程/序/员/软/件/开/发/书/库

和外星人一起学
基础·入门

明日科技◉编著

零基础学PHP

（全彩版）

立体化教学模式 0基础快速入门

- 书网合一 —— 扫描二维码，学习免费在线课程
- 全彩印刷 —— 还原真实开发环境，让编程学习更轻松
- 扫码答疑 —— 扫描e学码，深入学习开发技能
- 小白实战手册 —— 三个有趣的小项目，强化实战训练

数字电子书

吉林大学出版社

和外星人一起学
基础·入门

明日科技◉编著

零基础学PHP

（全彩版）

吉林大学出版社

内 容 简 介

《零基础学 PHP》是针对零基础编程学习者研发的 PHP 入门教程。从初学者角度出发，通过通俗易懂的语言、流行有趣的实例，详细地介绍了使用 PHP 进行程序开发需要掌握的知识和技术。全书共分 16 章，包括开发环境的搭建、PHP 语言基础、流程控制语句、字符串操作与正则表达式、PHP 数组、面向对象、PHP 与 Web 页面交互、MySQL 数据库基础以及 51 购商城等。书中所有知识都结合具体实例进行讲解，设计的程序代码给出了详细的注释，可以使读者轻松领会 PHP 程序开发的精髓，快速提高开发技能。

本书通过大量实例及一个完整项目案例，帮助读者更好地巩固所学知识，提升能力；随书附赠的《小白实战手册》中给出了 3 个流行的实用案例的详细开发流程，力求让学习者能学以致用，真正获得开发经验；附赠的光盘中给出视频讲解、实例及项目源码、代码查错器、练一练答案和动手纠错答案等，方便读者学习；书中设置了 200 多个二维码，扫描二维码观看视频讲解，解决学习疑难；不易理解的专业术语、代码难点只需手机扫描文字下方的 e 学码获得更多扩展解释，随时扫除学习障碍。此外，登录明日学院网站（www.mingrisoft.com）还可以获得更多学习资源和技术支持。

图书与《小白实战手册》+ 光盘 + 二维码 +e 学码 + 明日学院，实现立体化、全方位的教学模式，拉低编程门槛，让零基础者轻松跨入编程领域。

图书在版编目（CIP）数据

零基础学 PHP / 明日科技编著 . -- 长春 : 吉林大学出版社 , 2017.9
ISBN 978-7-5692-0868-9

Ⅰ . ①零… Ⅱ . ①明… Ⅲ . ① PHP 语言－程序设计 Ⅳ . ① TP312.8

中国版本图书馆 CIP 数据核字 (2017) 第 229142 号

书　　名：零基础学 PHP
　　LING JICHU XUE PHP
作　　者：明日科技 编著
策划编辑：殷丽爽　　责任编辑：张宏亮
责任校对：殷丽爽　　装帧设计：明日科技
出版发行：吉林大学出版社
社　　址：长春市人民大街 4059 号　　邮政编码：130021
发行电话：0431-89580026 / 0431-84978982
网　　址：http://www.jlup.com.cn　　电子邮箱：jdcbs@jlu.edu.cn
印　　刷：吉广控股有限公司
开　　本：850mm×1100mm　1/16　　印　　张：27
　　（附 DVD 光盘 1 张）
字　　数：850 千字　　印　　数：1 ～ 10000
版　　次：2017 年 9 月第 1 版　　印　　次：2017 年 9 月第 1 次
书　　号：ISBN 978-7-5692-0868-9
定　　价：79.80 元

如何使用本书

如何做编程的国王，而不是做编程的奴隶？

本书提供了编程学习的 4 大“利器”，助学习者轻松成为编程的国王！

☑ 视频讲解：精彩详尽的讲解，引导初学者快速入门

对于编程初学者来说，视频讲解就像一位贴身的老师，精彩的知识讲解和透彻的实例解析能够引导初学者快速入门，轻松理解和掌握知识要点。本书为每个章节都配备了精彩的视频讲解，读者可以通过扫描书中的“视频讲解”二维码或在光盘中 Video 文件夹中获取相应的视频课程。

☑ e 学码：更多拓展知识，扫除学习障碍

在学习时，经常会遇到难理解的术语和不知道如何使用的对象、语句、函数或控件等，通过 e 学码，可以很好地解决这些难题。例如，代码中“settype”是什么意思？扫描其下方的 e 学码“[illegible]”，如下图所示。（注：要使用 e 学码，请先扫描“e 学码使用方法”二维码）

```
$result = settype($num,'integer');
```

e学码

e学码使用方法

手机将会获得“settype”的扩展学习列表，如下图所示，单击选项即可查看详细内容。

☑ 小白实战手册：分阶段项目实战，提升开发技能

学编程最好的方法就是用编程，为使读者在学习一部分知识后，能进行一些力所能及的实战训练，小白实战手册针对本书的前三篇知识提供了趣味图片生成器、九宫格抽奖和模拟 12306 图片验证码三个有趣的实战训练项目。读者可以在每篇知识学习完成后，进行有针对性的实战训练，提升开发技能和编程思维。

第一篇训练项目

第二篇训练项目

第三篇训练项目

每个实战项目开发完成后，可以根据完成项目所用的时间，结合《小白实战手册》封三中提供的《编程能力测评表》，进行自我测评。

☑ 数字电子书：随时随地，想学就学

本书不仅内容丰富，图文并茂，还开发了配套的数字电子书。购买本书后，读者可以刮开封底的学习码涂层，然后扫描激活二维码，填写学习码注册会员，注册成功后即可查看本书的配套数字电子书及其他学习资源，随时随地，想学就学。注册过程如下图所示。

刮开封底的涂层，获得在线学习码

扫描封底的激活二维码

查看图书相关学习资源

注册明日会员，并激活在线学习码

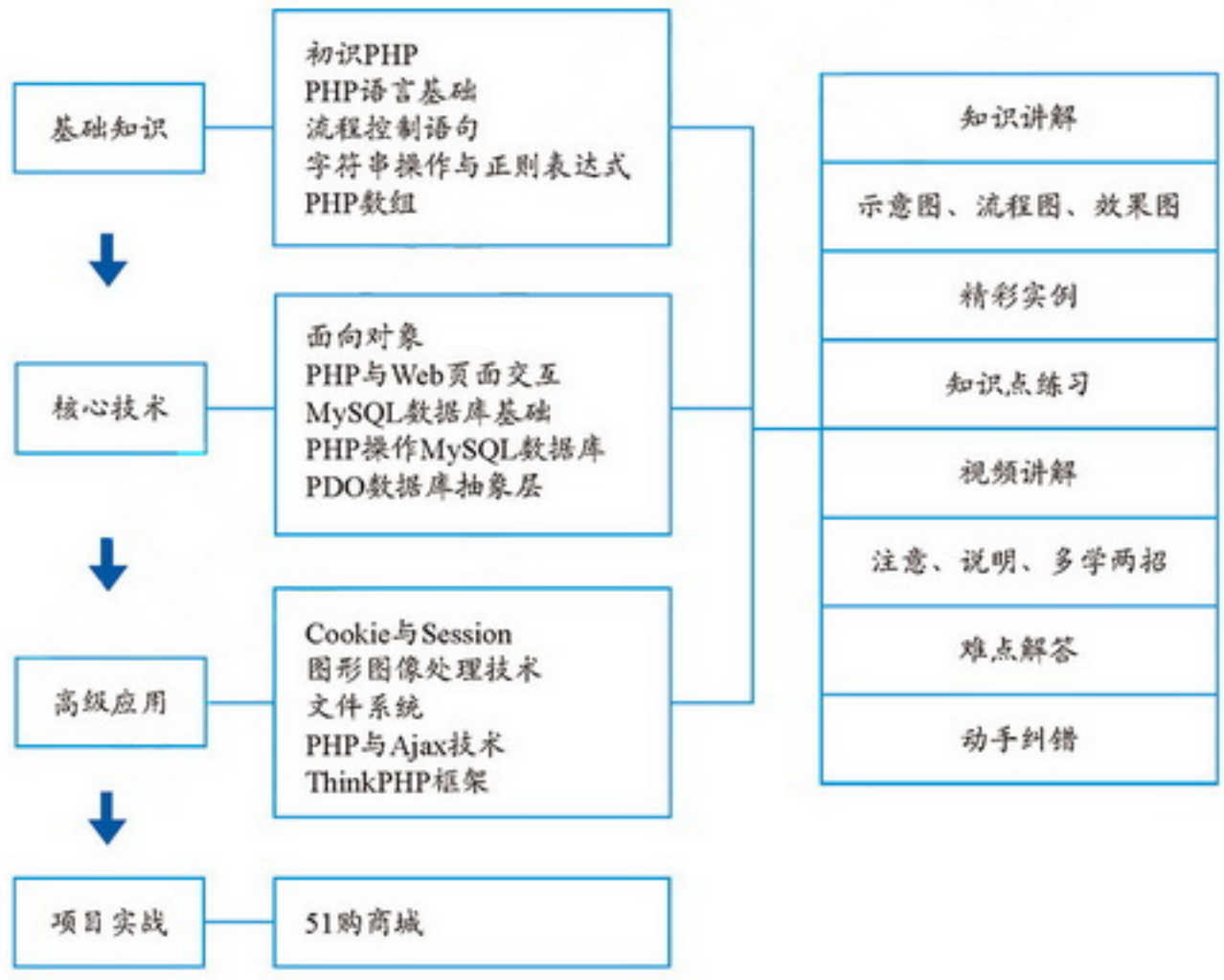
编程改变人生，代码改变世界。希望通过本书的学习，你不但可以成为编程的国王，更能通过编程，成就一番引以为豪的事业！

前　言

PHP 是一种在服务器端执行的嵌入 HTML 文档的脚本语言，是全球最普及、应用最广泛的互联网开发语言之一。PHP 语言因具有简单易学、源码开放、支持面向对象的编程，支持跨平台操作，而且完全免费等特点，越来越多地受到国内外知名企业及广大程序员的青睐，并迅速发展成为全球互联网社区领域应用位居第一的技术。

本书内容

本书从初学者角度出发，提供了从入门到成为程序开发高手所需要掌握的各方面知识和技术，图书知识体系如下图所示。



本书特色

☑ 由浅入深，编排合理——本书以零基础学习者为对象，采用图文结合、循序渐进的编排方式，由浅入深地讲解，适合初学者逐步掌握 Java 语言的语法规则和编程思想。

☑ 视频讲解，精彩详尽——书中每一章节都配有精彩详尽的视频讲解，知识点和实例讲解详尽到位，能够引导初学者快速入门，感受编程的快乐和成就感，快速成长为编程高手。

☑ 丰富实例，轻松易学——通过实例边学边练，是学习编程最有效的方式。本书通过“知识点＋精彩实例＋运行结果＋巩固练习＋项目实战”的模式，透彻解析程序开发中所需要的各方面知识，帮助初学者快速掌握编程技能。

☑ 贴心栏目，辅助学习——本书根据学习的需要，设置了“注意”“说明”“多学两招”“常见错误”等许多贴心小栏目，辅助读者轻松理解相关知识，避免不必要的错误，学会实用开发技巧。

☑ 纠错练习，巩固知识——书中每个实例后都配有练习题目，每个章节后都提供动手纠错练习，配合光盘中资源进行操作，读者可以进一步巩固所学知识点，更好地进行下一步学习。

☑ 编程词典（简易版）——本书为用户提供了明日科技研发的《编程词典（简易版）》，用户可以联系企业 QQ（4006751066）获取该资源。

读者对象

☑ 零基础的编程自学者
☑ 相关培训机构的老师和学生
☑ 编程爱好者
☑ 大中专院校的老师和学生
☑ 参加毕业设计的学生
☑ 初、中级程序开发人员

读者服务

为方便读者解决本书疑难问题，本书提供了企业服务 QQ：4006751066，明日学院网站（www.mingrisoft.com）还提供了社区服务和配套学习服务支持，我们将竭诚为您服务。

服务网站：www.mingrisoft.com
企业 Q Q：4006751066
服务信箱：mingrisoft@mingrisoft.com
客服电话：400 675 1066，0431-84978981，84978982

