变量 map，用来存放地图，而且该地图会显示在 HTML 网页中 id 属性为 "map_canvas" 的元素内，此处指的是第 43 行所插入的 元素。
- 17：调用 codeAddress() 函数。
- 20 ~ 38：定义 codeAddress() 函数，以将第 22 行所获取的地址转换成地理坐标，并使用 marker 在地图上标记该位置。
- 41：利用 <body> 元素的 onload 属性指定在网页加载的同时去执行 initialize() 函数。
- 42：在网页上插入一个隐藏字段，该字段的 id 属性为 "address"，value 属性为用户在 <farm.php> 网页中所核取之景点的地址，第 22 行就是利用这个元素获取要在地图上做标记的地址。
- 43：利用 元素插入一个 id 属性为 "map_canvas"、宽度为 500 像素、高度为 500 像素的区块，作为地图的容器。您也可以根据实际情况指定该区域的大小，假设要填充整个显示范围，那么可以将 500px 全部改为 100%。
- 44：插入一个超链接，供用户返回 <farm.php>，以继续查询其他景点的地图。

第 15 章
移动商品目录

15.1 设计移动版网站界面

15.2 完整的程序代码清单

15.1　设计移动版网站界面

在本章中，我们将使用 HTML 5、JavaScript、jQuery、jQuery Mobile、PHP、MySQL 等技术开发一个移动版商品目录网站，这个网站的浏览结果如以下各图所示。

图 15-1

图 15-2

图 15-3

图 15-4

图 15-5

图 15-6

- 首页包含页首、内容和页尾，内容的部分有一张图像和“快乐出版社简介”与“书籍介绍”两个按钮，如图 15-1 所示。
- 当单击“快乐出版社简介”时，会以对话框的形式显示出版社简介，如图 15-2 所示，

若要返回首页，可以单击左上角的关闭按钮。

- 当单击“书籍介绍”时，会以另一个内部页面显示书籍介绍，如图 15-3 所示。若要显示上一本书籍，可以单击导航条的“上一个”按钮；若要显示下一本书籍，可以单击导航条的“下一个”按钮，如图 15-4、图 15-5 和图 15-6 所示；若要返回首页，可以单击“回首页”按钮。

这个网站由下列几个文件所组成。

文件名	说明
ACL037300.jpg ACL039600.jpg AEB002800.jpg AEB003100.jpg AEN003400.jpg	这些 JPEG 图像文件是书籍产品的照片，如图 15-3 ~ 图 15-6 所示
mobile_store.jpg	这个 JPEG 图像文件是首页的标题图像，如图 15-1 所示
index.htm	移动商品目录的首页
get_book_info.php	这个程序负责读取书籍介绍的内容
mobile_store 数据库	这个数据库包含 product 数据表

我们使用了一个名称为 mobile_store 的数据库，里面包含一个 product 数据表，以保存产品数据，其字段结构如下。

字段名	数据类型	长度	主键	说明
book_id	INT	-	☑	流水号字段
image_name	VARCHAR	32	☐	图像文件名称字段
description	TEXT	-	☐	说明字段

您可以自己创建数据库或导入本书为您准备的数据库备份文件（位于下载资源的\samples\database\mobile_store.sql）。

15.2 完整的程序代码清单

❖ index.htm

<index.htm> 的页面结构包含“首页”、“快乐出版社简介”、“书籍介绍”3 个页面，id 分别为 "home"、"intro"、"book"。

\ch15\index.htm

```
01:<!doctype html>
02:<html>
```

```
<head>
    <meta charset="utf-8">
    <title>移动商品目录</title>
    <link rel="stylesheet" href="http://code.jquery.com/mobile/1.4.3/jquery.mobile-1.4.3.min.css" />
    <script src="http://code.jquery.com/jquery-1.11.1.min.js"></script>
    <script src="http://code.jquery.com/mobile/1.4.3/jquery.mobile-1.4.3.min.js"></script>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <script>
        var book_id = 0;

        function get_book_info(mode)
        {
            $.get("get_book_info.php", {book_id: book_id, mode: mode}, function(data){
            book_id = data.book_id;
            $("#bookimg").attr("src", data.image_name);
            $("#bookmsg").html(data.description);
          }, "json");
        }
    </script>
</head>
<body onload="javascript:get_book_info('next')">
    <div data-role="page" id="home">
        <div data-role="header" data-position="fixed">
            <h1>移动商品目录</h1>
        </div>
        <div data-role="content">
            <img src="mobile_store.jpg" width="100%">
            <a href="#intro" data-rel="dialog" data-role="button" data-icon="arrow-r">快乐出版社简介</a>
            <a href="#book" data-role="button" data-icon="arrow-r">书籍介绍</a>
        </div>
        <div data-role="footer" data-position="fixed"><h4>&copy;快乐出版社</h4></div>
    </div>
    <div data-role="page" id="intro">
        <div data-role="header"><h1>快乐出版社简介</h1></div>
        <div data-role="content">
            <p>快乐出版社专门发行各类计算机书籍，包括 Office、程序设计、
            绘图等系列图书，若您有任何问题，请来电与我们联系。</p>
        </div>
    </div>
    <div data-role="page" id="book">
        <div data-role="header"><h1>书籍介绍</h1></div>
        <div data-role="content">
            <img id="bookimg" width="100%">
```

```
46:          <p id="bookmsg" />
47:        </div>
48:        <div data-role="footer" data-position="fixed">
49:          <div data-role="navbar">
50:            <ul>
51:              <li><a href="#home" class="ui-btn-active ui-state-persist">回首页</a></li>
52:              <li><a href="javascript:get_book_info('prev')">上一个</a></li>
53:              <li><a href="javascript:get_book_info('next')">下一个</a></li>
54:            </ul>
55:          </div>
56:        </div>
57:      </div>
58:  </body>
59:</html>
```

- 06：使用 CDN 引用 jQuery Mobile 核心 CSS 文件。
- 07：使用 CDN 引用 jQuery 核心 JavaScript 文件。
- 08：使用 CDN 引用 jQuery Mobile 核心 JavaScript 文件。
- 10～21：这段 JavaScript 代码是本范例的精华所在，第 11 行声明变量 book_id 的初始值为 0，代表书籍介绍当前的书籍流水号为 0；第 13～20 行的 get_book_info() 函数用来获取前一本或后一本书的介绍信息，由 mode 来决定；第 15～19 行为 jQuery Ajax 的写法，其功能是以异步的方式调用服务器端的 get_book_info.php，并使用 POST 的方式传递 book_id 与 mode 参数给 get_book_info.php，当后端程序 get_book_info.php 执行成功时，会以 json 格式返回执行结果，返回的 json 对象会包含 image_name 与 description 属性，第 17～18 行用来将书籍图像与介绍设置为 json 对象的 image_name 与 description 属性，这样就能达成在用户单击图 15-3～图 15-6 中的“上一个”与“下一个”超链接时，变换书籍信息的目的。
- 23：在 body 的 onload 事件中调用 JavaScript 的 get_book_info() 函数，并传入参数 next，由于 book_id 初始值为 0，因此，此行的效果就等于要获取第 0 本书的下一本书信息。
- 24～34：这是“首页”页面，第 30 行指定以对话框的形式显示 intro 页面的快乐出版社简介，第 31 行指定以另一个内部页面显示 book 页面的书籍介绍。
- 35～41：这是“快乐出版社简介”页面。
- 42～57：这是“书籍介绍”页面，第 45～46 行分别放置一个空的 img 标签与 p 标签，其 id 为 bookimg 与 bookmsg，以供 get_book_info() 函数存取，用来放置书籍图像与介绍；第 48～56 行在页尾放入一个导航条，里面有“回首页”、“上一个”、“下一个”三个项目，“上一个”项目指定单击超链接时就调用 get_book_info() 函数，参数值为 prev，表示要获取上一本书的信息，同样地，“下一个”项目指定单击超链接时就调用 get_book_info() 函数，参数值为 next，表示要获取下一本书的信息。

❖ get_book_info.php

这个程序负责读取书籍介绍的内容。

\ch15\get_book_info.php

```
01:<?php
02:   require_once("dbtools.inc.php");
03:
04:   $book_id = $_GET["book_id"];
05:   $mode = $_GET["mode"];
06:   $link = create_connection();
07:
08:   if（$mode == "prev")
09:   {
10:      $sql = "select * from product where book_id < $book_id order by book_id desc limit 1";
11:      $result = execute_sql($link, "mobile_store", $sql);
12:      $total_records = mysqli_num_rows($result);
13:
14:      if（$total_records == 1)
15:      {
16:         $row = mysqli_fetch_assoc($result);
17:
18:         echo json_encode(array("book_id" => $row["book_id"], "image_name" =>
              $row["image_name"], "description" => $row["description"]));
19:      }
20:      else
21:      {
22:         $sql = "select * from product where book_id =（select max(book_id) from product)";
23:         $result = execute_sql($link, "mobile_store", $sql);
24:         $row = mysqli_fetch_assoc($result);
25:
26:         echo json_encode(array("book_id" => $row["book_id"], "image_name" =>
              $row["image_name"], "description" => $row["description"]));
27:      }
28:   }
29:   else if（$mode == "next")
30:   {
31:      $sql = "select * from product where book_id > $book_id order by book_id limit 1";
32:      $result = execute_sql($link, "mobile_store", $sql);
33:      $total_records = mysqli_num_rows($result);
34:
35:      if（$total_records == 1)
36:      {
```

```
37:         $row = mysqli_fetch_assoc($result);
38:
39:         echo json_encode(array("book_id" => $row["book_id"], "image_name" =>
              $row["image_name"], "description" => $row["description"]));
40:     }
41:     else
42:     {
43:         $sql = "select * from product where book_id =（select min(book_id) from product)";
44:         $result = execute_sql($link, "mobile_store", $sql);
45:         $row = mysqli_fetch_assoc($result);
46:
47:         echo json_encode(array("book_id" => $row["book_id"], "image_name" =>
              $row["image_name"], "description" => $row["description"]));
48:     }
49: }
50:
51: mysqli_free_result($result);
52: mysqli_close($link);
53:?>
```

- 08 ~ 28：用来获取上一本书籍的介绍，若已经是位于第一本书籍，则获取最后一本书籍。
- 29 ~ 49：用来获取下一本书籍介绍，若已经是位于最后一本书籍，则获取第一本书籍。
- 18、26、39、47：使用 json_encode() 函数，将获取的书籍信息转为 json 格式，获取的三个字段分别成为 json 对象的三个属性，并使用 echo 直接把 json 输出至浏览器端，这样浏览器端的 get_book_info() 函数执行成功时，即可读取回传的书籍信息。

第 16 章
访客留言板与讨论组

16.1 访客留言板

“访客留言板”(guestbook）是网页上常见的功能，浏览者可以在此张贴留言给站主或其他浏览者。图 16-1 是我们即将要制作的访客留言板，采用分页显示，每页显示 5 行记录，您也可以视实际情况做调整。网页中间会显示页数超链接，只要单击页数超链接，就会显示该页的记录内容，而且最晚输入的留言显示在最前面。

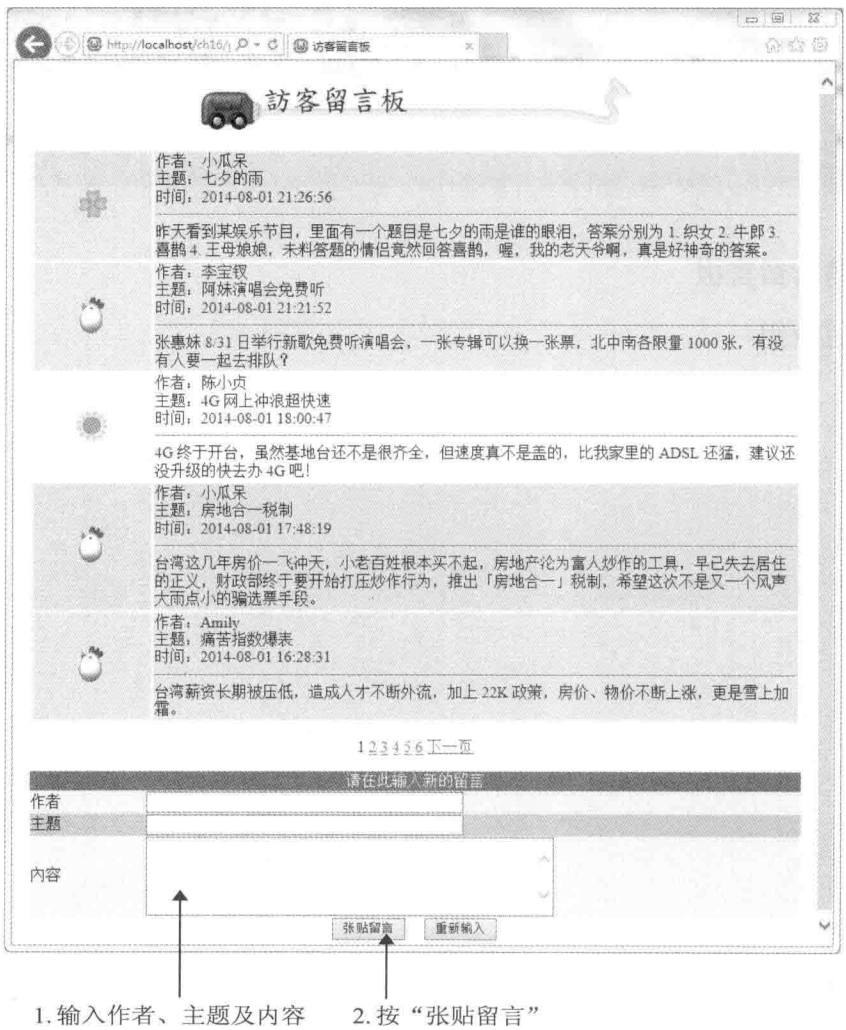


图 16-1

请注意，在您单击页数超链接后，网页上方所显示的该页数将不设置为超链接，表示此为当前正在显示的页数，以和其他页数超链接进行区分。

若要输入新留言，可以在网页下方的表单中输入新留言的作者、主题及内容，然后按“张贴留言”，新留言就会显示在第一页的第一行记录，如图 16-2 所示。若要再输入其他新留言，

只要将滚动条往下移动，在表单中输入即可。

图 16-2

16.1.1　组成网页的文件列表

这个访客留言板存放在下载资源的 \samples\ch16\guestbook\ 文件夹中，它用到下列文件。

文件名	说明
0.gif ~ 9.gif	这 10 个 GIF 动画图像文件用来作为留言板上的插图
fig.jpg	这个 JPEG 图像文件是网页的标题图像
index.php	这是访客留言板的主程序，除了负责从数据库读取留言、以分页方式显示留言，还提供表单让浏览者输入新留言
post.php	这个程序负责读取浏览者在 <index.php> 的表单中所输入的作者、主题及内容，然后写入数据库，最后再重定向到 <index.php>
guestbook 数据库	这个访客留言板使用了名称为 guestbook 的数据库，您必须先将下载资源的 \samples\database\guestbook.sql 导入 MySQL 数据库服务器

在这个访客留言板中，我们使用了名称为 guestbook 的数据库，包含一个 message 数据表，以存储留言数据，其字段结构如下。

字段名	数据类型	长度	主键	说明
id	INT	-	☑	编号字段 自动编号（auto_increment）
author	VARCHAR	10	☐	作者字段
subject	TINYTEXT	-	☐	主题字段
content	TEXT	-	☐	内容字段
date	DATETIME	-	☐	日期字段

您可以自己创建数据库或导入我们为您准备的数据库备份文件（位于下载资源的\samples\database\guestbook.sql），有关如何导入 MySQL 数据库，您可以参考第 11.3.7 小节的说明。

16.1.2 网页的运行流程

主程序 <index.php> 负责从 guestbook 数据库的 message 数据表读取所有记录，然后以分页的方式显示每个记录的作者（author）、主题（subject）、内容（content）及时间（date），预设的分页大小为 5 个记录，您可以自行改变每页显示的记录个数，而且记录的排列方式是由晚到早，也就是越晚输入的留言显示在越前面。

此外，浏览者还可以在网页下方的表单中输入新留言的作者、主题及内容，至于时间则为当时的时间，由 date() 函数自动获取系统当前时间，无须手动输入。待浏览者单击“张贴留言”，便调用 <post.php> 读取表单数据并写入 guestbook 数据库的 message 数据表，完毕后将网页重定向到 <index.php>，此时，新留言就会显示在第一页的第一个记录中。

网页的运行流程如图 16-3 所示。

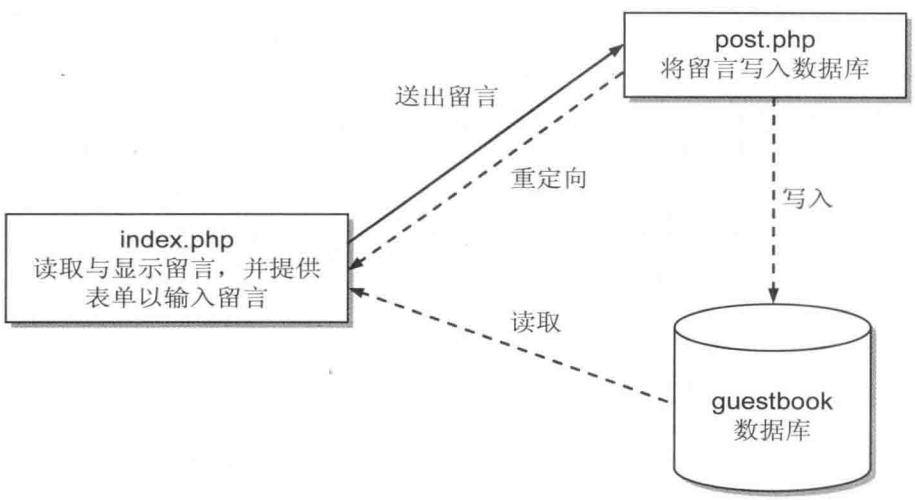


图 16-3

16.1.3 必须具备的背景知识

- 首先，您必须熟悉 HTML 语法或其他网页编辑软件，因为我们将用到许多表格。附录 A、附录 B、附录 C 中分别有 HTML 语法教学、标签与属性速查、特殊字符表，有需要的读者，请自行参考。
- 其次，您必须了解表单的制作方式及如何读取表单数据，我们在第 8 章介绍过。
- 其三，您必须懂得分页浏览数据库的技巧，我们在第 13.7 节介绍过。
- 其四，基本的 JavaScript 语法，我们将使用它来验证数据。

- 最后，您当然得熟悉 SQL 语法及如何访问数据库，第 12 章有 SQL 语法的讲解，有需要的读者，请自行参考。

16.1.4　完整的程序代码清单

index.php

这是访客留言板的主程序，除了负责从数据库读取留言、以分页方式显示留言之外，还提供表单让浏览者输入新留言。

\ch16\guestbook\index.php

```
001:<!doctype html>
002:<html>
003:  <head>
004:    <meta charset="utf-8">
005:    <title>访客留言表</title>
006:    <script type="text/javascript">
007:      function check_data()
008:      {
009:        if（document.myForm.author.value.length == 0)
010:          alert("作者字段不可以空白哦！");
011:        else if（document.myForm.subject.value.length == 0)
012:          alert("主题字段不可以空白哦！");
013:        else if（document.myForm.content.value.length == 0)
014:          alert("内容字段不可以空白哦！");
015:        else
016:          myForm.submit();
017:      }
018:    </script>
019:  </head>
020:  <body>
021:    <p align="center"><img src="fig.jpg"></p>
022:    <?php
023:      require_once("dbtools.inc.php");
024:
025:      //指定每页显示几行记录
026:      $records_per_page = 5;
027:
028:      //获取要显示第几页的记录
029:      if（isset($_GET["page"]))
030:        $page = $_GET["page"];
```

```
031:        else
032:            $page = 1;
033:
034:        //建立数据连接
035:        $link = create_connection();
036:
037:        //执行 SQL 命令
038:        $sql = "SELECT * FROM message ORDER BY date DESC";
039:        $result = execute_sql($link, "guestbook", $sql);
040:
041:        //获取记录数
042:        $total_records = mysqli_num_rows($result);
043:
044:        //计算总页数
045:        $total_pages = ceil($total_records / $records_per_page);
046:
047:        //计算本页第一个记录的序号
048:        $started_record = $records_per_page *（$page - 1);
049:
050:        //将记录指针移至本页第一个记录的序号
051:        mysqli_data_seek($result, $started_record);
052:
053:        //使用 $bg 数组来存储表格背景颜色
054:        $bg[0]= "#D9D9FF";
055:        $bg[1]= "#FFCAEE";
056:        $bg[2]= "#FFFFCC";
057:        $bg[3]= "#B9EEB9";
058:        $bg[4]= "#B9E9FF";
059:
060:        echo "<table width='800' align='center' cellspacing='3'>";
061:
062:        //显示记录
063:        $j = 1;
064:        while（$row = mysqli_fetch_assoc($result) and $j <= $records_per_page)
065:        {
066:            echo "<tr bgcolor='" . $bg[$j - 1]. "'>";
067:            echo "<td width='120' align='center'>
068:                    <img src='" . mt_rand(0, 9) . ".gif'></td>";
069:            echo "<td>作者：" . $row["author"]. "<br>";
070:            echo "主题：" . $row["subject"]. "<br>";
071:            echo "时间：" . $row["date"]. "<hr>";
072:            echo $row["content"]. "</td></tr>";
073:            $j++;
```

```
074:        }
075:        echo "</table>" ;
076:
077:        //产生导航条
078:        echo "<p align='center'>";
079:
080:        if（$page > 1)
081:           echo "<a href='index.php?page=".（$page - 1) . "'>上一页</a> ";
082:
083:        for（$i = 1; $i <= $total_pages; $i++)
084:        {
085:           if（$i == $page)
086:              echo "$i ";
087:           else
088:              echo "<a href='index.php?page=$i'>$i</a> ";
089:        }
090:
091:        if（$page < $total_pages)
092:           echo "<a href='index.php?page=".（$page + 1) . "'>下一页</a> ";
093:        echo "</p>";
094:
095:        //释放内存空间
096:        mysqli_free_result($result);
097:        mysqli_close($link);
098:     ?>
099:     <form name="myForm" method="post" action="post.php">
100:        <table border="0" width="800" align="center" cellspacing="0">
101:           <tr bgcolor="#0084CA" align="center">
102:              <td colspan="2">
103:                 <font color="#FFFFFF">请在此输入新的留言</font></td>
104:           </tr>
105:           <tr bgcolor="#D9F2FF">
106:              <td width="15%">作者</td>
107:              <td width="85%"><input name="author" type="text" size="50"></td>
108:           </tr>
109:           <tr bgcolor="#84D7FF">
110:              <td width="15%">主题</td>
111:              <td width="85%"><input name="subject" type="text" size="50"></td>
112:           </tr>
113:           <tr bgcolor="#D9F2FF">
114:              <td width="15%">内容</td>
115:              <td width="85%"><textarea name="content" cols="50" rows="5"></textarea></td>
116:           </tr>
```

```
117:          <tr>
118:            <td colspan="2" align="center">
119:              <input type="button" value="张贴留言" onClick="check_data()">
120:              <input type="reset" value="重新输入">
121:            </td>
122:          </tr>
123:        </table>
124:    </form>
125:  </body>
126:</html>
```

这个网页的源代码长达 4 页，看似复杂，其实很简单，所有概念在前面的章节中都已经讲解过，不同之处在于程序第 007 ~ 017 行的 check_data() 函数。

- 007 ~ 017：客户端 JavaScript，用来判断浏览者是否输入了留言。
- 009 ~ 010：判断作者（author）字段是否输入了数据。
- 011 ~ 012：判断主题（subject）字段是否输入了数据。
- 013 ~ 014：判断内容（content）字段是否输入了数据。
- 016：当浏览者输入了各个字段数据时，就会执行这行语句，将数据传送回服务器。
- 026：指定每页显示几笔记录，此处是 5，若您希望每页显示较多记录，可以设置变量 $records_per_page 的值。
- 029 ~ 032：设置要显示第几页的数据，首先会获取网址参数 page，我们使用 isset() 函数判断变量 $_GET["page"]是否获取了数值，若获取了的话，表示有浏览者指定要查看第 page 页的数据，就将变量 page 设置为获取的值；相反地，若没有获取到值的话，表示没有浏览者指定要显示第几页数据，就将变量 page 设置为 1，让网页显示第一页的数据。
- 035：建立数据连接。
- 038 ~ 039：对 guestbook 数据库执行 " SELECT * FROM message ORDER BY date DESC" 指令。
- 042：获取查询结果所包含的记录数。
- 045：计算总页数，此处使用了 ceil() 函数，若总页数出现小数点，就无条件进位。
- 048：计算当前要显示的这页数据的第一个记录位于查询结果的第几个记录。
- 051：使用 mysqli_data_seek() 函数将记录指针移至起始记录。
- 054 ~ 058：使用数组 $bg 存储表格每一列的背景颜色，好让每个记录的背景颜色均不相同。由于一页只显示 5 个记录，所以只需要 $bg[0]、$bg[1]、…$bg[4]，分别代表第 1 ~ 5 列的背景颜色。您可以自行变更为喜欢的颜色。若每页显示 10 个记录，则需要使用 $bg[0]、$bg[1]、…$bg[9]，依次类推。
- 063 ~ 074：这个部分用来显示记录的内容，我们在之前的例子中已经使用过很多次，相信您已经相当熟悉了。但有一点不同，我们并不是要显示所有记录，而是要显示某一区间范围的记录，请看第 064 行，这行的意思是当读取到记录且 $j <= $records_per_page 时，才会执行 while 循环内的程序代码来显示记录，其中 $j <=

$records_per_page 用来控制每页显示的记录数，此处为 $records_per_page = 5。

- 080 ~ 92：这个部分用来制作导航条，让浏览者能快速换页。第 080、081 行的意思是当目前页数大于第一页时，就插入“上一页”超链接，让浏览者浏览上一页；第 091、092 行的意思是当目前页数小于最后一页时，就插入“下一页”超链接，让浏览者浏览下一页；第 083 ~ 089 行用来产生所有页码，当前页数的页码为纯文本，不需要有超链接的功能，而非当前页数的页码，则具有超链接的功能，让浏览者跳至对应的页数。
- 96：释放查询结果所占用的内存。
- 97：关闭数据连接。
- 119：当浏览者单击[张贴留言]时，并不会马上送出数据，而是先执行 check_data() 函数检查浏览者是否输入了数据。

post.php

这个程序负责读取浏览者在 <index.php> 的表单中所输入的作者、主题及内容，然后写入数据库，最后再重定向到 <index.php>。

\ch16\guestbook\post.php

```
<?php
  require_once("dbtools.inc.php");

  $author = $_POST["author"];
  $subject = $_POST["subject"];
  $content = $_POST["content"];
  $current_time = date("Y-m-d H:i:s");

  //建立数据连接
  $link = create_connection();

  //执行 SQL 命令
  $sql = "INSERT INTO message(author, subject, content, date)
          VALUES('$author', '$subject', '$content', '$current_time')";
  $result = execute_sql($link, "guestbook", $sql);

  //关闭数据连接
  mysqli_close($link);

  //将网页重定向到 index.php
  header("location:index.php");
  exit();
?>
```

- 新留言表单中用来输入作者、主题及内容的字段名分别为 author、subject 和 content，至于时间则取决于 date() 函数。
- 在 <post.php> 的最后，我们将网页重定向到 <index.php>，重载访客留言板，此时，新留言会显示在第一页的第一个记录中。

16.2 讨论组

“讨论组”(newsgroup）和访客留言板相似，不同的是讨论组允许浏览者针对特定主题另辟版面做讨论。图 16-4 所示是我们即将要制作的讨论组，这个程序改写自访客留言板，故用户界面相似，一样是采用分页显示，每页默认的记录个数为 5，网页中间会显示页数超链接，只要单击页数超链接，就会显示该页的记录内容，而且越晚输入的讨论内容会显示在越前面。

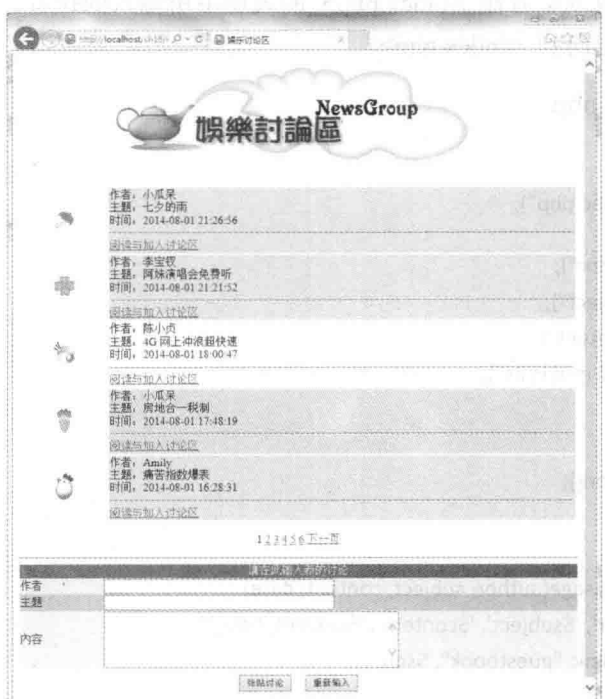


图 16-4

请注意，在您单击页数超链接后，网页上方所显示的该页数将不设置为超链接，表示此为当前正在显示的页数，以和其他页数超链接做区分。

若要输入新讨论，可以在网页下方的表单中输入新讨论的作者、主题及内容，然后单击[张贴讨论]，新讨论就会显示在第一页的第一个记录中；若要阅读留言或加入讨论，可以单击超链接文字“阅读与加入讨论”，屏幕上会出现如图 16-5 所示的界面，上面列出了原来的留言

内容和后来的讨论内容。

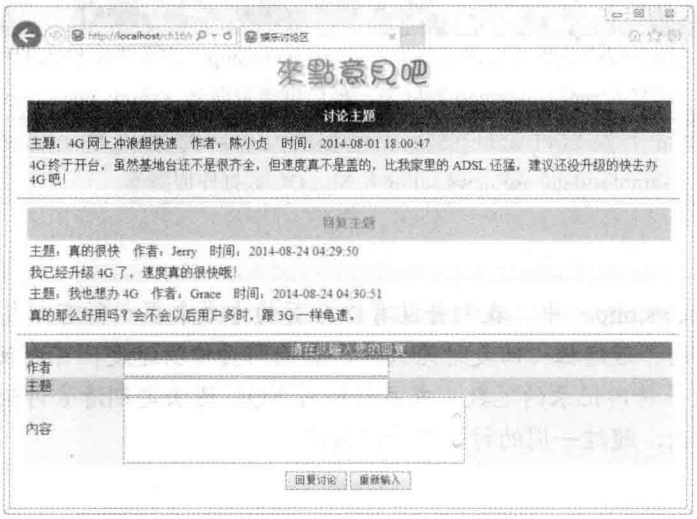


图 16-5

若要反映自己的意见，可以在网页下方的表单中输入作者、主题及内容，然后单击[回复讨论]，回复的数据会被加入 news 数据库的 reply_message 数据表，此时，网页会再度导向至 <show_news.php>，您就可以看到刚才回复的讨论内容。

16.2.1　组成网页的文件列表

这个讨论组存放在下载资源的 \samples\ch16\news\ 文件夹中它用到下列文件。

文件名	说明
0.gif ~ 9.gif	这 10 个 GIF 动画图像文件用来作为讨论区群组上的插图
fig.jpg	这个 JPEG 图像文件是 <index.php> 网页的标题图像
fig1.jpg	这个 JPEG 图像文件是 <show_news.php> 网页的标题图像
index.php	这是讨论组的主程序，除了负责从数据库读取讨论、以分页方式显示讨论之外，还提供表单让浏览者输入新讨论，完毕后单击[张贴讨论]，就调用表单处理程序 <post.php>
post.php	这个程序负责读取浏览者在 <index.php> 的表单中所输入的作者、主题及内容，然后写入 message 数据表，最后再重定向到 <index.php>
show_news.php	在浏览者单击超链接文字“阅读与加入讨论”后，就调用这个程序去 message 数据表读取原来讨论的作者（author）、主题（subject）、内容（content）及时间（date），然后显示出来；接着再根据原来留言的编号（id）去 reply_message 数据表读取看看有没有任何回复讨论。若有的话，就读取其作者（author）、主题（subject）、内容（content）及时间（date），然后显示出来；若要回复讨论，可以在网页下方的表单中做输入，完毕后单击[回复讨论]，就调用表单处理程序 <post_reply.php>

（续表）

文件名	说明
post_reply.php	这个程序负责读取浏览者在 <show_news.php> 表单中所输入的作者、主题及内容，然后写入 reply_message 数据表，最后再重定向到 <show_news.php>
news 数据库	这个娱乐讨论区使用了名称为 news 的数据库，您必须先将下载资源\samples\database\news.sql 导入 MySQL 数据库服务器

在 <show_news.php> 中，我们并没有以分页的方式来显示所有回复讨论，而是一次显示所有回复讨论，若您担心回复讨论太多，影响网页传输速度，可以使用 SQL 语法的 LIMIT 来限制最多传回记录的笔数，或另外编写 SQL 语法定期清除讨论，例如您可以只保留一周内的讨论，超过一周的讨论便予以清除。

在这个娱乐讨论区中，我们使用了名称为 news 的数据库，包含 message 和 reply_message 两个数据表，message 数据表用来存储讨论的内容，其字段结构如下。

字段名	数据类型	长度	主键	说明
id	INT	-	☑	编号字段 自动编号（auto_increment）
author	VARCHAR	10	☐	作者字段
subject	TINYTEXT	-	☐	主题字段
content	MEDIUMTEXT	-	☐	内容字段
date	DATETIME	-	☐	日期字段

reply_message 数据表用来存储回复讨论的内容，其字段结构如下：

字段名	数据类型	长度	主键	说明
id	INT	-	☑	编号字段 自动编号（auto_increment）
author	VARCHAR	10	☐	作者字段
subject	TINYTEXT	-	☐	主题字段
content	MEDIUMTEXT	-	☐	内容字段
date	DATETIME	-	☐	日期字段
reply_id	INT	-	☐	回复编号字段，用来存储原讨论主题的编号，以区分回复内容属于哪个讨论

您可以自己创建数据库或导入本书为您准备的数据库备份文件（位于下载资源的\samples\database\news.sql），有关如何导入 MySQL 数据库，可以参考第 11.3.7 小节的说明。

16.2.2　网页的运行流程

主程序 <index.php> 负责从 news 数据库的 message 数据表读取所有记录，然后以分页的方式显示每笔记录的作者（author）、主题（subject）及时间（date），但不包含内容（content），预设的分页大小为 5 个记录，而且记录的排列方式是由晚到早，也就是越晚的讨论显示在越前面。

若要输入新讨论，可以在网页下方的表单中输入，待单击[张贴讨论]后，就调用表单处理程序 <post.php> 读取表单数据并写入 message 数据表，然后将网页重定向到 <index.php>，此时，新讨论会显示在第一页的第一个记录中。