致读者

致读者

本书由明日科技 Java 程序开发团队策划并组织编写，主要编写人员有冯春龙、赛奎春、王国辉、李磊、申小琦、赵宁、王小科、张鑫、周佳星、白宏健、申野、贾景波、杨柳、葛忠月、隋妍妍、赵颖、李春林、辛洪郁、刘杰、宋万勇、杨丽、裴莹、刘媛媛、张云凯、吕玉翠、庞凤、孙巧辰、周艳梅、房雪坤、江玉贞、高春艳、梁英、胡冬、张宝华、何平、李菁菁、潘建羽、王博、张渤洋、岳彩龙等。在编写本书的过程中，我们本着科学、严谨的态度，力求精益求精，但疏漏之处在所难免，敬请广大读者批评指正。

感谢您阅读本书，希望本书能成为您编程路上的领航者。

零基础编程，一切皆有可能。

祝读书快乐！

编 者
2017 年 9 月

目 录
Content

第 5 章 PHP 数组 97

第 2 篇 核心技术

第 6 章 面向对象..... 118

第 4 篇　项目实战

第 1 篇

基础知识

- 第 1 章　初识 PHP
- 第 2 章　PHP 语言基础
- 第 3 章　流程控制语句
- 第 4 章　字符串操作与正则表达式
- 第 5 章　PHP 数组

第 1 章

初识 PHP

（视频讲解：1 小时）

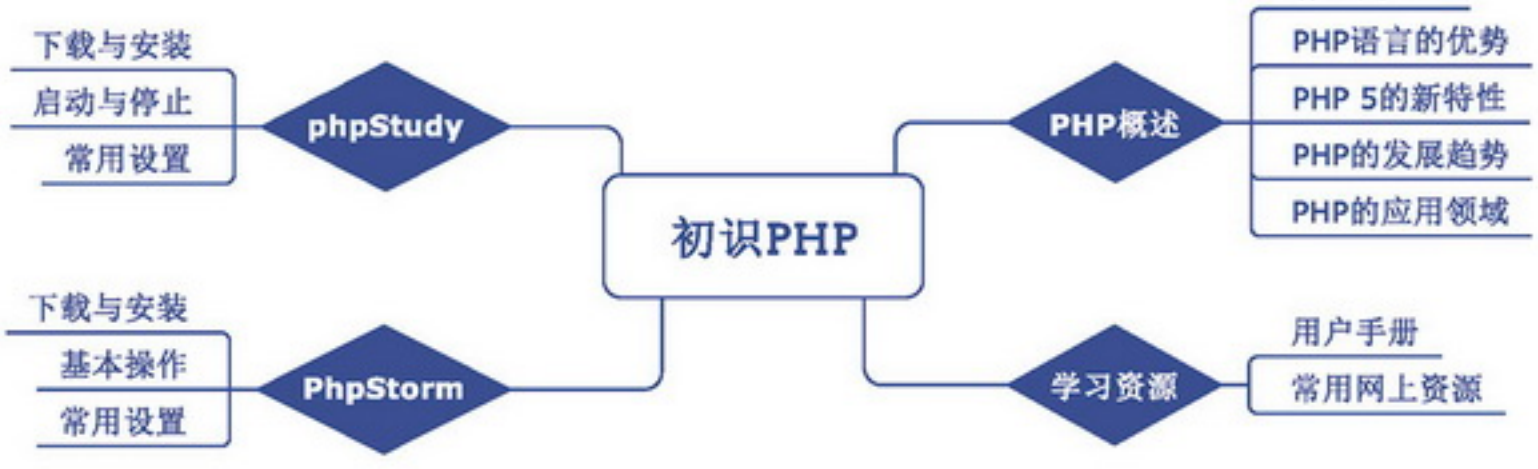
本章概览

PHP 是一种跨平台、HTML 嵌入式的服务器端脚本语言。PHP 语法结构简单，易于入门，很多功能只需一个函数即可实现。现在，PHP 已成为全球最受欢迎的脚本语言之一。

本章首先介绍了什么是 PHP、PHP 语言的优势、PHP 5 的新特性、PHP 的发展趋势、PHP 的应用领域，以及 PHP 的常用学习资源等。然后针对 Windows 用户，重点讲解了集成开发环境 phpStudy 的下载、安装、常用设置，以及 PhpStorm 的下载与安装、基本操作、常用设置等，并通过一个简单的“Hello World”程序引领读者体验 PHP 编程的过程。

“千里之行，始于足下！”赶快搭建好开发环境，开发属于你的 PHP 项目吧！

知识框架



1.1 PHP 概述

PHP 起源于 1995 年，由 Rasmus Lerdorf 开发。PHP 语法结构简单，易于入门，很多功能只需一个函数即可实现。到现在，PHP 已成为全球最受欢迎的脚本语言之一。

1.1.1 什么是 PHP

视频讲解：光盘\Video\01\1.1.1 什么是PHP.mp4

PHP 是 PHP:Hypertext Preprocessor（超文本预处理器）的缩写，是一种跨平台、HTML 嵌入式的服务器端脚本语言，其独特的语法混合了 C 语言、Java 语言和 Perl 语言的特点，是一种被广泛应用的、开源式的多用途脚本语言，尤其适合 Web 开发。

PHP 是 B/S（Browser/Server 的简写，即浏览器 / 服务器）体系结构，属于三层结构模式。服务器启动后，可以不使用相应的客户端软件，只使用浏览器即可访问，这样既保持了图形化的用户界面，又大大减少了应用维护量。

1.1.2 PHP 语言的优势

视频讲解：光盘\Video\01\1.1.2 PHP语言的优势.mp4

PHP 起源于自由软件，即开放源代码软件，使用 PHP 进行 Web 应用程序的开发具有以下优势：

◆ 安全性高：PHP 是开源软件，每个人都可以看到所有 PHP 的源代码，程序代码与 Apache 编译在一起的方式让它可以灵活地进行安全设定。PHP 具有公认的安全性能。

◆ 跨平台特性：PHP 几乎支持所有的操作系统平台（如 Windows 或 UNIX/Linux/Macintosh/FreeBSD/OS2 等），并且支持 Apache、Nginx、IIS 等多种 Web 服务器，因此广为流行。

◆ 支持广泛的数据库：可操作多种主流与非主流的数据库，如 MySQL、Access、SQL Server、Oracle、DB2 等，其中 PHP 与 MySQL 是目前最佳的组合，它们的组合可以跨平台运行。

◆ 易学性：PHP 嵌入在 HTML 语言中，以脚本语言为主，内置丰富函数，语法简单，书写容易，方便学习掌握。

◆ 执行速度快：PHP 占用系统资源少，代码执行速度快。

◆ 免费：在流行的企业应用平台 LAMP 中，Linux、Apache、MySQL、PHP 都是免费软件，这种开源免费的框架结构可以为网站经营者节省很大一笔开支。

1.1.3 PHP 5 的新特性

视频讲解：光盘\Video\01\1.1.3 PHP 5的新特性.mp4

如果刚开始学习 PHP，建议使用 PHP 5.5 或以上版本。PHP 近年来有了巨大的改进，增加了许多

强大的特性。虽然从 PHP 5.2 到 PHP 5.6 之间增加的版本号看上去很小，但它却代表了重大的改进。如果想查找某个函数及其用法，可以去官方手册中查找。以下是 PHP 5 中新增加的对象模式：

- 构造函数和析构函数
- 对象的引用
- 对象的克隆（clone）
- 对象中的公共、私有及受保护模式（public、private 和 protected 关键字）
- 接口（Interface）
- 抽象类
- 魔术方法
- 静态成员

1.1.4 PHP 的发展趋势

视频讲解：光盘\Video\01\1.1.4 PHP的发展趋势.mp4

由于 PHP 是一种面向对象的、完全跨平台的新型 Web 开发语言，所以无论从开发者角度考虑还是从经济角度考虑，都是非常实用的。PHP 语法结构简单，易于入门，很多功能只需一个函数就可以实现，并且很多机构都相继推出了用于开发 PHP 的 IDE 工具。

现在，越来越多的新公司或者新项目使用 PHP，这使得 PHP 相关社区越来越活跃，而这又反过来影响到很多项目或公司的选择，形成一个良性循环，因此，PHP 是国内大部分 Web 项目开发的首选语言。PHP 运行速度快，开发成本低，后期维护费用低，开源产品丰富，这些都是很多语言无法比拟的。而随着移动互联网技术的兴起，越来越多的 Web 应用也选择了 PHP 作为主流的开发语言。

W3Techs 网站提供了服务器端脚本语言的市场占有率，截止到 2016 年 12 月 12 日，PHP 在所有服务器端语言市场的占有率高达 82.4%，遥遥领先于其他编程语言，如图 1.1 所示。

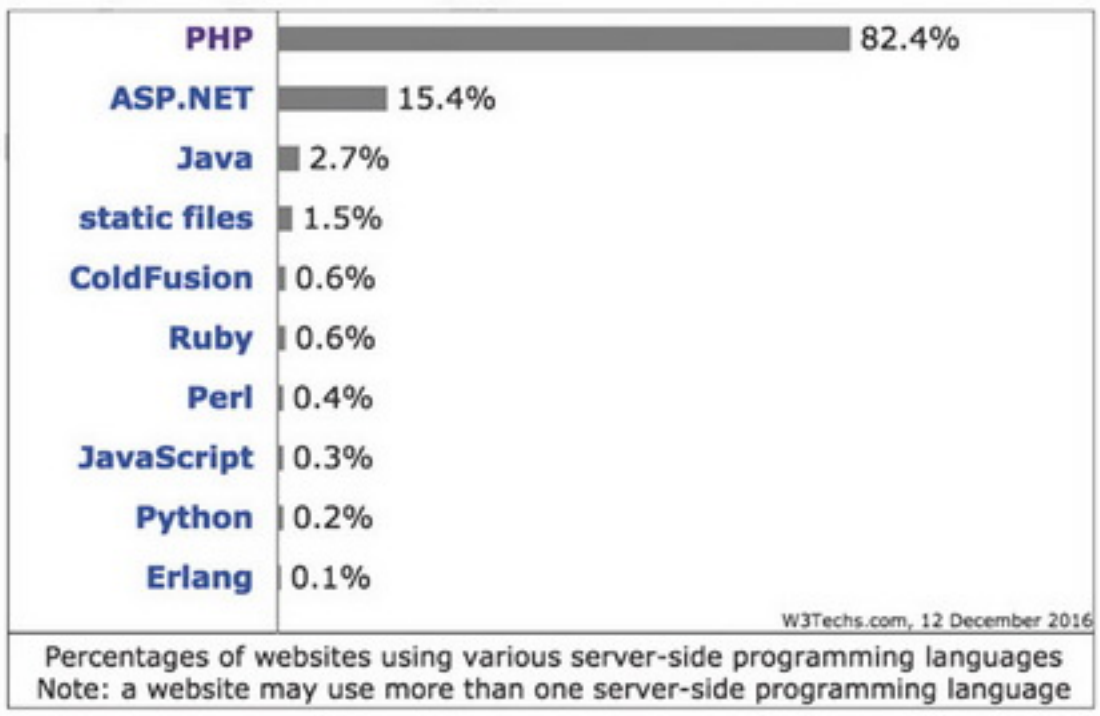


图 1.1　服务器端脚本语言市场占有率

PHP 的将来是由 PHP 7 决定的，再来看一下 PHP 7 的表现。Zend 公司发布了 PHP 与其他脚本语言运行效率的对比图，PHP 7 在动态语言运行效率中表现同样出色，如图 1.2 所示。

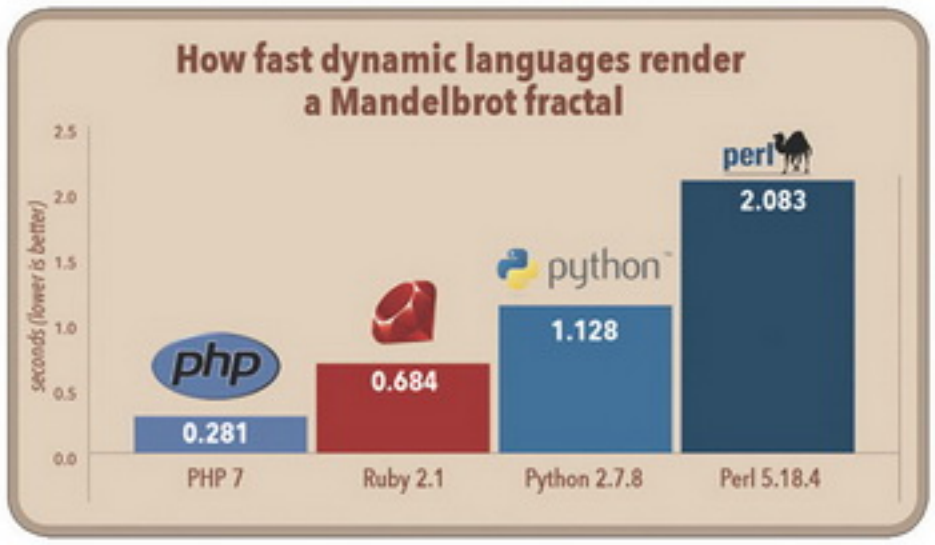


图 1.2　PHP 与其他脚本语言运行效率对比

1.1.5 PHP 的应用领域

视频讲解：光盘\Video\01\1.1.5 PHP的应用领域.mp4

在互联网高速发展的今天，PHP 的应用范围非常广泛，其应用领域主要包括：

- 中、小型网站的开发
- 大型网站的业务逻辑结果展示
- Web 办公管理系统
- 硬件管控软件的 GUI
- 电子商务应用
- Web 应用系统开发
- 多媒体系统开发
- 企业级应用开发
- 移动互联网开发

PHP 正吸引着越来越多的 Web 开发人员学习和使用。PHP 无处不在，它可应用于任何地方、任何领域，并且已拥有几百万个用户，其发展速度要快于在它之前的任何一种计算机语言。据最新数据统计，全世界有超过 2200 万个网站和 1.5 万家公司在使用 PHP 语言，包括百度、雅虎、Facebook、淘宝、腾讯、新浪、搜狐等知名网站。

1.2 学习资源

下面为读者推荐一些学习 PHP 的相关资源。使用这些资源，可以帮助读者找到掌握 PHP 的捷径。

1.2.1 PHP 用户手册

视频讲解：光盘\Video\01\1.2.1 PHP用户手册.mp4

学习 PHP 语言，配备《PHP 手册》是非常必要的，就像小学生在学习汉字时手中必须有一本《新

华字典》一样。《PHP 手册》对 PHP 的函数进行了详细的讲解和说明，并且还给出了一些简单的示例，同时还对 PHP 的安装与配置、语言参考、安全和特点等内容进行了介绍。

在 http://www.php.net/docs.php 网站上，提供了 PHP 的各种语言、格式和版本的参考手册，读者可以进行在线阅读，也可以下载。

《PHP 手册》不但对 PHP 的函数进行了解释和说明，而且还提供了快速查找的方法，让用户可以更加方便地查找到指定的函数。《PHP 手册》下载的 chm 版如图 1.3 所示。

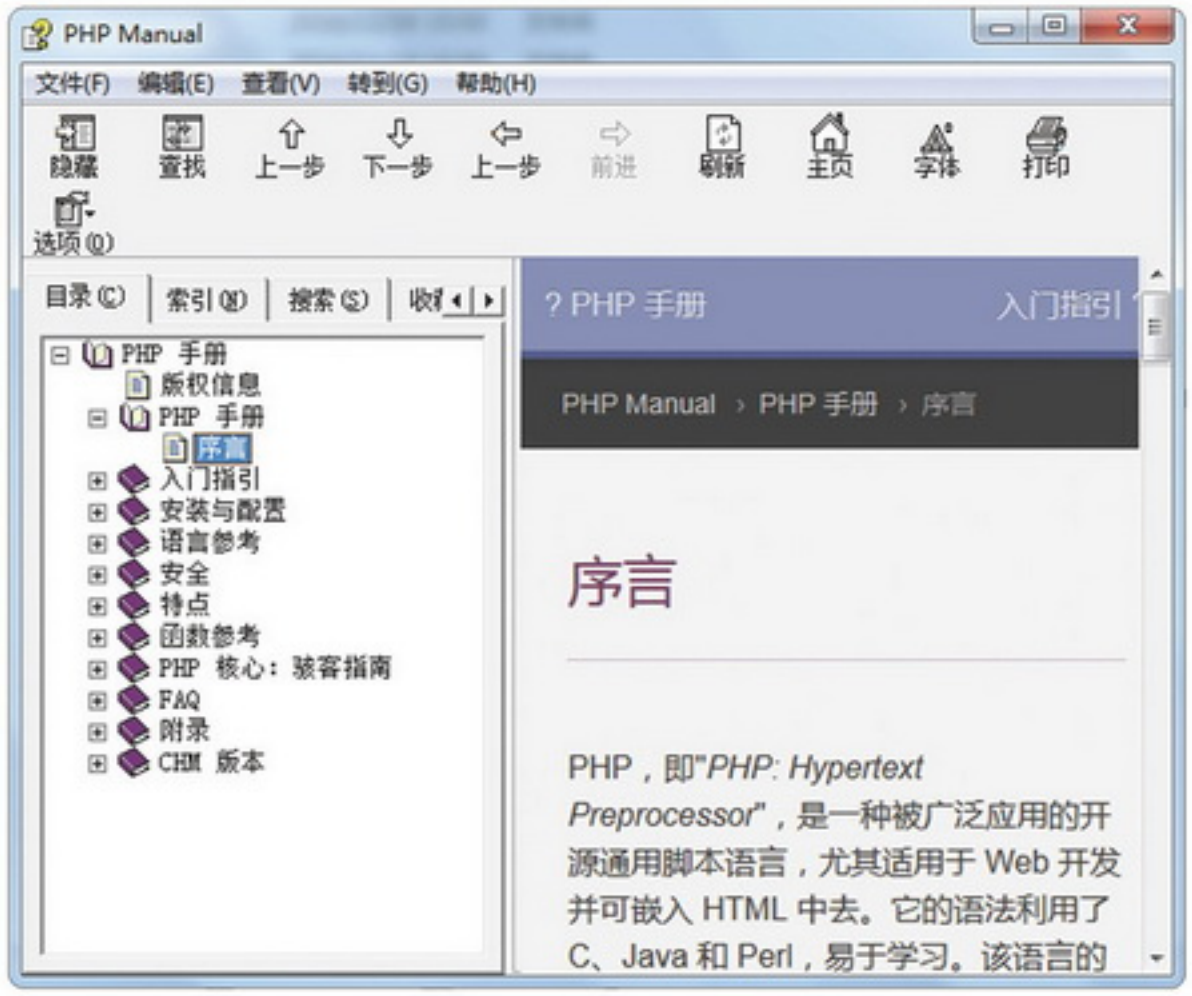


图 1.3 《PHP 手册》下载版

1.2.2 常用网上资源

视频讲解：光盘\Video\01\1.2.2 常用网上资源.mp4

下面提供一些大型的 PHP 技术网站，这些网站中提供的资源不但可以提高 PHP 编程者的技术水平，也是程序员学习和工作的好帮手。

1. PHP 官网

http://www.php.net

2. PHP 相关学习网站

◆ 明日学院官方网站

http://www.mingrisoft.com

◆ 慕课网

http://www.imooc.com

1.3 搭建 PHP 运行环境

在使用 PHP 进行开发前，首先需要搭建 PHP 运行环境。对于 PHP 语言的初学者来说，Apache、PHP 以及 MySQL 的安装和配置较为复杂，这时，可以选择集成安装环境快速安装、配置 PHP 服务器。集成安装环境就是将 Apache、PHP 和 MySQL 等服务器软件整合在一起，免去了单独安装配置服务器带来的麻烦，实现了 PHP 开发环境的快速搭建。

目前，比较常用的集成安装环境是 phpStudy、WampServer 和 AppServ 等，它们都集成了 Apache 服务器、PHP 预处理器以及 MySQL 服务器。本书以 phpStudy 为例介绍 PHP 服务器的安装与配置。由于 phpStudy 的版本会不断更新，因此，这里以常用的 phpStudy 2016（以下简称 phpStudy）为例介绍 phpStudy 的下载和安装。

1.3.1 phpStudy 的下载与安装

视频讲解：光盘\Video\01\1.3.1 phpStudy的下载与安装.mp4

phpStudy 官方网站的地址为：http://www.phpstudy.net，通过访问 phpStudy 的官方网站就可以下载 phpStudy。

下面以 Windows 7（64 位）系统为例，讲解 phpStudy 的安装步骤。

（1）下载完 phpStudy 安装文件的压缩包后，对该压缩包进行解压缩，然后双击 phpStudy2016.exe 安装文件，此时将弹出如图 1.4 所示的对话框。使用默认安装路径，单击“确定”按钮，运行效果如图 1.5 所示。

图 1.4　phpStudy 解压对话框

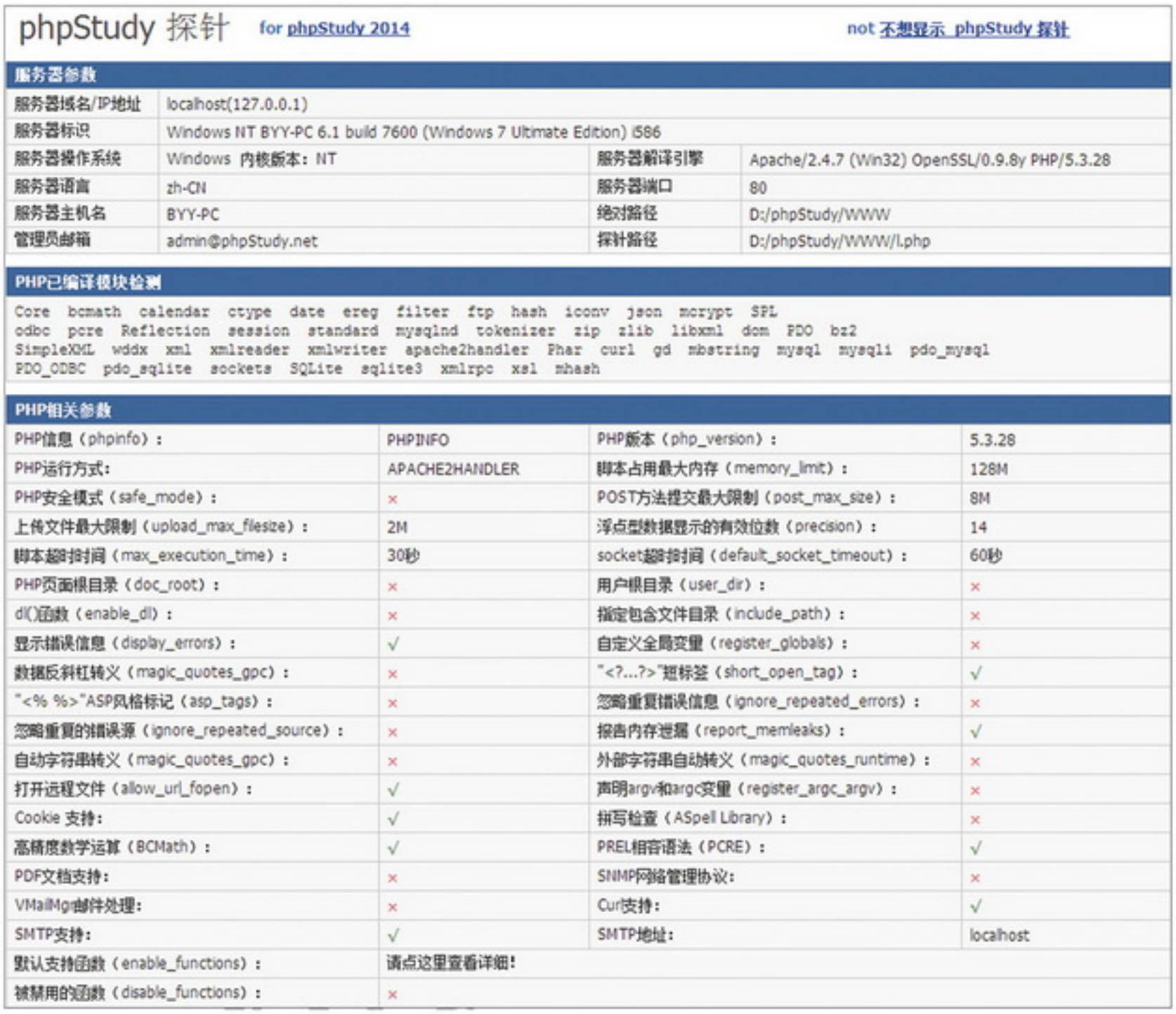
图 1.5　解压文件进度条

（2）解压文件完成后会弹出防止重复初始化的确认对话框，如图 1.6 所示。单击“是”按钮后进入 phpStudy 的启动界面，启动完成后的结果如图 1.7 所示。