若要阅读讨论内容或加入讨论，可以单击该留言的超链接文字“阅读与加入讨论”，此时会执行 <show_news.php>，这个程序除了显示原来讨论的内容，也会根据原来讨论的编号（id）到 reply_message 数据表搜索是否有相关的讨论，有的话就显示在网页上；若要加入讨论，可以在网页下方的表单中输入，待单击[回复讨论]后，就调用表单处理程序 <post_reply.php>，读取表单数据并写入 reply_message 数据表，然后将网页重定向到 <show_news.php>，您可以马上看到刚才回复的讨论内容。网页的运行流程如图 16-6 所示。

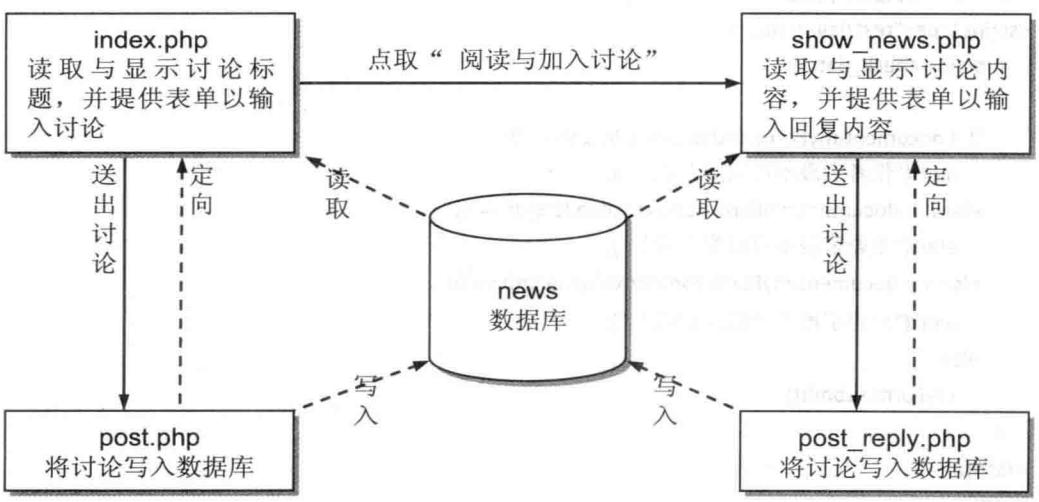


图 16-6

16.2.3　必须具备的背景知识

- 首先，您必须熟悉 HTML 语法或其他网页编辑软件。
- 其次，您必须了解表单的制作方式及如何读取表单数据，包括隐藏字段。
- 其三，您必须懂得分页浏览数据库的技巧。
- 其四，基本的 JavaScript 语法，我们将使用它来验证数据。
- 最后，您当然得熟悉 SQL 语法及如何访问数据库。

16.2.4 完整的程序代码清单

❖ index.php

这个程序大致上和访客留言板的 <index.php> 差不多，唯一做修改的是用来显示讨论的作者、主题、时间的地方，即程序代码反白的部分，只要将超链接设置得当，同时正确地将讨论的编号（id）当成参数一起传送给 <show_news.php> 即可。<show_news.php> 会根据讨论的编号（id）从 message 数据表读取讨论的内容，然后从 reply_message 数据表读取该主题的回复讨论并显示出来（若有的话）。

\ch16\news\index.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>娱乐讨论区</title>
    <script type="text/javascript">
      function check_data()
              {
        if（document.myForm.author.value.length == 0)
          alert("作者字段不可以空白哦！ ");
        else if（document.myForm.subject.value.length == 0)
          alert("主题字段不可以空白哦！ ");
        else if（document.myForm.content.value.length == 0)
          alert("内容字段不可以空白哦！ ");
        else
          myForm.submit();
      }
    </script>
  </head>
  <body>
    <p align="center"><img src="fig.jpg"></p>
    <?php
      require_once("dbtools.inc.php");

      //指定每页显示几个记录
      $records_per_page = 5;

      //获取要显示第几页的记录
      if（isset($_GET["page"]))
        $page = $_GET["page"];
      else
```

```
    $page = 1;

//建立数据连接
$link = create_connection();

//执行 SQL 命令
$sql = "SELECT id, author, subject, date FROM message ORDER BY date DESC";
$result = execute_sql($link, "news", $sql);

//获取记录数
$total_records = mysqli_num_rows($result);

//计算总页数
$total_pages = ceil($total_records / $records_per_page);

//计算本页第一个记录的序号
$started_record = $records_per_page *（$page - 1);

//将记录指针移至本页第一个记录的序号
mysqli_data_seek($result, $started_record);

echo "<table width='800' align='center' cellspacing='3'>";
//使用 $bg 数组来存储表格背景颜色
$bg[0]= "#D9D9FF";
$bg[1]= "#FFCAEE";
$bg[2]= "#FFFFCC";
$bg[3]= "#B9EEB9";
$bg[4]= "#B9E9FF";

//显示记录
$j = 1;
while（$row = mysqli_fetch_assoc($result) and $j <= $records_per_page)
{
   echo "<tr>";
   echo "<td width='120' align='center'><img src='" . mt_rand(0, 9) . ".gif'></td>";
   echo "<td bgcolor='" . $bg[$j - 1]. "'>作者： " . $row["author"]. "<br>";
   echo "主题： " . $row["subject"]. "<br>";
   echo "时间： " . $row["date"]. "<hr>";
   echo "<a href='show_news.php?id=";
   echo $row["id"]. "'>阅读与加入讨论</a></td></tr>";
   $j++;
}
echo "</table>" ;

//产生导航条
echo "<p align='center'>";
```

```
    if（$page > 1)
        echo "<a href='index.php?page=".（$page - 1) . "'>上一页</a> ";

    for（$i = 1; $i <= $total_pages; $i++)
    {
        if（$i == $page)
            echo "$i ";
        else
            echo "<a href='index.php?page=$i'>$i</a> ";
    }

    if（$page < $total_pages)
        echo "<a href='index.php?page=".（$page + 1) . "'>下一页</a> ";
    echo "</p>";

    //释放内存空间
    mysqli_free_result($result);
    mysqli_close($link);
?>

<hr>
<!- 显示输入新留言表单 -->
<form name="myForm" method="post" action="post.php">
    <table border="0" width="800" align="center" cellspacing="0">
        <tr bgcolor="#0084CA" align="center">
            <td colspan="2"><font color="white">请在此输入新的讨论</font></td>
        </tr>
        <tr bgcolor="#D9F2FF">
            <td width="15%">作者</td>
            <td width="85%"><input name="author" type="text" size="50"></td>
        </tr>
        <tr bgcolor="#84D7FF">
            <td width="15%">主题</td>
            <td width="85%"><input name="subject" type="text" size="50"></td>
        </tr>
        <tr bgcolor="#D9F2FF">
            <td width="15%">内容</td>
            <td width="85%"><textarea name="content" cols="50" rows="5"></textarea></td>
        </tr>
        <tr>
            <td colspan="2" height="40" align="center">
                <input type="button" value="张贴讨论" onClick="check_data()">
                <input type="reset" value="重新输入">
            </td>
        </tr>
```

```
    </table>
  </form>
 </body>
</html>
```

❖ post.php

这个程序负责读取浏览者在 <index.php> 的表单中所输入的新讨论，然后写入 news 数据库的 message 数据表，再将网页重定向到 <index.php>，此时，新讨论会显示在第一页的第一个记录。

\ch16\news\post.php

```
<?php
  require_once("dbtools.inc.php");

  $author = $_POST["author"];
  $subject = $_POST["subject"];
  $content = $_POST["content"];
  $current_time = date("Y-m-d H:i:s");

  //建立数据连接
  $link = create_connection();

  //执行 SQL 命令
  $sql = "INSERT INTO message(author, subject, content, date)
          VALUES（'$author', '$subject', '$content', '$current_time')";
  $result = execute_sql($link, "news", $sql);

  //关闭数据连接
  mysqli_close($link);

  //将网页重定向
  header("location:index.php");
  exit();
?>
```

❖ show_news.php

这个程序在浏览者单击超链接文字“阅读与加入讨论”后所调用，它会去 message 数据表读取原来讨论的作者（author）、主题（subject）、内容（content）及时间（date），然后显示出来；接着再根据原来讨论的编号（id）去 reply_message 数据表搜索看看有没有任何回

复的讨论。若有的话，就读取其作者（author）、主题（subject）、内容（content）及时间（date），然后显示出来；若要回复讨论，可以在网页下方的表单中输入，完毕后单击[回复讨论]，就调用表单处理程序 <post_reply.php>。

\ch16\news\show_news.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>娱乐讨论区</title>
    <script type="text/javascript">
      function check_data()
      {
        if（document.myForm.author.value.length == 0)
          alert("作者字段不可以空白哦！ ");
        else if（document.myForm.subject.value.length == 0)
          alert("主题字段不可以空白哦！ ");
        else if（document.myForm.content.value.length == 0)
          alert("内容字段不可以空白哦！ ");
        else
          myForm.submit();
      }
    </script>
  </head>
  <body>
    <center><img src="fig1.jpg"></center>
    <?php
      require_once("dbtools.inc.php");
      //获取要显示的记录
      $id = $_GET["id"];

      //建立数据连接
      $link = create_connection();

      //执行 SQL 命令
      $sql = "SELECT * FROM message WHERE id = $id";
      $result = execute_sql($link, "news", $sql);

      echo "<table width='800' align='center' cellpadding='3'>";
      echo "<tr height='40'><td colspan='2' align='center'
            bgcolor='#663333'><font color='white'>
            <b>讨论主题</b></font></td></tr>";
```

```
//显示原讨论主题的内容
while（$row = mysqli_fetch_assoc($result))
{
   echo "<tr>";
   echo "<td bgcolor='#CCFFCC'>主题：" . $row["subject"]. "    ";
   echo "作者：" . $row["author"]. "    ";
   echo "时间：" . $row["date"]. "</td></tr>";
   echo "<tr height='40'><td bgcolor='CCFFFF'>";
   echo $row["content"]. "</td></tr>";
}

echo "</table>";

//释放 $result 占用的内存空间
mysqli_free_result($result);

//执行 SQL 命令
$sql = "SELECT * FROM reply_message WHERE reply_id = $id";
$result = execute_sql($link, "news", $sql);
if（mysqli_num_rows($result) <> 0)
{
   echo "<hr>";
   echo "<table width='800' align='center' cellpadding='3'>";
   echo "<tr height='40'><td colspan='2' align='center'
            bgcolor='#99CCFF'><font color='#FF3366'>
            <b>回复主题</b></font></td></tr>";

   //显示回复主题的内容
   while（$row = mysqli_fetch_assoc($result))
   {
      echo "<tr>";
      echo "<td bgcolor='#FFFF99'>主题：" . $row["subject"]. "    ";
      echo "作者：" . $row["author"]. "    ";
      echo "时间：" . $row["date"]. "</td></tr>";
      echo "<tr><td bgcolor='CCFFFF'>";
      echo $row["content"]. "</td></tr>";
   }

   echo "</table>";
}
//释放内存空间
mysqli_free_result($result);
```

```
    mysqli_close($link);
  ?>
  <hr>
  <form name="myForm" method="post" action="post_reply.php">
    <input type="hidden" name="reply_id" value="<?php echo $id ?>">
    <table border="0" width="800" align="center" cellspacing="0">
      <tr bgcolor="#0084CA" align="center">
        <td colspan="2"><font color="white">请在此输入您的回复</font></td>
      </tr>
      <tr bgcolor="#D9F2FF">
        <td width="15%">作者</td>
        <td width="85%"><input name="author" type="text" size="50"></td>
      </tr>
      <tr bgcolor="#84D7FF">
        <td width="15%">主题</td>
        <td width="85%"><input name="subject" type="text" size="50"></td>
      </tr>
      <tr bgcolor="#D9F2FF">
        <td width="15%">内容</td>
        <td width="85%"><textarea name="content" cols="50" rows="5"></textarea></td>
      </tr>
      <tr>
        <td colspan="2" height="40" align="center">
          <input type="button" value="回复讨论" onClick="check_data()">
          <input type="reset" value="重新输入">
        </td>
      </tr>
    </table>
  </form>
 </body>
</html>
```

<show_news.php> 的回复讨论表单比 <index.php> 的新讨论表单多了一个名称为 reply_id 的隐藏字段，这主要用来记录原来讨论的编号（id)。

❖ post_reply.php

这个程序为回复讨论表单的处理程序，负责读取浏览者回复讨论时所输入的作者、主题及内容，然后写入 news 数据库的 reply_message 数据表，再将网页重定向到 <show_news.php>。

\ch16\news\post_reply.php

```
<?php
    require_once("dbtools.inc.php");

    $author = $_POST["author"];
    $subject = $_POST["subject"];
    $content = $_POST["content"];
    $current_time = date("Y-m-d H:i:s");
    $reply_id = $_POST["reply_id"];

    //建立数据连接
    $link = create_connection();

    //执行 SQL 命令
    $sql = "INSERT INTO reply_message(author, subject, content, date, reply_id)
            VALUES（'$author', '$subject', '$content', '$current_time', '$reply_id')";
    $result = execute_sql($link, "news", $sql);

    //关闭数据连接
    mysqli_close($link);

    //将网页重定向
    header("location:show_news.php?id=" . $reply_id);
    exit();
?>
```

第 17 章
文件上传

17.1　认识文件上传

“文件上传”是一个相当实用的功能，它允许浏览者将文件上传至 Web 服务器，通过 PHP 的内部函数，我们无须依赖任何软件，就可以轻松制作出具有文件上传功能的网页。

文件上传用户界面通常类似于如图 17-1 所示的界面，中间有文本框和[浏览]按钮，目的是让浏览者选择要上传的文件，由文件字段所组成。

当浏览者单击[上传]时，文件会被上传至 Web 服务器的暂存空间，我们必须自行编写后端的处理程序将文件存储在 Web 服务器的某特定文件夹；相反，当浏览者单击[重新设置]时，会清除文件上传字段的内容。

图 17-1

17.1.1　前置准备工作

PHP 支持文件上传功能，而且它的原始设置就已经打开这个功能，但在我们开始编写文件上传的程序之前，还是应该先确认 PHP 是否打开这个功能，现在，请您使用 Notepad++ 或其他文本编辑器打开 C:\wamp\bin\php\php5.5.12\php.ini 配置设置文件，然后设置下列参数。

参数	说明
file_uploads	设置是否允许通过 HTTP 通信协议进行文件上传，默认值为 On，表示允许
upload_tmp_dir	配置文件上传时所要使用的暂存目录，若省略不写，表示使用系统默认的暂存目录，Windows 7、Windows Vista、Windows XP/2003/2008 默认的暂存目录为 C:\Windows\Temp，您可以根据实际情况进行设置

（续表）

参数	说明
upload_max_filesize	设置允许上传多大的文件，若没设置，默认值为 2MB，表示上传的文件不能超过 2MB，如要加大或减小，可以变更这个参数的值，建议将此参数设为 64MB，以确保后续范例不会因上传文件过大而失败
post_max_size	设置当网页使用 POST 方式传送数据回 Web 服务器时，所允许传送的最大容量，若没设置，默认值为 8MB，表示每次返回 Web 服务器的数据最大不可超过 8MB 这个参数不能设得太小，切记不得小于 upload_max_filesize 参数的值，否则会使上传的文件大小比 POST 方式允许传送的最大容量还大，导致上传失败，建议将此参数设为 64MB，以确保后续范例不会因上传文件过大而失败
max_input_time	设置当网页使用 POST 方式传送数据回 Web 服务器时，所允许传送的最长时间，默认值为 60 秒，表示每次返回 Web 服务器的时间不得大于 60 秒。这个参数不能设得太小，一旦上传文件所花费的时间超过设置的时间，将会上传失败

请注意，php.ini 配置设置文件里的参数名称前面若是以分号（;）开头，表示此参数被注释起来，若要启用它，只要将“;”删除即可，设置完毕后，记住要重新启动 Apache Web Server，以使新配置设置生效。

17.1.2 编写前端的文件上传用户界面

文件上传用户界面主要包含一个或多个文件字段，文件字段和其他表单字段一样要放在表单里面，而且表单的编码方式必须设置为 "multipart/form-data"，如下所示：

```
<form method="post" action="..." enctype="multipart/form-data">
  文件字段放在此处
</form>
```

编写文件上传用户界面的重点在于如何插入文件字段，也就是使用如下的 HTML 标签：

```
<input type="file" name="...">
```

例如下面的语句是插入一个名称为 myfile 的文件字段：

```
<input type="file" name="myfile">
```

若要插入多个文件字段，可以写成如下的形式，其中 myfile[]是文件字段的名称：

```
< input type="file" name="myfile[]" size="50"><br>
< input type="file" name="myfile[]" size="50"><br>
```

```
< input type="file" name="myfile[]" size="50"><br>
< input type="file" name="myfile[]" size="50"><br>
```

您会发现，无论要插入多少文件字段，其实就是编写多少 <input type="file" name="..."> 语句而已，唯一不同的是 name 属性的设置，当我们插入多个文件字段时，每个字段的 name 属性都要相同，而且值必须为数组，例如 myfile[]。

另外，PHP 还有一个特殊功能，它可以在表单里面插入一个名称为 MAX_FILE_SIZE 的隐藏字段，用来设置允许上传的文件大小，单位为字节（bytes），其语法如下：

```
<input type="hidden" name="MAX_FILE_SIZE" value="1048576">
```

前述语法表示设置上传的文件大小不可以超过 1MB（1024 * 1024 = 1048576），MAX_FILE_SIZE 隐藏字段的优点是允许您可以在不同网页上设置不同的文件大小限制。

请注意，MAX_FILE_SIZE 隐藏字段设置的文件大小限制只对包含此行设置的网页有效，并不会改变 php.ini 配置设置文件内 upload_max_filesize 参数的值，换句话说，若网页上已设置 MAX_FILE_SIZE 隐藏字段，那么无论上传的文件大于 MAX_FILE_SIZE 隐藏字段所设置的大小，或大于 php.ini 配置设置文件内 upload_max_filesize 参数所设置的大小，均会上传失败；若网页上没有设置 MAX_FILE_SIZE 隐藏字段，那么只有在上传的文件大于 php.ini 配置设置文件内 upload_max_filesize 参数所设置的大小，才会上传失败。

MAX_FILE_SIZE 隐藏字段必须放在文件字段的前面，否则会设置失败，如下所示：

```
<form method="post" action="upload_01.php" enctype="multipart/form-data">
   <input type="hidden" name="MAX_FILE_SIZE" value="1048576">
   < input type="file" name="myfile" size="50"><br><br>
   < input type="submit" value="上传">
   < input type="reset" value="重新设置">
</form>
```

17.1.3　编写后端的处理程序

在浏览者选择好要上传的文件并按[上传]后，文件就会被上传至 Web 服务器的暂存文件夹，暂存文件的名称为 phpxxx.tmp，xxx 为流水号。

一、获取文件信息

后端的处理程序可以通过下列变量获取上传文件的信息。

变量	说明
$_FILES["*字段名*"]["name"]	获取上传文件的名称，若浏览者没有指定上传的文件，返回值为空白（empty）
$_FILES["*字段名*"]["type"]	获取上传文件的 MIME 类型，我们可以根据返回值决定是否让浏览者上传该类型的文件

（续表）

变量	说明
$_FILES["*字段名*"]["size"]	获取上传文件的大小，单位为字节（bytes），若浏览者没有指定上传的文件，返回值为 0
$_FILES["*字段名*"]["tmp_name"]	获取上传文件暂存的路径及文件名
$_FILES["*字段名*"]["error"]	获取上传时所发生的错误代码

使用 $_FILES["*字段名*"]["error"]获取错误代码时，其返回值有下列 5 种。

错误代码	说明
0	文件上传成功
1	文件大小超过 php.ini 配置设置文件内 upload_max_filesize 参数所设置的大小
2	文件大小超过网页上 MAX_FILE_SIZE 隐藏字段所设置的大小
3	文件上传不完整
4	找不到要上传的文件

当网页上允许浏览者一次上传多个文件时，获取文件信息的技巧略有不同，举例来说，假设我们要获取上传的第一个文件的名称，那么要写成 $_FILES["*字段名*"]["name"][0]，后面多一个[0]，表示第一个文件；同理，假设要获取上传的第 3 个文件的大小，那么要写成 $_FILES["*字段名*"]["size"][2]，以此类推。

二、移动文件

接下来，我们可以使用 move_uploaded_file() 函数将临时文件移至特定的文件夹，其语法如下：

```
move_uploaded_file(string filename, string destination)
```

- *filename*：临时文件的路径及文件名，这个参数可以通过 $_FILES["*字段名*"]["tmp_name"]来获取。
- *destination*：文件的目地路径及文件名。

move_uploaded_file() 函数会先检查参数 *filename* 指定的暂存文件是否为 HTTP 通信协议上传的文件，若不是的话，就直接返回 FALSE，表示文件移动失败，否则将文件移至参数 *destination* 指定的位置，并根据参数 *destination* 的设置来修改文件名。

不过，若因为其他因素造成 move_uploaded_file() 函数无法顺利移动临时文件，例如文件不存在或权限问题，也会直接返回 FALSE，表示文件移动失败。

此外，当参数 *destination* 指定的目标文件已经存在时，会覆盖原来的文件，而且当程序执行完毕时，无论有没有搬移暂存盘，暂存档都会自动被删除。

由于 move_uploaded_file() 在中文编码的操作系统里无法正确处理中文文件名，例如 Windows 操作系统，所以在上传文件时，文件名请勿出现非 ASCII 的字符，例如汉字，只要文件名包含汉字，上传就会失败。若您上传的文件包含中文名，请使用 iconv() 函数将文件名由 UTF-8 编码转换为中文编码，如此即可避免大部分的问题。

17.2　上传单一文件

我们先来示范如何上传单一文件，之后再来示范如何上传多个文件，图 17-2 所示是上传单一文件的流程。

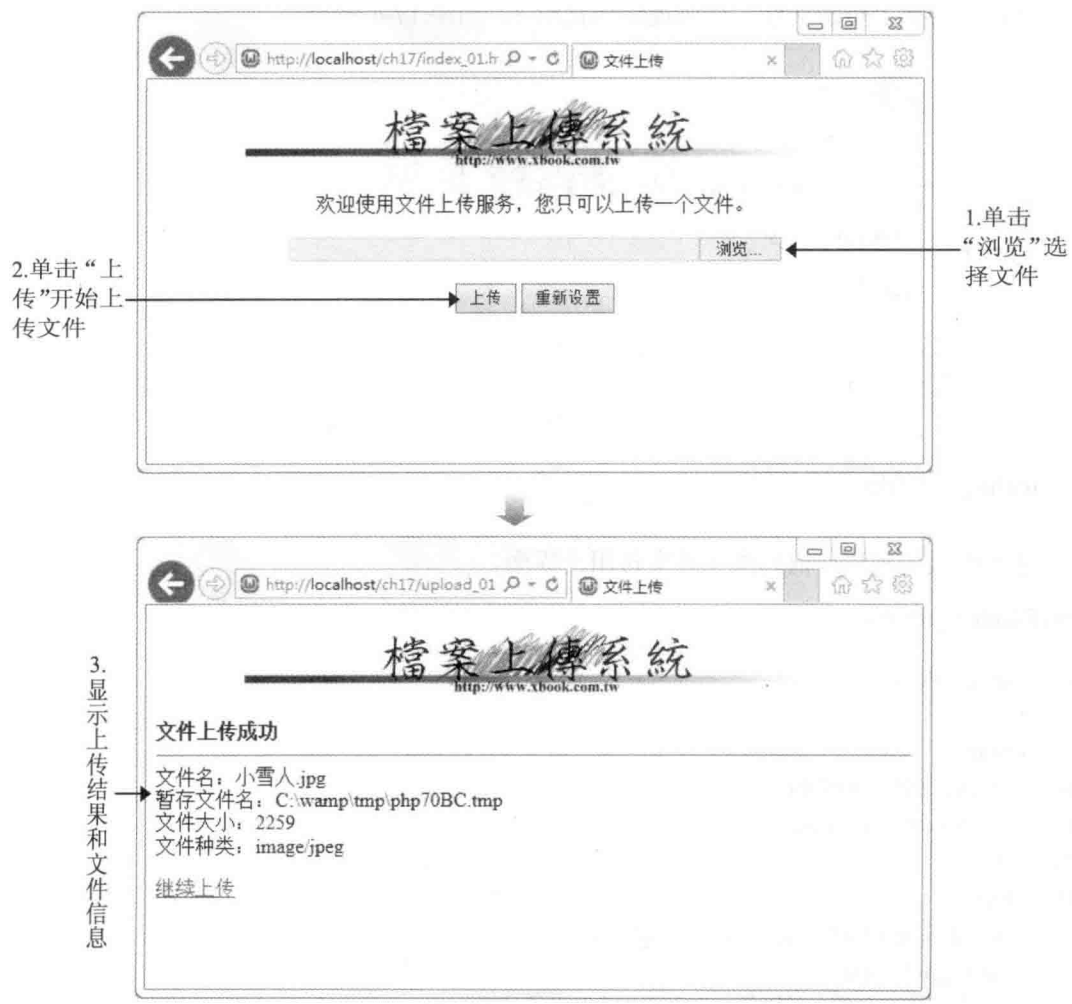


图 17-2

若文件大小超过 MAX_FILE_SIZE 隐藏字段所设置的大小，会出现错误代码 2，如图 17-3

所示。

图 17-3

若找不到要上传的文件，会出现错误代码 4，如图 17-4 所示。

图 17-4

❖ index_01.htm

这个网页用来制作前端的文件上传用户界面。

\ch17\index_01.htm

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <title>文件上传</title>
05:    <meta charset="utf-8">
06:  </head>
07:  <body>
08:    <p align="center"><img src="title.jpg"></p>
09:    <p align="center">
10:      欢迎使用文件上传服务，您只可以上传一个文件。
11:    </p>
12:    <p align="center">
```

```
13:       <form method="post" action="upload_01.php" enctype="multipart/form-data">
14:         <input type="hidden" name="MAX_FILE_SIZE" value="1048576">
15:         <input type="file" name="myfile" size="50"><br><br>
16:         <input type="submit" value="上传">
17:         <input type="reset" value="重新设置">
18:       </form>
19:     </P>
20:   </body>
21:</html>
```

- 13：设置表单处理程序为 "upload_01.php"，编码方式为 "multipart/form-data"。
- 14: 插入名称为 MAX_FILE_SIZE 的隐藏字段，配置文件大小限制为 1MB(1024 * 1024 = 1048576)，而且是放在文件字段的前面。
- 15：插入名称为 myfile 的文件字段，而且是放在隐藏字段的后面。

❖ upload_01.php

这个程序用来制作后端的处理程序。

\ch17\upload_01.php

```
01:<!doctype html>
02:<html>
03:   <head>
04:     <title>文件上传</title>
05:     <meta charset="utf-8">
06:   </head>
07:   <body>
08:     <p align="center"><img src="title.jpg"></p>
09:     <?php
10:       //指定文件保存的目录及文件名
11:       $upload_dir =   "./upload files/";
12:       $upload_file = $upload_dir . iconv("UTF-8", "Big5", $_FILES["myfile"]["name"]);
13:
14:       //将上传的文件由暂存目录移至指定的目录
15:       if（move_uploaded_file($_FILES["myfile"]["tmp_name"], $upload_file))
16:       {
17:         echo "<strong>文件上传成功</strong><hr>";
18:
19:         //显示文件信息
20:         echo "文件名：" . $_FILES["myfile"]["name"]. "<br>";
21:         echo "暂存文件名：" . $_FILES["myfile"]["tmp_name"]. "<br>";
22:         echo "文件大小：" . $_FILES["myfile"]["size"]. "<br>";
23:         echo "文件种类：" . $_FILES["myfile"]["type"]. "<br>";
```

```
24:            echo "<p><a href='javascript:history.back()'>继续上传</a></p>";
25:        }
26:        else
27:        {
28:            echo "文件上传失败（" . $_FILES["myfile"]["error"]. ")<br><br>";
29:            echo "<a href='javascript:history.back()'>重新上传</a>";
30:        }
31:    ?>
32:  </body>
33:</html>
```

- 10 ~ 11：设置用来保存文件的文件夹名称和文件名，为了避免上传中文文件名发生错误，我们使用 iconv() 函数将文件名转换为中文编码。
- 15 ~ 30：使用 move_uploaded_file() 函数将临时文件移至指定的文件夹并更改文件名；若移动成功，就执行第 17 ~ 24 行，显示文件信息，否则执行第 28 ~ 29 行，显示“文件上传失败”，并使用 $_FILES["myfile"]["error"]获取错误代码，其中第 29 行的 javascript:history.back() 用来返回上一页。

17.3 上传多个文件

无论要上传单一文件或多个文件，用户界面的编写方式完全相同，只是多插入几个文件字段而已，不同之处在于处理上传文件的程序代码，图 17-5 所示是上传多个文件的流程。

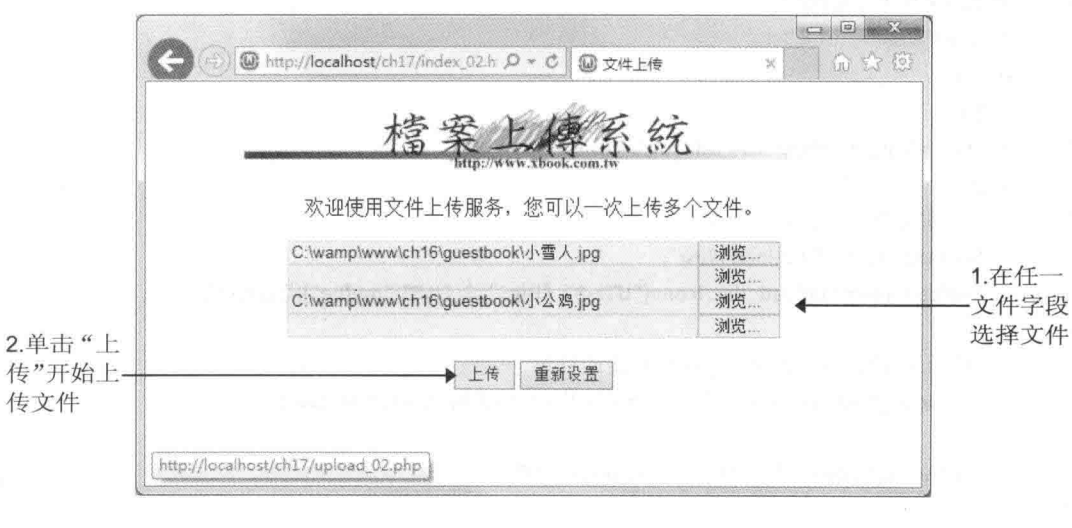


图 17-5

❖ index_02.htm

这个网页用来制作前端的文件上传用户界面，它和上传单一文件的 <\ch17\index_01.htm> 差不多，只是多插入几个文件字段而已，要注意的是所有文件字段的名称（即 name 属性）都必须相同，而且是数组，此处是设置为 myfile[]。

\ch17\index_02.htm

```
<!doctype html>
<html>
  <head>
    <title>文件上传</title>
    <meta charset="utf-8">
  </head>
  <body>
    <p align="center"><img src="title.jpg"></p>
    <p align="center">
      欢迎使用文件上传服务，您可以一次上传多个文件。
    </p>
    <p align="center">
    <form method="post" action="upload_02.php" enctype="multipart/form-data">
      <input type="file" name="myfile[]" size="50"><br>
      <input type="file" name="myfile[]" size="50"><br>
      <input type="file" name="myfile[]" size="50"><br>
      <input type="file" name="myfile[]" size="50"><br><br>
      <input type="submit" value="上传">
      <input type="reset" value="重新设置">
```

```
    </form>
    </P>
  </body>
</html>
```

upload_02.php

这个程序用来制作后端的处理程序。

\ch17\upload_02.php

```
<!doctype html>
<html>
  <head>
    <title>文件上传</title>
    <meta charset="utf-8">
  </head>
  <body>
    <p align="center"><img src="title.jpg"></p>
    <?php
      //指定保存文件的目录
      $upload_dir = "./upload files/";

      for（$i = 0; $i <= 3; $i++)
      {
        //若文件名不是空字符串，表示上传成功，将临时文件移至指定的目录。
        if（$_FILES["myfile"]["name"][$i]!= ""))
        {
          //搬移文件
          $upload_file = $upload_dir . iconv("UTF-8", "Big5", $_FILES["myfile"]["name"][$i]);
          move_uploaded_file($_FILES["myfile"]["tmp_name"][$i], $upload_file);

          //显示文件信息
          echo "文件名：" . $_FILES["myfile"]["name"][$i]. "<br>";
          echo "暂存文件名：" . $_FILES["myfile"]["tmp_name"][$i]. "<br>";
          echo "文件大小：" . $_FILES["myfile"]["size"][$i]. "<br>";
          echo "文件种类：" . $_FILES["myfile"]["type"][$i]. "<br>";
          echo "<hr>";
        }
      }
    ?>
  </body>
</html>
```

第 18 章

在线寄信服务与电子贺卡

18.1 在线寄信服务

在线寄信服务在网页设计中是相当常见的应用，例如下面的网页提供了一个人性化界面，让浏览者直接在网页上输入发件人名称及电子邮件信箱、收件人名称及电子邮件信箱、邮件格式、主题、内容等数据，而且还能设置附加文件，设置完毕后单击“发送邮件”按钮，就可以替浏览者寄出邮件。

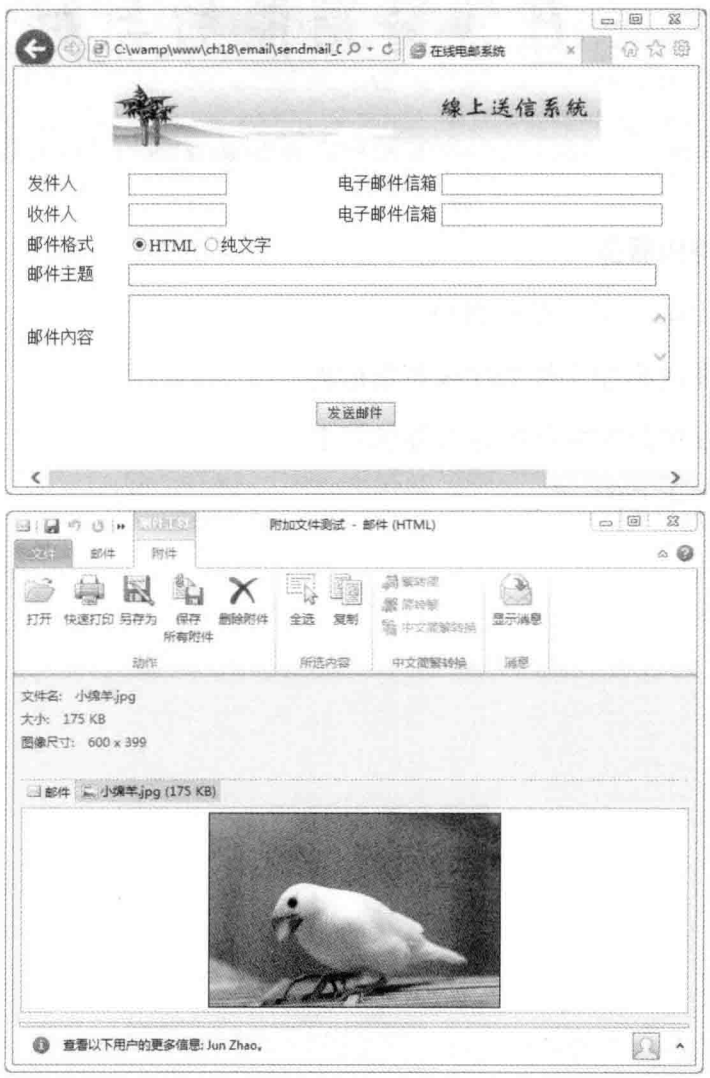


图 18-1

PHP 内建的 mail() 函数可以让我们使用 SMTP（Simple Mail Transfer Protocol）服务器发送邮件，但在此之前，我们必须先在 php.ini 配置设置文件中设置 SMTP 服务器的相关参数，现在，请您使用 Notepad++ 或其他文本编辑器打开 C:\wamp\bin\php\php5.5.12\php.ini 配置设

置文件，然后设置下列参数，如表 18-1 所示。

表 18-1　要设置的参数

参数	说明
SMTP	指定要使用的 SMTP 服务器的主机名或 IP 地址，localhost 表示本机计算机，这对自行配置 STMP 服务器的用户来说是正确的，若您要使用 ISP 提供的 SMTP 服务器，例如您是 Hinet 的用户，那么请修改为 msa.hinet.net，同理，若您的 ISP 为 SeedNet，那么您的 SMTP 服务器是 seed.net.tw，请按照实际情况判断自己的 SMTP 服务器是什么，务必指定一台可以用来发送 E-mail 的 SMTP 服务器
smtp_port	指定 SMTP 服务器使用的端口编号，默认值为 25，无论您使用哪家 ISP 的 SMTP 服务器，端口编号的默认值都是 25，除非 ISP 有特别指定，若是这样的话，就必须改变此参数的设置，举例来说，假设 SMTP 服务器的端口编号为 235，那么就必须将此参数设置为 235
sendmail_from	指定预设的发件人 e-mail，此为非必要参数，您可以自行决定是否设置，若要设置，请记得删除前面的分号（;），在此我们将之设置为 jean@seed.net.tw，请您按照实际情况进行设置
sendmail_path	此参数为 unix-like（例如 Linux、FreeBSD、Solaris、OpenBSD）的 SMTP 服务器专用，Windows 的 SMTP 服务器无须设置此参数，若要进行设置，请记得删除前面的分号（;）

请注意，php.ini 配置文件里的参数名称前面若是以分号（;）开头，表示此参数被注释起来，若要启用它，只要将分号删除即可，设置完毕后，记得要重新启动 Apache Web Server，以使新的配置设置生效。

18.2　使用 mail() 函数发送邮件

18.2.1　发送纯文本邮件

PHP 内置的 mail() 函数可以用来发送邮件，它的缺点是只能发送给收件人，若要发送邮件给副本及密件抄送收件人，则必须通过邮件头信息来指定副本及密件密本的收件人，其语法如下，若邮件发送成功，就返回 TRUE，否则返回 FALSE：