图 1.6　防止重复初始化确认对话框

图 1.7　phpStudy 启动界面

在 Apache 服务和 MySQL 服务启动成功之后，即完成了 phpStudy 的安装操作。打开浏览器，在地址栏中输入“http://localhost/l.php”或者“http://127.0.0.1/l.php”后按下 <Enter> 键，如果运行后出现如图 1.8 所示的页面，则说明 phpStudy 安装成功。



phpStudy 探针 for phpStudy 2014 not 不想显示 phpStudy 探针

服务器参数			
服务器域名/IP地址	localhost(127.0.0.1)		
服务器标识	Windows NT BYY-PC 6.1 build 7600 (Windows 7 Ultimate Edition) i586		
服务器操作系统	Windows 内核版本：NT	服务器解译引擎	Apache/2.4.7 (Win32) OpenSSL/0.9.8y PHP/5.3.28
服务器语言	zh-CN	服务器端口	80
服务器主机名	BYY-PC	绝对路径	D:/phpStudy/WWW
管理员邮箱	admin@phpStudy.net	探针路径	D:/phpStudy/WWW/l.php

PHP已编译模块检测

Core bcmath calendar ctype date ereg filter ftp hash iconv json mcrypt SPL
odbc pcre Reflection session standard mysqlnd tokenizer zip zlib libxml dom PDO bz2
SimpleXML wddx xml xmlreader xmlwriter apache2handler Phar curl gd mbstring mysql mysqli pdo_mysql
PDO_ODBC pdo_sqlite sockets SQLite sqlite3 xmlrpc xsl mhash

PHP相关参数			
PHP信息（phpinfo）：	PHPINFO	PHP版本（php_version）：	5.3.28
PHP运行方式：	APACHE2HANDLER	脚本占用最大内存（memory_limit）：	128M
PHP安全模式（safe_mode）：	×	POST方法提交最大限制（post_max_size）：	8M
上传文件最大限制（upload_max_filesize）：	2M	浮点型数据显示的有效位数（precision）：	14
脚本超时时间（max_execution_time）：	30秒	socket超时时间（default_socket_timeout）：	60秒
PHP页面根目录（doc_root）：	×	用户根目录（user_dir）：	×
dl()函数（enable_dl）：	×	指定包含文件目录（include_path）：	×
显示错误信息（display_errors）：	√	自定义全局变量（register_globals）：	×
数据反斜杠转义（magic_quotes_gpc）：	×	"<?...?>"短标签（short_open_tag）：	√
"<% %>"ASP风格标记（asp_tags）：	×	忽略重复错误信息（ignore_repeated_errors）：	×
忽略重复的错误源（ignore_repeated_source）：	×	报告内存泄漏（report_memleaks）：	√
自动字符串转义（magic_quotes_gpc）：	×	外部字符串自动转义（magic_quotes_runtime）：	×
打开远程文件（allow_url_fopen）：	√	声明argv和argc变量（register_argc_argv）：	×
Cookie 支持：	√	拼写检查（ASpell Library）：	×
高精度数学运算（BCMath）：	√	PREL相容语法（PCRE）：	√
PDF文档支持：	×	SNMP网络管理协议：	×
VMailMgr邮件处理：	×	Curl支持：	√
SMTP支持：	√	SMTP地址：	localhost
默认支持函数（enable_functions）：	请点这里查看详细！		
被禁用的函数（disable_functions）：	×		

图 1.8　phpStudy 安装成功运行页面

📖 **说明**：如果提示“没有安装 VC 9 运行库”，则需要到微软官方网站下载。下载地址为：http://www.microsoft.com/zh-CN/download/details.aspx?id=5582。

（3）phpStudy 启动失败时的解决方法如下：

◆ 失败原因：防火墙拦截

为了减少出错，安装路径不得有汉字。如有防火墙开启，会提示是否信任 httpd、mysqld 运行，请选择全部允许。

◆ 失败原因：80 端口已经被其他程序占用，如 IIS、迅雷等

由于端口问题无法启动时，请选择 phpStudy 界面中的“其他选项菜单”→“环境端口检测”→“检测端口”→“尝试强制关闭相关进程并启动”菜单项，如图 1.9 所示。

📖 **说明**：如强制关闭相关进程并启动后仍无法启动 phpStudy，请检测端口被哪一个进程所占用，然后手动关闭该进程。

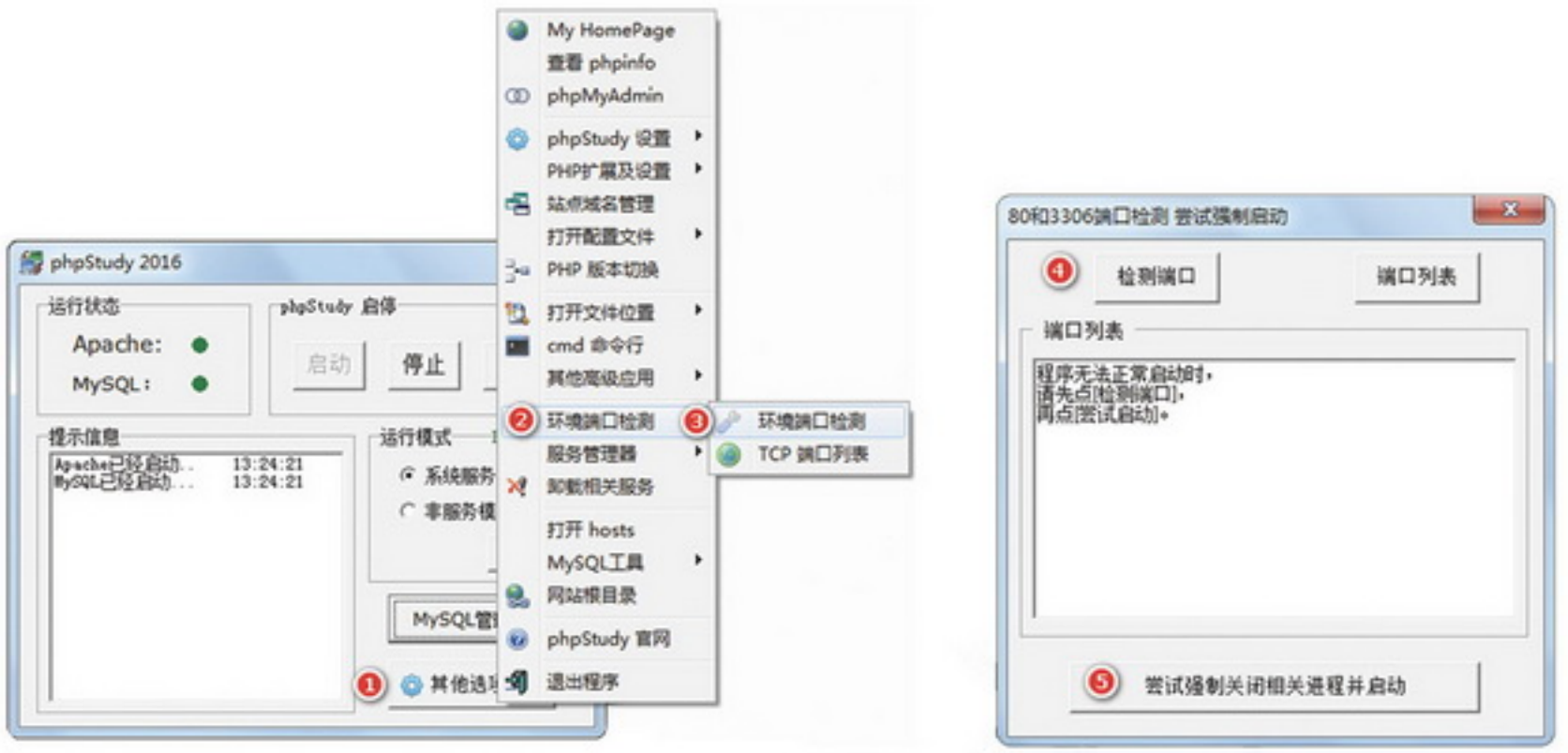


图 1.9　phpStudy 检测端口

1.3.2 PHP 服务器的启动与停止

视频讲解：光盘\Video\01\1.3.2 PHP服务器的启动与停止.mp4

PHP 服务器主要包括 Apache 服务器和 MySQL 服务器。重新启动计算机后，在默认状态下，Apache 服务和 MySQL 服务是停止的，下面介绍在 phpStudy 中启动与停止这两种服务器的方法。

1. 启动和停止服务器

双击 phpStudy 快捷方式图标打开 phpStudy，打开后的界面如图 1.10 所示，单击“启动”按钮即可同时启动 Apache 服务和 MySQL 服务，启动后的结果如图 1.11 所示。

图 1.10　phpStudy 的打开界面

图 1.11　启动服务

如果想要停止 Apache 服务和 MySQL 服务，只需要单击图 1.11 中的“停止”按钮即可。另外，单击图 1.11 中的“重启”按钮还可以重启这两种服务。

2. 设置开机自动启动服务

在 phpStudy 的启动界面，只需单击“系统服务”单选按钮，然后单击“应用”按钮即可实现开机自动启动服务的功能，如图 1.12 所示。

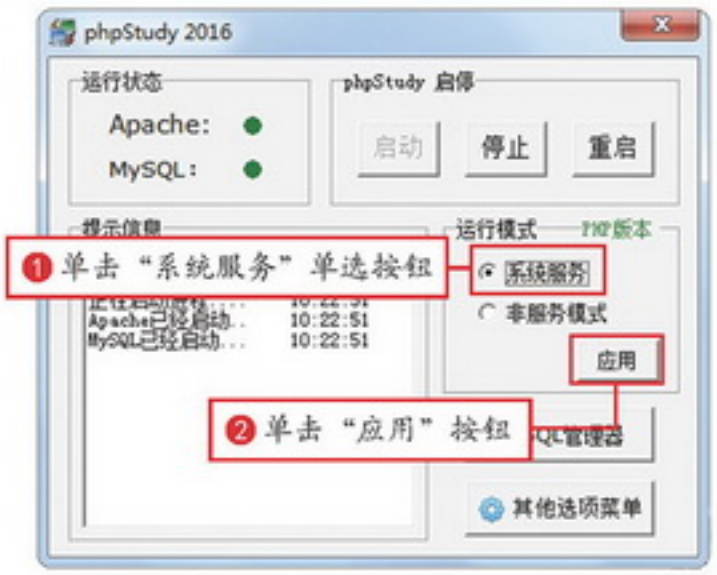


图 1.12　设置开机自动启动服务

1.3.3 phpStudy 的常用设置

视频讲解：光盘\Video\01\1.3.3 phpStudy的常用设置.mp4

phpStudy 的强大之处在于它配置的灵活性，用户可以根据个人需求，方便快捷地进行相关设置。下面就介绍 phpStudy 的一些常用设置。

1. PHP 版本切换

phpStudy 启动后，默认使用的 PHP 版本是 Apache + PHP 5.3，如果需要使用其他的服务器（如 Nginx）或其他的 PHP 版本，可以使用 phpStudy 快速切换。在 phpstudy 启动界面，选择“其他选项菜单”→“PHP 版本切换”→“选择版本”→“应用”即可，如图 1.13 所示。