```
mail(string to, string subject, string message[, string headers[, string parameters]])
```

- *to*：指定收件人的电子邮件地址，若有多个收件人，请使用逗号（,）隔开，例如 "jerry.php@m2k.com.tw, jean.php@m2k.com.tw"。请注意，参数 *to* 并不支持类似 "Jerry<jerry.php@m2k.com.tw>" 的格式，若您希望邮件出现的收件人是收件人的名称，而不是收件人的电子邮件地址，必须通过参数 *headers* 来指定邮件头信息。
- *subject*：指定邮件主题。
- *message*：指定邮件内容，在预设的情况下，若您在邮件内容中加入 HTML 标签，收

件人看到的邮件内容将是包含 HTML 标签的源代码，而不是格式化的结果，如果要传送 HTML 格式的邮件，也必须通过参数 *headers* 来指定邮件头信息。

- *headers*：用来指定额外的邮件头信息，若有多个邮件头信息，必须以 \r\n 字符隔开。邮件头信息的用途很广，例如邮件要传送附加文件、发送邮件给副本及密件密本收件人、指定邮件编码方式等，都必须通过邮件头信息。
- *parameters*：用来传递额外的参数给 sendmail 服务。

下面是一个例子，寄信给 army_zhao@hotmail.com、jerry.php@m2k.com.tw 和 jean.php@m2k.com.tw 三位收件人。下面程序的运行结果如图 18-2 所示。

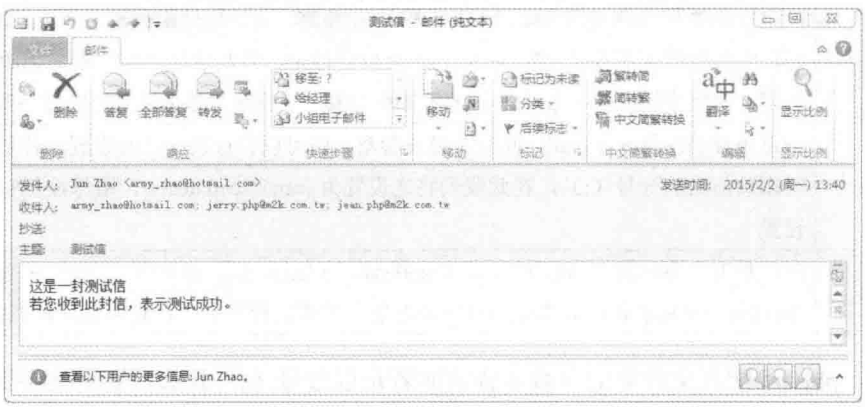


图 18-2

\ch18\mail_01.php

```
01:<?php
02:  //指定收件人
03:  $to = "army_zhao@hotmail.com, jerry.php@m2k.com.tw, jean.php@m2k.com.tw";
04:
05:  //指定邮件主题
06:  $subject = "测试信";
07:  $subject = "=?utf-8?B?" . base64_encode($subject) . "?=";
08:
09:  //指定邮件内容
10:  $message = "这是一封测试信\n\n 若您收到此封信，表示测试成功。";
11:
12:  //发送邮件
13:  mail($to, $subject, $message);
14:?>
```

- 03：指定邮件的收件人地址。
- 06～07：指定邮件主题，第 07 行的写法比较特别，用来指定邮件标题的编码方式为 UTF-8，并将标题进行 base64 编码，若不这么写，邮件标题及内容都会是乱码。
- 10：指定邮件的内容。

- 13：使用 mail() 函数发送邮件。

18.2.2　传送 HTML 格式的邮件

若要传送 HTML 格式的邮件，必须指定 Content-type 标头信息，下面是一个例子。

\ch18\mail_02.php

```
01:<?php
02:  //指定收件人
03:  $to = "army_zhao@hotmail.com";
04:
05:  //指定邮件主题
06:  $subject = "=?utf-8?B?" . base64_encode("HTML 格式测试信") . "?=";
07:
08:  //指定邮件内容
09:  $message = "
10:     <!doctype html>
11:     <html>
12:        <head>
13:           <title> </title>
14:        </head>
15:        <body bgcolor='#FFFFCC'>
16:           <p><strong>这是一封 HTML 格式的邮件</strong></p>
17:           <p><font color='Blue'>您可以使用任何 HTML 标签</font></p>
18:        </body>
19:     </html>
20:  ";
21:
22:  //若要发送 HTML 格式的邮件，须指定 Content-type 标头信息
23:  $headers  = "MIME-Version: 1.0\r\n";
24:  $headers .= "Content-type: text/html; charset=utf-8\r\n";
25:
26:  //发送邮件
27:  mail($to, $subject, $message, $headers);
28:?>
```

- 03：指定邮件收件人的电子邮件地址，若有多个收件人，必须使用逗号（,）隔开。
- 06：指定邮件主题。
- 09～20：指定邮件内容，您可以使用任何 HTML 标签，将邮件内容予以格式化。
- 23：指定邮件头信息，"MIME-Version: 1.0\r\n" 为 MIME 版本。
- 24：指定邮件头信息，Content-type:text/html 表示邮件内容支持 HTML 标签，

charset=utf-8 表示邮件编码方式为 UTF-8，若没有指定这些邮件头信息，收件人看到的邮件内容将是包含 HTML 标签的源代码，而不是格式化的结果。

- 27: 使用 mail() 函数发送邮件。

这个程序执行完毕后，收件人将收到如下邮件，如图 18-3 所示。

图 18-3

18.2.3 发送邮件给副本及密件抄送收件人

若要传送邮件给副本及密件抄送收件人，必须通过邮件头信息来指定副本及密件密本收件人，我们在第 18.2.2 小节已经介绍过 MIME-Version 和 Content-type 标头信息的用途，本节将告诉您下列几个标头信息的用途。

- To: 指定收件人的电子邮件地址，若有多个收件人，必须使用逗号（,）隔开，例如 "jerry.php@m2k.com.tw, jean.php@m2k.com.tw"。
  正如前面所说，mail() 函数的参数 *to* 不支持 "Jerry<jerry.php@m2k.com.tw>" 的格式，但邮件头信息 To 支持。若您希望邮件显示收件人的名称，而不是收件人的电子邮件地址，可以写成 "Jerry<jerry.php@m2k.com.tw>, Jean<jean.php@m2k.com.tw>" 的格式。
- From: 指定发件人的电子邮件地址，邮件头信息 From 也支持 "小绵羊 <grace.php@m2k.com.tw>" 的格式，但请注意，因为 "小绵羊" 是 ASCII 以外的字符，必须按照邮件主题一样的方式，对 "小绵羊" 进行编码，才不会出现乱码。
  不知道您是否已经发现，在前几个例子中，发件人的电子邮件地址一律等于 php.ini 配置设置文件内 sendmail_from 参数所设置的电子邮件地址，这种结果有点不太合理，因为不同的发件人本来就应该显示不同的发件人电子邮件地址，此时，我们可以使用邮件头信息 From 来决解这个问题。
- Cc: 指定副本收件人的电子邮件地址，若有多个收件人，必须使用逗号（,）隔开，例如 "jerry.php@m2k.com.tw, jean.php@m2k.com.tw"，邮件头信息 Cc 也支持

"Jerry<jerry.php@m2k.com.tw>" 的格式。

- Bcc：指定密件抄送收件人的电子邮件地址，若有多个收件人，必须使用逗号（,）隔开，例如 "jerry.php@m2k.com.tw, jean.php@m2k.com.tw"。
- Reply-To：指定回信的电子邮件地址，Reply-To 邮件头信息支持 "Jerry<jerry.php@m2k.com.tw>" 的格式。

以下面的程序代码 <\ch18\mail_03.php> 为例，它会将邮件寄给 army_zhao@hotmail.com, jerry.php@m2k.com.tw 和 jean.php@m2k.com.tw，显示的收件人为 "jerry" 和 "jean"，副本收件人为 ping.php@m2k.com.tw，密件抄送收件人为 liyu.php@m2k.com.tw，发件人为 army_zhao@hotmail.com，并显示为 "Jun Zhao"，邮件内容如图 18-4 所示。

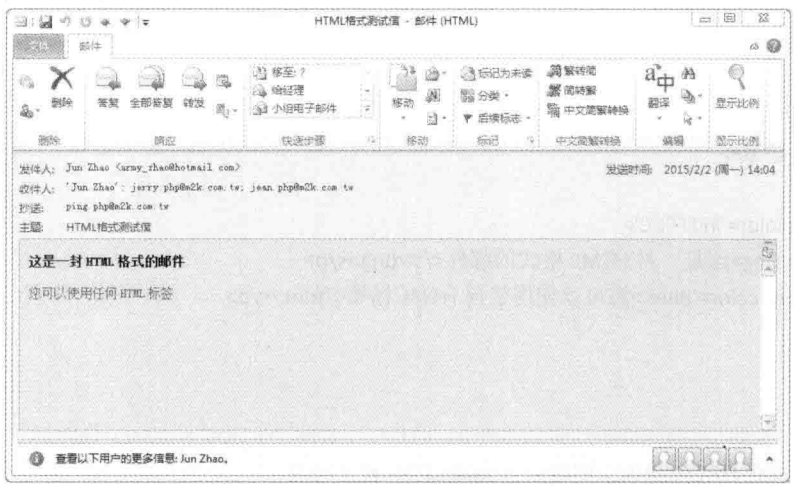


图 18-4

此时，若用鼠标双击发件人或收件人的名称，就会出现如下对话框，里面会显示其电子邮件地址，如图 18-5 所示。

图 18-5

\ch18\mail_03.php

```
<?php
  //指定收件人
  $to = "jerry.php@m2k.com.tw,jean.php@m2k.com.tw";

  //指定邮件主题
  $subject = "=?utf-8?B?" . base64_encode("HTML 格式测试信") . "?=";
  $from_name = "=?utf-8?B?" . base64_encode("小绵羊") . "?=";

  //指定邮件内容
  $message = "
    <!doctype html>
    <html>
      <head>
        <title></title>
      </head>
      <body bgcolor='#FFFFCC'>
        <p><strong>这是一封 HTML 格式的邮件</strong></p>
        <p><font color='Blue'>您可以使用任何 HTML 标签</font></p>
      </body>
    </html>
  ";

  //若要发送 HTML 格式的邮件，须指定 Content-type 标头信息
  $headers = "MIME-Version: 1.0\r\n";
  $headers .= "Content-type: text/html; charset=utf-8\r\n";
  $headers .= "To: Jerry<jerry.php@m2k.com.tw>, Jean<jean.php@m2k.com.tw>\r\n";
  $headers .= "From: $from_name <army_zhao@hotmail.com>\r\n";
  $headers .= "Cc: ping.php@m2k.com.tw\r\n";
  $headers .= "Bcc: liyu.php@m2k.com.tw\r\n";

  //发送邮件
  mail($to, $subject, $message, $headers);
?>
```

18.2.4 发送有附加文件的邮件

使用 mail() 函数发送有附加文件的邮件也是通过邮件头信息完成的，过程有点复杂，我们必须建立 MIME（Multipurpose Internet Mail Extensions）邮件，也就是包含多种区段内容的邮件。由于邮件拥有多个区段，所以我们只能使用邮件头信息来创建所有区段的内容，为了让读者便于了解，我们直接以下面的例子来进行说明。

\ch18\mail_04.php

```
01:<?php
02:   //指定收件人
03:   $to = "jerry.php@m2k.com.tw";
04:
05:   //指定发件人
06:   $from = "jean.php@m2k.com.tw";
07:
08:   //指定邮件主题
09:   $subject = "=?utf-8?B?" . base64_encode("附加文件测试") . "?=";
10:
11:   //指定邮件内容
12:   $message ="
13:      <!doctype html>
14:      <html>
15:         <head>
16:            <title> </title>
17:         </head>
18:         <body >
19:            <p><strong>这封邮件可以发送附加文件</strong></p>
20:            <p><font color='Blue'>您可以附加任何类型的文件</font></p>
21:         </body>
22:      </html>
23:   ";
24:
25:   //指定要发送的附加文件
26:   $file_name = "D:\小绵羊.jpg";
27:   $big5_file_name = iconv("UTF-8", "Big5", $file_name);
28:
29:   //建立 MIME 边界字符串
30:   $mime_boundary = md5(uniqid(mt_rand(), TRUE));
31:
32:   //打开指定的文件
33:   $fp = fopen($big5_file_name, "rb");
34:
35:   //读取文件内容
36:   $data = fread($fp, filesize($big5_file_name));
37:
38:   //使用 MIME base64 来对  $data 编码
39:   $data = chunk_split(base64_encode($data));
40:
41:   //创建邮件头信息
42:   $header = "From: $from \r\n";
```

```
43:  $header.= "To: $to\r\n";
44:  $header.= "MIME-Version: 1.0\r\n";
45:  $header.= "Content-Type: multipart/mixed; boundary=$mime_boundary\r\n";
46:
47:  //创建邮件内容
48:  $content = "This is a multi-part message in MIME format.\r\n";
49:  $content .= "--$mime_boundary\r\n";
50:  $content .= "Content-Type: text/html; charset=utf-8\r\n";
51:  $content .= "Content-Transfer-Encoding: 8bit\r\n\r\n";
52:  $content .= "$message\r\n";
53:
54:  //加入附加文件
55:  $content .= "--$mime_boundary\r\n";
56:  $content .= "Content-Type: image/pjpeg; name=". basename($file_name) . "\r\n";
57:  $content .= "Content-Disposition: attachment; filename=" . basename($file_name) ."\r\n";
58:  $content .= "Content-Transfer-Encoding: base64 \r\n\r\n";
59:  $content .= "$data\r\n";
60:  $content .= "--$mime_boundary--\r\n";
61:
62:  //发送邮件
63:  mail($to, $subject, $content, $header);
64:?>
```

- 03～09：指定收件人及发件人的电子邮件地址，并设置邮件主题。
- 12～23：指定邮件内容。
- 26～27：指定要发送的附加文件，您可以按照实际情况指定一个 JPG 图像文件。
- 30：建立 MIME 边界字符串，由于 MIME 邮件具有多个区段，所以这行语句的目的是随机产生一个区段与区段之间的分界字符串。mt_rand() 函数用来产生一个随机的数字；uniqid() 函数用来产生一个独一无二的标识符；md5() 函数用来产生一个 MD5 哈希码，这行语句所产生的哈希码是英文字母与阿拉伯数字的组合，类似于 af5e1072fb1e4221a7525bac031dc53e。
- 33：使用 fopen() 函数以只读的方式打开要发送的文件。
- 36：使用 fread() 函数以二进制的方式，将文件内容全部读取出来，filesize() 函数可以获取文件的大小。
- 39：附加文件只能以 base64 编码来传送，所以使用 base64_encode() 函数对 $data 进行 MIME base64 编码，chunk_split() 函数则用来将编码后的 $data 分割成每一行 76 个字符。
- 42 ～ 45：建立邮件头信息，第 42 行用来指定发件人的电子邮件地址，第 43 行指定收件人的电子邮件地址，第 44 行用来指定 MIME 版本，第 45 行用来指定邮件的 Content-Type 为 "multipart/mixed"，表示混合多种区段的邮件，并指定变量 $boundary 为 mime_boundary 的值，表示区段与区段之间的分界字符串为变量 $mime_boundary，

它的值是第 30 行随机产生的。

- 48～52：建立邮件第一个区段的内容，即文字的部分，其中第 48 行用来注明这是一封具有多个区段的 MIME 邮件；第 49 行用来标记边界，表示此区段的内容由这行语句开始；第 50 行的 Content-Type: text/html; 用来指定此区段的 Content-Type 为 "text/html"，表示接受 HTML 语法，若不想支持 HTML 语法，可以改为 "text/plain"，而 charset=utf-8 表示邮件编码方式为 UTF-8；第 52 行用来指定邮件内容为变量 $message 的值。
- 55～60：创建邮件第二个区段的内容，即附加文件的部分，其中第 55 行用来标记边界，表示此区段的内容由这行语句开始；第 56 行用来指定此区段的 Content-Type 为 "image/pjpeg"，且名称（name）为 basename($fileName)，basename() 函数用来获取文件名，若您不是发送 JPG 图像文件，请改为对应的 MIME 文件类型；第 57 行用来指定此区段的处理方式为 attachment（附加文件）；第 58、59 行用来指定此区段的编码方式为 base64，内容为变量 $data 的值；第 60 行标记边界，表示邮件内容到此结束。
- 63：使用 mail() 函数发送邮件。

我们可以将 <\ch18\mail_04.php> 改写成函数，以便重复使用。

\ch18\mail_05.php

```
<?php
  //指定收件人
  $to = "jerry.php@m2k.com.tw";

  //指定发件人
  $from = "jean.php@m2k.com.tw";

  //指定邮件主题
  $subject = "=?utf-8?B?" . base64_encode("附加文件测试") . "?=";

  //指定邮件内容
  $message = "
    <!doctype html>
    <html>
      <head>
        <title> </title>
      </head>
      <body >
        <p><strong>这封邮件可以发送附加文件</strong></p>
        <p><font color='Blue'>您可以附加任何类型的文件</font></p>
      </body>
    </html>
  ";
```

```
//指定要发送的附加文件
$file_name = "D:\小绵羊.jpg";

//调用 mail_attach() 函数来发送邮件
mail_attach($to, $from, $subject, $message, $file_name);

//自定义 mail_attach() 函数来发送邮件
function mail_attach($to, $from, $subject, $message, $file_name)
{
  $big5_file_name = iconv("UTF-8", "Big5", $file_name);

  //创建 MIME 边界字符串
  $mime_boundary = md5(uniqid(mt_rand(), TRUE));

  //打开指定的文件
  $fp = fopen($big5_file_name, "rb");

  //读取文件内容
  $data = fread($fp, filesize($big5_file_name));

  //使用 MIME base64 来对 $data 编码
  $data = chunk_split(base64_encode($data));

  //创建邮件头信息
  $header = "From: $from\r\n";
  $header.= "To: $to\r\n";
  $header.= "MIME-Version: 1.0\r\n";
  $header.= "Content-Type: multipart/mixed; boundary=$mime_boundary\r\n";

  //创建邮件内容
  $content = "This is a multi-part message in MIME format.\r\n";
  $content .= "--$mime_boundary\r\n";
  $content .= "Content-Type: text/html; charset=utf-8\r\n";
  $content .= "Content-Transfer-Encoding: 8bit\r\n\r\n";
      $content .= "$message\r\n";
  //加入附加文件
  $content .= "--$mime_boundary\r\n";
  $content .= "Content-Type: image/pjpeg; name=". basename($file_name) . "\r\n";
  $content .= "Content-Disposition: attachment; filename=" . basename($file_name) ."\r\n";
  $content .= "Content-Transfer-Encoding: base64 \r\n\r\n";
  $content .= "$data\r\n";
  $content .= "--$mime_boundary--\r\n";
```

```
    //发送邮件
    mail($to, $subject, $content, $header);
  }
?>
```

无论您执行 <\ch18\mail_04.php> 还是 <\ch18\mail_05.php>，收件人均可以收到如下邮件，里面包含一个附加文件 "小绵羊.jpg"，程序运行结果如图 18-6 所示。

图 18-6

18.3　无法发送附加文件的在线寄信服务

在本节中，我们将制作如下的在线寄信服务网页，浏览者必须逐一填写[发件人]和[电子邮件信箱]、[收件人]和[电子邮件信箱]，选择[邮件格式]，输入[邮件主题]和[邮件内容]，然后单击[发送邮件]按钮，就可以将邮件发送出去，要注意的是这个网页无法发送附加文件。

当浏览者单击[发送邮件]按钮，这个网页会先通过 JavaScript 程序检查浏览者是否有字段忘记填写，在所有字段均填写的情况下，才将邮件发送出去，否则会出现对话框通知浏览者有哪些字段忘记了填写。

请注意，在邮件发送出去后，网页上并不会出现任何信息通知浏览者，若您希望有这项功能，可以补写相关的程序代码。

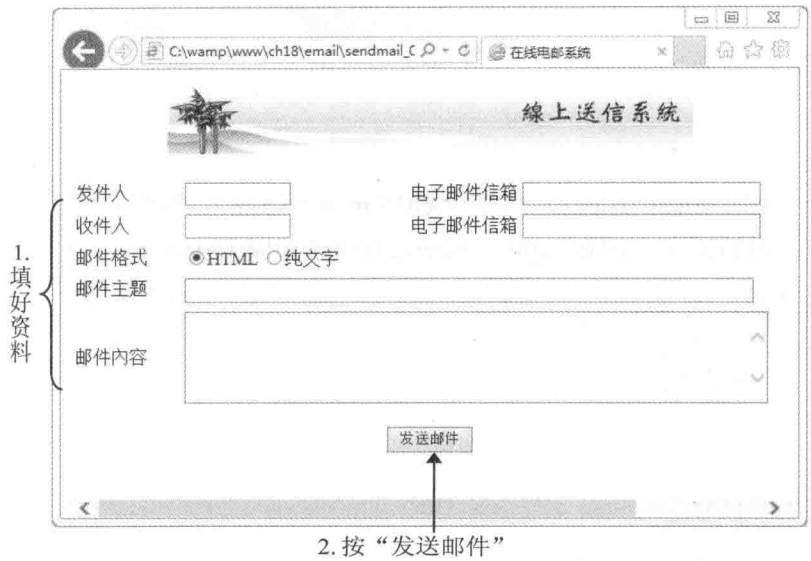


图 18-7

❖ sendmail_01.htm

这是在线寄信服务网页的用户界面，里面使用的功能均已介绍过，相信您很快就可以理解。

\ch18\email\sendmail_01.htm

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>在线电邮系统</title>
    <script type="text/javascript">
      function send()
      {
        if（document.myForm.from_name.value.length == 0)
          alert("发件人字段不可以空白哦！ ");
        else if（document.myForm.from_email.value.length == 0)
          alert("发件人电子邮件字段不可以空白哦！ ");
        else if（document.myForm.to_name.value.length == 0)
          alert("收件人字段不可以空白哦！ ");
        else if（document.myForm.to_email.value.length == 0)
          alert("收件人电子邮件字段不可以空白哦！ ");
        else if（document.myForm.subject.value.length == 0)
          alert("邮件主题字段不可以空白哦！ ");
        else if（document.myForm.message.value.length == 0)
          alert("邮件内容字段不可以空白哦！ ");
```

```
        else
          myForm.submit();
      }
    </script>
</head>
<body>
    <p align="center"><img src="title.jpg"></p>
    <form name="myForm" action="send_01.php" method="post">
      <table width="800" align="center" cellspacing="2" cellpadding="3">
        <tr bgcolor="#FFFFA1">
          <td width="12%">发件人</td>
          <td width="25%">
            <input type="text" name="from_name" size="10">
          </td>
          <td width="63%">电子邮件信箱
            <input type="text" name="from_email" size="30">
          </td>
        </tr>
        <tr bgcolor="#FFFFF0">
          <td>收件人</td>
          <td><input type="text" name="to_name" size="10"></td>
          <td>电子邮件信箱
            <input name="to_email" type="text" size="30">
          </td>
        </tr>
        <tr bgcolor="#FFFFA1">
          <td>邮件格式 </td>
          <td colspan="2">
            <input type="radio" name="format" value="html" checked>HTML
            <input type="radio" name="format" value="plain">纯文本
          </td>
        </tr>
        <tr bgcolor="#FFFFA1">
          <td>邮件主题</td>
          <td colspan="2">
            <input type="text" name="subject" size="80">
          </td>
        </tr>
        <tr bgcolor="#FFFFF0">
          <td>邮件内容</td>
          <td colspan="2">
            <textarea name="message" cols="62" rows="5"></textarea>
          </td>
        </tr>
```

```
        </table>
        <p align="center"><input type="button" value="发送邮件" onClick="send()"></p>
      </form>
    </body>
  </html>
```

❖ send_01.php

这是在线寄信服务网页的表单处理程序，里面使用的功能均已介绍过，相信您很快就可以理解。

\ch18\email\send_01.php

```
<?php
  //获取表单数据
  $from_name = "=?utf-8?B?" . base64_encode($_POST["from_name"]) . "?=";
  $from_email = $_POST["from_email"];
  $to_name = "=?utf-8?B?" . base64_encode($_POST["to_name"]) . "?=";
  $to_email = $_POST["to_email"];
  $format = $_POST["format"];
  $subject = "=?utf-8?B?" . base64_encode($_POST["subject"]) . "?=";
  $message = $_POST["message"];
  $message = "
    <!doctype html>
    <html>
      <head>
        <title></title>
      </head>
      <body>
        $message
      </body>
    </html>
  ";

  //指定标头信息
  $headers = "MIME-Version: 1.0\r\n";
  $headers .= "Content-type: text/$format; charset=utf-8\r\n";
  $headers .= "To: $to_name<$to_email>\r\n";
  $headers .= "From: $from_name<$from_email>\r\n";

  //发送邮件
  mail($to_email, $subject, $message, $headers);
?>
```

18.4　能够发送附加文件的在线寄信服务

在本节中，我们将制作如下的在线寄信服务网页，这个网页与上一节的网页主要差别在于能够发送附加文件。首先，浏览者必须一一填写[发件人]和[电子邮件信箱]、[收件人]和[电子邮件信箱]，选择[邮件格式]，输入[邮件主题]和[邮件内容]；接着，浏览者可以单击“附加文件”字段的[浏览]按钮，指定要发送的附加文件；最后再单击[发送邮件]按钮，就可以将邮件发送出去。

图 18-8

当浏览者单击[发送邮件]按钮时，这个网页会先通过 JavaScript 程序检查浏览者是否有字段忘记填写，在所有字段均填写的情况下，才将邮件寄出，否则会出现对话框通知浏览者有哪些字段忘记了填写。

由于发送附加文件时必须先将文件上传至服务器，因此，您必须熟悉文件上传的功能，这一内容第 17 章介绍过。此外，在邮件发送出去后，网页上并不会出现任何信息通知浏览者，若您希望有这项功能，可以补写相关的程序代码。

❖　sendmail_02.htm

这是在线寄信服务网页的用户界面，里面使用的功能均已介绍过。

\ch18\email\sendmail_02.htm

```
<!doctype html>
<html>
  <head>
```

```
    <meta charset="utf-8">
    <title>在线邮寄系统</title>
    <script type="text/javascript">
      function send()
              {
        if（document.myForm.from_name.value.length == 0)
          alert("发件人字段不可以空白哦！ ");
        else if（document.myForm.from_email.value.length == 0)
          alert("发件人电子邮件字段不可以空白哦！ ");
        else if（document.myForm.to_name.value.length == 0)
          alert("收件人字段不可以空白哦！ ");
        else if（document.myForm.to_email.value.length == 0)
          alert("收件人电子邮件字段不可以空白哦！ ");
        else if（document.myForm.subject.value.length == 0)
          alert("邮件主题字段不可以空白哦！ ");
        else if（document.myForm.message.value.length == 0)
          alert("邮件内容字段不可以空白哦！ ");
        else
          myForm.submit();
      }
    </script>
  </head>
  <body>
    <p align="center"><img src="title.jpg"></p>
    <form action="send_02.php" method="post" enctype="multipart/form-data"
      name="myForm">
      <table width="800" align="center" cellspacing="2" cellpadding="3">
              <tr bgcolor="#FFFFA1">
          <td width="12%">发件人</td>
          <td width="25%">
            <input type="text" name="from_name" size="10">
          </td>
          <td width="63%">电子邮件信箱
            <input type="text" name="from_email" size="30">
          </td>
        </tr>
        <tr bgcolor="#FFFFF0">
          <td>收件人</td>
          <td><input type="text" name="to_name" size="10"></td>
          <td>电子邮件信箱
            <input name="to_email" type="text" size="30">
          </td>
```

```
        </tr>
        <tr bgcolor="#FFFFA1">
          <td>邮件格式 </td>
          <td colspan="2">
            <input type="radio" name="format" value="html" checked>HTML
            <input type="radio" name="format" value="plain">纯文本
          </td>
        </tr>
        <tr bgcolor="#FFFFA1">
          <td>邮件主题</td>
          <td colspan="2">
            <input type="text" name="subject" size="80">
          </td>
        </tr>
        <tr bgcolor="#FFFFF0">
          <td>邮件内容</td>
          <td colspan="2">
            <textarea name="message" cols="62" rows="5"></textarea>
          </td>
        </tr>
        <tr bgcolor="#CCCCFF">
              <td>附加文件</td>
          <td colspan="2">
            <input type="file" name="myfile" size="80">
          </td>
        </tr>
      </table>
      <p align="center">
        <input type="button" value="发送邮件" onClick="send()">
      </p>
    </form>
  </body>
</html>
```

send_02.php

这是在线寄信服务网页的表单处理程序，里面使用的功能均已介绍过。

\ch18\email\send_02.php

```
<?php
  //获取表单数据
  $from_name = "=?utf-8?B?" . base64_encode($_POST["from_name"]) . "?=";
```

```
$from_email = $_POST["from_email"];
$to_name = "=?utf-8?B?" . base64_encode($_POST["to_name"]) . "?=";
$to_email = $_POST["to_email"];
$format = $_POST["format"];
$subject = "=?utf-8?B?" . base64_encode($_POST["subject"]) . "?=";
$message = $_POST["message"];

//创建 MIME 边界字符串
$mime_boundary = md5(uniqid(mt_rand(), TRUE));

//创建邮件头信息
$header = "From: $from_name<$from_email>\r\n";
$header .= "To: $to_name<$to_email>\r\n";
$header .= "MIME-Version: 1.0\r\n";
$header .= "Content-Type: multipart/mixed; boundary=". $mime_boundary . "\r\n";

//创建邮件内容
$content = "This is a multi-part message in MIME format.\r\n";
$content .= "--$mime_boundary\r\n";
$content .= "Content-Type: text/$format; charset=utf-8\r\n";
$content .= "Content-Transfer-Encoding: 8bit\r\n\r\n";
$content .= "$message\r\n";
$content .= "--$mime_boundary\r\n";

//若文件名称不是空白，表示上传成功，新增附加文件
if（$_FILES{"myfile"}{"name"} != ""）
{
   $file = $_FILES{"myfile"}{"tmp_name"};
   $file_name = $_FILES{"myfile"}{"name"};
   $content_type = $_FILES{"myfile"}{"type"};

   //打开文件
   $fp = fopen($file, "rb");

   //读取文件内容
   $data = fread($fp, filesize($file));

   //使用 MIME base64 来对 $data 编码
   $data = chunk_split(base64_encode($data));

   //加入附加文件
   $content .= "Content-Type: $content_type; name=$file_name\r\n";
   $content .= "Content-Disposition: attachment; filename=$file_name\r\n";
```

```
    $content .= "Content-Transfer-Encoding: base64\r\n\r\n";
    $content .= "$data\r\n";
    $content .= "--$mime_boundary--\r\n";
}

//发送邮件
mail($to_email, $subject, $content, $header);
?>
```

18.5　电子贺卡 DIY

因特网上存在各种电子贺卡网站，您是否羡慕那些会编写电子贺卡网站的设计人员呢？其实编写电子贺卡网站并不困难，本节就是要告诉您相关的技巧。以下图为例，首先，选择一种卡片样式，如果您觉得卡片太小看不清楚，可以单击卡片，打开大图来查看；接着，在下方填入卡片标题、寄信人与收件人的姓名及电子邮件信箱、选择音乐、选择卡片字体、卡片内容；最后，单击[卡片预览]按钮，预览刚才设置的卡片，若发现错误，可以返回上页进行修改，确认无误后才寄出。