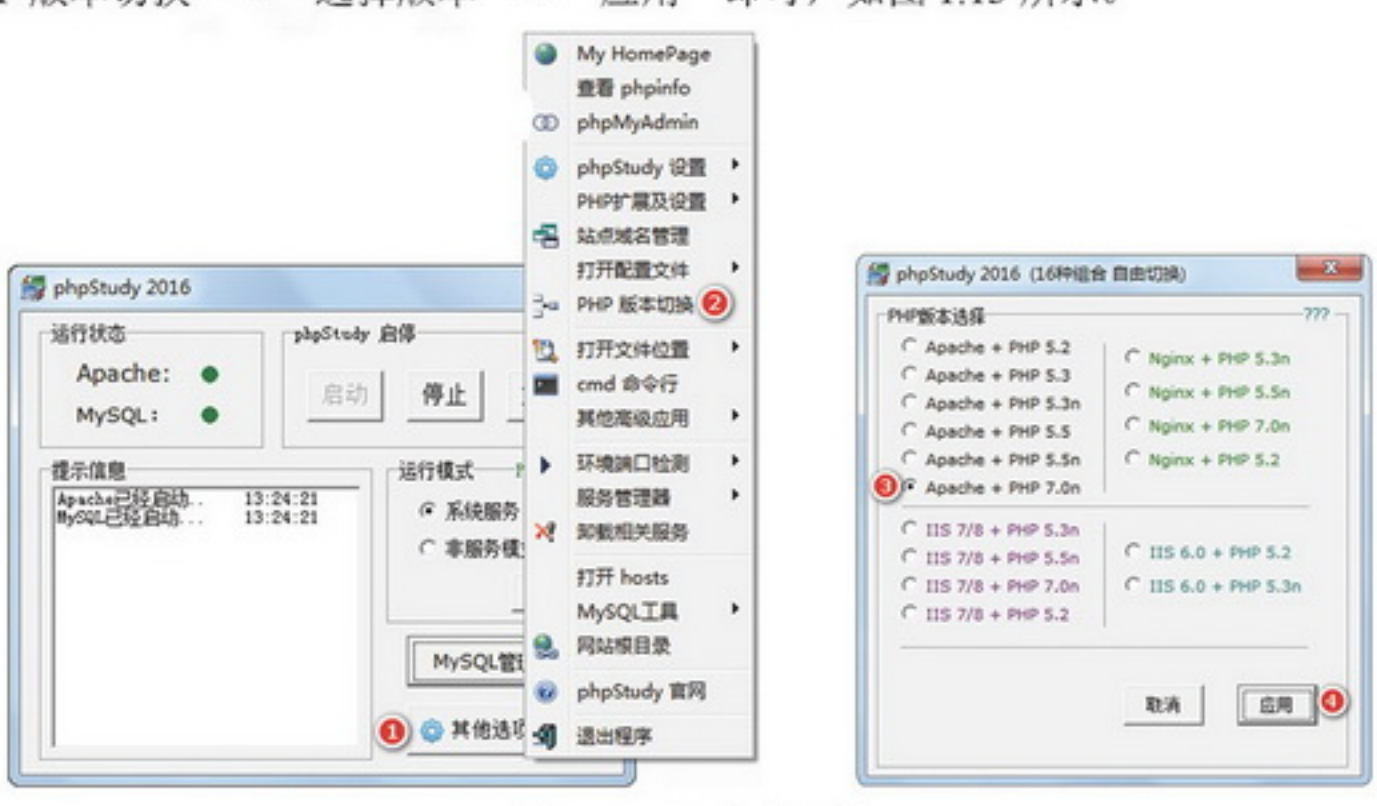


图 1.13　PHP 版本切换

注意：PHP 5.3、PHP 5.4 和 Apache 都是用 VC 9 编译，使用时必须安装 VC 9 运行库才能运行；PHP 5.5、PHP 5.6 是用 VC 11 编译，使用时必须安装 VC 11 运行库；PHP 7.0、PHP 7.1 是用 VC 14 编译，使用时必须安装 VC 14 运行库。

2. 开启 PHP 扩展设置

在开发某些项目时，会使用 PHP 扩展库中的扩展。通常情况下，如果要开启某个扩展，以 php_fileinfo.dll(Bzip 2 压缩函数库) 为例，则需要打开 php.ini 文件，修改后代码如下：

```
extension=php_fileinfo.dll                    //去除前面的分号
```

现在，使用 phpStudy 开启扩展，操作过程将变得非常简单，只需选择“其他选项菜单”→“PHP 扩展及设置”→“PHP 扩展”→“勾选相应的扩展”即可，如图 1.14 所示。

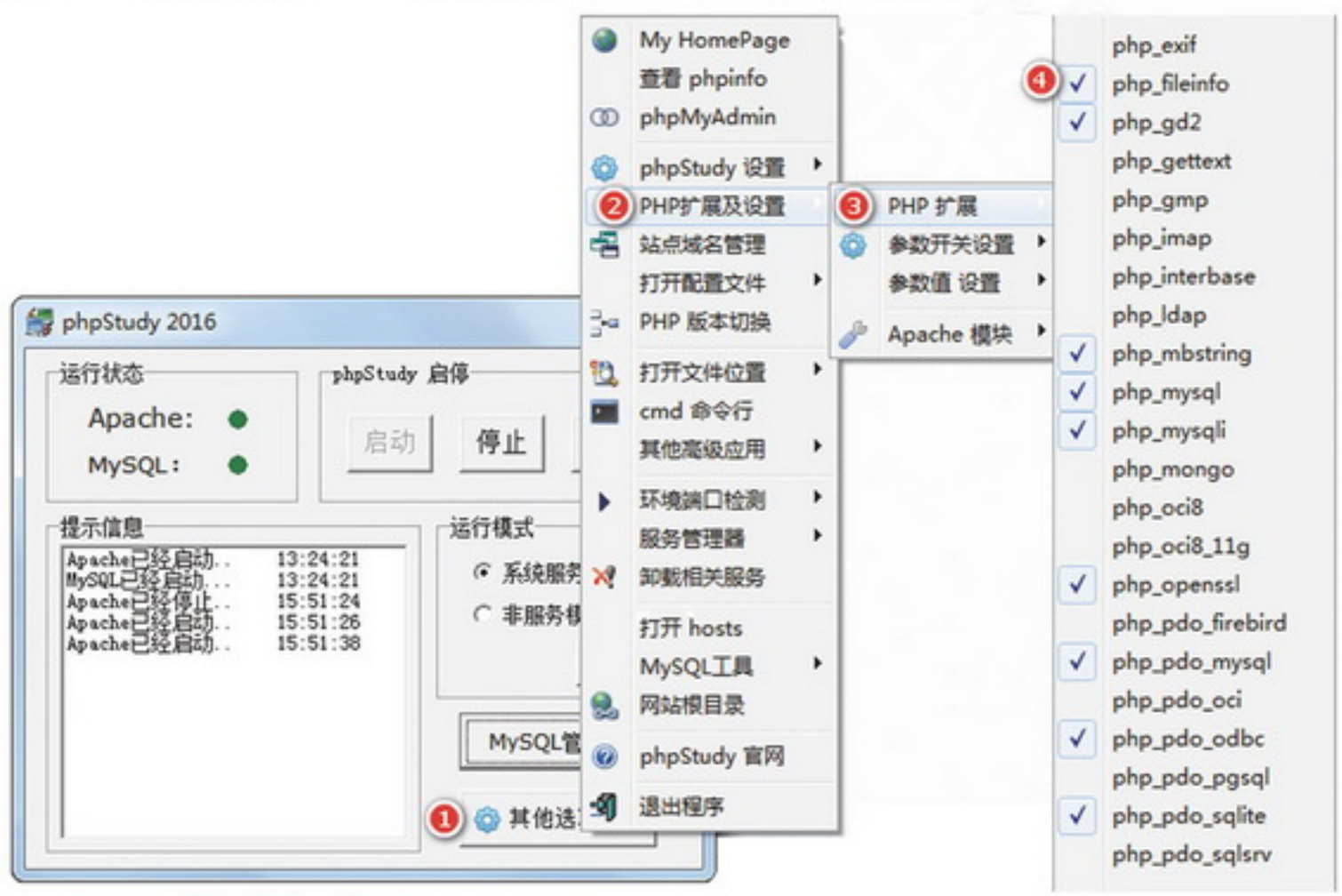


图 1.14　开启 PHP 扩展

1.4 PhpStorm 的下载与安装

PHP 的开发工具很多，每种开发工具都有其各自的优势。在编写程序时，一款好的开发工具会使开发人员的编码过程更加轻松、有效和快捷，达到事半功倍的效果。本书中是以 PhpStorm 为开发工具对 PHP 程序进行开发的。应用 PhpStorm 开发 PHP 程序有许多优点，它可以提高用户效率，提供智能代码补全、快速导航以及即时错误检查的功能。由于 PhpStorm 的版本会不断更新，因此这里以常用的 PhpStorm 9.0.3（以下简称 PhpStorm）为例，介绍 PhpStorm 的下载和安装。

1.4.1 PhpStorm 的下载

视频讲解：光盘\Video\01\1.4.1 PhpStorm的下载.mp4

PhpStorm 是 JetBrains 公司开发的一款商业的 PHP 集成开发工具，其不同版本可以通过官方网站进行下载。下载地址为：http://www.jetbrains.com/phpstorm。

下载 PhpStorm 的步骤如下：

（1）在浏览器中输入“http://www.jetbrains.com/phpstorm”，按下 <Enter> 键进入 PhpStorm 的主页面，如图 1.15 所示。

图 1.15　PhpStorm 的主页面

（2）在 PhpStorm 主页面中，单击如图 1.15 所示的 Download 按钮，在打开的页面中找到 Previous versions 超链接，如图 1.16 所示。

图 1.16　找到 Previous versions 超链接

（3）单击如图 1.16 所示的 Previous versions 超链接，进入 PhpStorm 不同版本的下载页面，在下载页面中找到 PhpStorm 9.0.3 的下载链接，如图 1.17 所示。

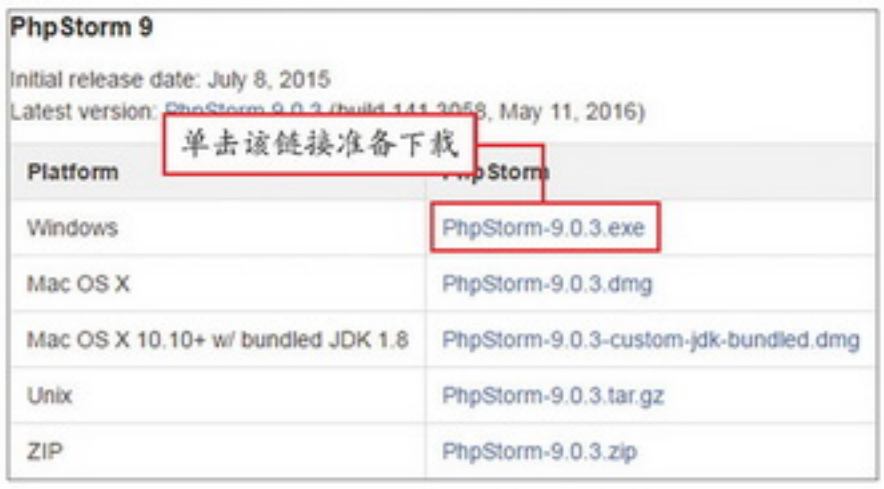


图 1.17　PhpStorm 9.0.3 的下载页面

（4）单击如图 1.17 所示的 PhpStorm-9.0.3.exe 超链接弹出下载对话框，单击对话框中的“下载”按钮即可将 PhpStorm 的安装文件下载到本地计算机上。

1.4.2 PhpStorm 的安装

视频讲解：光盘\Video\01\1.4.2 PhpStorm的安装.mp4

PhpStorm 的安装步骤如下：

（1）PhpStorm 下载完成后，双击 PhpStorm-9.0.3.exe 安装文件，打开 PhpStorm 的安装欢迎界面，如图 1.18 所示。

（2）单击 Next 按钮，打开 PhpStorm 的许可协议界面，如图 1.19 所示。

图 1.18　PhpStorm 安装欢迎界面

图 1.19　PhpStorm 许可协议界面

（3）单击 I Agree 按钮，打开 PhpStorm 的选择安装路径界面，如图 1.20 所示。在该界面中可以设置 PhpStorm 的安装路径，这里将安装路径设置为“D:\PhpStorm 9.0.3”。

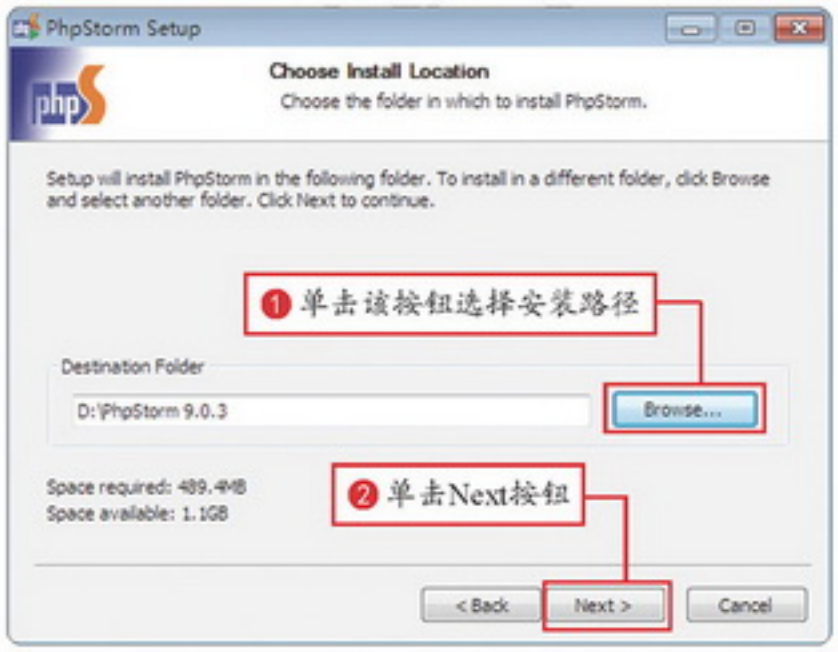


图 1.20　PhpStorm 选择安装路径界面

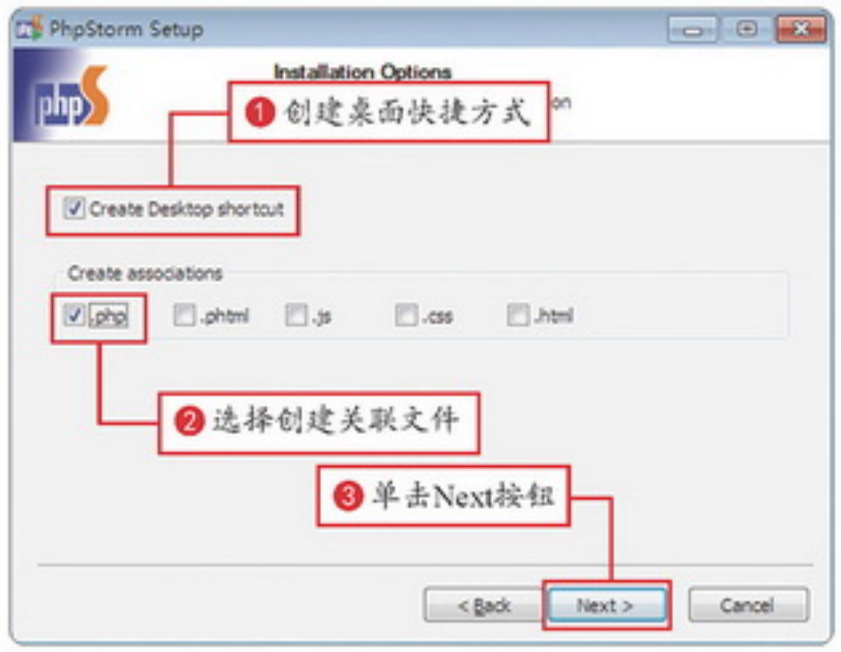


图 1.21　PhpStorm 安装选项界面

（4）设置好 PhpStorm 的安装路径后，单击 Next 按钮，打开 PhpStorm 的安装选项界面，如图 1.21 所示。在该界面中可以设置是否创建 PhpStorm 的桌面快捷方式，以及选择创建关联文件。

（5）设置完成后，单击 Next 按钮，打开 PhpStorm 的选择开始菜单文件夹界面，如图 1.22 所示。

（6）单击如图 1.22 所示的 Install 按钮开始安装 PhpStorm，正在安装界面如图 1.23 所示。

图 1.22　PhpStorm 的选择开始菜单文件夹界面

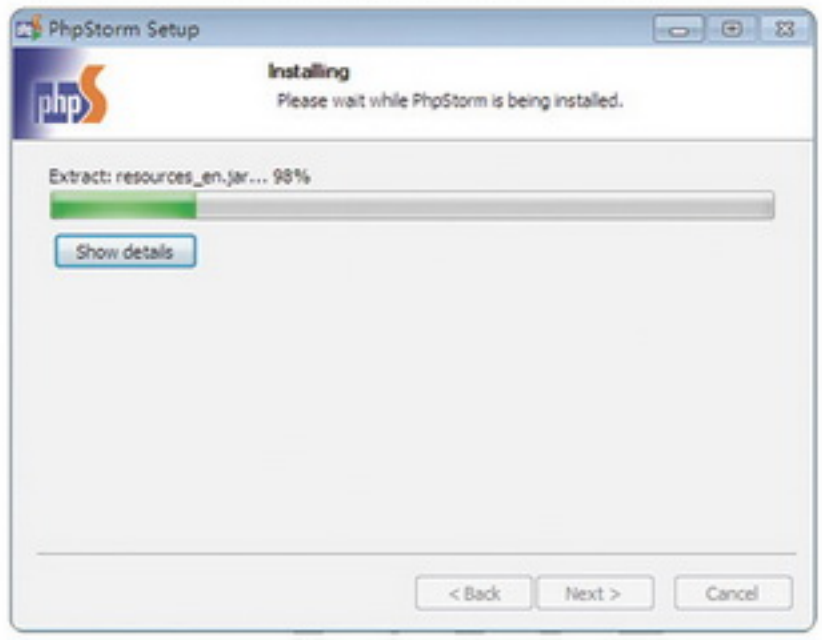


图 1.23　PhpStorm 正在安装界面

（7）安装结束后会打开如图 1.24 所示的完成安装界面，在该界面中选中 Run PhpStorm 前面的复选框，然后单击 Finish 按钮即可运行 PhpStorm。

图 1.24　PhpStorm 完成安装界面

（8）首次运行 PhpStorm 时，会弹出如图 1.25 所示的对话框，提示用户是否需要导入 PhpStorm 上一版本的配置，这里保持默认选项即可。

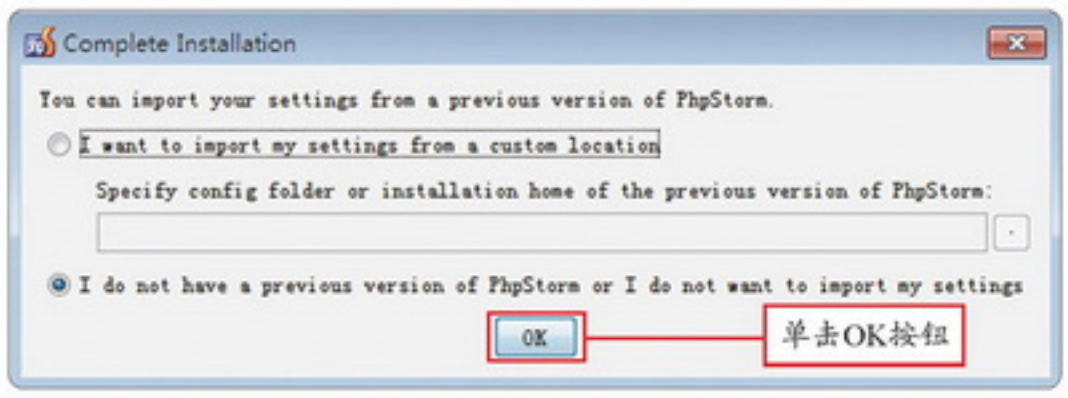


图 1.25　是否导入上一版本配置提示对话框

（9）单击如图 1.25 所示的 OK 按钮，打开 PhpStorm 的许可证激活界面，如图 1.26 所示。由于 PhpStorm 是收费软件，因此，这里选择的是 30 天试用版。如果想使用正式版，可以通过官方渠道购买。

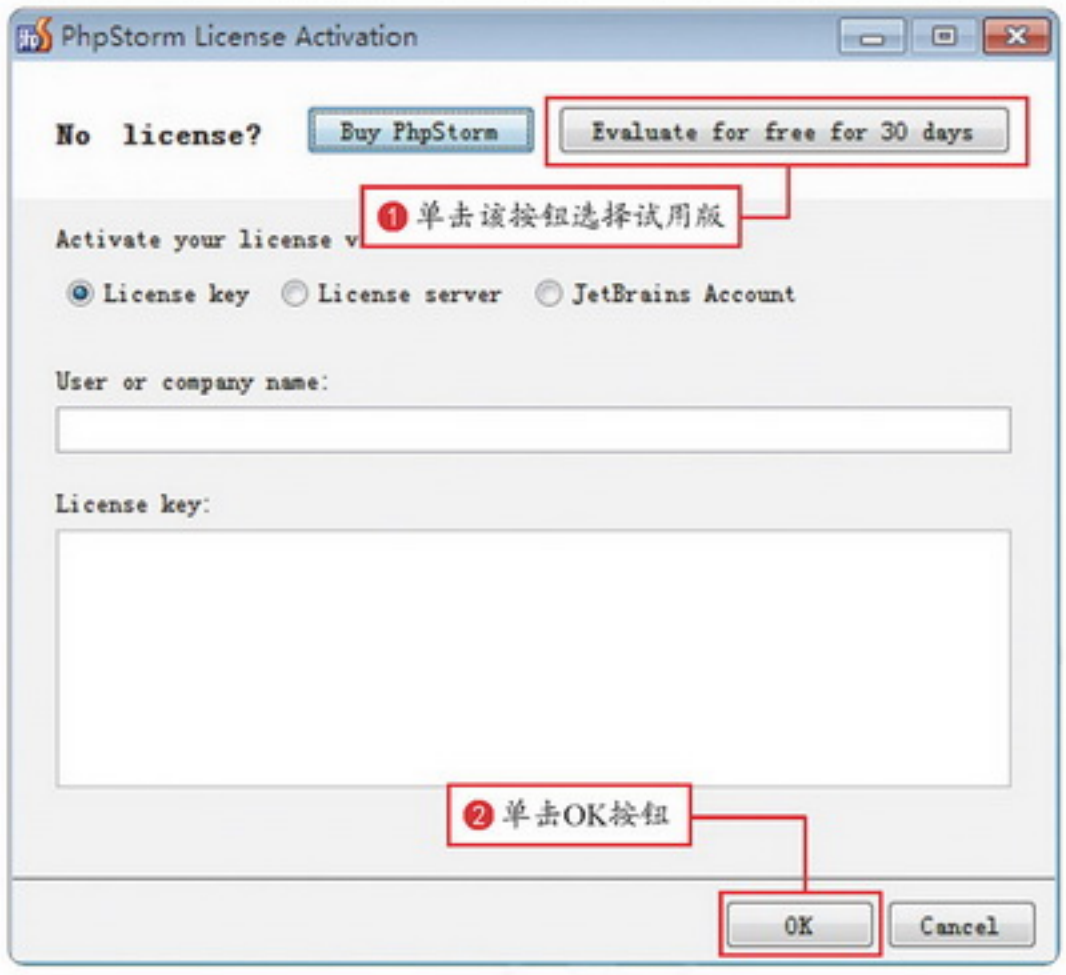


图 1.26　PhpStorm 许可证激活界面

（10）单击 Evaluate for free for 30 days 按钮选择 30 天试用版，然后单击 OK 按钮，将打开 PhpStorm 的许可协议界面，如图 1.27 所示。

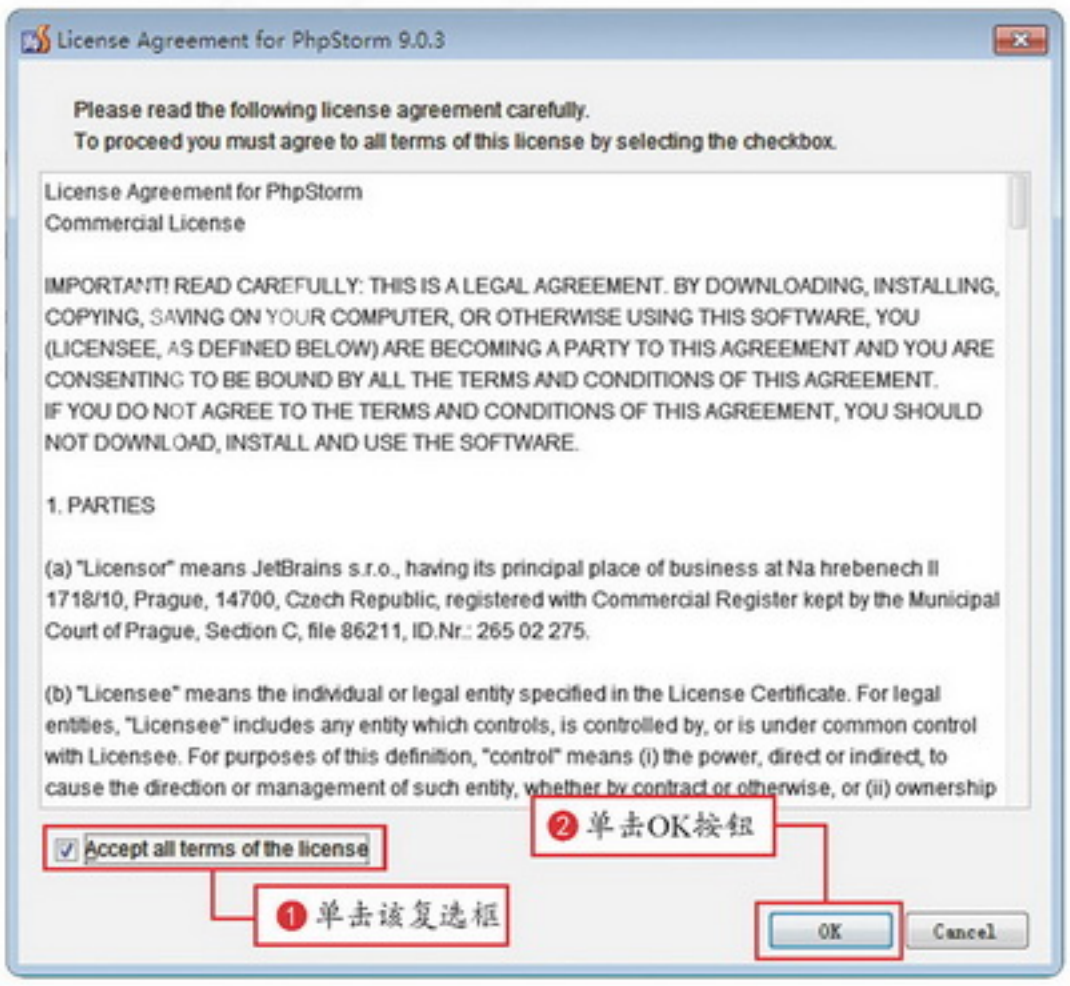


图 1.27　PhpStorm 许可协议界面

（11）单击如图 1.27 所示的 Accept all terms of the license 复选框接受许可协议，然后单击 OK 按钮，如图 1.28 所示。这里保持默认选项即可。