图 18-9

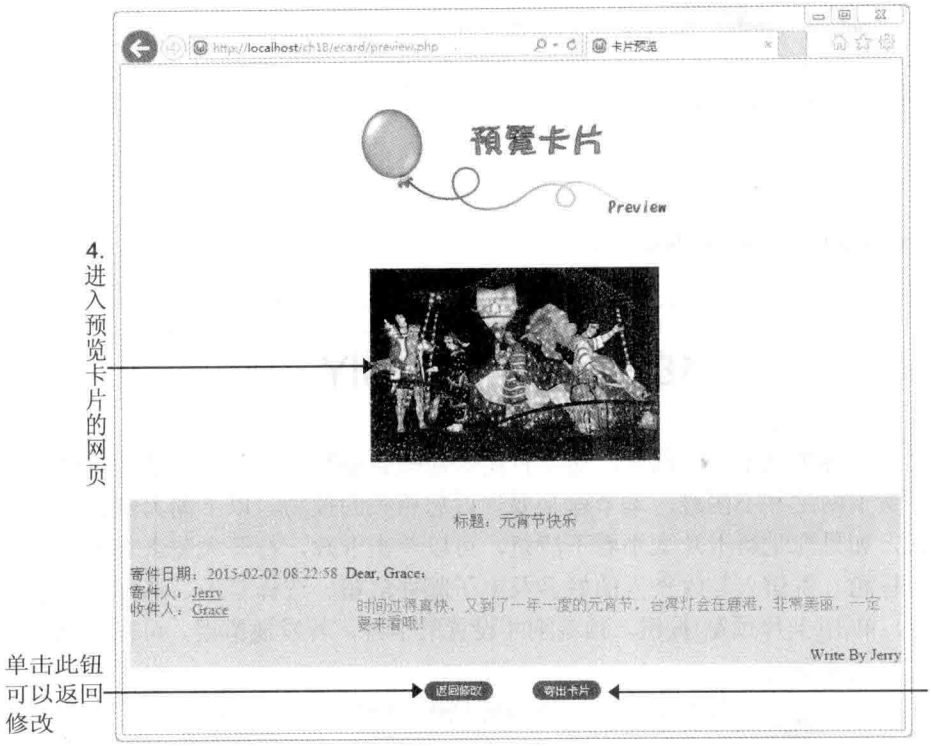

4.进入预览卡片的网页
预览卡片
Preview
标题：元宵节快乐
寄件日期：2015-02-02 08:22:58 Dear, Grace:
寄件人：Jerry
收件人：Grace
时间过得真快，又到了一年一度的元宵节。台湾灯会在鹿港，非常美丽，一定要来看哦！
Write By Jerry
返回修改
寄出卡片
单击此钮可以返回修改
单击此钮可以寄出卡片

图 18-10

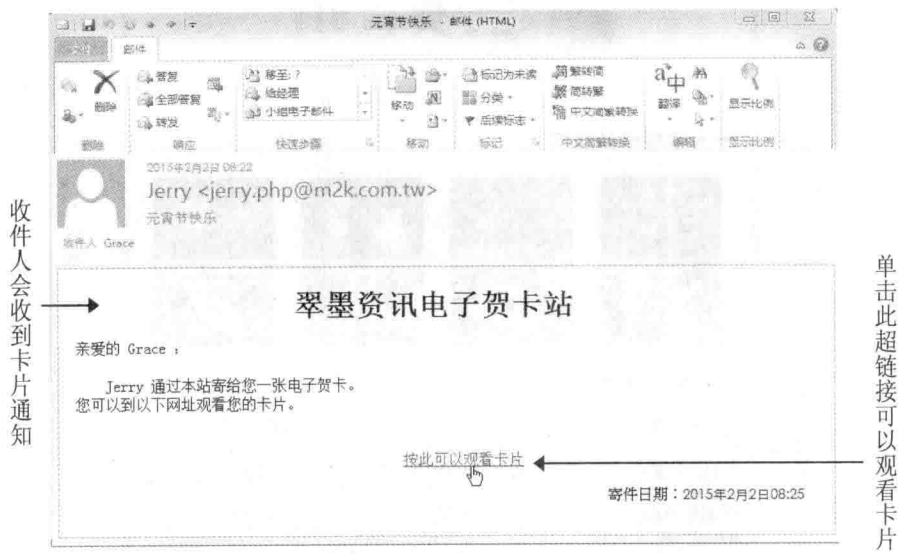

元宵节快乐 - 邮件 (HTML)
Jerry <jerry.php@m2k.com.tw>
元宵节快乐
收件人 Grace
翠墨资讯电子贺卡站
亲爱的 Grace ：
Jerry 通过本站寄给您一张电子贺卡。
您可以到以下网址观看您的卡片。
按此可以观看卡片
寄件日期：2015年2月2日08:25
收件人会收到卡片通知
单击此超链接可以观看卡片

图 18-11

图 18-12

图 18-13

18.5.1　组成网页的文件列表

这个电子贺卡网站存放在的 \samples\ch18\ecard\ 文件夹下，它用到如表 18-2 所示的文件。

表 18-2　电子贺卡网站用到的文件

文件名	说明
1.jpg ~ 8.jpg	这 8 个 JPEG 图像文件是卡片图像文件
1s.jpg ~ 8s.jpg	这 8 个 JPEG 图像文件是前述 8 个图像文件的缩略图
其他图像文件	有 preview.jpg、title.jpg、view.jpg、步骤一.jpg、步骤二.jpg 5 个图像文件，用来作为页标题，此外，modify.gif、sent.gif 则用来作为按钮图标
MIDI 音乐文件	由于是有声电子贺卡，所以我们使用 4 个 MIDI 音乐文件作为背景音乐
index.php	这是电子贺卡的主程序，可以让浏览者选择卡片样式并提供表单以输入卡片的收件人、主题及内容，然后判断数据是否完整，执行界面如前面的图(一)
preview.php	这个程序负责读取浏览者选择的卡片样式并读取在表单中所输入的收件人、主题及内容，然后将结果呈现给发件人查看，满意的话就单击[寄出卡片]按钮，否则单击[返回修改]按钮回到上一页进行修改，执行界面如前面的图(二)
ecard.php	在发件人单击[寄出卡片]按钮后，就会执行这个程序，它会读取<preview.php> 的表单数据，然后写入 ecard 数据库并寄出卡片通知邮件，前面的图(三)为卡片通知邮件的内容
get_ecard.php	这个程序提供给收件人查看卡片，执行界面如前面的图(四)
ecard 数据库	这个数据库包含一个名称为 card_message 的数据表，里面有 13 个字段

在这个电子贺卡网站中，我们使用了名称为 ecard 的数据库，里面包含一个 card_message 数据表，以存储电子贺卡的内容，其字段结构如表 18-3 所示。

表 18-3　card_message 数据表的字段结构

字段名	数据类型	长度	主键	说明
id	INT	-	☑	编号字段 自动编号（auto_increment）
card_id	VARCHAR	30	☐	卡片样式字段，用来记录卡片文件名
subject	TINYTEXT	-	☐	主题字段，用来记录卡片标题
from_name	VARCHAR	30	☐	发件人姓名字段
from_email	VARCHAR	50	☐	发件人电子邮件地址字段
to_name	VARCHAR	30	☐	收件人姓名字段
to_email	VARCHAR	50	☐	收件人电子邮件地址字段
music	VARCHAR	30	☐	卡片音乐字段，用来记录音乐文件名
style	VARCHAR	20	☐	样式字段，用来记录卡片字体样式
size	VARCHAR	10	☐	大小字段，用来记录卡片字体大小
color	VARCHAR	20	☐	颜色字段，用来记录卡片字体颜色
message	MEDIUMTEXT	-	☐	卡片内容字段
date	DATETIME	-	☐	日期字段

您可以自己创建数据库或导入本书为您准备的数据库备份文件（位于下载资源的

\samples\database\ecard.sql），有关如何导入 MySQL 数据库，可以参考第 11.3.7 小节的说明。

18.5.2　网页的运行流程

这个电子贺卡网站有些复杂，图 18-14 是整个程序的运行流程，稍微花点儿时间体会一下，就可以掌握其中的精髓。

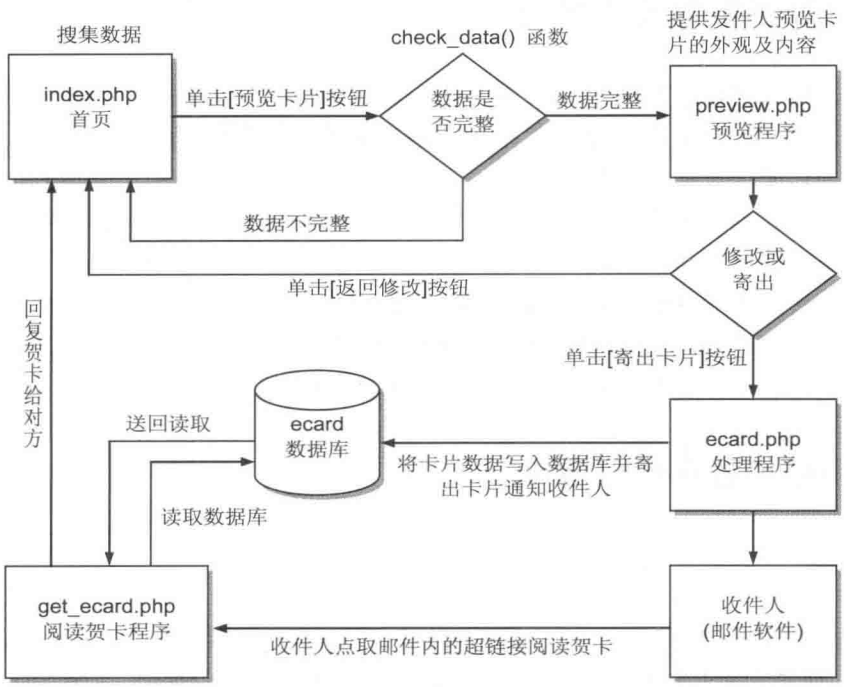


图 18-14

主程序 <index.php> 负责搜集卡片样式、发件人姓名及电子邮件信箱、收件人姓名及电子邮件信箱、卡片通知标题、卡片内容、声音、字体样式等数据，<index.php> 本身具有数据验证的功能，当浏览者单击[预览卡片]按钮时，会执行 check_data() 函数检查数据是否填写完整，若不完整，就请求浏览者再次填写，否则送出表单数据并调用表单处理程序 <preview.php>。

<preview.php> 会先读取 <index.php> 表单数据，让发件人预览刚才设置的卡片，若发件人要修改资料，可以单击[返回修改]按钮回到 <index.php> 修改卡片内容，确定要寄出卡片再单击[寄出卡片]按钮，请留意 <preview.php> 里面有隐藏的表单字段。

当发件人单击[寄出卡片]按钮时，会送出表单数据并调用表单处理程序 <ecard.php>，将数据写入 ecard 数据库，然后使用 mail() 函数传送卡片通知邮件。

在寄出卡片后，收件人只要接收邮件，就会收到一封卡片通知邮件，里面会留下类似 http://127.0.0.1/ch18/ecard/get_ecard.php?id=2&subject=%E85%E4BA 的超链接，收件人只要单击此超链接，就可以连接阅读卡片的网页 <get_ecard.php>，此时，<get_ecard.php> 会向 ecard

数据库请求卡片数据并返回查询结果，若找到符合的数据，就显示卡片数据，否则显示“没有您的卡片或卡片已经删除...”。

在阅读卡片时，若收件人想回复卡片给发件人，可以单击网页上的[我也要寄卡片给他（她）]超链接，就会自动打开 <index.php>，并自动填入收信者及电子邮件信箱、寄信者及电子邮件信箱、卡片通知标题等字段，收件人只要选择卡片样式和输入卡片内容即可。

18.5.3 必须具备的背景知识

- 首先，您必须熟悉 HTML 语法或其他网页编辑软件，因为我们将用到许多表格。附录 A、附录 B 和附录 C 分别有 HTML 语法教学、标签与属性速查、特殊字符表，有需要的读者，请自行参考。
- 其次，您必须了解表单的制作方式及如何读取表单数据，包括隐藏字段的应用，我们在第 8 章介绍过。
- 其三，您必须懂得如使用 mail() 函数发送邮件，我们在前几节介绍过。
- 其四，基本的 JavaScript 语法，我们将使用它来验证数据。
- 最后，您当然得熟悉 SQL 语法及如何访问数据库。

18.5.4 完整的程序代码清单

❖ index.php

这是电子贺卡的主程序，可以让浏览者选择卡片样式，并提供表单以输入卡片的收件人、标题及内容，然后判断数据是否完整，执行界面如前面的图 18-9 所示。

\ch18\ecard\index.php

```
001:<?php
002:   $from_name = "";
003:   $from_email = "";
004:   $to_name = "";
005:   $to_email = "";
006:   $subject = "";
007:
008:   if（isset($_GET["from_name"]))
009:   {
010:      $from_name = $_GET["from_name"];
011:      $from_email = $_GET["from_email"];
012:      $to_name = $_GET["to_name"];
013:      $to_email = $_GET["to_email"];
014:      $subject = $_GET["subject"];
```

若浏览者是直接浏览 <index.php>，那么这些变量将是空字符串；若浏览者选取 <get_ecard.php> 的“我也要寄卡片给他(她)”超链接才连上此网页，那么这些变量会记录寄信人姓名、电子邮件信箱等数据，然后分别成为第 070、074、076、080、082 行等单行文本框的默认值。

```
015:  }
016:?>
017:<!doctype html>
018:<html>
019:  <head>
020:    <title>电子贺卡 DIY</title>
021:    <meta charset="utf-8">
022:    <script type="text/javascript">
023:      function check_data()
024:      {
025:        if（document.eCard.subject.value.length == 0)
026:          alert("卡片通知标题字段不可以空白哦！ ");
027:        else if（document.eCard.from_name.value.length == 0)
028:          alert("寄件人姓名字段不可以空白哦！ ");
029:        else if（document.eCard.from_email.value.length == 0)
030:          alert("寄件人电子邮件字段不可以空白哦！ ");
031:        else if（document.eCard.to_name.value.length == 0)
032:          alert("收件人姓名字段不可以空白哦！ ");
033:        else if（document.eCard.to_email.value.length == 0)
034:          alert("收件人电子邮件字段不可以空白哦！ ");
035:        else if（document.eCard.message.value.length == 0)
036:          alert("卡片内容字段不可以空白哦！ ");
037:        else
038:          eCard.submit();
039:      }
040:    </script>
041:  </head>
042:  <body>
043:    <p align="center"><img src="images/title.jpg"></p>
044:    <form name="eCard" method="post" action="preview.php" >
045:      <p><img src="images/步骤一.jpg"></p>
046:      <table align="center" width="750">
047:        <?php
048:          $card_id = 0;
049:
050:          for（$i = 1; $i <= 2; $i++)
051:          {
052:            for（$j = 1; $j <= 4; $j++)
053:            {
054:              $card_id += 1;
055:              echo "<td>";
056:              echo "<input type='radio' name='card_id' value='";
057:              echo $card_id .".jpg' checked>";
```

```
                echo "<a href='images/" . $card_id . ".jpg' target='_blank'>";
                echo "<img src='images/" . $card_id . "s.jpg' border='1'></a>";
                echo "</td>";
            }
            echo "</tr>";
        }
    ?>
</table>
<p><img src="images/步骤二.jpg"></p>
<table width="750" align="center" cellspacing="0" cellpadding="4">
    <tr bgcolor="#F36B9D">
        <td colspan="2">1.卡片通知标题：
            <input name="subject" size="50" value="<?php echo $subject ?>"></td>
    </tr>
    <tr bgcolor="#FEA1C1">
        <td>2. 寄信人姓名：
            <input name="from_name" size="10" value="<?php echo $from_name ?>"></td>
        <td>电子邮件信箱：
            <input name="from_email" size="30" value="<?php echo $from_email ?>"></td>
    </tr>
    <tr bgcolor="#F36B9D">
        <td>3. 收信人姓名：
            <input name="to_name" size="10" value="<?php echo $to_name ?>"></td>
        <td>电子邮件信箱：
            <input name="to_email" size="30" value="<?php echo $to_email ?>"></td>
    </tr>
    <tr bgcolor="#FEA1C1">
        <td colspan="2">4. 选择音乐：
            <select name="music">
                <option value="music_01.mid" selected>如果云知道</option>
                <option value="music_02.mid">往事随风</option>
                <option value="music_03.mid">流浪的小孩</option>
                <option value="music_04.mid">棋子</option>
            </select>
        </td>
    </tr>
    <tr bgcolor="#F36B9D">
        <td colspan="2">5. 选择卡片字体 ：
            <select name="style">
                <option value="normal" selected>一般</option>
                <option value="italic">斜体</option>
            </select>
            大小：
```

```
101:              <select name="size">
102:                <option value="8 pt">8 pt</option>
103:                <option value="10 pt">10 pt</option>
104:                <option value="12 pt" selected>12 pt</option>
105:                <option value="14 pt">14 pt</option>
106:                <option value="16 pt">16 pt</option>
107:                <option value="20 pt">20 pt</option>
108:                <option value="24 pt">24 pt</option>
109:              </select>
110:              颜色：
111:              <select name="color">
112:                <option value="#000000" selected>黑色</option>
113:                <option value="#FF0000">红色</option>
114:                <option value="#FF6633">橙色</option>
115:                <option value="#339900">绿色</option>
116:                <option value="#0099FF">蓝色</option>
117:                <option value="#9933FF">紫色</option>
118:                <option value="#BB0000">暗红色</option>
119:                <option value="#005500">深绿色</option>
120:                <option value="#333399">深蓝色</option>
121:                <option value="#663399">深紫色</option>
122:                <option value="#770000">咖啡色</option>
123:              </select>
124:            </td>
125:          </tr>
126:          <tr bgcolor="#FEA1C1">
127:            <td colspan="2">6.卡片内容：<br>    
128:              <textarea cols="72" name="message" rows="8"></textarea>
129:            </td>
130:          </tr>
131:        </table><br>
132:        <p align="center">
133:          <input type="button" value="卡片预览" onClick="check_data()">
134:          <input type="reset" value="重新填写">
135:        </p>
136:      </form>
137:    </body>
138:</html>
```

- 023 ~ 039：使用 JavaScript 编写 check_data() 函数，用来检查 eCard 表单所送出的数据是否完整，若不完整，就出现对话框通知浏览者。
- 050 ~ 063：使用嵌套循环产生一个 2 × 4 的表格，然后在每个单元格中建立一个单选

按钮并插入一张卡片。

- 070、074、076、080、082：这些单行文本框的默认值分别为第 002～015 行的变量，若浏览者是直接浏览 <index.php>，那么变量将是空字符串。
- 086～091：创建一个名称为 music 的下拉式菜单，里面有 4 个选项，用来设置卡片音乐。
- 096～099：创建一个名称为 style 的下拉式菜单，里面有两个选项，用来设置卡片字体样式。
- 101 ～ 109：创建一个名称为 size 的下拉式菜单，里面有 7 个选项，用来设置卡片字体大小。
- 111～123：创建一个名称为 color 的下拉式菜单，里面有 11 个选项，用来设置卡片字体颜色。
- 133：当浏览者单击[卡片预览]按钮时，会调用 check_data() 函数。

❖ preview.php

这个程序负责读取浏览者选择的卡片样式并读取在表单中输入的收件人、卡片通知标题及卡片内容，然后将结果呈现给发件人查看，满意的话就单击[寄出卡片]按钮，否则单击[返回修改]按钮回到上一页进行修改，执行界面如前面的图 18-10 所示。

\ch18\ecard\preview.php

```
01:<?php
02:   //获取表单数据
03:   $card_id = $_POST["card_id"];
04:   $subject = $_POST["subject"];
05:   $from_name = $_POST["from_name"];
06:   $from_email = $_POST["from_email"];
07:   $to_name = $_POST["to_name"];
08:   $to_email = $_POST["to_email"];
09:   $music = $_POST["music"];
10:   $style = $_POST["style"];
11:   $size = $_POST["size"];
12:   $color = $_POST["color"];
13:   $message = $_POST["message"];
14:?>
15:<!doctype html>
16:<html>
17:   <head>
18:      <title>卡片预览</title>
19:      <meta charset="utf-8">
20:      <embed src="music/<?php echo $music ?>" hidden loop="true"></embed>
21:   </head>
```

```
<body>
  <p align="center"><img src="images/preview.jpg"></p>
  <table width="800" align="center" cellpadding="0" cellspacing="0" border="0">
    <tr>
      <td colspan="2" bgcolor="#FFFFCC">
        <p align="center"><br>
          <img src="images/<?php echo $card_id ?>"><br><br></p></td>
    </tr>
    <tr>
      <td colspan="2" height="22" bgcolor="#CCCCFF" valign="middle">
        <p align="center"><font color="#9900CC">标题：<?php echo $subject ?></font></p>
      </td>
    </tr>
    <tr>
      <td width="28%" bgcolor="#FFC6E2" valign="top">
        <br><font color="#9900CC">
        寄件日期：<?php echo date("Y-m-d H:i:s") ?><br>
        寄件人：<a href="MAILTO:<?php echo $from_email ?>">
          <?php echo $from_name ?></a><br>
        收件人：<a href="MAILTO:<?php echo $to_email ?>">
          <?php echo $to_name ?></a></font>
      </td>
      <td bgcolor="#FFC6E2">
        <br><font color="#0000FF">Dear, <?php echo $to_name ?>：</font>
        <p style="font-style:<?php echo $style ?>; color:<?php echo $color ?>;
          font-size:<?php echo $size ?>; padding-left:3mm; padding-right:3mm">
          <?php echo $message ?>
        </p>
        <div align="right">
          <font color="#0000FF">Write By <?php echo $from_name ?></font>
        </div>
      </td>
    </tr>
  </table>
  <form action="ecard.php" method="post" >
    <input type="hidden" name="card_id" value="<?php echo $card_id ?>">
    <input type="hidden" name="subject" value="<?php echo $subject ?>">
    <input type="hidden" name="from_name" value="<?php echo $from_name ?>">
    <input type="hidden" name="from_email" value="<?php echo $from_email ?>">
    <input type="hidden" name="to_name" value="<?php echo $to_name ?>">
    <input type="hidden" name="to_email" value="<?php echo $to_email ?>">
    <input type="hidden" name="music" value="<?php echo $music ?>">
    <input type="hidden" name="style" value="<?php echo $style ?>">
```

```
65:       <input type="hidden" name="size" value="<?php echo $size ?>">
66:       <input type="hidden" name="color" value="<?php echo $color ?>">
67:       <input type="hidden" name="message" value="<?php echo $message ?>">
68:       <p align="center">
69:         <a href="javascript:history.back"><img src="images/modify.gif" border="0"></a>
70:         <input type="image" src="images/sent.gif">
71:       </p>
72:     </form>
73:   </body>
74:</html>
```

- 03 ~ 13：获取 <index.php> 的表单数据。
- 19：指定网页的编码方式为 UTF-8。
- 20：使用 <embed>...</embed> 标签插入背景音乐（卡片音乐），src 属性用来指定音乐文件的位置及文件名，hidden 属性表示将音乐控制面板隐藏起来，loop="true" 属性表示将播放次数设置为无限次。
- 38：使用 date() 函数获取系统目前的日期与时间。
- 46 ~ 47：使用 CSS（Cascading Style Sheet）指定区块样式，font-style 用来指定文字是否加斜体，font-size 用来指定文字大小，color 用来指定文字颜色，padding-left 用来指定区块文字与区块左边的间距，padding-right 用来指定区块文字与区块右边的间距。
- 56 ~ 72：创建一个表单，当浏览者单击[寄出卡片]按钮时，必须将一开始从<index.php>搜集来的数据传送出去，所以我们使用表单来达成这个目的，但又因为表单所要传送的数据不需要我们输入，故使用隐藏字段（input type="hidden"）的方式。
- 69：加入一个图像超链接，图像文件来源为 modify.gif，当浏览者单击此图像时，会执行 JavaScript 语法的 history.back 指令，返回上一页。
- 70：原则上，我们应该使用 <input type="submit" value="寄出卡片"> 来送出表单数据，而此处的 <input type="image" src="sent.gif"> 也可以达到相同的目的，只是按钮形式由文字变成图像，且图像文件来源为 sent.gif。

❖ ecard.php

在发件人单击[寄出卡片]按钮后，就会执行这个程序，它会读取 <preview.php> 的表单数据，然后写入 ecard 数据库并寄出卡片通知邮件，通知收件人到指定的网页查看卡片，卡片通知邮件的内容如前文的图 18-11 所示。

\ch18\ecard\ecard.php

```
01:<?php
02:   require_once("dbtools.inc.php");
03:
```

```
04:  //获取表单数据
05:  $card_id = $_POST["card_id"];
06:  $subject = $_POST["subject"];
07:  $from_name = $_POST["from_name"];
08:  $from_email = $_POST["from_email"];
09:  $to_name = $_POST["to_name"];
10:  $to_email = $_POST["to_email"];
11:  $music = $_POST["music"];
12:  $style = $_POST["style"];
13:  $size = $_POST["size"];
14:  $color = $_POST["color"];
15:  $message = $_POST["message"];
16:  $date = date("Y-m-d H:i:s");
17:
18:  //建立数据连接
19:  $link = create_connection();
20:
21:  //执行 INSERT INTO 语句来将贺卡内容写入 card_message 数据表
22:  $sql = "INSERT INTO card_message（card_id, subject, from_name, from_email,
23:          to_name, to_email, music, style, size, color, message, date)
24:          VALUES（'$card_id', '$subject', '$from_name', '$from_email',
25:          '$to_name', '$to_email', '$music', '$style', '$size', '$color',
26:          '$message', '$date')";
27:  $result = execute_sql($link, "ecard", $sql);
28:
29:  //执行 SELECT 语句取出刚才加入记录的 id 字段
30:  $sql = "SELECT id FROM card_message WHERE subject='$subject' AND date='$date'";
31:  $result = execute_sql($link, "ecard", $sql);
32:
33:  //获取 id 字段的值
34:  $id = mysqli_fetch_object($result)->id;
35:
36:  //关闭数据连接
37:  mysqli_free_result($result);
38:  mysqli_close($link);
39:
40:  //设置查看电子贺卡的网址
41:  $current_url = "http://" . $_SERVER["REMOTE_ADDR"]. $_SERVER["PHP_SELF"];
42:  $get_ecard_url = str_replace("ecard.php", "get_ecard.php", $current_url);
43:  $get_ecard_url .= "?id=" . $id. "&subject=" . urlencode($subject);
44:
45:  //指定邮件内容
46:  $message = "<h1 align='center'>翠墨信息电子贺卡站</h1>";
```

```
47:  $message .= "<p>亲爱的【" . $to_name . "】：</p>";
48:  $message .= "<p>【" . $from_name . "】通过本站寄给您一张电子贺卡</p>";
49:  $message .= "<p>您可以到以下网址查看您的卡片：</p>";
50:  $message .= "<p align='center'><a href='$get_ecard_url'>";
51:  $message .= "按此可以查看卡片</a></p>";
52:  $message .= "<p align='right'>寄件日期：$date</p>";
53:
54:  $subject = "=?utf-8?B?" . base64_encode($subject) . "?=";
55:  $from_name = "=?utf-8?B?" . base64_encode($from_name) . "?=";
56:  $to_name = "=?utf-8?B?" . base64_encode($to_name) . "?=";
57:
58:  //若要发送 HTML 格式的邮件，须指定 Content-type 标头信息
59:  $headers  = "MIME-Version: 1.0\r\n";
60:  $headers .= "Content-type: text/html; charset=utf-8\r\n";
61:  $headers .= "From: $from_name<$from_email>\r\n";
62:  $headers .= "To: $to_name<$to_email>\r\n";
63:
64:  //发送邮件
65:  mail($to_email, $subject, $message, $headers);
66:?>
```

- 05 ~ 16：获取 <preview.php> 传送出来的数据。
- 22 ~ 27：将卡片内容写入 card_message 数据表。
- 29 ~ 34：由于在将卡片内容写入 card_message 数据表时，id 字段会自动编号，而且它是主键，具有唯一的特性，因此，我们可以通过这几行语句取出 id 字段的值。
- 41 ~ 42：用来产生查看卡片网页 <get_ecard.php> 的网址及参数，第 41 行用来获取当前网页 <ecard.php> 的网址，变量 $_SERVER["REMOTE_ADDR"]会返回网页服务器的IP地址，例如127.0.0.1，变量 $_SERVER["PHP_SELF"]会返回 <ecard.php> 的路径，例如 /ch18/ecard/ecard.php，因此，变量 current_url 存储的字符串是 "http://127.0.0.1/ch18/ecard/ecard.php"；第 42 行使用 str_replace() 函数取代字符串，其中我们是将变量 current_url 的值 "ecard.php" 取代为 "get_ecard.php"，所以变量 get_ecard_url 存储的字符串是 "http://127.0.0.1/ch18/ecard/get_ecard.php"，也就是查看卡片网页 <get_ecard.php> 的网址。
- 第 43 行用来指定 <get_ecard.php> 的参数，包括 id（编号）及 subject（卡片通知标题），其实只要使用 id 来识别即可，但由于 id 字段以数字递增 1 的方式来编号，容易被有心人猜中，进而偷窥卡片内容，所以多了一个 subject 参数，只有两者均符合才能查看卡片内容，以防他人任意查看其他人的卡片内容。此外，urlencode($subject) 表示对 $subject 变量进行网址编码，以避免网址出现 ACSII 以外的字符。
- 46 ~ 52：指定邮件内容。
- 54 ~ 56：将 $subject、$from_name、$to_name 3 个变量进行 base64 编码。

- 59 ~ 62，指定标头信息，我们将邮件格式设置为 HTML，这样才可以在邮件内使用 HTML 标签，至于编码方式则为 UTF-8。
- 65：使用 mail() 函数寄出卡片通知邮件。

get_ecard.php

这个程序提供给收件人查看卡片，执行前面的图 18-12。

\ch18\ecard\get_ecard.php

```
<?php
  require_once("dbtools.inc.php");

  //获取电子贺卡的编号及卡片标题
  $id = $_GET["id"];
  $subject = $_GET["subject"];

  //建立数据连接
  $link = create_connection();

  //执行 SQL 命令
  $sql = "SELECT * FROM card_message WHERE id = $id AND subject = '$subject'";
  $result = execute_sql($link, "ecard", $sql);

  //获取贺卡各字段的值
  if（mysqli_num_rows($result) != 0)
  {
    $row = mysqli_fetch_object($result);
    $id = $row->id;
    $card_id = $row->card_id;
    $subject = $row->subject;
    $from_name = $row->from_name;
    $from_email = $row->from_email;
    $to_name = $row->to_name;
    $to_email = $row->to_email;
    $music = $row->music;
    $style = $row->style;
    $size = $row->size;
    $color = $row->color;
    $message = $row->message;
    $date = $row->date;

    //释放 $result 占用的内存
```

```
        mysqli_free_result($result);
      }
    else
    {
        echo "没有您的卡片或卡片已经删除...";
        exit();
    }

    //关闭数据连接
    mysqli_close($link);

    //设置回复电子贺卡的参数
    $parameter = "from_name=" . urlencode($to_name) . "&to_email=" . $from_email;
    $parameter .= "&to_name=" . urlencode($from_name) . "&from_email=" . $to_email;
    $parameter .= "&subject=Re:" . urlencode($subject);
?>
<!doctype html>
<html>
    <head>
        <title>查看卡片</title>
        <meta charset="utf-8">
        <embed src="music/<?php echo $music ?>" hidden loop="true"></embed>
    </head>
    <body>
        <p align="center"><img src="images/view.jpg"></p>
        <table width="800" align="center" cellpadding="0" cellspacing="0">
            <tr>
                <td colspan="2" bgcolor="#FFFFCC">
                    <p align="center"><br><img src="images/<?php echo $card_id ?>"><br><br></p></td>
            </tr>
            <tr>
                <td colspan="2" height="22" bgcolor="#CCCCFF" valign="middle">
                    <p align="center"><font color="#9900CC">标题：<?php echo $subject ?></font></p>
                        </td>
            </tr>
            <tr>
                <td width="28%" bgcolor="#FFC6E2" valign="top">
                    <font color="#9900CC"><br>
                    寄件日期：<?php echo $date ?><br>
                    寄件人：<a href="mailto:<?php echo $from_email ?>">
                    <?php echo $from_name ?></a><br>
                    收件人：<a href=" mailto:<?php echo $to_email ?>">
```

```
            <?php echo $to_name ?></a></font>
          </td>
          <td bgcolor="#FFC6E2">
            <font color="#0000FF"><br>Dear, <?php echo $to_name ?>：</font>
            <p style="font-style:<?php echo $style ?>; color:<?php echo $color ?>;
              font-size:<?php echo $size ?>; padding-left:3mm; padding-right:3mm">
              <?php echo $message ?>
            </p>
            <div align="right">
              <font color="#0000FF">Write By <?php echo $from_name ?></font>
            </div>
          </td>
        </tr>
      </table>
      <p align="center">
      <a href="index.php?<?php echo $parameter ?>">我也要寄卡片给他（她）</a></p>
    </body>
</html>
```

第 19 章
会员管理系统

19.1　认识会员管理系统

“会员管理系统”也是网页上相当常用的系统，浏览者要进入某个网站时，必须先申请加入该网站的会员，通常加入会员都是免费的。

以下各图是我们即将要制作的会员管理系统，第一次拜访这个网站的人都必须先加入会员，等拥有一组独一无二的账号与密码后，就可以从首页登录网站了，如图 19-1 所示，成功登录后，可以进一步修改自己的资料、删除自己的资料或从该网站注销帐号，如图 19-2 所示。至于其他会员独享的功能，就请您自己创作吧！

另外要说明的是当会员忘记密码时，可以单击首页的“查询密码”超链接来查询自己的密码，有使用邮件通知和网页显示密码两种选择。

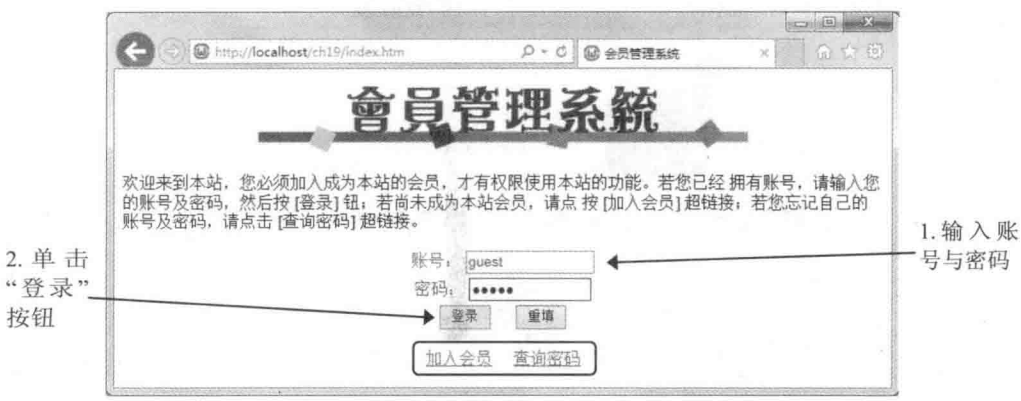


图 19-1

图 19-2

图 19-3

图 19-4

图 19-5

19.2 组成网页的文件列表

这个会员管理系统存放在下载资源的 \samples\ch19\ 文件夹中，它用到如下文件，如表 19-1 所示。

表 19-1 会员管理系统用到的文件

文件名	说明
所有 JPEG 图像文件	这些 JPEG 图像文件用来作为各个网页的标题图像
index.htm	这是首页，浏览者可以加入会员、登录网站或查询密码，执行界面如图 19-1
join.htm	这是加入会员的主程序，执行界面如图 19-3
addmember.php	这个程序负责搜集与处理 <join.htm> 传送出来的数据，若账号已经有人使用，就显示信息通知申请者，然后回到上一页让申请者重新输入账号，没有的话表示申请成功，执行界面如图 19-4
checkpwd.php	当浏览者从首页登录网站时，会启动这个程序，它会检查输入的账号与密码是否正确，若正确的话，就把某些重要数据写入 Cookie，然后重定向到 <main.php>

（续表）

文件名	说明
main.php	在会员输入正确的账号与密码后，会定向到这个程序，会员可以在此修改或删除自己的资料，执行界面如图 19-2
modify.php	当会员在 <main.php> 单击“修改会员资料”超链接时会链接到这个程序，用以修改会员资料
update.php	这个程序负责搜集从 <modify.php> 传送出来的会员资料，然后更新会员资料
delete.php	当会员在 <main.php> 单击“删除会员资料”超链接时会链接到这个程序，用以删除会员资料
search_pwd.htm	这个网页用来查询密码，执行界面如图 19-5
search.php	获取 <search_pwd.htm> 的表单数据来查询密码，并根据会员选择的查询方式，将密码显示在网页上或以电子邮件寄给会员
member 数据库	这个数据库包含一个名称为 users 的数据表，共有 14 个字段

在这个会员管理系统中，我们使用了名称为 member 的数据库，里面包含一个 users 数据表，用来存储会员资料，其字段结构如下。

字段名	数据类型	长度	主键	说明
id	INT	-	☑	编号字段 自动编号（auto_increment）
account	VARCHAR	10	☐	账号字段
password	VARCHAR	10	☐	密码字段
name	VARCHAR	10	☐	姓名字段
sex	CHAR	2	☐	性别字段
year	TINYINT	-	☐	出生年字段
month	TINYINT	-	☐	出生月字段
day	TINYINT	-	☐	出生日字段
telephone	VARCHAR	20	☐	电话字段
cellphone	VARCHAR	20	☐	移动电话字段
address	VARCHAR	50	☐	地址字段
email	VARCHAR	50	☐	电子邮件字段
url	VARCHAR	50	☐	个人网址字段
comment	TEXT		☐	备忘字段

您可以自己创建数据库或导入我们为您准备的数据库备份文件（位于下载资源的\samples\database\member.sql），有关如何导入 MySQL 数据库，可以参考第 11.3.7 小节的说明。

19.3 网页的运行流程

图 19-3 是整个会员管理系统的运行流程，您可以由此了解整个程序是如何工作的。

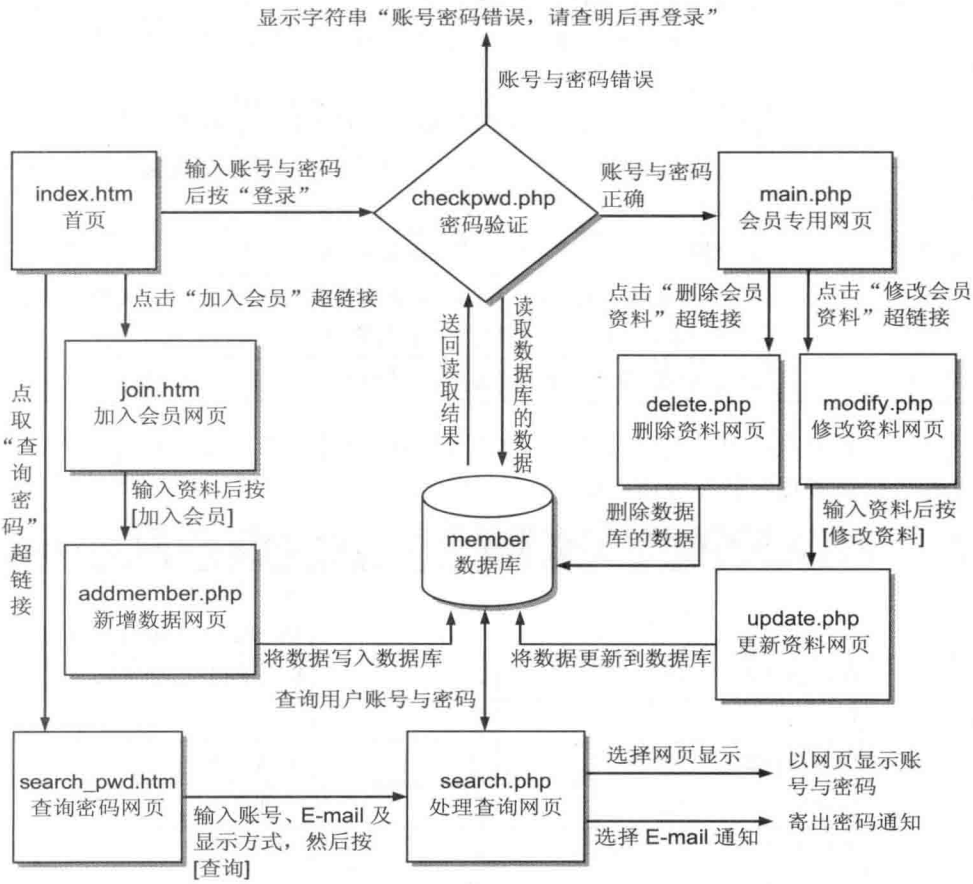


图 19-6

在主程序 <index.htm> 中，会员只要输入账号与密码，然后按[登录]，就可以登录网站；非会员可以单击“加入会员”超链接，注册成为会员；若会员忘记密码，可以单击“查询密码”超链接来查询密码。

当非会员在 <index.htm>中单击“加入会员”超链接时，会链接到 <join.htm>，待输入会员资料（* 符号表示一定要填写）并按[加入会员]后，就会调用 <join.htm> 的 check_data() 函数，这个函数用来验证会员资料是否正确，正确的话，就执行 <addmember.php>，将数据添加剂数据库。

当会员在 <index.htm>中输入账号与密码并按[登录]时，会执行 <checkpwd.php>，检查账号与密码是否正确，错误的话，就显示“账号密码错误，请查明后再登录”，否则定向到 <main.php>，此时，会员可以修改或删除自己的资料。当会员单击“修改会员资料”超链接时，会链接到 <modify.php>，在会员输入数据并按[修改资料]后，就执行 <update.php>，将

会员资料更新到数据库；当会员单击“删除会员资料”超链接时，会链接到 <delete.php>，将会员资料从数据库中删除。

当会员在 <index.htm>中单击“查询密码”超链接时，会链接到 <search_pwd.htm>，会员只要输入自己的账号与 E-mail 账号，然后选择一种显示方式，再按[查询]，就会执行 <search.php>，这个程序会根据输入的账号与 E-mail 账号，到数据库对比每一笔数据，若找到符合的数据，就根据会员选择的显示方式，将会员账号与密码显示在网页上或以邮件通知密码。

19.4　必须具备的背景知识

- 首先，您必须熟悉 HTML 语法或其他网页编辑软件。
- 其次，您必须了解表单的制作方式及如何读取表单数据，我们在第 8 章介绍过。
- 其三，您必须懂得如何使用 mail() 函数发送邮件，我们在第 18 章介绍过。
- 其四，您必须懂得如何存取 Cookie，我们在第 8 章介绍过。
- 其五，基本的 JavaScript 语法，我们将使用它来验证数据。
- 最后，您当然得熟悉 SQL 语法及如何访问数据库。

19.5　完整的程序代码清单

❖　index.htm

这是首页，浏览者可以加入会员、登录网站或查询密码，执行界面如前面的图 19-1 所示。

\ch19\index.htm

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>会员管理系统</title>
    <script type="text/javascript">
      function check_data()
      {
        if（document.myForm.account.value.length == 0)
          alert("账号字段不可以空白哦！ ");
        else if（document.myForm.password.value.length == 0)
          alert("密码字段不可以空白哦！ ");
        else
          myForm.submit();
      }
```

```
    </script>
  </head>
  <body>
    <p align="center"><img src="member.jpg"></p>
    <p>欢迎来到本站，您必须加入成为本站的会员，才有权限使用本站的功能。若您已经
      拥有账号，请输入您的账号及密码，然后按[登录]按钮；若尚未成为本站会员，请点
      按[加入会员]超链接；若您忘记自己的账号及密码，请单击[查询密码]超链接。</p>
    <form action="checkpwd.php" method="post" name="myForm">
      <table width="40%" align="center">
        <tr>
          <td align="center">
            <font color="#3333FF">账号：</font>
                <input name="account" type="text" size="15">
          </td>
        </tr>
        <tr>
          <td align="center">
            <font color="#3333FF">密码：</font>
            <input name="password" type="password" size="15">
          </td>
        </tr>
        <tr>
          <td align="center">
            <input type="button" value="登录" onClick="check_data()">
            <input type="reset" value="重填">
          </td>
        </tr>
      </table>
    </form>
    <p align="center">
      <a href="join.htm">加入会员</a>
      <a href="search_pwd.htm">查询密码</a></p>
  </body>
</html>
```

❖ join.htm

这是加入会员的主程序，执行界面如前面的图 19-3 所示。

\ch19\join.htm

```
001:<!doctype html>
002:<html>
```

```
<head>
  <title>加入会员</title>
  <meta charset="utf-8">
  <script type="text/javascript">
    function check_data()
    {
      if（document.myForm.account.value.length == 0)
      {
        alert("“用户账号”一定要填写哦...");
        return false;
      }
      if（document.myForm.account.value.length > 10)
      {
        alert("“用户账号”不可以超过 10 个字符哦...");
        return false;
      }
      if（document.myForm.password.value.length == 0)
      {
        alert("“用户密码”一定要填写哦...");
        return false;
      }
      if（document.myForm.password.value.length > 10)
      {
        alert("“用户密码”不可以超过 10 个字符哦...");
        return false;
      }
      if（document.myForm.re_password.value.length == 0)
      {
        alert("“密码确认”字段忘了填哦...");
        return false;
      }
      if（document.myForm.password.value != document.myForm.re_password.value)
      {
        alert("“密码确认”字段与“用户密码”字段一定要相同...");
        return false;
      }
      if（document.myForm.name.value.length == 0)
      {
        alert("您一定要留下真实姓名哦！ ");
        return false;
      }
      if（document.myForm.year.value.length == 0)
      {
```

```
046:            alert("您忘了填“出生年份”字段了...");
047:            return false;
048:        }
049:        if（document.myForm.month.value.length == 0)
050:        {
051:            alert("您忘了填“出生月份”字段了...");
052:            return false;
053:        }
054:        if（document.myForm.month.value > 12 | document.myForm.month.value < 1)
055:        {
056:            alert("“出生月份”应该介于 1~12 之间哦！");
057:            return false;
058:        }
059:        if（document.myForm.day.value.length == 0)
060:        {
061:            alert("您忘了填“出生日期”字段了...");
062:            return false;
063:        }
064:        if（document.myForm.month.value == 2 & document.myForm.day.value > 29)
065:        {
066:            alert("二月只有 28 天，最多 29 天");
067:            return false;
068:        }
069:        if（document.myForm.month.value == 4 | document.myForm.month.value == 6
070:            | document.myForm.month.value == 9 | document.myForm.month.value == 11)
071:        {
072:            if（document.myForm.day.value > 30)
073:            {
074:                alert("4 月、6 月、9 月、11 月只有 30 天哦！");
075:                return false;
076:            }
077:        }
078:        else
079:        {
080:            if（document.myForm.day.value > 31)
081:            {
082:                alert("1 月、3 月、5 月、7 月、8 月、10 月、12 月只有 31 天哦！");
083:                return false;
084:            }
085:        }
086:        if（document.myForm.day.value > 31 | document.myForm.day.value < 1)
087:        {
088:            alert("出生日期应该在 1~31 之间");
```

```
089:             return false;
090:         }
091:         myForm.submit();
092:     }
093:   </script>
094: </head>
095: <body>
096:   <p align="center"><img src="join.jpg"></p>
097:   <form action="addmember.php" method="post" name="myForm">
098:     <table border="2" align="center" bordercolor="#6666FF">
099:       <tr>
100:         <td colspan="2" bgcolor="#6666FF" align="center">
101:           <font color="#FFFFFF">请填入下列资料（标记“*”字段请务必填写)</font>
102:         </td>
103:       </tr>
104:       <tr bgcolor="#99FF99">
105:         <td align="right">*用户账号：</td>
106:         <td><input name="account" type="text" size="15">
107:         （请使用英文或数字键)</td>
108:       </tr>
109:       <tr bgcolor="#99FF99">
110:         <td align="right">*用户密码：</td>
111:         <td><input name="password" type="password" size="15">
112:         （请使用英文或数字键)</td>
113:       </tr>
114:       <tr bgcolor="#99FF99">
115:         <td align="right">*密码确认：</td>
116:         <td><input name="re_password" type="password" size="15">
117:         （再输入一次密码)</td>
118:       </tr>
119:       <tr bgcolor="#99FF99">
120:         <td align="right">*姓名：</td>
121:         <td><input name="name" type="text" size="8"></td>
122:       </tr>
123:       <tr bgcolor="#99FF99">
124:         <td align="right">*性别：</td>
125:         <td>
126:           <input type="radio" name="sex" value="男" checked>男
127:           <input type="radio" name="sex" value="女">女
128:         </td>
129:       </tr>
130:       <tr bgcolor="#99FF99">
```

```
131:            <td align="right">*生日：</td>
132:            <td>民国
133:              <input name="year" type="TEXT" size="2">年
134:              <input name="month" type="TEXT" size="2">月
135:              <input name="day" type="TEXT" size="2">日
136:            </td>
137:          </tr>
138:          <tr bgcolor="#99FF99">
139:            <td align="right">电话：</td>
140:            <td><input name="telephone" type="text" size="20"></td>
141:          </tr>
142:          <tr bgcolor="#99FF99">
143:            <td align="right">移动电话：</td>
144:            <td><input name="cellphone" type="text" size="20"></td>
145:          </tr>
146:          <tr bgcolor="#99FF99">
147:            <td align="right">地址：</td>
148:            <td><input name="address" type="text" size="45"></td>
149:          </tr>
150:          <tr bgcolor="#99FF99">
151:            <td align="right">E-mail 账号：</td>
152:            <td><input name="email" type="text" size="30"></td>
153:          </tr>
154:          <tr bgcolor="#99FF99">
155:            <td align="right">个人网站：</td>
156:            <td><input name="url" type="text" value="http://" size="40"></td>
157:          </tr>
158:          <tr bgcolor="#99FF99">
159:            <td align="right">备注：</td>
160:            <td><textarea name="comment" cols="45" rows="4" ></textarea></td>
161:          </tr>
162:          <tr bgcolor="#99FF99">
163:            <td align="center" colspan="2">
164:              <input type="button" value="加入会员" onClick="check_data()">
165:              <input type="reset" value="重新填写">
166:            </td>
167:          </tr>
168:        </table>
169:      </form>
170:  </body>
```

```
171:</html>
```

这个网页的源代码长达 6 页，看似复杂，其实很简单，所有概念我们都讲解过，最重要的是程序第 007 ~ 092 行的 check_data() 函数，其他说明如下。

- 054 ~ 058: if 控制结构用来判断输入的月份是否在 1 ~ 12 之间，若不在范围内，表示输入的月份错误。
- 064 ~ 068: if 控制结构用来判断当输入的月份是 2 月时，若日期超过 29 日，表示输入的日期错误，若日期为 29 日，可能对也可能错，因为只有闰年才会出现 2 月 29 日。
- 069 ~ 085: if 控制结构用来判断当输入的月份是 4、6、9、11 月时，若日期超过 30 日，表示输入的日期错误，因为这几个月只有 30 天，而当输入的月份是 1、3、5、7、8、10、12 月时，若日期超过 31 日，表示输入的日期错误，因为这几个月只有 31 天。
- 086 ~ 090: if 控制结构用来判断输入的日期是否在 1 ~ 31 之间，若不在范围内，表示输入的日期错误。

addmember.php

这个程序会先读取 <join.htm> 的表单数据，以用户账号（account）和数据库内的 account 字段做对比，若没有相同的 account 数据（即 mysqli_num_rows($result) == 0），表示该账号无人使用，那么将获取的数据写入数据库，然后显示加入会员成功信息；若对比结果有相同的 account 数据（即 mysqli_num_rows($result) != 0），表示该账号已经有人使用，那么会执行 if 控制结构内的程序代码，其中 JavaScript 指令 history.back() 用来使浏览界面返回上一页。

\ch19\addmember.php

```
<?php
  require_once("dbtools.inc.php");
  //获取表单数据
  $account = $_POST["account"];
  $password = $_POST["password"];
  $name = $_POST["name"];
  $sex = $_POST["sex"];
  $year = $_POST["year"];
  $month = $_POST["month"];
  $day = $_POST["day"];
  $telephone = $_POST["telephone"];
  $cellphone = $_POST["cellphone"];
  $address = $_POST["address"];
  $email = $_POST["email"];
  $url = $_POST["url"];
  $comment = $_POST["comment"];
```

```
    //建立数据连接
    $link = create_connection();

    //检查账号是否有人申请
    $sql = "SELECT * FROM users Where account = '$account'";
    $result = execute_sql($link, "member", $sql);

    //若账号已经有人使用
    if（mysqli_num_rows($result) != 0)
    {
      //释放 $result 占用的内存
      mysqli_free_result($result);

      //显示信息请求用户更换账号名称
      echo "<script type='text/javascript'>";
      echo "alert('您所指定的账号已经有人使用，请使用其他账号');";
      echo "history.back();";
      echo "</script>";
    }

    //若账号没人使用
    else
    {
      //释放 $result 占用的内存
      mysqli_free_result($result);

      //执行 SQL 命令，添加此账号
      $sql = "INSERT INTO users（account, password, name, sex,
              year, month, day, telephone, cellphone, address,
              email, url, comment) VALUES（'$account', '$password',
              '$name', '$sex', $year, $month, $day, '$telephone',
              '$cellphone', '$address', '$email', '$url', '$comment')";
      $result = execute_sql($link, "member", $sql);
    }
    //关闭数据连接
    mysqli_close($link);
  ?>
  <!doctype html>
  <html>
    <head>
      <meta charset="utf-8">
      <title>添加账号成功</title>
    </head>
```

```
<body bgcolor="#FFFFFF">
  <p align="center"><img src="Success.jpg">
  <p align="center">恭喜您已经注册成功了，您的资料如下：（请勿按重新刷新按钮）<br>
    账号：<font color="#FF0000"><?php echo $account ?></font><br>
    密码：<font color="#FF0000"><?php echo $password ?></font><br>
    请记下您的账号及密码，然后<a href="index.htm">登录网站</a>。
  </p>
 </body>
</html>
```

❖ checkpwd.php

当浏览者从首页登录网站时，会启动这个程序，它会检查输入的账号与密码是否正确，正确的话，就把数据写入 Cookie，然后定向到 <main.php>。

\ch19\checkpwd.php

```
01:<?php
02:   require_once("dbtools.inc.php");
03:   header("Content-type: text/html; charset=utf-8");
04:
05:   //获取表单数据
06:   $account = $_POST["account"];
07:   $password = $_POST["password"];
08:   //建立数据连接
09:   $link = create_connection();
10:
11:   //检查账号密码是否正确
12:   $sql = "SELECT * FROM users Where account = '$account' AND password = '$password'";
13:   $result = execute_sql($link, "member", $sql);
      14:
15:   //若账号密码错误
16:   if（mysqli_num_rows($result) == 0)
17:   {
18:      //释放 $result 占用的内存
19:      mysqli_free_result($result);
20:      //关闭数据连接
21:      mysqli_close($link);
22:      //显示信息请求用户输入正确的账号密码
23:      echo "<script type='text/javascript'>";
24:      echo "alert('账号密码错误，请查明后再登录');";
25:      echo "history.back();";
26:      echo "</script>";
```

```
27:  }
28:  //若账号密码正确
29:  else
30:  {
31:     //获取 id 字段
32:     $id = mysqli_fetch_object($result)->id;
33:
34:     //释放 $result 占用的内存
35:     mysqli_free_result($result);
36:
37:     //关闭数据连接
38:     mysqli_close($link);
39:     //将用户数据加入 cookies
40:     setcookie("id", $id);
41:     setcookie("passed", "TRUE");
42:     header("location:main.php");
43:  }
44:?>
```

- 16～27: 若在数据库内找不到账号与密码符合的数据，就显示错误信息并停止执行。
- 30～42: 若在数据库内找到账号与密码符合的数据，就将变量 id、passed 写入 Cookie，以利后续的验证。

❖ main.php

在会员输入正确的账号与密码后，会重定向到这个程序，会员可以在此修改或删除自己的资料，执行界面如前面的图 19-2 所示。

\ch19\main.php

```
01:<?php
02:  /*检查 cookie 中的 passed 变量是否不等于 TRUE，是的话，表示尚未登录网站，就重定向到首页 index.htm
*/
03:  $passed = $_COOKIE["passed"];
04:  if（$passed != "TRUE")
05:  {
06:     header("location:index.htm");
07:     exit();
08:  }
09:?>
10:<!doctype html>
11:<html>
12:  <head>
```

```
13:     <title>会员管理</title>
14:     <meta charset="utf-8">
15:  </head>
16:  <body>
17:     <p align="center"><img src="management.jpg"></p>
18:     <p align="center">
19:        <a href="modify.php">修改会员资料</a>     
20:        <a href="delete.php">删除会员资料</a>     
21:        <a href="logout.php">从网站退出</a></p>
22:  </body>
23:</html>
```

第 03 ~ 08 行用来判断 Cookie 变量 passed 是否不等于 TRUE，请您回想 <checkpwd.php> 的第 41 行，当会员输入的账号与密码和数据库内的数据相符时，表示为合法会员，会执行 setcookie("passed", "TRUE")，因此，当 $_COOKIE["passed"]不等于 TRUE 时，表示浏览者没有通过密码证验，此时会重定向到 <index.htm>，让浏览者输入账号与密码，这项功能用来防止有人直接从会员网页进入。

❖ modify.php

当浏览者在 <main.php> 中单击“修改会员资料”超链接时，会链接到这个程序以修改会员资料，在会员输入资料并按[修改资料]后会执行 <update.php>，<update.php> 会先检查是否为合法会员，是的话，再获取这个程序的表单数据，然后将会员资料更新到数据库。

\ch19\modify.php

```
001:<?php
002:  //检查 cookie 中的 passed 变量是否等于 TRUE
003:  $passed = $_COOKIE{"passed"};
004:
005:  //若 cookie 中的 passed 变量不等于 TRUE
006:  //表示尚未登录网站，将用户导向首页 index.htm
007:  if（$passed != "TRUE")
008:  {
009:     header("location:index.htm");
010:     exit();
011:  }
012:
013:  //若 cookie 中的 passed 变量等于 TRUE
014:  //表示已经登录网站，获取用户资料
015:  else
016:  {
017:     require_once("dbtools.inc.php");
```

```
018:
019:     $id = $_COOKIE{"id"};
020:
021:     //建立数据连接
022:     $link = create_connection();
023:
024:     //执行 SELECT 语句获取用户数据
025:     $sql = "SELECT * FROM users Where id = $id";
026:     $result = execute_sql($link, "member", $sql);
027:
028:     $row = mysqli_fetch_assoc($result);
029:?>
030:<!doctype html>
031:<html>
032:  <head>
033:     <title>修改会员资料</title>
034:     <meta charset="utf-8">
035:     <script type="text/javascript">
036:        function check_data()
037:        {
038:           if（document.myForm.password.value.length == 0)
039:           {
040:              alert("“用户密码”一定要填写哦...");
041:              return false;
042:           }
043:           if（document.myForm.password.value.length > 10)
044:           {
045:              alert("“用户密码”不可以超过 10 个字符哦...");
046:              return false;
047:           }
048:           if（document.myForm.re_password.value.length == 0)
049:           {
050:              alert("“密码确认”字段忘了填哦...");
051:              return false;
052:           }
053:           if（document.myForm.password.value != document.myForm.re_password.value)
054:           {
055:              alert("“密码确认”字段与“用户密码”字段一定要相同...");
056:              return false;
057:           }
058:           if（document.myForm.name.value.length == 0)
059:           {
060:              alert("您一定要留下真实姓名哦！");
061:              return false;
```

```
062:        }
063:        if（document.myForm.year.value.length == 0)
064:        {
065:          alert("您忘了填“出生年份”字段了...");
066:          return false;
067:        }
068:        if（document.myForm.month.value.length == 0)
069:        {
070:          alert("您忘了填“出生月份”字段了...");
071:          return false;
072:        }
073:        if（document.myForm.month.value > 12 | document.myForm.month.value < 1)
074:        {
075:          alert("“出生月份”应该介于 1~12 之间哦！");
076:          return false;
077:        }
078:        if（document.myForm.day.value.length == 0)
079:        {
080:          alert("您忘了填“出生日期”字段了...");
081:          return false;
082:        }
083:        if（document.myForm.month.value == 2 & document.myForm.day.value > 29)
084:        {
085:          alert("二月只有 28 天，最多 29 天");
086:          return false;
087:        }
088:        if（document.myForm.month.value == 4 | document.myForm.month.value == 6
089:          | document.myForm.month.value == 9 | document.myForm.month.value == 11)
090:        {
091:          if（document.myForm.day.value > 30)
092:          {
093:            alert("4 月、6 月、9 月、11 月只有 30 天哦！");
094:            return false;
095:          }
096:        }
097:        else
098:        {
099:          if（document.myForm.day.value > 31)
100:          {
101:            alert("1 月、3 月、5 月、7 月、8 月、10 月、12 月只有 31 天哦！");
102:            return false;
103:          }
104:        }
105:        if（document.myForm.day.value > 31 | document.myForm.day.value < 1)
```

```
106:          {
107:            alert("出生日期应该在 1~31 之间");
108:            return false;
109:          }
110:          myForm.submit();
111:        }
112:    </script>
113: </head>
114: <body>
115:    <p align="center"><img src="modify.jpg"></p>
116:    <form name="myForm" method="post" action="update.php" >
117:      <table border="2" align="center" bordercolor="#6666FF">
118:        <tr>
119:          <td colspan="2" bgcolor="#6666FF" align="center">
120:            <font color="#FFFFFF">请填入下列资料(标记“*”字段请务必填写)</font>
121:          </td>
122:        </tr>
123:        <tr bgcolor="#99FF99">
124:          <td align="right">*用户账号：</td>
125:          <td><?php echo $row{"account"} ?></td>
126:        </tr>
127:        <tr bgcolor="#99FF99">
128:          <td align="right">*用户密码：</td>
129:          <td>
130:            <input type="password" name="password" size="15"
131:              value="<?php echo $row{"password"} ?>">
132:            （请使用英文或数字键，勿使用特殊字符)
133:          </td>
134:        </tr>
135:        <tr bgcolor="#99FF99">
136:          <td align="right">*密码确认：</td>
137:          <td>
138:            <input type="password" name="re_password" size="15"
139:              value="<?php echo $row{"password"} ?>">
140:            （再输入一次密码，并记下您的用户名称与密码)
141:          </td>
142:        </tr>
143:        <tr bgcolor="#99FF99">
144:          <td align="right">*姓名：</td>
145:          <td><input type="text" name="name" size="8"
146:            value="<?php echo $row{"name"} ?>"></td>
147:        </tr>
148:        <tr bgcolor="#99FF99">
149:          <td align="right">*性别：</td>
```

```
150:          <td>
151:            <input type="radio" name="sex" value="男" checked>男
152             <input type="radio" name="sex" value="女">女
153:          </td>
154:        </tr>
155:        <tr bgcolor="#99FF99">
156:          <td align="right">*生日：</td>
157:          <td>
158:            <input type="text" name="year" size="2" value="<?php echo $row{"year"} ?>">年
159:            <input type="text" name="month" size="2"
160:              value="<?php echo $row{"month"} ?>">月
161:            <input type="text" name="day" size="2" value="<?php echo $row{"day"} ?>">日
162:          </td>
163:        </tr>
164:        <tr bgcolor="#99FF99">
165:          <td align="right">电话：</td>
166:          <td>
167:            <input type="text" name="telephone" size="20"
168:              value="<?php echo $row{"telephone"} ?>">
169:            （按照（02) 2311-3836 格式 or（04) 657-4587)
170:          </td>
171:        </tr>
172:        <tr bgcolor="#99FF99">
173:          <td align="right">移动电话：</td>
174:          <td>
175:            <input type="text" name="cellphone" size="20"
176:              value="<?php echo $row{"cellphone"} ?>">
177:            （按照（0922) 302-228 格式)
178:          </td>
179:        </tr>
180:        <tr bgcolor="#99FF99">
181:          <td align="right">地址：</td>
182:          <td><input type="text" name="address" size="45"
183:            value="<?php echo $row{"address"} ?>"></td>
184:        </tr>
185:        <tr bgcolor="#99FF99">
186:          <td align="right">E-mail 账号：</td>
187:          <td><input type="text" name="email" size="30"
188:            value="<?php echo $row{"email"} ?>"></td>
189:        </tr>
190:        <tr bgcolor="#99FF99">
191:          <td align="right">个人网站：</td>
192:          <td><input type="text" name="url" size="40"
193:            value="<?php echo $row{"url"} ?>"></td>
```

```
194:          </tr>
195:          <tr bgcolor="#99FF99">
196:            <td align="right">备注：</td>
197:            <td><textarea name="comment" rows="4" cols="45">
198:              <?php echo $row{"comment"} ?></textarea></td>
199:          </tr>
200:          <tr bgcolor="#99FF99">
201:            <td colspan="2" align="CENTER">
202:              <input type="button" value="修改资料" onClick="check_data()">
203:              <input type="reset" value="重新填写">
204:            </td>
205:          </tr>
206:        </table>
207:      </form>
208:  </body>
209:</html>
210:<?php
211:      //释放资源及关闭数据连接
212:      mysqli_free_result($result);
213:      mysqli_close($link);
214:  }
215:?>
```

- 007 ~ 011：用来判断 Cookie 变量 passed 是否等于 TRUE，请您回想一下 <checkpwd.php> 的第 41 行，当会员输入的账号与密码和数据库内的数据符合时，表示为合法会员，会执行 setcookie("passed", "TRUE")，因此，当 $_COOKIE["passed"] 不等于 TRUE 时，表示浏览者没有通过密码验证，也就不是合法的进入者，此时会重定向到 <index.htm>，让浏览者输入账号与密码，这项功能用来防止有人直接从某一会员网页进入。
- 019 ~ 028：获取 Cookie 变量 id 的值，以得知当前要修改资料的是哪个账号的浏览者，接着打开数据库，并筛选出数据库内哪个 id 字段的值等于变量 id 的值，这样就能取出当前要修改的会员资料。
- 116 ~ 207：这段程序代码和 <join.htm> 的程序代码类似，只是将每个文字输入字段的初始值设置为原会员资料，此处不再重复讲解。

update.php

这个程序负责搜集从 <modify.php> 传送出来的会员资料，然后更新会员资料。

\ch19\update.php

```
<?php
  //检查 cookie 中的 passed 变量是否等于 TRUE
```

```
$passed = $_COOKIE["passed"];

/* 若 cookie 中的 passed 变量不等于 TRUE，
   表示尚未登录网站，将用户导向首页 index.htm */
if（$passed != "TRUE")
{
  header("location:index.htm");
  exit();
}

/* 若 cookie 中的 passed 变量等于 TRUE，
   表示已经登录网站，则获取用户数据 */
else
{
  require_once("dbtools.inc.php");

  //获取 modify.php 网页的表单数据
  $id = $_COOKIE["id"];
  $password = $_POST["password"];
  $name = $_POST["name"];
  $sex = $_POST["sex"];
  $year = $_POST["year"];
  $month = $_POST["month"];
  $day = $_POST["day"];
  $telephone = $_POST["telephone"];
  $cellphone = $_POST["cellphone"];
  $address = $_POST["address"];
  $email = $_POST["email"];
  $url = $_POST["url"];
  $comment = $_POST["comment"];

  //建立数据连接
  $link = create_connection();

  //执行 UPDATE 语句来更新用户数据
  $sql = "UPDATE users SET password = '$password', name = '$name',
          sex = '$sex', year = $year, month = $month, day = $day,
          telephone = '$telephone', cellphone = '$cellphone',
          address = '$address', email = '$email', url = '$url',
          comment = '$comment' WHERE id = $id";
  $result = execute_sql($link, "member", $sql);

  //关闭数据连接
  mysqli_close($link);
```

```
    }
?>
<!doctype html>
<html>
  <head>
    <title>修改会员资料成功</title>
    <meta charset="utf-8">
  </head>
  <body>
    <center>
      <img src="revise.jpg"><br><br>
      <?php echo $name ?>，恭喜您已经修改资料成功了。
      <p><a href="main.php">回会员专用网页</a></p>
    </center>
  </body>
</html>
```

❖ delete.php

当会员在 <main.php> 单击“删除会员资料”超链接时，会链接到这个程序，将会员资料从会员数据库中删除。

\ch19\delete.php

```
<?php
  //检查 cookie 中的 passed 变量是否等于 TRUE
  $passed = $_COOKIE["passed"];

  /*      若 cookie 中的 passed 变量不等于 TRUE，
      表示尚未登录网站，将用户导向首页 index.htm */
  if（$passed != "TRUE")
  {
    header("location:index.htm");
    exit();
  }
  /*      若 cookie 中的 passed 变量等于 TRUE，
      表示已经登录网站，将用户的账号删除 */
  else
  {
    require_once("dbtools.inc.php");

    $id = $_COOKIE["id"];
```

```
    //建立数据连接
    $link = create_connection();

    //删除账号
    $sql = "DELETE FROM users Where id = $id";
    $result = execute_sql($link, "member", $sql);

    //关闭数据连接
    mysqli_close($link);
  }
?>
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>删除会员资料成功</title>
  </head>
  <body bgcolor="#FFFFFF">
    <p align="center"><img src="erase.jpg"></p>
    <p align="center">
      您的资料已从本站中删除，若要再次使用本站台服务，请重新申请，谢谢。
    </p>
    <p align="center"><a href="index.htm">回首页</a></p>
  </body>
</html>
```

search_pwd.htm

这个网页用来查询密码，执行界面如前面的图 19-5 所示。

\ch19\search_pwd.htm

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>查询密码</title>
  </head>
  <body>
    <p align="center"><img src="inquire.jpg"></p>
    <p>请输入您的姓名及 E-mail 账号，并选择一种显示方式，然后按[查询]按钮。</p>
    <form method="post" action="search.php" >
      <table align="center">
        <tr><td>账号：</td>
```

```
            <td><input type="text" name="account" size="10"></td>
          </tr>
          <tr><td>电子邮件账号：</td>
            <td><input type="text" name="email" size="30"></td>
          </tr>
          <tr><td>显示方式：</td>
            <td>
              <select name="show_method">
                <option value="网页显示">网页显示</option>
                <option value="E-mail 传送">E-mail 传送</option>
              </select>
            </td>
          </tr>
          <tr>
            <td colspan="2" align="center">
              <input type="submit" value="查询"><input type="reset" value="重填">
            </td>
          </tr>
        </table>
      </form>
    </body>
  </html>
```