图 1.28　PhpStorm 初始配置对话框

（12）单击如图 1.28 所示的 OK 按钮关闭初始配置对话框，将打开 PhpStorm 的欢迎界面，如图 1.29 所示。这时，就表示 PhpStorm 启动成功。

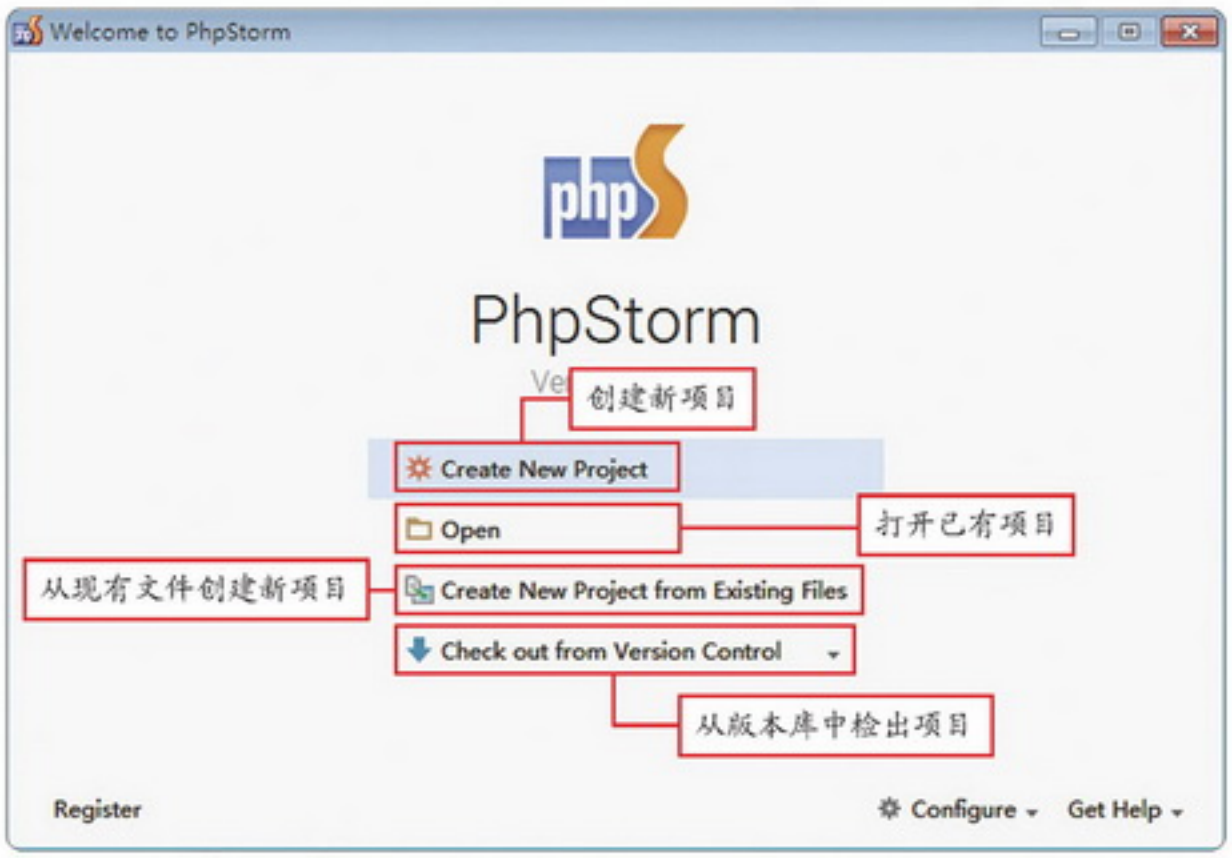


图 1.29　PhpStorm 欢迎界面

1.5 PhpStorm 基本操作

1.5.1 创建 PHP 项目

视频讲解

视频讲解：光盘\Video\01\1.5.1 创建PHP项目.mp4

PhpStorm 安装完成后，如果还没有创建项目，在首次启动时将进入如图 1.29 所示的欢迎界面。在

该界面可以进行创建新项目、打开已经存在的项目等操作。

创建 PHP 项目的具体步骤如下：

（1）在 PhpStorm 的欢迎界面中单击 Create New Project 按钮，进入创建新项目对话框，如图 1.30 所示。在对话框中首先选择项目存储路径，将项目文件夹存储在“D:\phpStudy\WWW”目录下，然后输入新创建的项目名称“myProject”，最后单击 OK 按钮即可完成 PHP 新项目的创建。

图 1.30　创建新项目对话框

（2）创建项目后会打开 PhpStorm 的主界面，在主界面的左侧显示新建项目的名称以及自动生成的文件，如图 1.31 所示。同时会弹出如图 1.32 所示的提示框，单击 Close 按钮将其关闭。

图 1.31　创建后的项目目录

图 1.32　提示框

📖 **说明**：默认情况下，在每次打开 PhpStorm 时都会弹出如图 1.32 所示的提示框。如果不想弹出该提示框，将图 1.32 中的 Show Tips on Startup 复选框的勾去掉即可。

如果应用 PhpStorm 创建过项目，打开 PhpStorm 后会进 PhpStorm 的主界面，在主界面中会默认打开之前创建过的项目，并弹出如图 1.32 所示的提示框，可以单击 Close 按钮将其关闭，然后新建一个 PHP 项目。具体步骤如下：

（1）选择菜单栏中 File 菜单下的 New Project 选项，如图 1.33 所示。单击该选项，此时会弹出如图 1.34 所示的创建新项目对话框。

（2）在图 1.34 所示的对话框中首先选择项目存储路径，将项目文件夹存储在“D:\phpStudy\WWW”目录下，然后输入新创建的项目名称“test”，最后单击 OK 按钮创建项目，这时会弹出打开项目对话框，如图 1.35 所示。单击 This Window 按钮在当前窗口打开创建的项目。此时在主界面的左侧会显示新建项目的名称以及自动生成的文件，如图 1.36 所示。

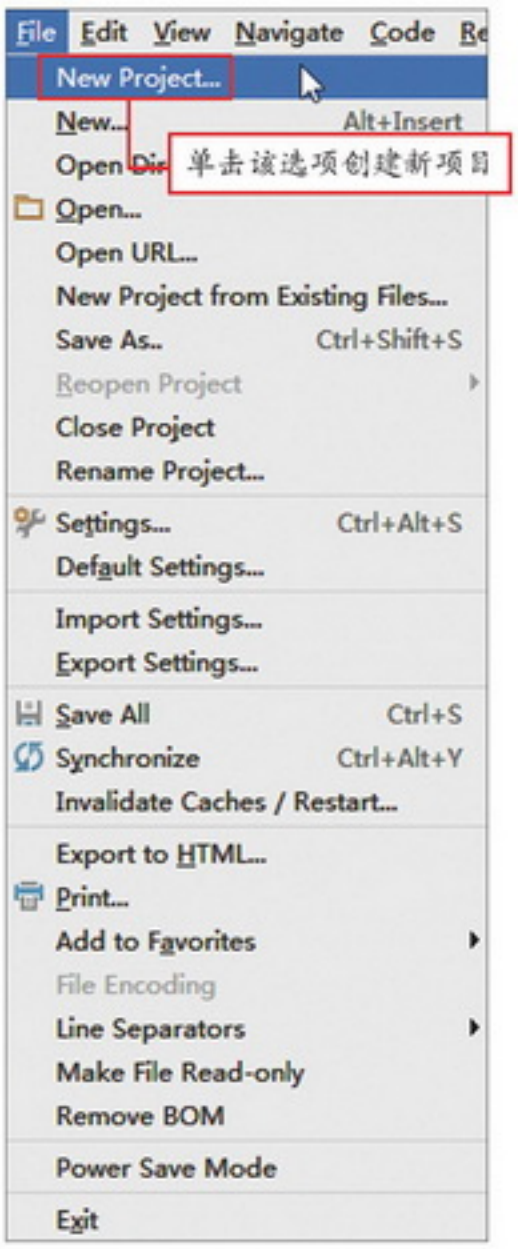


图 1.33　选择 New Project 选项

图 1.34　创建新项目对话框

图 1.35　打开项目对话框

图 1.36　新建的项目目录

📖 **说明**：如果在创建项目时弹出如图 1.37 所示的对话框，则说明“WWW”目录下已经存在该名称的项目文件夹，此时单击 Yes 按钮将其替换即可。

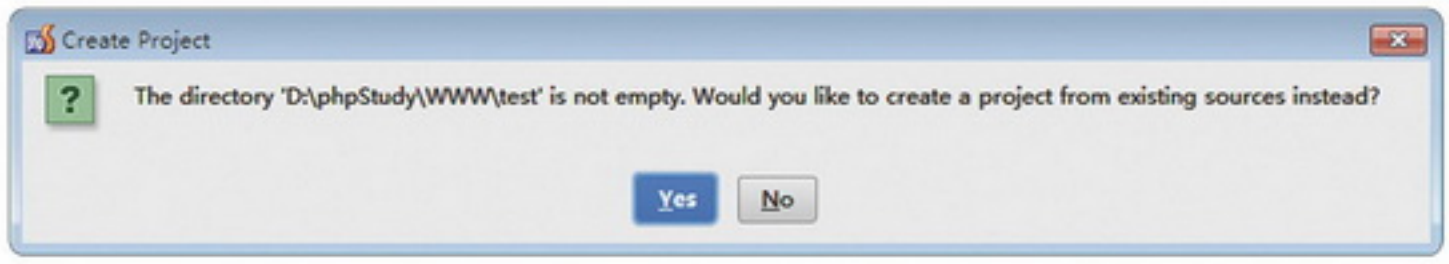


图 1.37　提示用户是否替换已存在的目录

1.5.2 打开已有项目

视频讲解：光盘\Video\01\1.5.2 打开已有项目.mp4

应用 PhpStorm 还可以打开已经存在的项目，具体方法如下：

（1）选择菜单栏中 File 菜单下的 Open Directory 选项，如图 1.38 所示。单击该选项，此时会弹出如图 1.39 所示的选择项目路径对话框。