❖ search.php

获取 <search_pwd.htm> 的表单数据来查询密码，并根据会员选择的查询方式，将密码显示在网页上或用电子邮件寄给会员。

\ch19\search.php

```
<?php
  require_once("dbtools.inc.php");
  header("Content-type: text/html; charset=utf-8");

  //获取表单数据
  $account = $_POST["account"];
  $email = $_POST["email"];
  $show_method = $_POST["show_method"];

  //建立数据连接
  $link = create_connection();

  //检查查询的账号是否存在
  $sql = "SELECT name, password FROM users WHERE
```

```
            account = '$account' AND email = '$email'";
$result = execute_sql($link, "member", $sql);

//若账号不存在
if（mysqli_num_rows($result) == 0)
{
  //显示信息告知用户，查询的账号并不存在
  echo "<script type='text/javascript'>
            alert('您所查询的数据不存在，请检查是否输入错误。');
            history.back();
          </script>";
}
else  //若账号存在
{
  $row = mysqli_fetch_object($result);
  $name = $row->name;
  $password = $row->password;
  $message = "
    <!doctype html>
    <html>
      <head>
        <title></title>
        <meta http-equiv='Content-Type' content='text/html; charset=utf-8'>
      </head>
      <body>
        $name 您好，您的账号数据如下：<br><br>
            账号：$account<br>
            密码：$password<br><br>
          <a href='http://localhost/ch19/index.htm'>按此登录本站</a>
        </body>
    </html>";

  if（$show_method == "网页显示")
  {
    echo $message;      //显示信息告知用户账号密码
  }
  else
  {
    $subject = "=?utf-8?B?" . base64_encode("账号通知") . "?=";
    $headers   = "MIME-Version: 1.0\r\nContent-type: text/html; charset=utf-8\r\n ";
    mail($email, $subject, $message, $headers);

    //显示信息告知账号密码已寄至其电子邮件了
```

```
            echo "$name 您好，您的账号资料已经寄至 $email<br><br>
                    <a href='index.htm'>按此登录本站</a>";
        }
    }

    //释放 $result 占用的内存
    mysqli_free_result($result);
    //关闭数据连接
    mysqli_close($link);
?>
```

第 20 章
在线投票系统

20.1 认识在线投票系统

您想设计一个在线投票系统吗？本章可以让您如愿哦！您可以选择自己喜欢的明星，然后输入身份证号码，再按[投票]，即可完成投票，如图 20-1 所示，若里面没有您喜欢的明星，可以按[推荐候选人]来推荐候选人，如图 20-3 所示，同时您也可以按[查看投票结果]来查看投票结果如图 20-2 所示。

图 20-1

图 20-2

图 20-3

20.2　组成网页的文件列表

这个在线投票系统存放在网络下载资源的 \samples\ch20\ 文件夹中，它用到如表 20-1~表 20-3 所示的文件。

表 20-1　在线投票系统用到的文件

文件名	说明
4 个 JPEG 图像文件	其中 3 个 JPEG 图像文件用来作为各个网页的标题图像，另一个 JPEG 图像文件用来作为直方图
index.php	这是在线投票系统的首页，浏览者可以在此投票，执行界面如图 20-1 所示
recommend.htm	这是推荐候选人的主程序，执行界面如图 20-3 所示
recommend.php	这是推荐候选人网页的后端处理程序，它会检查您所推荐的人是否已经在候选人名单中，若候选人已经存在，就不用推荐了
result.php	这是用来显示投票结果的网页，执行界面如图 20-2 所示
vote.php	当浏览者在 <index.php> 按[投票]时会执行这个程序，它负责将被投票人的得票数加一
vote 数据库	这个数据库包含 candidate 和 id_number 两个数据表

在这个在线投票系统中，我们使用了名称为 vote 的数据库，里面包含 candidate 和 id_number 数据表，以存储候选人数据及投票人的身份证号码，其字段结构如下。（本范例程序采用的是台湾地区的身份证号码为例）

字段名	数据类型	长度	主键	说明
id	INT	-	☑	编号字段 自动编号（auto_increment）
Name	VARCHAR	20	☐	候选人姓名字段

（续表）

字段名	数据类型	长度	主键	说明
introduction	TEXT	-	□	候选人简介字段
score	INT	-	□	分数字段
id	VARCHAR	10	□	身份证号码字段

您可以自己创建数据库或导入本书为您准备的数据库备份文件（位于下载资源的\samples\database\vote.sql），有关如何导入 MySQL 数据库，可以参考第 11.3.7 小节的说明。

20.3 网页的运行流程

网页的运行流程如图 20-4 所示。

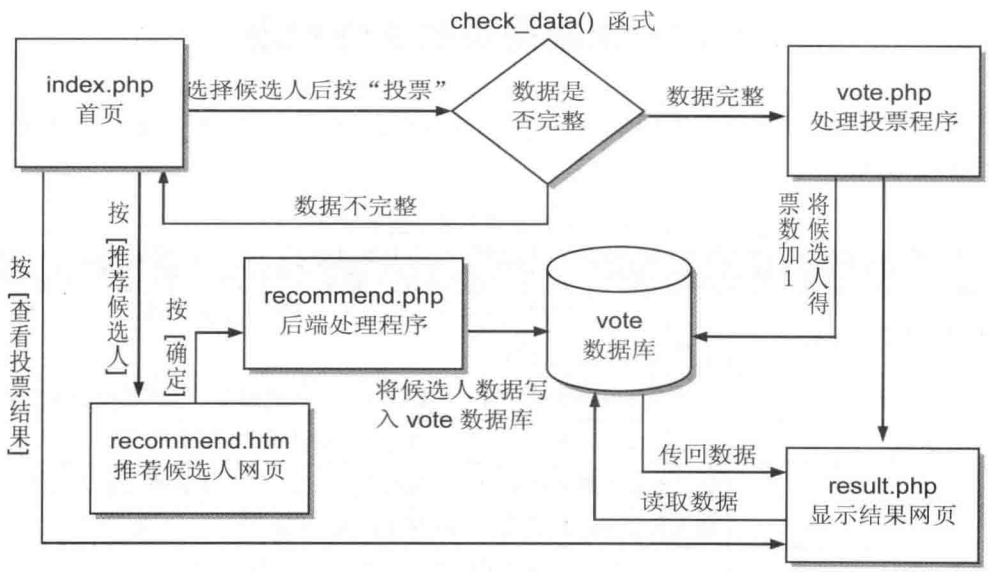


图 20-4

20.4 必须具备的背景知识

- 首先，您必须熟悉 HTML 语法或其他网页编辑软件，因为我们将用到许多表格。
- 其次，您必须了解表单的制作方式及如何读取表单数据，我们在第 8 章介绍过。
- 其三，基本的 JavaScript 语法，我们将使用它来验证身份证号码。
- 最后，您当然得熟悉 SQL 语法及如何访问数据库。

20.5　完整的程序代码清单

❖ index.php

这是在线投票系统的首页，浏览者可以在此投票，执行界面如前面的图 20-1 所示。

\ch20\index.php

```
001:<!doctype html>
002:<html>
003:  <head>
004:    <title>在线投票</title>
005:    <meta charset="utf-8">
006:    <script type="text/javascript">
007:      function check_data()
008:      {
009:        var id = document.myForm.id.value;
010:        var tab = "ABCDEFGHJKLMNPQRSTUVWXYZIO";
011:        var A1 = new Array（1,1,1,1,1,1,1,1,1,1,2,2,2,2,2,2,2,2,2,3,3,3,3,3,3 );
012:        var A2 = new Array（0,1,2,3,4,5,6,7,8,9,0,1,2,3,4,5,6,7,8,9,0,1,2,3,4,5 );
013:        var Mx = new Array（9,8,7,6,5,4,3,2,1,1);
014:
015:        //将身份证号码的英文字母转换为大写
016:        id = id.toUpperCase();
017:
018:        //验证身份证号码是否为 10 位
019:        if（id.length != 10)
020:        {
021:          alert("身份证号码共有 10 位");
022:          return false;
023:        }
024:
025:        //验证身份证号码的第一位是否为英文字母
026:        var i = tab.indexOf(id.charAt(0));
027:        if（i == -1)
028:        {
029:          alert("身份证号码第一位为英文字母");
030:          return false;
031:        }
032:
033:        //验证身份证号码高级规则
034:        var sum = A1[i]+ A2[i]* 9;
035:        var v;
```

```
036:          for（i = 1; i < 10; i++)
037:          {
038:             v = parseInt(id.charAt(i));
039:             if（isNaN(v))
040:             {
041:                alert("身份证号码末九位必须为数字");
042:                return false;
043:             }
044:
045:             sum = sum + v * Mx[i];
046:          }
047:
048:          if（sum % 10 != 0)
049:          {
050:             alert("不合法的身份证号码");
051:             return false;
052:          }
053:
054:          //此部份用来验证浏览者是否有选择候选人
055:          for（var i = 0;i < document.myForm.elements.length; i++)
056:          {
057:             var e = document.myForm.elements[i];
058:             if（e.name == "name" && e.checked == true)
059:             {
060:                var found = true;
061:                break;
062:             }
063:          }
064:
065:          if（found != true)
066:          {
067:             alert("您没有选择候选人");
068:             return false;
069:          }
070:
071:          myForm.submit();
072:       }
073:    </script>
074: </head>
075: <body bgcolor="#FFE6CC">
076:    <p align="center"><img src="title_1.jpg"></p>
077:    <form name="myForm" action="vote.php" method="post" >
078:       <table width="75%" align="center" border="2" bordercolor="#999999">
```

```
079:          <tr bgcolor="#0033CC">
080:            <td align="center"><font color="#FFFFFF">候选人</font></td>
081:            <td align="center"><font color="#FFFFFF">候选人介绍</font></td>
082:          </tr>
083:            <?php
084:              require_once("dbtools.inc.php");
085:
086:              //建立数据连接
087:              $link = create_connection();
088:              //执行 Select 语句获取候选人数据
089:              $sql = "SELECT * FROM candidate";
090:              $result = execute_sql($link, "vote", $sql);
091:
092:              while（$row = mysqli_fetch_object($result))
093:              {
094:                echo "<tr>";
095:                echo "<td bgcolor='#CCFFCC'> ";
096:                echo "<input type='radio' name='name'" .
097:                  "value='$row->name'>$row->name</td>";
098:                echo "<td bgcolor='#FFCCCC'>$row->introduction</td>";
099:                echo "</tr>";
100:              }
101:
102:              //关闭数据连接
103:              mysqli_close($link);
104:            ?>
105:          <tr bgcolor="#FFFF99">
106:            <td colspan="2" align="right">请输入您的身份证号码：
107:            <input type="text" name="id" size="10"></td>
108:          </tr>
109:        </table>
110:        <p align="center">
111:          <input type="button" value="投票" onClick="check_data()">
112:          <input type="reset" value="重新设置">
113:          <input type="button" value="推荐候选人"
114:            onclick="javascript:window.open('recommend.htm','_self')">
115:          <input type="button" value="查看投票结果"
116:            onclick="javascript:window.open('result.php','_self')">
117:        </p>
118:      </form>
119:    </body>
120:      </html>
```

- 007～072：check_data() 函数用来审核投票者输入的身份证号码是否正确，以及判断投票者是否选择了候选人，必须通过这两项验证，才能继续执行。
- 009：获取表单中 id 字段的数据，然后赋值给变量 id。
- 011～013：建立三个数组来存放验证身份证号码的数据。
- 016：调用 toUpperCase() 函数将变量 id 的英文字母转换成大写。
- 019 ～ 023：这个部分用来验证身份证号码是否等于 10 个字，若不等于 10 个字，就显示“身份证号码共有 10 位”。
- 026 ～ 031：检查身份证号码的第一个字是否等于大写的英文字母，若不是的话，就显示“身份证号码第一位为英文字母”。
- 034～052：身份证号码的命名规则除了要有 10 个字，首字为大写英文字母之外，还有一个非常重要的命名规则，就是身份证号码的第 2～10 个字必须为数字，第 039～043 即是验证此规则。

  此外，我们还必须套用身份证验证公式计算出一个数字，第 045 行的变量 sum 就用来存放这个数字，而且这个数字必须能被 10 整除，第 048～052 即是验证此规则，必须通过所有条件，才算是一个合法的身份证号码。
- 055～069：这个部分用来验证投票者是否选择了候选人。

❖ vote.php

当浏览者在 <index.php> 中按[投票]时会执行这个程序，它负责将被投票人的得票数加一。

\ch20\vote.php

```
01:<?php
02:   require_once("dbtools.inc.php");
03:   header("Content-type: text/html; charset=utf-8");
04:   $id = strtoupper($_POST["id"]);
05:   $name = $_POST["name"];
06:
07:   //建立数据连接
08:   $link = create_connection();
09:
10:   //检查身份证号码是否投过票
11:   $sql = "SELECT * FROM id_number Where id = '$id'";
12:   $result = execute_sql($link, "vote", $sql);
13:
14:   //若身份证号码已经投过票
15:   if（mysqli_num_rows($result) != 0)
16:   {
17:      //释放 $result 占用的内存
```

```
18:     mysqli_free_result($result);
19:
20:     //显示信息请求用户更换账号名称
21:     echo "<script type='text/javascript'>";
22:     echo "alert('您已经参加过本次活动了');";
23:     echo "history.back();";
24:     echo "</script>";
25:     exit();
26:  }
27:
28:  //若身份证号码没有投过票
29:  else
30:  {
31:     //释放 $result 占用的内存
32:     mysqli_free_result($result);
33:
34:     /* 执行 INSERT INTO 语句，将此浏览者的身份证号码加
35:        入 id_number 数据表，表示此身份证号码已投票 */
36:     $sql = "INSERT INTO id_number（id) VALUES（'$id')";
37:     $result = execute_sql($link, "vote", $sql);
38:     //使用 UPDATE 语句将票数 + 1
39:     $sql = "UPDATE candidate SET score = score + 1 WHERE name = '$name'";
40:     $result = execute_sql($link, "vote", $sql);
41:  }
42:
43:  //关闭数据连接
44:  mysqli_close($link);
45:  //将用户导向到`result.php 网页
46:  header("location:result.php");
47:?>
```

- 04～05：获取 <index.php> 表单中的身份证号码（id）及候选人（name），然后调用 strtoupper() 函数将变量 $id 的英文字母转换成大写。
- 11～41：使用身份证号码来验证投票人是否投过了票，变量 id 用来与 vote 数据库内 id_number 数据表的 id 字段做对比，若有一样的身份证号码，表示已经投过票，就不能再投票；若没有一样的身份证号码，第 36 行、第 37 行将变量 id 的值写入 vote 数据库内 id_number 数据表的 id 字段，这个方法可以防止同一人投多次票，以防投票结果被灌水；第 39 行、第 40 行用来将被投票者的票数加一，并更新到 candidate 数据库。
- 46：将浏览者重定向到投票结果网页 <result.php>。

❖ result.php

这是用来显示投票结果的网页，执行界面如前面的图 20-2 所示。

\ch20\result.php

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <title>投票结果</title>
05:    <meta charset="utf-8">
06:  </head>
07:  <body>
08:    <p align='center'><img src='title_3.jpg'></p>
09:    <table align='center' width='600' border='1' bordercolor='#990033'>
10:      <tr bgcolor='#CC66FF'>
11:        <td align='center'><b><font color='#FFFFFF'>候选人</font></b></td>
12:        <td align='center'><b><font color='#FFFFFF'>得票数</font></b></td>
13:        <td align='center'><b><font color='#FFFFFF'>得票百分比</font></b></td>
14:        <td align='center'><b><font color='#FFFFFF'>直方图</font></b></td>
15:      </tr>
16:      <tr bgcolor='#FFCCFF'>
17:      <?php
18:        require_once("dbtools.inc.php");
19:
20:        //建立数据连接
21:        $link = create_connection();
22:
23:        //执行 SELECT 语句来获取候选人数据
24:        $sql = "SELECT * FROM candidate";
25:        $result = execute_sql($link, "vote", $sql);
26:
27:        //计算总记录数
28:        $total_records = mysqli_num_rows($result);
29:
30:        //计算总票数
31:          $total_score = 0;
32:        while（$row = mysqli_fetch_object($result))
33:          $total_score += $row->score;
34:
35:        /* 当前记录指针已经在数据表尾端，我们使用
36:            mysqli_data_seek() 函数将记录指针移至第 1 笔记录 */
37:        mysqli_data_seek($result, 0);
38:
```

```
39:         //列出所有候选人得票数据
40:         for（$j = 0; $j < $total_records; $j++)
41:         {
42:           //获取候选人数据
43:           $row = mysqli_fetch_assoc($result);
44:
45:           //计算候选人得票百分比
46:           $percent = round($row["score"]/ $total_score, 4) * 100;
47:
48:           //显示候选人各字段的数据
49:           echo "<tr>";
50:           echo "<td align='center'>" . $row["name"]. "</td>";
51:           echo "<td align='center'>" . $row['score']. "</td>";
52:           echo "<td align='center'>" . $percent . "%</td>";
53:           echo "<td height='35'><img src='bar.jpg' width='" .
54:             $percent * 3 . "' height='20'></tr>";
55:           echo "</tr>";
56:         }
57:
58:         //释放资源及关闭数据连接
59:         mysqli_free_result($result);
60:         mysqli_close($link);
61:       ?>
62:       <tr bgcolor='#FFCCFF'>
63:         <td align='center'>总计</td>
64:         <td align='center'><?php echo $total_score ?></td>
65:         <td align='center'>100%</td>
66:         <td><img src='bar.jpg' width='300' height='20'></td>
67:       </tr>
68:     </table>
69:     <p align='center'><a href='index.php'>回首页</a></p>
70:   </body>
71:</html>
```

- 31～33：计算总投票数。
- 37：当前记录指针已经在数据表尾端，我们使用 mysqli_data_seek() 函数将记录指针移至第 1 笔记录。
- 40～56：将每个候选人的名称、得票数、得票百分比及直方图显示出来。

❖ recommand.htm

这是推荐候选人的主程序，执行界面如前面的图 20-3 所示。

\ch20\recommand.htm

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>推荐候选人</title>
  </head>
  <body>
    <p align="center"><img src="title_2.jpg"></p>
    <form action="recommend.php" method="post" >
      <table width="650" align="center" border="1">
        <tr bgcolor="#FFFFCC">
          <td align="right">候选人姓名：</td>
          <td><input type="text" name="name" size="10"></td>
        </tr>
        <tr>
          <td bgcolor="#CCFFCC" align="right">候选人简介：</td>
          <td bgcolor="#CCFFCC">
            <textarea name="introduction" cols="40" rows="3"></textarea>
          </td>
        </tr>
      </table>
      <br>
      <div align="center">
        <input type="submit" value="确定">
        <input type="reset" value="重新输入">
      </div>
    </form>
    <p align="center"><a href="javascript:history.back()">回首页</a></p>
  </body>
</html>
```

recommand.php

这是推荐候选人网页的后端处理程序，它会检查您所推荐的人是否已经在候选人名单中，若候选人已经存在，就不用推荐了。

\ch20\recommand.php

```
<?php
  require_once("dbtools.inc.php");
  header("Content-type: text/html; charset=utf-8");
```

```
//获取表单数据
$name = $_POST["name"];
$introduction = $_POST["introduction"];

//建立数据连接
$link = create_connection();

//执行 SELECT 语句来获取候选人数据
$sql = "SELECT * FROM candidate WHERE name='$name'";
$result = execute_sql($link, "vote", $sql);

//检查被推荐人是否有在候选人列表中
if（mysqli_num_rows($result) == 0)
{
  //释放资源
  mysqli_free_result($result);

  //将候选人数据加入数据库
  $sql = "INSERT INTO candidate（name , introduction , score)
          VALUES（'$name', '$introduction', 0)";
  $result = execute_sql($link, "vote", $sql);

  //将用户导向到在线投票首页
  header("location:index.php");
}
else
{
  //显示被推荐人已经是候选人的信息
  echo "<p align='center'>您推荐的人已经在候选人名单中了，不需要再推荐。</p>";
  echo "<p align='center'><a href ='javascript:history.back()'>回上一页</a>";
  echo "      <a href='index.php'>回首页</a></p>";
}

//关闭数据连接
mysqli_close($link);
?>
```

第 21 章

购物车

21.1　认识购物车

在线购物是目前非常流行的趋势，您无须准备大笔的创业资金就可以创建网站，开设网络商店。以下各图是我们即将要制作的购物车，首先，您得在图 21-1 中输入名字以登录网站，接着，您会进入图 21-2 所示的界面。

图 21-1

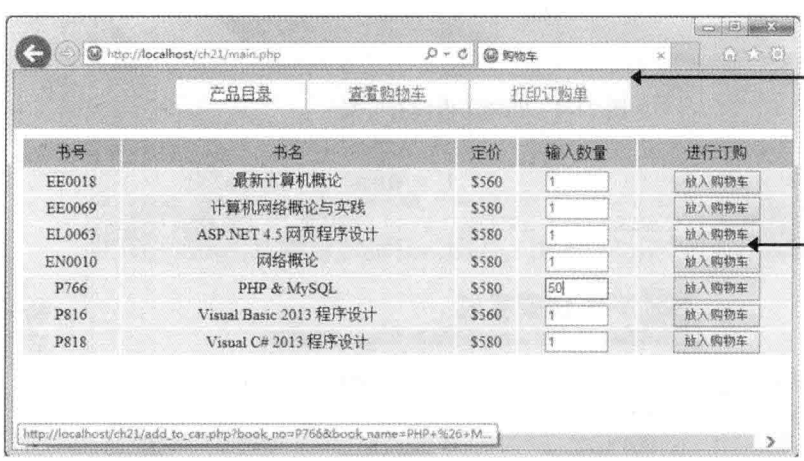


图 21-2

若您在图 21-3 的上方框架中单击“查看购物车”超链接，下方框架会显示您放入购物车的所有产品，图 21-4 是购物车内没有产品的情况，图 21-5 是购物车内有产品的情况。

3.产品已放入购物车

图 21-3

图 21-4　购物车内没有产品

单击此钮可以退回所有商品

输入数量后按“修改”，可以变更订购数量，若数量为0或空白，表示取消订购该产品

图 21-5　购物车内有产品

若您在图 21-5 的上方框架中单击“打印订购单”超链接，下方框架会显示如图 21-6 所示的订购单，请详细检查上面的订单明细，不正确的话，可以回到购物车去做修改，正确的话，就把它打印出来，然后按照指示使用邮政划拨或信用卡方式进行订购。

图 21-6

21.2　组成网页的文件列表

这个购物车网站程序存放在下载资源的 \samples\ch21\ 文件夹中，它用到如表 21-1 中所示文件。

表 21-1　购物车网站程序用到的文件

文件名	说明
fig1.jpg	这个 JPEG 图像文件是“快乐书城”标题图像
bg1.jpg	这个 JPEG 图像文件是打印订购单界面（图 21-6）的背景图像
index.htm	这个网页为购物车第一个界面，如图 21-1 所示，负责提供表单让浏览者输入名字，然后按[登录]，就会调用表单处理程序 <main.php>

（续表）

文件名	说明
main.php	这个网页为购物车第二个界面的框架网页，如图 21-2 所示，负责指定上方框架的高度为 60 像素，来源网页为 <show_link.htm>，下方框架的来源网页为 <catalog.php>。
show_link.htm	这个网页是 <main.php> 的上方框架网页，用来显示“产品目录”、“查看购物车”、“打印订购单”三个超链接，分别链接至 <catalog.php>、<shopping_car.php>、<print_order.php>
catalog.php	这个网页是 <main.php> 预设的下方框架网页，用来读取并显示 store 数据库内 product_list 数据表的所有记录，同时浏览者可以在这个网页中输入要订购的产品数量，然后按[放入购物车]，就会调用 <add_to_car.php>，将指定的产品放入购物车（即写入 Cookie）
add_to_car.php	在浏览者输入订购数量并按[放入购物车]后会调用此程序，将指定的产品及数量写入 Cookie，并显示成功信息，如图 21-3 所示
shopping_car.php	这个网页的用途是从 Cookie 读取浏览者放入购物车的产品并显示出来，如图 21-4、21-5 所示，同时各个产品后面有一个[修改]按钮，浏览者只要在网页上输入数量，然后按[修改]，就可以变更订购数量，若数量为 0 或空白，表示取消订购该产品。
change.php	当浏览者在购物车内按[修改]变更订购数量时，会调用这个程序在购物车内修改指定产品的数量
delete_order.php	当浏览者在购物车内按[退回所有商品]时，会调用这个程序清除所有购物车的数据
print_order.php	这个网页会根据购物车内的产品显示订购单，让浏览者打印出来，以进行邮政划拨订购或信用卡订购（图 21-6）
store 数据库	这个数据库包含 product_list 数据表

在这个购物车中，我们使用了名称为 store 的数据库，里面包含一个 product_list 数据表，以存储产品数据，其字段结构如下。

字段名	数据类型	长度	主键	说明
book_no	VARCHAR	20	☑	书号字段
book_name	VARCHAR	30	☐	书名字段
price	INT	-	☐	定价字段

您可以自己创建数据库或导入本书为您准备的数据库备份文件（位于下载资源的\samples\database\store.sql），有关如何导入 MySQL 数据库，可以参考第 11.3.7 小节的说明。

21.3 网页的运行流程

首先，执行 <index.htm>，请求浏览者输入名字，然后按[登录]（图 21-1）；接着，调用

表单处理程序 <main.php>，这是构成购物车第二个界面的框架网页（图 21-2），负责指定上方框架的高度为 60 像素，来源网页为 <show_link.htm>，下方框架的来源网页为 <catalog.php>；此外，<main.php> 会使用 Cookie 记录浏览者的名字。

当浏览者单击“产品目录”超链接时，会执行 <catalog.php> 列出所有产品；当浏览者在产品目录中按[放入购物车]时，会执行 <add_to_car.php> 将浏览者选择的产品及数量加入购物车（图 21-3，在加入购物车之前，会先检查购物车内是否有相同的产品，有的话，就将购物车内的数量加上当前所选购的数量，没有的话，才加入新的一笔数据。

当浏览者单击“查看购物车”或“打印订购单”超链接时，会分别执行 <shopping_car.php>、<print_order.php>，这两个网页均会读取 Cookie 的数据，然后将购物车内的数据显示出来（图 21-4、21-5、21-6）。

当浏览者在 <shopping_car.php> 按[修改]时，会执行 <change.php> 在购物车内修改指定产品的数量；当浏览者按[退回所有商品]时，会执行 <delete_order.php>，清除所有购物车的数据。

网页运行流程如图 21-7 所示。

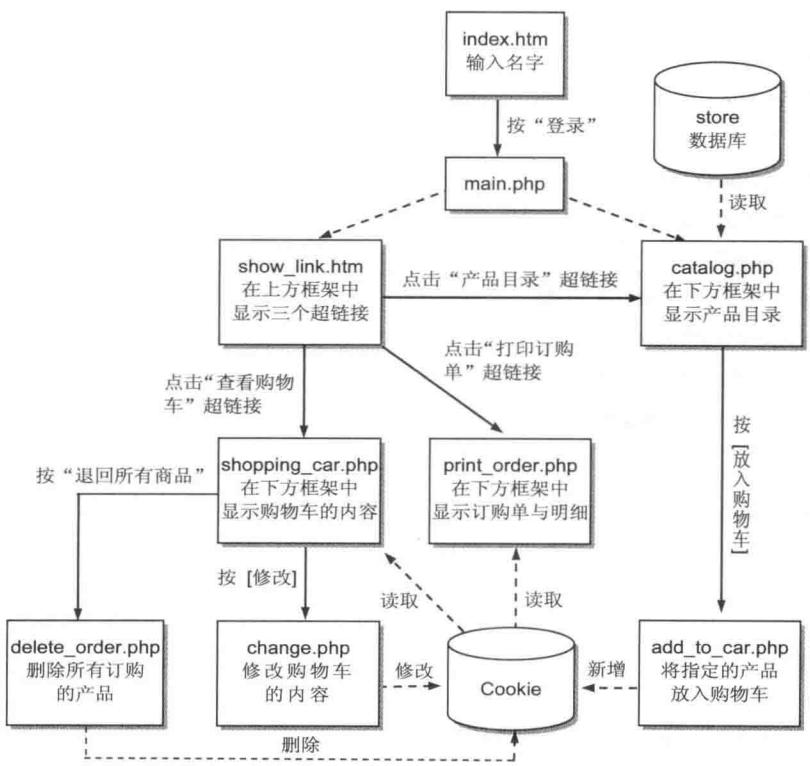


图 21-7

21.4　您必须具备的背景知识

- 首先，您必须熟悉 HTML 语法或其他网页编辑软件。

- 其次，您必须了解表单的制作方式及如何读取表单数据。
- 其三，您必须懂得如何存取 Cookie。
- 最后，您当然得熟悉 SQL 语法及如何访问数据库。

21.5 完整的程序代码清单

index.htm

这个网页为购物车第一个界面，如前面的图 21-1 所示，负责提供表单让浏览者输入名字，然后按“登录”，就会调用表单处理程序 <main.php>。

\ch21\index.htm

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>购物车</title>
  </head>
  <body bgcolor="lightyellow">
    <form method="post" action="main.php">
      <p align="center">
        <img src="fig1.jpg"><br><br>请输入您的名字：
        <input type="text" name="name" size="20">
        <input type="submit" value="登录">
        <input type="reset" value="重新输入">
      </P>
    </form>
  </body>
</html>
```

main.php

这个网页为购物车第二个界面的框架网页，如前面的图 21-2 所示，负责指定上方框架的高度为 60 像素，来源网页为 <show_link.htm>，下方框架的来源网页为 <catalog.php>。请注意第 2 行，它的作用是将浏览者输入的名字记录在 Cookie 变量 name 中。

\ch21\main.php

```
<?php
  setcookie("name", $_POST["name"]);
```

```
?>
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
    <title>购物车</title>
  </head>
  <frameset rows="60, *" border="0">
    <frame name="top" noresize scrolling="no" src="show_link.htm">
    <frame name="bottom" noresize src="catalog.php">
  </frameset>
</html>
```

show_link.htm

这个网页是 <main.php> 的上方框架网页，用来显示“产品目录”、“查看购物车”、“打印订购单”三个超链接，分别链接至 <catalog.php>、<shopping_car.php>、<print_order.php>。

\ch21\show_link.htm

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body bgcolor="#9CCDCD">
    <table align="center" width="60%" border="0">
      <tr height="30" bgcolor="#EDF5F5" align="center">
        <td><a href="catalog.php" target="bottom">产品目录</a></td>
        <td><a href="shopping_car.php" target="bottom">查看购物车</a></td>
        <td><a href="print_order.php" target="bottom">打印订购单</a></td>
      </tr>
    </table>
  </body>
</html>
```

catalog.php

这个网页是 <main.php> 预设的下方框架网页，用来读取并显示 store 数据库内 product_list 数据表的所有记录，同时浏览者可以在这个网页输入要订购的产品数量，然后按 [放入购物车]，就会调用 <add_to_car.php>，将指定的产品放入购物车（即写入 Cookie）。

\ch21\catalog.php

```
<!doctype html>
<html>
  <head>
    <meta charset="utf-8">
  </head>
  <body bgcolor="lightyellow">
    <table border="0" align="center" width="800" cellspacing="2">
      <tr bgcolor="#BABA76" height="30" align="center">
        <td>书号</td>
        <td>书名</td>
        <td>定价</td>
        <td>输入数量</td>
        <td>进行订购</td>
      </tr>
        <?php
          require_once("dbtools.inc.php");

          //建立数据连接
          $link = create_connection();

          //筛选出所有产品数据
          $sql = "SELECT * FROM product_list";
          $result = execute_sql($link, "store", $sql);

          //计算总记录数
          $total_records = mysqli_num_rows($result);

          //列出所有产品数据
          for（$i = 0; $i < $total_records; $i++)
          {
            //获取产品数据
            $row = mysqli_fetch_assoc($result);

            //显示产品各字段的数据
            echo "<form method='post' action='add_to_car.php?book_no=" .
              $row["book_no"]. "&book_name=" . urlencode($row["book_name"]) .
              "&price=" . $row["price"]. "'>";
            echo "<tr align='center' bgcolor='#EDEAB1'>";
            echo "<td>" . $row["book_no"]. "</td>";
            echo "<td>" . $row["book_name"]. "</td>";
            echo "<td>$" . $row["price"]. "</td>";
```

```
42:                 echo "<td><input type='text' name='quantity' size='5' value='1'></td>";
43:                 echo "<td><input type='submit' value='放入购物车'></td>";
44:                 echo "</tr>";
45:                 echo "</form>";
46:             }
47:
48:             //释放资源及关闭数据连接
49:             mysqli_free_result($result);
50:             mysqli_close($link);
51:         ?>
52:     </table>
53: </body>
54:</html>
```

- 15～51：用来读取 product_list 数据表的所有记录，其中第 35～45 行用来插入表单和单行文本框，而且在调用表单处理程序 <add_to_car.php> 的同时会传送 book_no、book_name、price 三个参数，分别表示书号、书名及定价；第 39～41 用来显示所有字段的数据。

❖ add_to_car.php

在浏览者输入订购数量并按[放入购物车]后，会调用这个程序，将指定的产品及数量写入 Cookie，然后显示成功信息，如前面的图 21-3 所示。

\ch21\add_to_car.php

```
01:<?php
02:  //获取表单数据
03:  $book_no = $_GET["book_no"];
04:  $book_name = $_GET["book_name"];
05:  $price = $_GET["price"];
06:  $quantity = $_POST["quantity"];
07:  if（empty($quantity)) $quantity = 1;
08:
09:  //若购物车没有任何项目，就直接加入产品数据
10:  if（empty($_COOKIE["book_no_list"]))
11:  {
12:     setcookie("book_no_list", $book_no);
13:     setcookie("book_name_list", $book_name);
14:     setcookie("price_list", $price);
15:     setcookie("quantity_list", $quantity);
16:  }
17:  else
```

```
18:  {
19:     //获取购物车数据
20:     $book_no_array = explode(",", $_COOKIE["book_no_list"]);
21:     $book_name_array = explode(",", $_COOKIE["book_name_list"]);
22:     $price_array = explode(",", $_COOKIE["price_list"]);
23:     $quantity_array = explode(",", $_COOKIE["quantity_list"]);
24:
25:     //判断选择的物品有在购物车中
26:     if（in_array($book_no, $book_no_array))
27:     {
28:        //若选择的物品已经在购物车中，变更物品数量即可
29:        $key = array_search($book_no, $book_no_array);
30:        $quantity_array[$key]+= $quantity;
31:     }
32:     else
33:     {
34:        //若选择的物品没有在购物车中，就将物品数据加入购物车
35:        $book_no_array[]= $book_no;
36:        $book_name_array[]= $book_name;
37:        $price_array[]= $price;
38:        $quantity_array[]= $quantity;
39:     }
40:
41:     //存储购物车数据
42:     setcookie("book_no_list", implode(",", $book_no_array));
43:     setcookie("book_name_list", implode(",", $book_name_array));
44:     setcookie("price_list", implode(",", $price_array));
45:     setcookie("quantity_list", implode(",", $quantity_array));
46:  }
47:?>
48:<!doctype html>
49:<html>
50:  <head>
51:     <meta charset="utf-8">
52:     </head>
53:  <body bgcolor="lightyellow">
54:     <p align="center"><img src="fig1.jpg"></p>
55:     <p align="center">您所选取的商品及数量已成功地放入购物车！ </p>
56:     <p align="center"><a href="catalog.php">继续购物</a></p>
57:  </body>
58:</html>
```

- 在解说这个网页之前，我们必须先让您了解如何保存购物车的数据，此处是以 Cookie

中来存放数据，假设浏览者陈小贞购买了下列几个产品：

书号	书名	定价	数量
P766	PHP & MySQL	$580	50
P816	Visual Basic 2013 程序设计	$560	10
P818	Visual C# 2013 程序设计	$580	20

那么存放在 Cookie 中的数据将如下所示，您可以看到，我们是以逗号（,）区分不同的产品：

键名	数值
book_no_list	P766,P816,P818
book_name_list	PHP & MySQL,Visual Basic 2013 程序设计,Visual C# 2013 程序设计
price_list	580,560,580
quantity_list	50,10,20

- 03 ~ 07：用来获取表单数据，由于 book_no、book_name、price 三个参数以 GET 的方式传送出来，所以必须通过变量 $_GET 获取其值，而第 07 行表示当浏览者将订购数量保持空白但又单击[放入购物车]时，产品数量会自动设置为 1，表示订购一个单位。
- 10 ~ 16：若 Cookie 变量 book_no_list 为空白（即购物车是空的），就直接将浏览者的订购数据加入 Cookie（即购物车）。
- 19 ~ 45：若 Cookie 变量 book_no_list 不是空白（即购物车不是空的），所需要做的处理。
- 20 ~ 23：用来读取购物车的数据并存放在字符串数组中，由于我们是以逗号(,) 区分不同的产品，因此，$_COOKIE["book_no_list"]的返回值会类似于"P766,P816,P818"的形式。

Explode（*separator*, *string*）函数用来分割字符串，它会以参数 *separator* 作为分隔符，将参数 *string* 分割为多个字符串，然后返回字符串数组，数组的每个元素代表一个分离出来的字符串。

就前面的例子来说，第 20 行的 explode(",", $_COOKIE["book_no_list"]) 会返回一个包含 3 个元素的数组，然后将数组赋值给变量 book_no_array，其中 $book_no_array[0]为 "P766"、$book_no_array[1]为 "P816"、$book_no_array[2]为 "P818"。

- 26 ~ 39：第 26 行使用 in_array() 函数判断浏览者所选购的产品是否已经在购物车内，若已经在购物车内，就执行第 28 ~ 30 行，否则执行第 34 ~ 38 行。
- 28 ~ 30：第 29 行用来获取浏览者选购的产品在购物车内的键值，然后将键值存放在变量 key 中；第 30 行用来变更购物车的数量。
- 34 ~ 38：若浏览者所选购的产品不在购物车内，就会执行这些程序代码，将产品数

据加入购物车。

- 42～45：$book_no_array、$book_name_array、$price_array、$quantity_array 4 个数组变量用来暂存购物车的数据，而第 42～45 行用来将购物车数据存回 Cookie，其中 implode(*glue*, *pieces*) 函数刚好与 explode() 函数相反，它可以将参数 *pieces* 指定之数组的数据，以参数 *glue* 指定的分隔符转换成单一字符串，因此，存放在 Cookie 中的数据又变成以逗号（,）分隔的字符串。

❖ shopping_car.php

这个网页的用途是从 Cookie 读取浏览者放入购物车的产品并显示出来，如前面的图 21-4、21-5 所示，同时各个产品后面有一个[修改]按钮，浏览者只要在网页上输入数量，然后按[修改]，就会调用 <change.php> 变更订购数量，若数量为 0 或空白，表示取消订购该产品。

此外，网页最下方有一个[退回所有商品]按钮，只要单击这个按钮，就会调用 <delete_order.php> 将所有产品从购物车内删除。

\ch21\shopping_car.php

```
01:<!doctype html>
02:<html>
03:  <head>
04:    <meta charset="utf-8">
05:  </head>
06:  <body bgcolor="LightYellow">
07:    <p align="center"><img src='fig1.jpg'></p>
08:    <table border="0" align="center" width="800">
09:      <tr bgcolor="#ACACFF" height="30" align="center">
10:        <td>书号</td>
11:        <td>书名</td>
12:        <td>定价</td>
13:        <td>数量</td>
14:        <td>小计</td>
15:        <td>变更数量</td>
16:      </tr>
17:        <?php
18:          //若购物车是空的，就显示 "当前购物车内没有任何商品及数量"
19:          if（empty($_COOKIE["book_no_list"]))
20:          {
21:            echo "<tr align='center'>";
22:            echo "<td colspan='6'>当前购物车内没有任何商品及数量！</td>";
23:            echo "</tr>";
24:          }
25:          else
```

```
26:         {
27:             //获取购物车数据
28:             $book_no_array = explode(",", $_COOKIE["book_no_list"]);
29:             $book_name_array = explode(",", $_COOKIE["book_name_list"]);
30:             $price_array = explode(",", $_COOKIE["price_list"]);
31:             $quantity_array = explode(",", $_COOKIE["quantity_list"]);
32:
33:             //显示购物车内容
34:             $total = 0;
35:             for（$i = 0; $i < count($book_no_array); $i++)
36:             {
37:                 //计算小计
38:                 $sub_total = $price_array[$i]* $quantity_array[$i];
39:
40:                 //计算总计
41:                 $total += $sub_total;
42:
43:                 //显示各字段数据
44:                 echo "<form method='post' action='change.php?book_no=" .
45:                     $book_no_array[$i]. "'>";
46:                 echo "<tr bgcolor='#EDEAB1'>";
47:                 echo "<td align='center'>" . $book_no_array[$i]. "</td>";
48:                 echo "<td align='center'>" . $book_name_array[$i]. "</td>";
49:                 echo "<td align='center'>$" . $price_array[$i]. "</td>";
50:                 echo "<td align='center'><input type='text' name='quantity' value='" .
51:                     $quantity_array[$i]. "' size='5'></td>";
52:                 echo "<td align='center'>$" . $sub_total . "</td>";
53:                 echo "<td align='center'><input type='submit' value='修改'></td>";
54:                 echo "</tr>";
55:                 echo "</form>";
56:             }
57:
58:             echo "<tr align='right' bgcolor='#EDEAB1'>";
59:             echo "<td colspan='6'>总金额 = " . $total . "</td>";
60:             echo "</tr>";
61:             echo "<tr align='center'>";
62:             echo "<td colspan='6'>" . "<br><input type='button' value='退回所有商品'
63:                 onClick=\"javascript:window.open('delete_order.php','_self')\">";
64:             echo "</td>";
65:             echo "</tr>";
66:         }
67:     ?>
68: </table>
```

```
69:  </body>
70:</html>
```

- 17～67：用来从 Cookie 读取购物车数据并显示出来。
- 19 ～ 24：若购物车是空的，就显示字符串“当前购物车内没有任何商品及数量!”，执行界面如前面的图 21-4 所示。
- 27～65：若购物车不是空的，就显示购物车的内容，执行界面如前面的图 21-5 所示。
- 28～31：获取购物车的内容。
- 34～56：插入表单和单行文本框，而且在调用表单处理程序 <change.php> 的同时会传送 book_no 参数，表示书号；第 47 ～ 52 用来显示所有字段的数据，其中第 52 行的变量 sub_total 是我们在第 38 行计算出来的单一产品小计金额。
- 53：插入一个 submit 按钮，上面的文字为“修改”。
- 58～60：显示第 41 行计算的购物车总金额。
- 62～63：插入[退回所有商品]按钮，链接至 <delete_order.php> 程序，以删除整个购物车的数据。

❖ change.php

当浏览者在 <shopping_car.php>中输入数量并按[修改]时，就会执行 <change.php>，在购物车内修改指定产品的数量，若数量为 0 或空白，表示取消订购该产品。

\ch21\change.php

```
01:<?php
02:  //获取表单数据
03:  $book_no = $_GET["book_no"];
04:  $quantity = $_POST["quantity"];
05:
06:  //获取购物车数据
07:  $book_no_array = explode(",", $_COOKIE["book_no_list"]);
08:  $book_name_array = explode(",", $_COOKIE["book_name_list"]);
09:  $price_array = explode(",", $_COOKIE["price_list"]);
10:  $quantity_array = explode(",", $_COOKIE["quantity_list"]);
11:
12:  $key = array_search($book_no, $book_no_array);
13:
14:  //若数量等于 0，就将该产品从购物车中删除
15:  if（$quantity == 0 || empty($quantity))
16:  {
17:     unset($book_no_array[$key]);
18:     unset($book_name_array[$key]);
19:     unset($price_array[$key]);
```

```
20:     unset($quantity_array[$key]);
21:   }
22:   else
23:   {
24:     //若数量不等于 0，就更新数量
25:     $quantity_array[$key]= $quantity;
26:   }
27:
28:   //存储购物车数据
29:   setcookie("book_no_list", implode(",", $book_no_array));
30:   setcookie("book_name_list", implode(",", $book_name_array));
31:   setcookie("price_list", implode(",", $price_array));
32:   setcookie("quantity_list", implode(",", $quantity_array));
33:
34:   //导向到 shopping_car.php 网页
35:   header("location:shopping_car.php");
36:?>
```

- 03 ~ 04：获取表单数据，由于 book_no 参数是以 GET 的方式传送出来的，所以必须通过变量 $_GET 来获取其值。
- 07 ~ 10：获取购物车数据，我们已经在本章前面解释过 explode() 函数的用途。
- 12：获取浏览者要修改的产品在购物车内的键值，然后将键值存放在变量 key 中。
- 15 ~ 21：若浏览者输入的数量是 0 或空白，表示要删除该产品，第 17 ~ 20 行使用 unset() 函数将该产品从数组中删除。
- 23 ~ 26：若浏览者输入的数量不是 0 或空白，就直接更改订购数量。
- 29 ~ 32：$book_no_array、$book_name_array、$price_array、$quantity_array 数组变量用来暂存购物车数据，第 29 ~ 32 行用来将购物车数据存回 Cookie，我们已经在本章前面解释过 implode() 函数的用途。
- 35：将浏览者导向到 <shopping_car.php>。

delete_order.php

当浏览者在购物车内按[退回所有商品]时，会调用这个程序清除所有购物车的数据，然后将浏览者导向到 <shopping_car.php>。

\ch21\delete_order.php

```
<?php
  //清除购物车数据
  setcookie("book_no_list", "");
  setcookie("book_name_list", "");
  setcookie("price_list", "");
```

```
    setcookie("quantity_list", "");

    //导向到 shopping_car.php 网页
    header("location:shopping_car.php");
?>
```

❖ print_order.php

这个网页会根据购物车内的产品显示订购单，让浏览者打印出来，以进行邮政划拨订购或信用卡订购（如前面的图 21-6）所示。基本上，这个网页虽然冗长，但却没有使用特殊的程序技巧，最费时的就是调整订购单的格式，好让界面看起来整齐美观，因此，您可以直接套用，也可以自行设计订购单的格式。

\ch21\print_order.php

```
<?php
    //若购物车是空的，就显示 "你尚未选购任何产品"
    if（empty($_COOKIE["book_no_list"]))
    {
        echo "<script type='text/javascript'>";
        echo "alert('你尚未选购任何产品');";
        echo "history.back();";
        echo "</script>";
    }
?>
<!doctype html>
<html>
    <head>
        <meta charset="utf-8">
    </head>
    <body background="bg1.jpg">
        <h3>注意事项</h3>
        <ol type="1">
            <li>
                订阅方法一：请填好信用卡专用订阅单后装订，免贴邮票，
                直接投邮即可，亦可发送传真至 02-23695588。
            </li>
            <li>
                订阅方法二：请利用邮局划拨单，填好姓名、户名、杂志、起讫月
                份、电话，直接至邮局划拨付款。账号：12345678 户名：快乐书城。
            </li>
```

```
    <li>
      寄书与补书：您将于邮件寄出之后的 10 日内收到书籍，若没有收到
      ，请来电咨询 02-23695599。
    </li>
</ol>
<hr>
<table border="1" bgcolor="white" rules="cols" align="center" cellpadding="5">
<tr height="25">
    <td colspan="4" align="Center" bgcolor="#CCCC00">个人数据</td>
</tr>
<tr height="25">
    <td colspan="4">姓名：<u><?php echo $_COOKIE["name"]?>
      <?php for（$i = 0; $i <= 100 - 2* strlen($_COOKIE["name"]); $i++) echo " "; ?></u>
    </td>
</tr>
<tr height="25">
    <td colspan="4">电话：
      <u><?php for（$i = 0; $i <= 100; $i++) echo " "; ?></u>
    </td>
</tr>
<tr height="25">
    <td colspan="4">地址：
      <u><?php for（$i = 0; $i <= 100; $i++) echo " "; ?></u>
    </td>
</tr>
<tr height="25">
    <td colspan="4">
      邮寄方式：□国内限时    □国内挂号（另加 20 元邮资)
    </td>
</tr>
<tr height="25">
    <td colspan="4">
      付款方式：□JCB CARD   □VISA CARD   
        □MASTER CARD
    </td>
</tr>
<tr height="25">
    <td colspan="4">
      信用卡卡号：<u><?php for（$i = 0; $i <= 89; $i++) echo " "; ?></u>
    </td>
</tr>
<tr height="25">
```

```
    <td colspan="4">
      有效日期：<u>公元<?php for（$i = 0; $i <= 85; $i++) echo " "; ?></u>
    </td>
  </tr>
  <tr height="25">
    <td colspan="4">
      签名(与信用卡签名相同)：<u><?php for（$i = 0; $i <= 66; $i++) echo " "; ?></u>
    </td>
  </tr>
  <tr height="25">
    <td colspan="4">
      支付总金额：<u><?php for（$i = 0; $i <= 89; $i++) echo " "; ?></u>
    </td>
  </tr>      <tr height="25">
    <td colspan="4">
      开立发票：□二联式    □三联式
    </td>
  </tr>
  <tr height="25">
    <td colspan="4">
      发票地址：<u><?php for（$i = 0; $i <= 93; $i++) echo " "; ?></u>
    </td>
  </tr>
  <tr height="25">
    <td colspan="4">
      统一编号：<u><?php for（$i = 0; $i <= 93; $i++) echo " "; ?></u>
    </td>
  </tr>
  <tr height="25">
    <td colspan="4" align="center" bgcolor="#CCCC00">订单明细</td>
  </tr>
  <tr height="25" align="center" bgcolor="FFFF99">
    <td>书名</td>
    <td>定价</td>
    <td>数量</td>
    <td>小计</td>
  </tr>
    <?php
      //获取购物车数据
      $book_name_array = explode(",", $_COOKIE["book_name_list"]);
      $price_array = explode(",", $_COOKIE["price_list"]);
      $quantity_array = explode(",", $_COOKIE["quantity_list"]);
      //显示购物车内容
```

```
        $total = 0;
        for（$i = 0; $i < count($book_name_array); $i++)
        {
          //计算小计
          $sub_total = $price_array[$i]* $quantity_array[$i];

          //计算总计
          $total += $sub_total;

          //显示各字段数据
          echo "<tr>";
          echo "<td align='center'>" . $book_name_array[$i]. "</td>";
          echo "<td align='center'>$" . $price_array[$i]. "</td>";
          echo "<td align='center'>" . $quantity_array[$i]. "</td>";
          echo "<td align='center'>$" . $sub_total . "</td>";
          echo "</tr>";
        }
        echo "<tr align='right' bgcolor='#CCCC00'>";
        echo "<td colspan='4'>总金额 = " . $total . "</td>";
        echo "</tr>";
      ?>
    </table>
  </body>
</html>
```

第 22 章
网络相册

22.1　认识网络相册

随着数字相机的普及，网络相册（又称为电子相册）已经成为目前非常流行的应用，以下各图是本章即将要制作的网络相册。

图 22-1

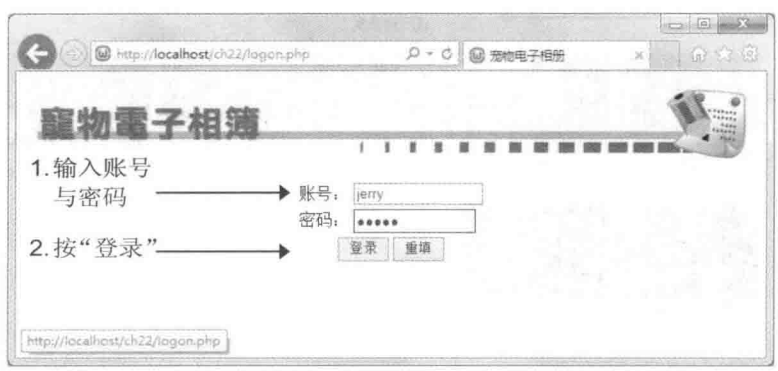


图 22-2


http://localhost/ch22/index.php
宠物电子相册
寵物電子相簿
9 Albums
元元精选集 1
10 Pictures
编辑 删除
元元精选集 2
7 Pictures
编辑 删除
小绵羊丑小鸭篇
11 Pictures
编辑 删除
小绵羊天鹅篇
6 Pictures
编辑 删除
小绵羊成熟篇
7 Pictures
编辑 删除
小绵羊餐团篇
27 Pictures
编辑 删除
爱情鸟撒野篇
6 Pictures
爱情鸟洗澡篇
7 Pictures
瓜瓜肉鸡
5 Pictures
编辑 删除
新增相册 注销【 Jerry 】
单击相册封面或相册名称可以进入相册
相册主人可以编辑或删除相册
选择此超链接来新增相册
选择此超链接来从网络相册系统注销

图 22-3


http://localhost/ch22/showAlbum.php?album_id=2
宠物电子相册
寵物電子相簿
元元精选集 2
IMG_0173.jpg
编辑 删除
IMG_0176.jpg
编辑 删除
IMG_0178.jpg
编辑 删除
IMG_0179.jpg
编辑 删除
IMG_0180.jpg
编辑 删除
IMG_0181.jpg
编辑 删除
IMG_0182.jpg
编辑 删除
上传相片 回首页
相册名称
相片缩略图
相片名称
单击相片缩略图或相片名称可以查看该相片
相册主人可以编辑或删除相片
选择此超链接来上传相片
选择此超链接可以回到网络相册首页

图 22-4


http://localhost/ch22/photoDetail.php?album=2&photo=30
宠物电子相册
寵物電子相簿
元元精选集 2
嗨... 我只是虚胖啦
回首页 回【元元精选集 2】相册
原始相片
相片导览区，框红线的相片为目前显示的相片，单击其他相片缩略图可以变更显示的相片

图 22-5

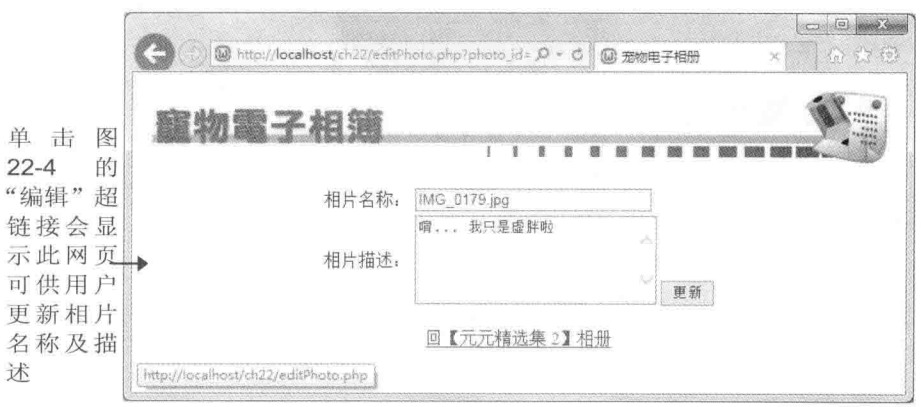

http://localhost/ch22/editPhoto.php?photo_id=
宠物电子相册
寵物電子相簿
相片名称：
IMG_0179.jpg
嗨... 我只是虚胖啦
相片描述：
更新
回【元元精选集 2】相册
http://localhost/ch22/editPhoto.php
单击图 22-4 的“编辑”超链接会显示此网页可供用户更新相片名称及描述

图 22-6

单击图 22-4 的“上传相片”超链接会显示此网页可供用户上传相片

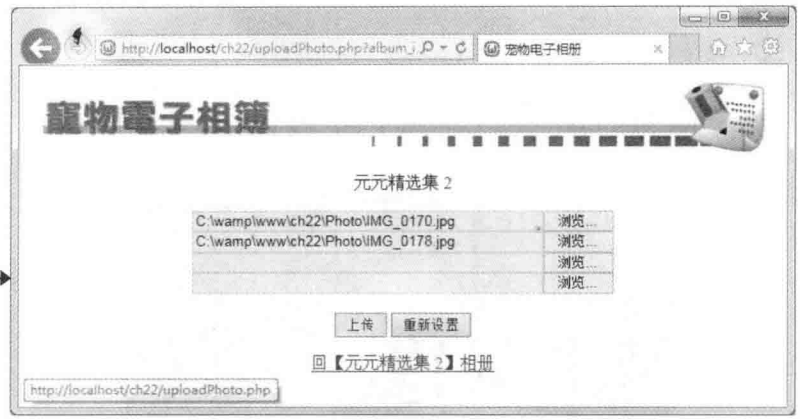


图 22-7

单击图 22-3 的“新增相册”超链接会显示此网页可供用户建立新的相册

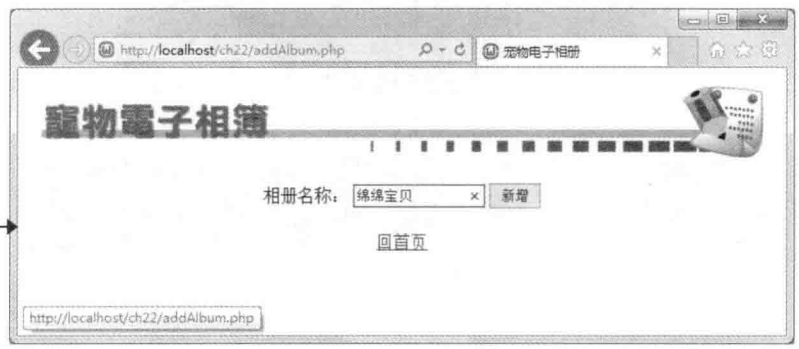


图 22-8

单击图 22-3 的“编辑”超链接会显示此网页可供用户更新相册的名称

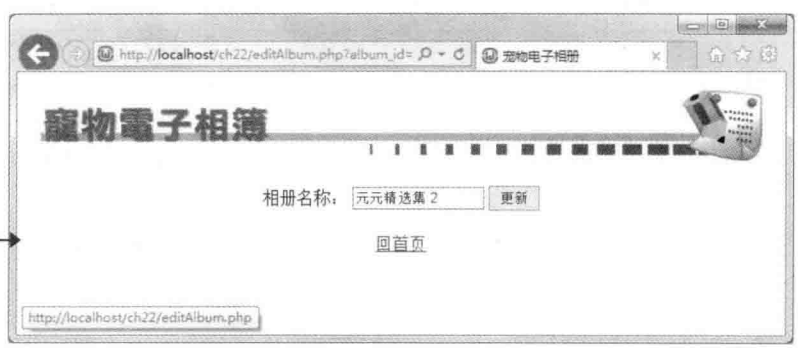


图 22-9

22.2 组成网页的文件列表

这个“网络相册”程序存放在下载资源的 \samples\ch22\ 文件夹中，它用到如表 22-1 所示的文件。

表 22-1　“网络相册”程序用到的文件

文件名	说明
Title.png	这个 PNG 图像文件是“网络相册”的标题图像
index.php	这个网页为网络相册的首页，如图 22-1，用来显示全部的相册，单击[登录]超链接，就会调用<logon.php>，提供用户登录网络相册
logon.php	这个网页提供用户登录网络相册，如图 22-2 所示，登录相册后会自动将用户导向回 <index.php>，如图 22-3 所示
showAlbum.php	这个网页用来显示相册中所有相片的缩略图，如图 22-4，提供用户编辑、删除或上传相片
photoDetail.php	这个网页用来查看相片，如图 22-5 所示，并在网页下方建立相片预览区，框红线的相片缩略图代表当前显示的相片，单击其他相片缩略图可以变更欲查看的相片
editPhoto.php	这个网页在用户单击 <showAlbum.php> 的[编辑]超链接时执行，用来编辑相片名称及描述，如图 22-6 所示
uploadPhoto.php	这个网页在用户单击 <showAlbum.php> 的[上传相片]超链接时执行，用来上传相片，如图 22-7 所示
addAlbum.php	这个网页在用户单击 <index.php> 的[新增相册]超链接时执行，用来建立新的相册，如图 22-8 所示
editAlbum.php	这个网页在用户单击 <index.php> 的[编辑]超链接时执行，用来编辑相册名称，如图 22-9 所示
delAlbum.php	这个网页在用户单击 <index.php> 的[删除]超链接时执行，用来删除特定相册
delPhoto.php	这个网页在用户单击 <showAlbum.php> 的[删除]超链接时执行，用来删除特定相片
logout.php	这个网页在用户单击 <index.php> 的[注销]超链接时执行，用来从网络相册退出
album 数据库	这个数据库包含 album、photo 和 user 三个数据表

在这个网络相册中，我们使用了名称为 album 的数据库，里面包含 album、photo、user 三个数据表，分别用来存储相册数据、相片数据及用户数据，album 数据表用来存储相册数据，其结构如下。

字段名	数据类型	长度	主键	说明
id	INT	-	☑	相册编号字段
name	VARCHAR	256	☐	相册名称字段
owner	VARCHAR	32	☐	相册主人字段

photo 数据表用来存储相片数据，其结构如下。

字段名	数据类型	长度	主键	说明
id	INT	-	☑	相片编号字段 自动编号（auto_increment）
name	VARCHAR	64	☐	相片名称字段
filename	VARCHAR	64	☐	文件名字段
comment	VARCHAR	512	☐	相片描述字段
album_id	INT	-	☐	相册编号字段（用来记录相片属于哪本相册）

user 数据表用来存储用户的账号与密码数据，其结构如下，我们已经事先建立 jerry/jerry 和 jean/jean 两组账号。

字段名	数据类型	长度	主键	说明
account	VARCHAR	32	☑	账号字段
password	VARCHAR	32	☐	密码字段
name	VARCHAR	32	☐	用户名称字段

您可以自己创建数据库或导入本书为您准备的数据库备份文件（位于下载资源的\samples\database\album.sql），有关如何导入 MySQL 数据库，可以参考第 11.3.7 小节的说明。

22.3 网页的运行流程

网页的运行流程如图 22-10 所示。

首先，执行 <index.php>，这是网络相册的首页，它会显示全部的相册及各相册包含的相片数目，如图 22-1 所示，这个网络相册具有权限控制和管理的功能，相册的主人可以添加、编辑或删除相册，也可以上传、编辑或删除相册内的相片，其他用户则只能浏览相册及相片。

若要登录网络相册系统，可以单击 <index.php> 的[登录]超链接，随即会导向到 <logon.php>，如图 22-2 所示，您只要输入账号与密码，按[登录]钮即可登录，登录后会导向回 <index.php>，如图 22-3 所示；此时，您所建立的相册下方会显示[编辑]与[删除]超链接，供您编辑与删除相册，界面最下方会显示[添加相册]超链接，供您建立新的相册。

若要从网络相册系统注销出来（即退出），可以单击 <index.php> 的[注销【xx】]（xx 为用户的名称）超链接，随即会执行 <logout.php>，此网页没有单独的用户界面用于从网络相册系统注销，注销后会自动导向到 <index.php>。

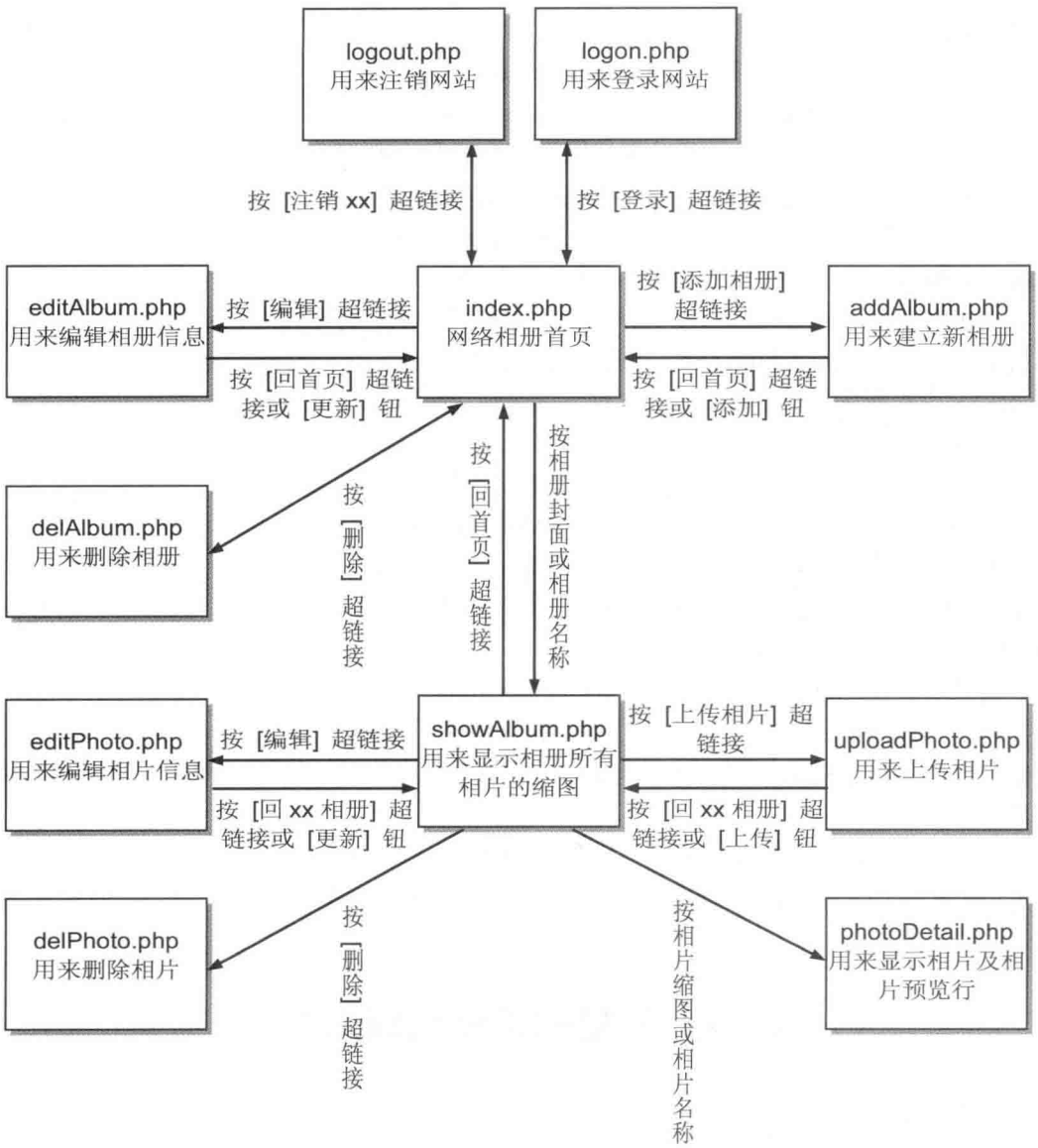


图 22-10

若要创建新的相册，可以单击 <index.php> 的[添加相册]超链接，接着会自动导向到 <addAlbum.php>，如图 22-8，您只要输入相册的名称，按[添加]钮即可创建新相册，相册创建后会导向回 <index.php>，若要放弃创建新相册，可以单击[回首页]超链接。

若要修改相册名称，可以单击 <index.php> 的[编辑]超链接，接着会自动导向到 <editAlbum.php>，如图 22-9，您只要输入相册的名称，按[更新]钮即可更新相册名称，更新完成会导向回 <index.php>，若要放弃更新相册名称，可以单击[回首页]超链接。

若要删除相册（同时会删除相册内的所有相片），只要单击 <index.php> 的[删除]超链接，接着会导向到 <delAlbum.php>，此网页没有单独的用户界面用来删除特定相册，删除完

毕会自动导向回 <index.php>。

若要查看特定相册内的相片，可以单击 <index.php> 界面中的相册封面或相册名称，随即会导向到 <showAlbum.php>，此网页用来显示相册中所有的相片缩略图及相片名称，如图 22-4 所示。若您是相册的主人，相片缩略图下方会显示[编辑]与[删除]超链接，供您编辑与删除相片，界面最下方会显示[上传相片]超链接，供您上传新的相片。

若要上传新的相片，可以单击 <showAlbum.php> 的[上传相片]超链接，随即会自动导向到 <uploadPhoto.php>，如图 22-7 所示，您只要选择要上传的相片，然后按[上传]钮即可上传相片。上传时会自动变更相片大小，并且制作缩略图，然后将之保存在服务器中。上传成功后会自动导向回 <showAlbum.php> 并显示原相册的内容。若要放弃上传相片，可以单击[回 xx 相册](xx 为相册名称) 超链接，这样也可以返回 <showAlbum.php> 并显示原相册内容。

若要修改相片的名称及描述，可以单击 <showAlbum.php> 相片缩略图下方的[编辑]超链接，界面会自动导向到 <editPhoto.php>，如图 22-6 所示，您只要输入相片的名称及描述，然后按[更新]钮，更新完成会自动导向回 <showAlbum.php> 并显示原相册内容。

若要删除相片，只要单击 <showAlbum.php> 相片缩略图下方的[删除]超链接，随即会执行 <delPhoto.php>，此网页没有单独的用户界面用来删除特定相片，删除后会自动导向回 <showAlbum.php> 并显示原相册内容。

若要查看特定相片，可以单击 <showAlbum.php> 界面中的相片缩略图或相片名称，随即会导向到 <photoDetail.php>，并显示您单击的相片，如图 22-5，同时，界面下方会建立相片预览区，用来显示相册中其他相片的缩略图，当您单击其他相片缩略图时，就会变更显示的相片。

此外，框红线的相片缩略图代表当前显示的相片，无法单击，若您要返回原相册，可以单击[回 xx 相册]（xx 为相册名称）超链接，若要回到网络相册的首页，可以单击[回首页]超链接。

22.4 完整的程序代码清单

❖ logon.php

这个网页用来登录网络相册系统，如前面的图 22-2 所示，负责提供表单让用户输入账号与密码，然后按[登录]。

\ch22\logon.php

```
01:<?php
02:   if（isset($_POST["account"]))
03:   {
04:     require_once("dbtools.inc.php");
05:
06:     //获取登录数据
```

```
07:    $login_user = $_POST["account"];
08:    $login_password = $_POST["password"];
09:
10:    //建立数据连接
11:    $link = create_connection();
12:
13:    //检查账号与密码是否正确
14:    $sql = "SELECT account, name FROM user Where account = '$login_user'
15:                AND password = '$login_password'";
16:    $result = execute_sql($link, "album", $sql);
17:
18:    //若没找到数据，表示账号与密码错误
19:    if（mysqli_num_rows($result) == 0)
20:    {
21:       //释放 $result 占用的内存
22:       mysqli_free_result($result);
23:
24:       //关闭数据连接
25:       mysqli_close($link);
26:
27:       //显示信息请求用户输入正确的账号与密码
28:       echo "<script type='text/javascript'>alert('账号密码错误，请查明后再登录')</script>";
29:    }
30:    else        //若账号与密码正确
31:    {
32:       //将用户数据加入 Session
33:       session_start();
34:       $row = mysqli_fetch_object($result);
35:       $_SESSION["login_user"]= $row->account;
36:       $_SESSION["login_name"]= $row->name;
37:
38:       //释放 $result 占用的内存
39:       mysqli_free_result($result);
40:
41:       //关闭数据连接
42:       mysqli_close($link);
43:
44:       header("location:index.php");
45:    }
46:  }
47:?>
48:<!doctype html>
49:<html>
```

```
50:  <head>
51:    <meta charset="utf-8">
52:    <title>宠物电子相册</title>
53:  </head>
54:  <body>
55:    <p align="center"><img src="Title.png"></p>
56:    <form action="logon.php" method="post" name="myForm">
57:      <table align="center">
58:        <tr>
59:          <td>
60:            账号:
61:          </td>
62:          <td>
63:           <input type="text" name="account" size="15">
64:          </td>
65:        </tr>
66:        <tr>
67:          <td>
68:            密码:
69:          </td>
70:          <td>
71:           <input type="password"name="password" size="15">
72:          </td>
73:        </tr>
74:        <tr>
75:          <td align="center" colspan="2">
76:            <input type="submit" value="登录">
77:            <input type="reset" value="重填">
78:          </td>
79:        </tr>
80:      </table>
81:    </form>
82:  </body>
83:</html>
```

- 48～83：建立表单供用户输入账号与密码，按[登录]按钮会执行 <logon.php>。
- 02～46：用来验证用户输入的账号与密码是否有效，第 02 行的 if 条件使用 isset() 函数判断 $_POST["account"]是否有值，有值的话，表示用户按[登录]按钮，此时会执行第 04～44 行来验证账号与密码是否有效，账号与密码错误会执行第 21～28 行来显示错误信息，若通过验证，就执行第 32～44 行，用来将用户的登录账号与名称保存在 Session 中，并将用户导向回网络相册首页 <index.php>。

❖ logout.php

这个网页用来从网络相册系统注销，我们只要清除 Session，然后将用户导向回网络相册首页 <index.php> 即可。

\ch22\logout.php

```
<?php
  session_start();
  session_unset();
  header("location:index.php");
?>
```

❖ index.php

这个网页为网络相册系统的第一个界面，如前面的图 22-1 所示，用来显示全部的相册，并提供相册主人添加、编辑或删除相册，其他用户则只能浏览相册。

\ch22\index.php

```
001:<!doctype html>
002:<html>
003:  <head>
004:    <title>宠物电子相册</title>
005:    <meta charset="utf-8">
006:    <script type="text/javascript">
007:      function DeleteAlbum(album_id)
008:      {
009:        if（confirm("请确认是否删除此相册？"))
010:          location.href = "delAlbum.php?album_id=" + album_id;
011:      }
012:    </script>
013:  </head>
014:  <body>
015:    <p align="center"><img src="Title.png"></p>
016:    <?php
017:      require_once("dbtools.inc.php");
018:
019:      //获取登录者账号与名称
020:      session_start();
021:      if（isset($_SESSION["login_user"]))
022:      {
023:        $login_user = $_SESSION["login_user"];
024:        $login_name = $_SESSION["login_name"];
```

```
025:        }
026:
027:        $link = create_connection();
028:
029:        //获取所有相册的数据
030:        $sql = "SELECT id, name, owner FROM album order by name";
031:        $album_result = execute_sql($link, "album", $sql);
032:
033:        //获取相册的数目
034:        $total_album = mysqli_num_rows($album_result);
035:
036:        echo "<p align='center'>$total_album Albums</p>";
037:        echo "<table border='0' align='center'>";
038:
039:        //指定每行显示几本相册
040:        $album_per_row = 5;
041:
042:        //显示相册列表
043:        $i = 1;
044:        while（$row = mysqli_fetch_assoc($album_result))
045:        {
046:            //获取相册编号、名称及相册的主人
047:            $album_id = $row["id"];
048:            $album_name = $row["name"];
049:            $album_owner = $row["owner"];
050:
051:            $sql = "SELECT filename FROM photo WHERE album_id = $album_id";
052:            $photo_result = execute_sql($link, "album", $sql);
053:
054:            //获取相册的相片数目
055:            $total_photo = mysqli_num_rows($photo_result);
056:
057:            //若相片数目大于 0，就以第一张相片当做相册封面，否则以 None.png 当做封面
058:            if（$total_photo > 0)
059:               mysqli_fetch_object($photo_result)->filename;
060:            else
061:               $cover_photo = "None.png";
062:
063:            mysqli_free_result($photo_result);
064:
065:          if（$i % $album_per_row == 1)
066:             echo "<tr align='center' valign='top'>";
067:
```

```
068:          echo "<td width='160px'>
069:                <a href='showAlbum.php?album_id=$album_id'>
070:                <img src='Thumbnail/$cover_photo'
071:                  style='border-color:Black;border-width:1px'>
072:                <br>$album_name</a><br>$total_photo Pictures";
073:
074:          if（isset($login_user) && $album_owner == $login_user)
075:          {
076:             echo "<br><a href='editAlbum.php?album_id=$album_id'>编辑</a>
077:                   <a href='#' onclick='DeleteAlbum($album_id)'>删除</a>";
078:          }
079:
080:          echo "<p></td>";
081:
082:          if（$i % $album_per_row == 0 || $i == $total_album)
083:             echo "</tr>";
084:
085:          $i++;
086:       }
087:
088:       echo "</table>" ;
089:
090:       //释放内存并关闭数据连接
091:       mysqli_free_result($album_result);
092:       mysqli_close($link);
093:       echo "<hr><p align='center'>";
094:
095:       //若 isset($login_name) 返回 FALSE，表示用户尚未登录系统
096:       if（!isset($login_name))
097:          echo "<a href='logon.php'>登录</a>";
098:       else
099:       {
100:          echo "<a href='addAlbum.php'>添加相册</a>
101:                <a href='logout.php'>注销【 $login_name 】</a>";
102:       }
103:    ?>
104:    </p>
105: </body>
106:</html>
```

- 07 ~ 11：使用 JavaScript 定义 DeleteAlbum() 函数，此函数在单击[删除]超链接时会被执行，用来删除指定的相册。
- 20 ~ 25：获取用户登录的账号与名称。

- 30～31：读取数据库里所有的相册数据，并根据相册名称来排序。
- 34～36：获取并显示相册的数目。
- 40：指定每行显示几本相册。
- 43～86：用来显示相册封面、相册名称及相册包含的相片数目。
- 47～49：获取相册编号、相册名称及相册的主人。
- 51～52：获取相册内所有相片的文件名。
- 55：获取相册包含的相片数目。
- 58～61：用来决定哪张相片要当做相册的封面相片，若相册内的相片数目大于 0，就以第一张相片的缩略图当作相册封面，否则以 None.png 当作封面。
- 65～66、82～83：用来控制每行只显示 $album_per_row 变量指定的相册数目，由于 $album_per_row 等于 5，所以这个网络相册每行会显示 5 个相册。
- 68～72：显示相册的封面相片、相册名称及相册包含的相片数目。
- 74～78：若用户已经登录系统且为相册的主人，就显示[编辑]与[删除]超链接，单击[编辑]超链接将链接至 <editAlbum.php>，用来编辑相册名称，单击[删除]超链接会执行 JavaScript 的 DeleteAlbum() 方法，用来删除特定的相册。
- 96～102：若用户尚未登录系统，会显示[登录]超链接，链接至 <logon.php>，供用户登录系统；相反的，若用户已经登录系统，就显示[添加相册]与[注销【xx】]超链接，单击[添加相册]超链接将链接至 <addAlbum.php>，用来创建新的相册，单击[注销【xx】]超链接将链接至 <logout.php>，用来从网络相册系统注销。

❖ addAlbum.php

这个网页用来创建新的相册，如前面的图 22-8 所示。

\ch22\addAlbum.php

```
01:<?php
02:   if（isset($_POST["album_name"]))
03:   {
04:      require_once("dbtools.inc.php");
05:         $album_name = $_POST["album_name"];
06:
07:      //获取登录者账号
08:      session_start();
09:      $login_user = $_SESSION["login_user"];
10:
11:      $link = create_connection();
12:
13:         //添加相册
14:         $sql = "SELECT ifnull(max(id), 0) + 1 AS album_id FROM album";
15:         $result = execute_sql($link, "album", $sql);
```

```
16:        $album_id = mysqli_fetch_object($result)->album_id;
17:
18:        $sql = "INSERT INTO album(id, name, owner)
19:                VALUES($album_id, '$album_name', '$login_user')";
20:        execute_sql($link, "album", $sql);
21:
22:     //释放内存并关闭数据连接
23:     mysqli_free_result($result);
24:     mysqli_close($link);
25:
26:     header("location:showAlbum.php?album_id=$album_id");
27:  }
28:?>
29:<!doctype html>
30:<html>
31:  <head>
32:     <meta charset="utf-8">
33:     <title>宠物电子相册</title>
34:  </head>
35:  <body>
36:     <p align="center"><img src="Title.png"></p>
37:     <form action="addAlbum.php" method="post">
38:        <table align="center">
39:           <tr>
40:              <td>
41:                 相册名称：
42:              </td>
43:              <td>
44:               <input type="text" name="album_name" size="15">
45:               <input type="submit" value="添加">
46:              </td>
47:           </tr>
48:           <tr>
49:              <td colspan="3" align="center">
50:                 <br><a href="index.php">回首页</a>
51:              </td>
52:           </tr>
53:        </table>
54:     </form>
55:  </body>
56:</html>
```

- 02、27：这个 if 条件使用 isset() 函数判断 $_POST["album_name"]是否有值，有值的

话，表示用户按[添加]钮，此时会执行第 04～26 行，将新相册的数据写入数据库。

- 05：获取新相册的名称。
- 08～09：获取登录者的账号。
- 14～16：获取当前系统里最大的相册编号，再将编号 + 1 当做新相册的编号，当系统里没有任何相册时，max(id) 会返回 null，null + 1 仍是 null，ifnull(max(id), 0) 的目的是当 max(id) 返回 null 时，自动将 null 取代为 0，如此即可避免 null + 1 的错误情况。
- 18～20：将新相册的数据写入 album 数据表。
- 26：将用户导向到 <showAlbum.php>，并显示刚才创建的相册。

editAlbum.php

这个网页用来编辑相册名称，如前面的图 22-9 所示。

\ch22\editAlbum.php

```
01:<?php
02:   require_once("dbtools.inc.php");
03:
04:   //获取登录者账号
05:   session_start();
06:   $login_user = $_SESSION["login_user"];
07:
08:   //建立数据连接
09:   $link = create_connection();
10:
11:   if（!isset($_POST["album_id"]))
12:   {
13:       $album_id = $_GET["album_id"];
14:
15:     //获取相册名称及相册所有者的账号
16:     $sql = "SELECT name, owner FROM album where id = $album_id";
17:     $result = execute_sql($link, "album", $sql);
18:     $row = mysqli_fetch_object($result);
19:     $album_name = $row->name;
20:     $album_owner = $row->owner;
21:
22:     //释放 $result 占用的内存
23:     mysqli_free_result($result);
24:
25:     //关闭数据连接
26:     mysqli_close($link);
```

```
27:
28:     if（$album_owner != $login_user)
29:     {
30:         echo "<script type='text/javascript'>";
31:         echo "alert('您不是相册的主人，无法修改相册名称。$album_owner');";
32:         echo "</script>";
33:     }
34: }
35: else
36: {
37:         $album_id = $_POST["album_id"];
38:         $album_name = $_POST["album_name"];
39:
40:         $sql = "UPDATE album SET name = '$album_name'
41:                 WHERE id = $album_id AND owner = '$login_user'";
42:         execute_sql($link, "album", $sql);
43:
44:     //关闭数据连接
45:     mysqli_close($link);
46:
47:     header("location:index.php");
48: }
49:?>
50:<!doctype html>
51:<html>
52:  <head>
53:     <meta charset="utf-8">
54:     <title>宠物电子相册</title>
55:  </head>
56:  <body>
57:     <p align="center"><img src="Title.png"></p>
58:     <form action="editAlbum.php" method="post">
59:         <table align="center">
60:             <tr>
61:                 <td>
62:                     相册名称：
63:                 </td>
64:                 <td>
65:                  <input type="text" name="album_name" size="15"
66:                     value="<?php echo $album_name ?>">
67:                  <input type="hidden" name="album_id" value="<?php echo $album_id ?>">
68:                  <input type="submit" value="更新"
69:                     <?php if（$album_owner != $login_user) echo 'disabled' ?>>
```

```
70:             </td>
71:           </tr>
72:           <tr>
73:             <td colspan="2" align="center">
74:               <br><a href="index.php">回首页</a>
75:             </td>
76:           </tr>
77:         </table>
78:       </form>
79:   </body>
80:</html>
```

- 05～06：获取登录者的账号。
- 11、48：这个 if 条件使用 isset() 函数判断 $_POST["album_id"]是否有值，没有值的话，表示用户单击了 <index.php> 的[编辑]超链接，才被导向到此网页，此时会执行第 13～33 行；相反的，有值的话，就执行第 37 ～ 47 行，更新相册名称。
- 16～20：获取相册的名称及相册的主人。
- 28～33：若登录者不是相册的主人，就显示错误信息。
- 37～38：获取相册编号及新的相册名称。
- 40～42：将新的相册名称更新到 album 数据表。
- 47：将用户导向到<index.php>。
- 69：若登录者不是相册的主人，就取消[更新]按钮的功能，使之无法单击，这样可以避免非相册主人更改相册的名称。

delAlbum.php

这个网页用来删除相册，删除相册会同时删除存储在数据库里的相片信息及保存在硬盘里的相片文件。

\ch22\delAlbum.php

```
01:<?php
02:   require_once("dbtools.inc.php");
03:   $album_id = $_GET["album_id"];
04:
05:   //获取登录者账号
06:   session_start();
07:   $login_user = $_SESSION["login_user"];
08:
09:   //建立数据连接
10:   $link = create_connection();
11:
```

```
12:  //删除保存在硬盘的相片
13:  $sql = "SELECT filename FROM photo WHERE album_id = $album_id
14:          AND EXISTS(SELECT '*' FROM album WHERE id = $album_id AND owner = '$login_user')";
15:  $result = execute_sql($link, "album", $sql);
16:
17:  while（$row = mysqli_fetch_assoc($result))
18:  {
19:      $file_name = $row["filename"];
20:      $photo_path = realpath("./Photo/$file_name");
21:      $thumbnail_path = realpath("./Thumbnail/$file_name");
22:
23:    if（file_exists($photo_path))
24:      unlink($photo_path);
25:
26:    if（file_exists($thumbnail_path))
27:      unlink($thumbnail_path);
28:  }
29:
30:  //删除存储在数据库的相片信息
31:  $sql = "DELETE FROM photo WHERE album_id = $album_id
32:          AND EXISTS(SELECT '*' FROM album WHERE id = $album_id AND owner = '$login_user')";
33:  execute_sql($link, "album", $sql);
34:
35:  //删除存储在数据库的相册信息
36:  $sql = "DELETE FROM album WHERE id = $album_id AND owner = '$login_user'";
37:  execute_sql($link, "album", $sql);
38:
39:  //释放内存并关闭数据连接
40:  mysqli_free_result($result);
41:  mysqli_close($link);
42:
43:  header("location:index.php");
44:?>
```

- 03：获取相册编号。
- 06～07：获取登录者的账号。
- 13～15：获取相册内所有相片的文件名，SQL 语法中的 EXISTS(SELECT '*' FROM album WHERE id = $album_id AND owner = '$login_user') 是为了预防相片不会被非相册主人删除。
- 17～28：删除相册内的所有相片及缩略图的文件。
- 31～33：删除存储在数据库里的相片信息。

- 36～37：删除存储在数据库里的相册信息。
- 43：将用户导向到 <index.php>。

❖ showAlbum.php

这个网页用来显示相册内所有相片的缩略图，并供用户编辑、删除或上传相片，如前面的图 22-4 所示。

\ch22\showAlbum.php

```
01:<!doctype html>
02:<html>
03:  <head>
04:      <title>宠物电子相册</title>
05:    <meta charset="utf-8">
06:    <script type="text/javascript">
07:      function DeletePhoto(album_id, photo_id)
08:      {
09:            if（confirm("请确认是否删除此相片？"))
10:              location.href = "delPhoto.php?album_id=" + album_id + "&photo_id=" + photo_id;
11:      }
12:    </script>
13:  </head>
14:  <body>
15:    <p align="center"><img src="Title.png"></p>
16:    <?php
17:      require_once("dbtools.inc.php");
18:      $album_id = $_GET["album_id"];
19:
20:      //获取登录者账号
21:      $login_user = "";
22:
23:      session_start();
24:      if（isset($_SESSION["login_user"]))
25:        $login_user = $_SESSION["login_user"];
26:
27:      //建立数据连接
28:      $link = create_connection();
29:
30:      //获取相册名称及相册的主人
31:      $sql = "SELECT name, owner FROM album WHERE id = $album_id";
32:      $result = execute_sql($link, "album", $sql);
33:      $row = mysqli_fetch_object($result);
```

```
34:     $album_name = $row->name;
35:     $album_owner = $row->owner;
36:     echo "<p align='center'>$album_name</p>";
37:
38:     //获取相册内所有相片的缩略图
39:     $sql = "SELECT id, name, filename FROM photo WHERE album_id = $album_id";
40:     $result = execute_sql($link, "album", $sql);
41:     $total_photo = mysqli_num_rows($result);
42:
43:     echo "<table border='0' align='center'>";
44:
45:     //指定每行显示几张相片
46:     $photo_per_row = 5;
47:
48:     //显示相片缩略图
49:     $i = 1;
50:     while（$row = mysqli_fetch_assoc($result))
51:     {
52:         $photo_id = $row["id"];
53:         $photo_name = $row["name"];
54:         $file_name = $row["filename"];
55:
56:       if（$i % $photo_per_row == 1)
57:         echo "<tr align='center'>";
58:
59:       echo "<td width='160px'><a href='photoDetail.php?album=$album_id&photo=$photo_id'>
60:             <img src='Thumbnail/$file_name' style='border-color:Black;border-width:1px'>
61:             <br>$photo_name</a>";
62:
63:       if（$album_owner == $login_user)
64:         echo "<br><a href='editPhoto.php?photo_id=$photo_id'>编辑</a>
65:               <a href='#' onclick='DeletePhoto($album_id, $photo_id)'>删除</a>";
66:
67:       echo "<p></td>";
68:
69:       if（$i % $photo_per_row == 0 || $i == $total_photo)
70:         echo "</tr>";
71:       $i++;
72:     }
73:
74:     echo "</table>" ;
75:
76:     //释放资源并关闭数据连接
77:     mysqli_free_result($result);
```

```
78:         mysqli_close($link);
79:
80:         echo "<hr><p align='center'>";
81:         if（$album_owner == $login_user)
82:             echo "<a href='uploadPhoto.php?album_id=$album_id'>上传相片</a> ";
83:     ?>
84:     <a href='index.php'>回首页</a></p>
85:  </body>
86:</html>
```

- 07～11：使用 JavaScript 定义 DeletePhoto() 函数，此函数在单击[删除]超链接时会被执行，用来删除指定的相片。
- 21～25：获取用户登录的账号。
- 31～35：获取相册名称及相册的主人。
- 39～40：获取相册内所有相片的缩略图。
- 46：指定每行显示几张相片。
- 49～72：用来显示相片缩略图及相片名称。
- 52～54：获取相片编号、名称及实体文件名。
- 56～57、69～70：用来控制每行只显示 $photo_per_row 变量指定的相片数目，由于 $photo_per_row 等于 5，所以这个网络相册每行会显示 5 张相片。
- 59～61：显示相片缩略图及相片名称。
- 63～65：若用户已经登录系统且为相册的主人，就显示[编辑]与[删除]超链接，单击[编辑]超链接将链接至 <editPhoto.php>，用来编辑相片名称及描述，单击[删除]超链接会执行 JavaScript 的 DeletePhoto() 方法，用来删除特定相片。
- 81～82：若用户已经登录系统且为相册的主人，就显示[上传相片]超链接，单击此超链接将链接至 <uploadPhoto.php>，用来上传新的相片。

editPhoto.php

这个网页用来编辑相片名称及描述，如前面的图 22-6 所示。

\ch22\editPhoto.php

```
01:<?php
02:  require_once("dbtools.inc.php");
03:
04:  //获取登录者账号
05:  session_start();
06:  $login_user = $_SESSION["login_user"];
07:
08:  $link = create_connection();
```

```
09:
10:  if（!isset($_POST["photo_name"]))
11:  {
12:      $photo_id = $_GET["photo_id"];
13:
14:     //获取相册名称及相册的主人等信息
15:     $sql = "SELECT a.name, a.filename, a.comment, a.album_id, b.name AS album_name,
16:             b.owner FROM photo a, album b where a.id = $photo_id and b.id = a.album_id";
17:     $result = execute_sql($link, "album", $sql);
18:     $row = mysqli_fetch_object($result);
19:     $album_id = $row->album_id;
20:     $album_name = $row->album_name;
21:     $album_owner = $row->owner;
22:     $photo_name = $row->name;
23:     $file_name = $row->filename;
24:     $photo_comment = $row->comment;
25:
26:     //释放 $result 占用的内存
27:     mysqli_free_result($result);
28:
29:     //关闭数据连接
30:     mysqli_close($link);
31:
32:     if（$album_owner != $login_user)
33:     {
34:        echo "<script type='text/javascript'>";
35:        echo "alert('您不是相片的主人，无法修改相片名称。')";
36:        echo "</script>";
37:     }
38:  }
39:  else
40:  {
41:      $album_id = $_POST["album_id"];
42:      $photo_id = $_POST["photo_id"];
43:      $photo_name = $_POST["photo_name"];
44:      $photo_comment = $_POST["photo_comment"];
45:
46:      $sql = "UPDATE photo SET name = '$photo_name', comment = '$photo_comment'
47:              WHERE id = $photo_id AND EXISTS(SELECT '*' FROM album
48:              WHERE id = $album_id AND owner = '$login_user')";
49:      execute_sql($link, "album", $sql);
50:
51:    mysqli_close($link);
```

```
52:     header("location:showAlbum.php?album_id=$album_id");
53:  }
54:?>
55:<!doctype html>
56:<html>
57:  <head>
58:     <meta charset="utf-8">
59:     <title>宠物电子相册</title>
60:  </head>
61:  <body>
62:     <p align="center"><img src="Title.png"></p>
63:     <form action="editPhoto.php" method="post">
64:        <table align="center">
65:           <tr>
66:              <td>
67:                 相片名称：
68:              </td>
69:              <td>
70:               <input type="text" name="photo_name" size="31"
71:                  value="<?php echo $photo_name ?>">
72:              </td>
73:           </tr>
74:           <tr>
75:              <td>
76:                 相片描述：
77:              </td>
78:              <td>
79:               <textarea name="photo_comment" rows="5" cols="25">
80:               <?php echo $photo_comment ?></textarea>
81:               <input type="hidden" name="photo_id" value="<?php echo $photo_id ?>">
82:               <input type="hidden" name="album_id" value="<?php echo $album_id ?>">
83:               <input type="submit" value="更新"
84:                  <?php if（$album_owner != $login_user) echo 'disabled' ?>>
85:              </td>
86:           </tr>
87:           <tr>
88:              <td colspan="2" align="center">
89:                 <br><a href="showAlbum.php?album_id=<?php echo $album_id ?>">
90:                    回【<?php echo $album_name ?>】相册</a>
91:              </td>
92:           </tr>
93:        </table>
94:     </form>
```

```
95:  </body>
96:</html>
```

- 05～06：获取登录者的账号。
- 10、53：这个 if 条件使用 isset() 函数判断 $_POST["photo_name"]是否有值，没有值的话，表示用户是单击 <showAlbum.php> 的[编辑]超链接，才被导向到此网页，此时会执行第 12 ～ 37 行，相反的，有值的话，就执行第 41 ～ 52 行，用来更新相片名称及描述。
- 15～24：获取相册编号、相册名称、相册的主人、相片名称、文件名及相片描述。
- 32～37：若登录者不是相册的主人，就显示错误信息。
- 41～44：获取相册编号、相片编号、相片名称、相片描述。
- 46～49：将新的相片名称及描述更新至 photo 数据表。
- 52：将用户导向至 <showAlbum.php>。

❖ delPhoto.php

这个网页用来删除相片。

\ch22\delPhoto.php

```
01:<?php
02:   require_once("dbtools.inc.php");
03:   $album_id = $_GET["album_id"];
04:   $photo_id = $_GET["photo_id"];
05:
06:   //获取登录者账号
07:   session_start();
08:   $login_user = $_SESSION["login_user"];
09:
10:   //建立数据连接
11:   $link = create_connection();
12:   //删除保存在硬盘的相片
13:   $sql = "SELECT filename FROM photo WHERE id = $photo_id
14:           AND EXISTS(SELECT '*' FROM album WHERE id = $album_id AND owner = '$login_user')";
15:   $result = execute_sql($link, "album", $sql);
16:
17:   $file_name = mysqli_fetch_object($result)->filename;
18:   $photo_path = realpath("./Photo/$file_name");
19:   $thumbnail_path = realpath("./Thumbnail/$file_name");
20:
```

```
21:  if（file_exists($photo_path))
22:     unlink($photo_path);
23:  if（file_exists($thumbnail_path))
24:     unlink($thumbnail_path);
25:
26:  //删除存储在数据库的相片信息
27:  $sql = "DELETE FROM photo WHERE id = $photo_id
28:          AND EXISTS(SELECT '*' FROM album WHERE id = $album_id AND owner = '$login_user')";
29:  execute_sql($link, "album", $sql);
30:
31:  //释放内存并关闭数据连接
32:  mysqli_free_result($result);
33:  mysqli_close($link);
34:
35:  header("location:showAlbum.php?album_id=$album_id");
36:?>
```

- 03、04：获取相册编号及相片编号。
- 07～08：获取用户的登录账号。
- 13～15：获取相片的文件名，SQL 语法中的 EXISTS(SELECT '*' FROM album WHERE id = $album_id AND owner = '$login_user'）是为了预防相片不会被非相册主人删除。
- 21～24：删除保存在硬盘里的相片及相片缩略图文件。
- 27～29：删除存储在数据库里的相片信息。
- 35：将用户导向至 <showAlbum.php>。

❖ uploadPhoto.php

这个网页用来上传相片。

\ch22\uploadPhoto.php

```
001:<?php
002:  require_once("dbtools.inc.php");
003:
004:  //建立数据连接
005:  $link = create_connection();
006:
007:  if（!isset($_POST["album_id"]))
008:  {
009:       $album_id = $_GET["album_id"];
010:
```

```
011:      //获取相册名称及相册的主人
012:      $sql = "SELECT name, owner FROM album WHERE id = $album_id";
013:      $result = execute_sql($link, "album", $sql);
014:    $row = mysqli_fetch_object($result);
015:    $album_name = $row->name;
016:    $album_owner = $row->owner;
017:
018:    //释放 $result 占用的内存
019:    mysqli_free_result($result);
020: }
021: else
022: {
023:      $album_id = $_POST["album_id"];
024:           $album_owner = $_POST["album_owner"];
025:    //获取登录者账号
026:       session_start();
027:       $login_user = $_SESSION["login_user"];
028:
029:    if（isset($login_user) && $album_owner == $login_user)
030:    {
031:         for（$i = 0; $i <= 3; $i++)
032:      {
033:              //若文件名不是空字符串，表示上传成功，将临时文件移至指定的目录
034:          if（$_FILES["myfile"]["name"][$i]!= "")
035:          {
036:              $src_file = $_FILES["myfile"]["tmp_name"][$i];
037:              $src_file_name = $_FILES["myfile"]["name"][$i];
038:              $src_ext = strtolower(strrchr($_FILES["myfile"]["name"][$i], "."));
039:              $desc_file_name = uniqid() . ".jpg";
040:
041:          $photo_file_name = "./Photo/$desc_file_name";
042:              $thumbnail_file_name = "./Thumbnail/$desc_file_name";
043:
044:              resize_photo($src_file, $src_ext, $photo_file_name, 600);
045:              resize_photo($src_file, $src_ext, $thumbnail_file_name, 150);
046:
047:              $sql = "insert into photo(name, filename, album_id) values('$src_file_name',
048:                   '$desc_file_name', $album_id)";
049:              execute_sql($link, "album", $sql);
050:          }
051:      }
052:    }
053:
```

```
    //关闭数据连接
    mysqli_close($link);

    header("location:showAlbum.php?album_id=$album_id");
}
function resize_photo($src_file, $src_ext, $dest_name, $max_size)
{
    switch（$src_ext)
    {
        case ".jpg":
            $src = imagecreatefromjpeg($src_file);
            break;
        case ".png":
            $src = imagecreatefrompng($src_file);
            break;
        case ".gif":
            $src = imagecreatefromgif($src_file);
            break;
    }

    $src_w = imagesx($src);
    $src_h = imagesy($src);

    //建立新的空白图像
    if($src_w > $src_h)
    {
        $thumb_w = $max_size;
        $thumb_h = intval($src_h / $src_w * $thumb_w);
    }
    else
    {
        $thumb_h = $max_size;
        $thumb_w = intval($src_w / $src_h * $thumb_h);
    }

    $thumb = imagecreatetruecolor($thumb_w, $thumb_h);

    //进行复制并生成缩略图
        imagecopyresized($thumb, $src, 0, 0, 0, 0, $thumb_w, $thumb_h, $src_w, $src_h);
    //存储相片
    imagejpeg($thumb, $dest_name, 100);

    //释放影像占用的内存
```

```
097:      imagedestroy($src);
098:      imagedestroy($thumb);
099:  }
100:?>
100:<!doctype html>
102:<html>
103:  <head>
104:    <title>宠物电子相册</title>
105:    <meta charset="utf-8">
106:  </head>
107:  <body>
108:    <p align="center"><img src="Title.png"></p>
109:    <p align="center">
110:     <?php echo $album_name ?>
111:    <form method="post" action="uploadPhoto.php" enctype="multipart/form-data">
112:      <input type="file" name="myfile[]" size="50"><br>
113:           <input type="file" name="myfile[]" size="50"><br>
114:           <input type="file" name="myfile[]" size="50"><br>
115:           <input type="file" name="myfile[]" size="50"><br><br>
116:           <input type="hidden" name="album_id" value="<?php echo $album_id ?>">
117:           <input type="hidden" name="album_owner" value="<?php echo $album_owner ?>">
118:           <input type="submit" value="上传">
119:           <input type="reset" value="重新设置">
120:         </form>
121:      <a href="showAlbum.php?album_id=<?php echo $album_id ?>">
122:        回【<?php echo $album_name ?>】相册</a>
123:    </P>
124:  </body>
125:</html>
```

- 007、058：这个 if 条件使用 isset() 函数判断 $_POST["album_id"]是否有值，没有值的话，表示用户单击了 <showAlbum.php> 的[上传相片]超链接，才被导向到此网页，此时会执行第 009 ~ 019 行；相反的，有值的话，表示单击[上传]钮，此时会执行第 23 ~ 57 行，用来上传相片。
- 012 ~ 016：获取相册名称及相册的主人。
- 023 ~ 024：获取相册编号及相册的主人。
- 026 ~ 027：获取用户的登录账号。
- 029 ~ 052：若用户已经登录系统且为相册的主人，就开始处理上传的相片，第 39 行使用 uniqid() 函数是为了产生独一无二的文件名，第 044、045 行调用 resize_photo() 函数，第 44 行用来存储相片，第 45 行用来制作一张缩略图，第 047 ~ 049 用来将相片信息写入 photo 数据表。

- 057：将用户导向至 <showAlbum.php>。
- 059～099：定义 resize_photo() 函数，用来对相片进行缩图，然后保存在服务器中。

photoDetail.php

这个网页用来查看特定的相片，如前面的图 22-5 所示，并在网页下方建立相片预览区，框红线的相片缩略图代表目前显示的相片，单击其他相片缩略图可以变更要查看的相片。

\ch22\photoDetail.php

```
01:<!doctype html>
02:<html>
03:  <head>
04:      <title>宠物电子相册</title>
05:    <meta charset="utf-8">
06:  </head>
07:  <body>
08:    <p align="center"><img src="Title.png"></p>
09:    <?php
10:      require_once("dbtools.inc.php");
11:      $album_id = $_GET["album"];
12:      $photo_id = $_GET["photo"];
13:
14:      //建立数据连接
15:      $link = create_connection();
16:
17:      //获取相册名称
18:      $sql = "SELECT name FROM album WHERE id = $album_id";
19:      $result = execute_sql($link, "album", $sql);
20:      $album_name = mysqli_fetch_object($result)->name;
21:
22:      echo "<p align='center'>$album_name</p>";
23:
24:      //获取并显示相片数据
25:      $sql = "SELECT filename, comment FROM photo WHERE id = $photo_id";
26:      $result = execute_sql($link, "album", $sql);
27:      $row = mysqli_fetch_object($result);
28:      $file_name = $row->filename;
29:      $comment = $row->comment;
30:
31:      echo "<p align='center'><img src='Photo/$file_name'
32:              style='border-style:solid;border-width:1px'></p>";
33:      echo "<p align='center'>$comment</p>";
```

```
34:
35:      //获取并建立相片预览数据
36:      $sql = "SELECT a.id, a.filename FROM（SELECT id, filename FROM photo
37:              WHERE album_id = $album_id AND（id <= $photo_id)
38:              ORDER BY id desc limit 3) a ORDER BY a.id";
39:      $result = execute_sql($link, "album", $sql);
40:
41:      echo "<hr><p align='center'>";
42:      while（$row = mysqli_fetch_assoc($result))
43:      {
44:        if（$row["id"]== $photo_id)
45:      {
46:              echo "<img src='Thumbnail/" . $row["filename"].
47:                  "' style='border-style:solid;border-color: Red;border-width:2px'>   ";
48:          }
49:          else
50:          {
51:              echo "<a href='photoDetail.php?album=$album_id&photo=" . $row["id"].
52:                  "'><img src='Thumbnail/" . $row["filename"].
53:                  "' style='border-style:solid;border-color:Black;border-width:1px'></a>   ";
54:          }
55:      }
56:
57:      $record_to_get = 5 - mysqli_num_rows($result);
58:      $sql = "SELECT id, filename FROM photo WHERE album_id = $album_id AND（id > $photo_id)
59:              ORDER BY id limit $record_to_get";
60:      $result = execute_sql($link, "album", $sql);
61:
62:      while（$row = mysqli_fetch_assoc($result))
63:      {
64:          echo "<a href='photoDetail.php?album=$album_id&photo=" . $row["id"].
65:                  "'><img src='Thumbnail/" . $row["filename"].
66:                  "' style='border-style:solid;border-color:Black;border-width:1px'></a>   ";
67:      }
68:
69:      echo "</p>";
70:
71:      //释放内存并关闭数据连接
72:      mysqli_free_result($result);
73:      mysqli_close($link);
74:  ?>
75:  <p align="center">
76:    <a href="index.php">回首页</a>
```

```
77:         <a href="showAlbum.php?album_id=<?php echo $album_id ?>">
78:             回【<?php echo $album_name ?>】相册
79:       </p>
80:   </body>
81: </html>
```

PHP & MySQL

跨设备网站开发实例精粹

超强范例集

在网页上导入Ajax技术

构建Google地图应用网站

丰富的范例帮助您设计和制作跨智能设备（电脑，智能手机，平板电脑等）的网站，各类应用包括：

- 支持移动设备上的网店应用：商品目录，购物车
- 包含移动版网页的：访客留言板，讨论组，文件上传，在线邮件服务，电子贺卡，会员管理系统，在线投票系统，网络相册等。

清华大学出版社数字出版网站
WQBook 书文局泉
www.wqbook.com

碁峰 www.gotop.com.tw

ISBN 978-7-302-40045-5
9 787302 400455 >
定价：79.00元

[General Information]
书名=PHP&MYSQL跨设备网站开发实例精粹
作者=陈惠贞，陈俊荣编著
页数=478
SS号=13868245
DX号=
出版日期=2015.08
出版社=清华大学出版社