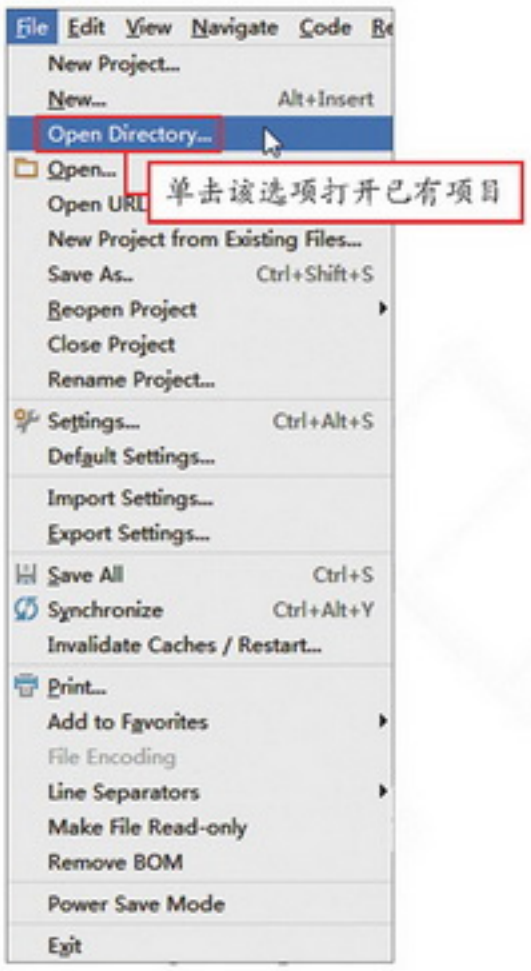


图 1.38　选择 Open Directory 选项

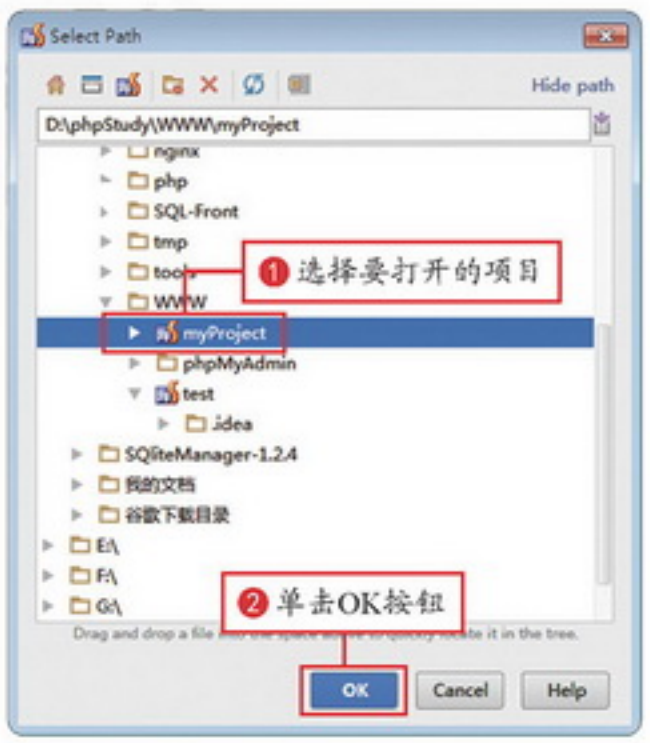


图 1.39　选择要打开的项目

（2）在图 1.39 所示的对话框中选择要打开的项目，然后单击 OK 按钮，会弹出打开项目对话框，如图 1.40 所示。在该对话框中可以对项目打开方式进行选择，单击 This Window 按钮即可在当前窗口打开项目。

图 1.40　打开项目对话框

1.5.3 在项目中创建文件夹和文件

视频讲解：光盘\Video\01\1.5.3 在项目中创建文件夹和文件.mp4

在 PHP 项目创建完成之后，接下来就可以在项目中创建文件夹和文件了。下面介绍在项目目录中创建文件夹以及文件的方法。

1. 在项目中创建文件夹

在项目目录 myProject 中创建一个名为“css”的文件夹，具体步骤如下：

（1）在项目名称“myProject”上单击鼠标右键，然后依次选择 New → Directory 菜单项，如图 1.41 所示。

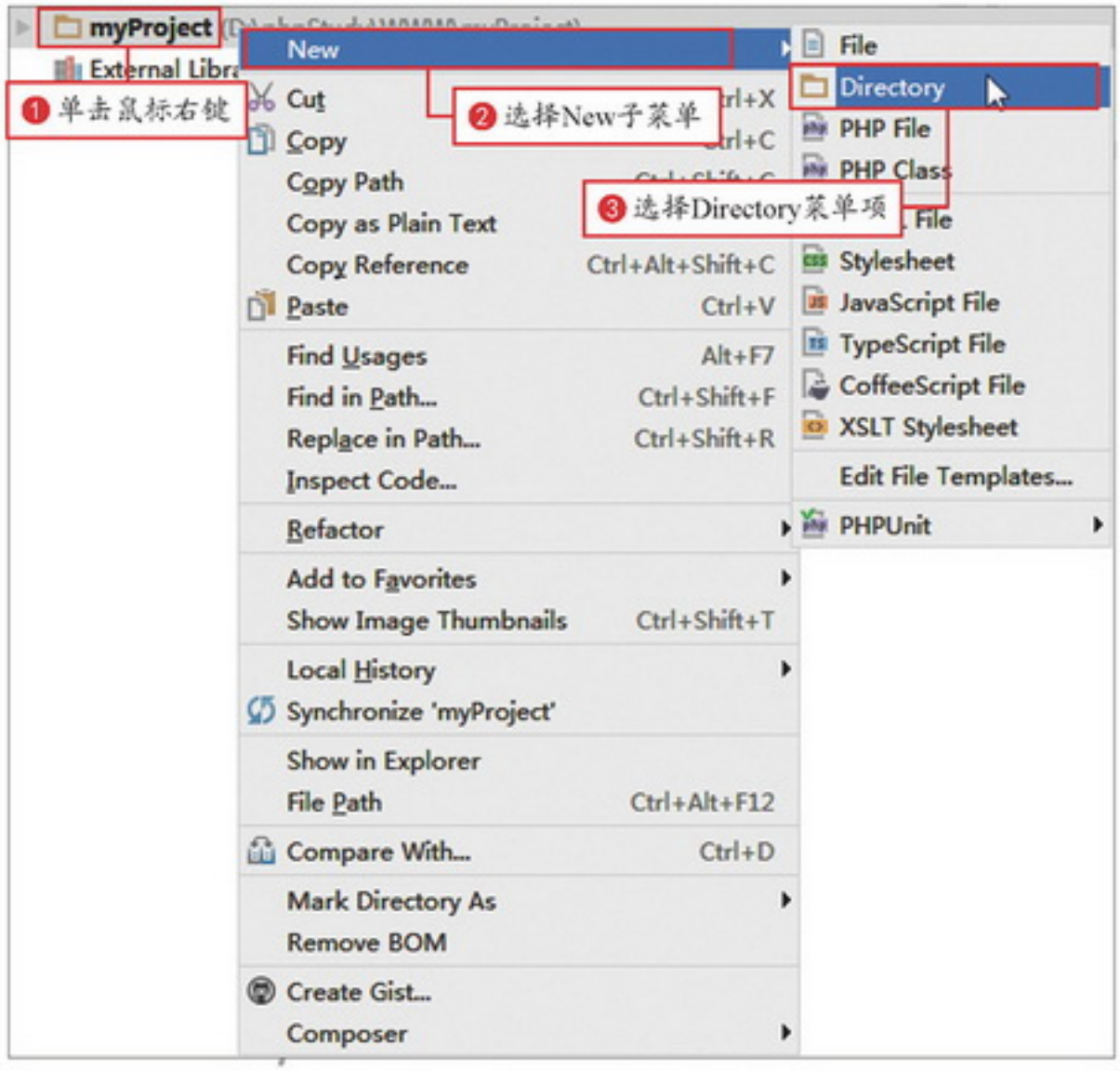


图 1.41　在项目中创建目录

（2）单击 Directory 选项后，弹出新建目录对话框，如图 1.42 所示，在文本框中输入新建目录的名称“css”，然后单击 OK 按钮，完成文件夹 css 的创建，创建后的项目目录结构如图 1.43 所示。

图 1.42　输入新建目录名称

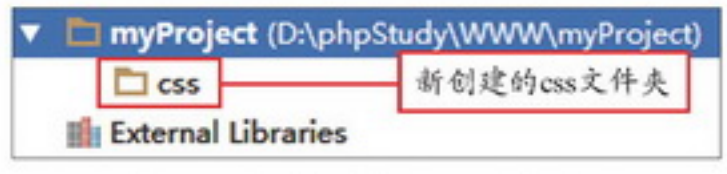


图 1.43　创建后的项目目录结构

2. 在项目中创建 PHP 文件

在项目目录 myProject 中创建一个 PHP 文件 index.php，具体步骤如下：

（1）在项目名称“myProject”上单击鼠标右键，然后依次选择 New → PHP File 菜单项，如图 1.44 所示。

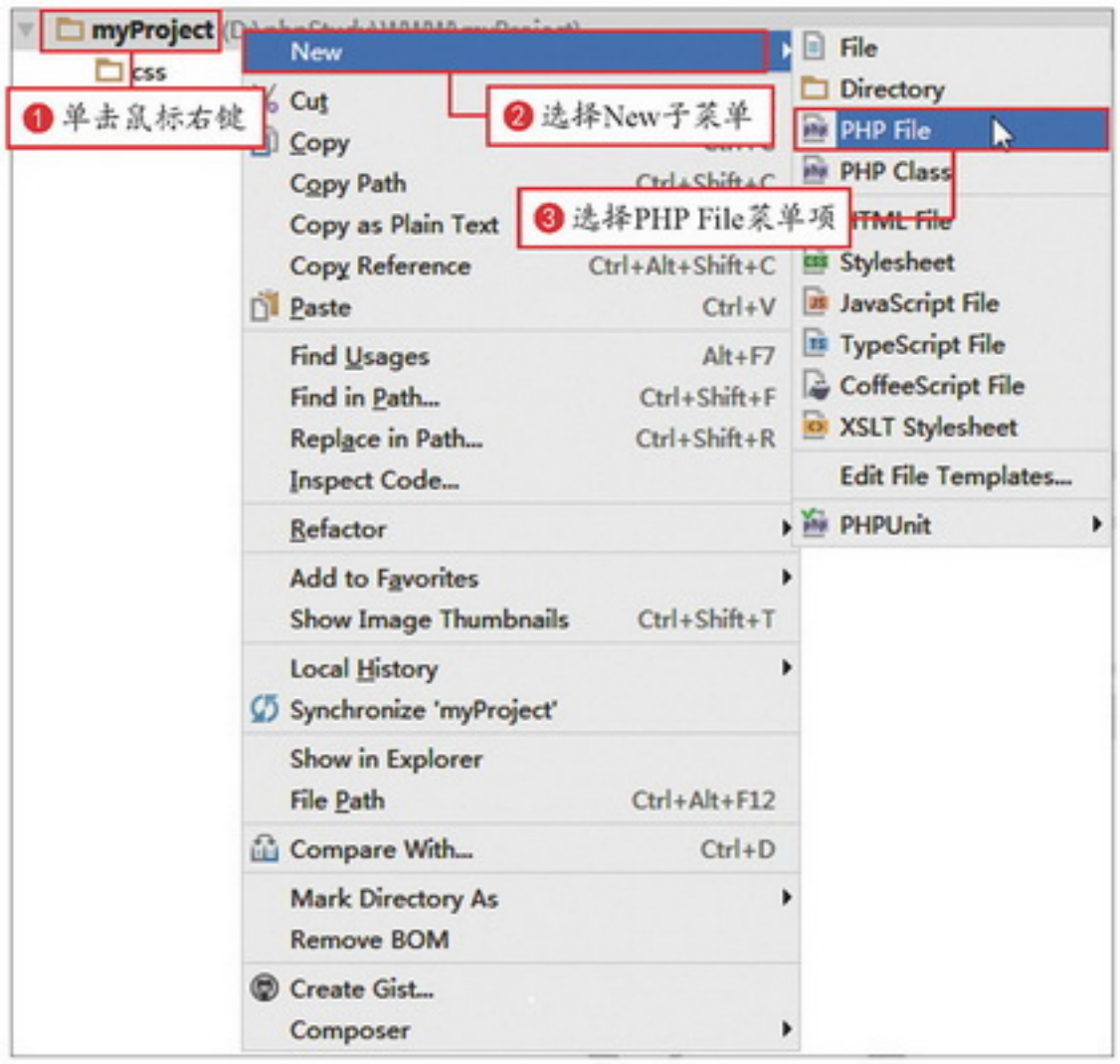


图 1.44　在项目中创建 PHP 文件

图 1.45　输入 PHP 文件名称

（2）单击 PHP File 选项后，弹出新建 PHP 文件对话框，如图 1.45 所示，在文本框中输入 PHP 文件的名称“index”，然后单击 OK 按钮，完成 index.php 文件的创建。此时，开发工具会自动打开创建的文件，如图 1.46 所示。

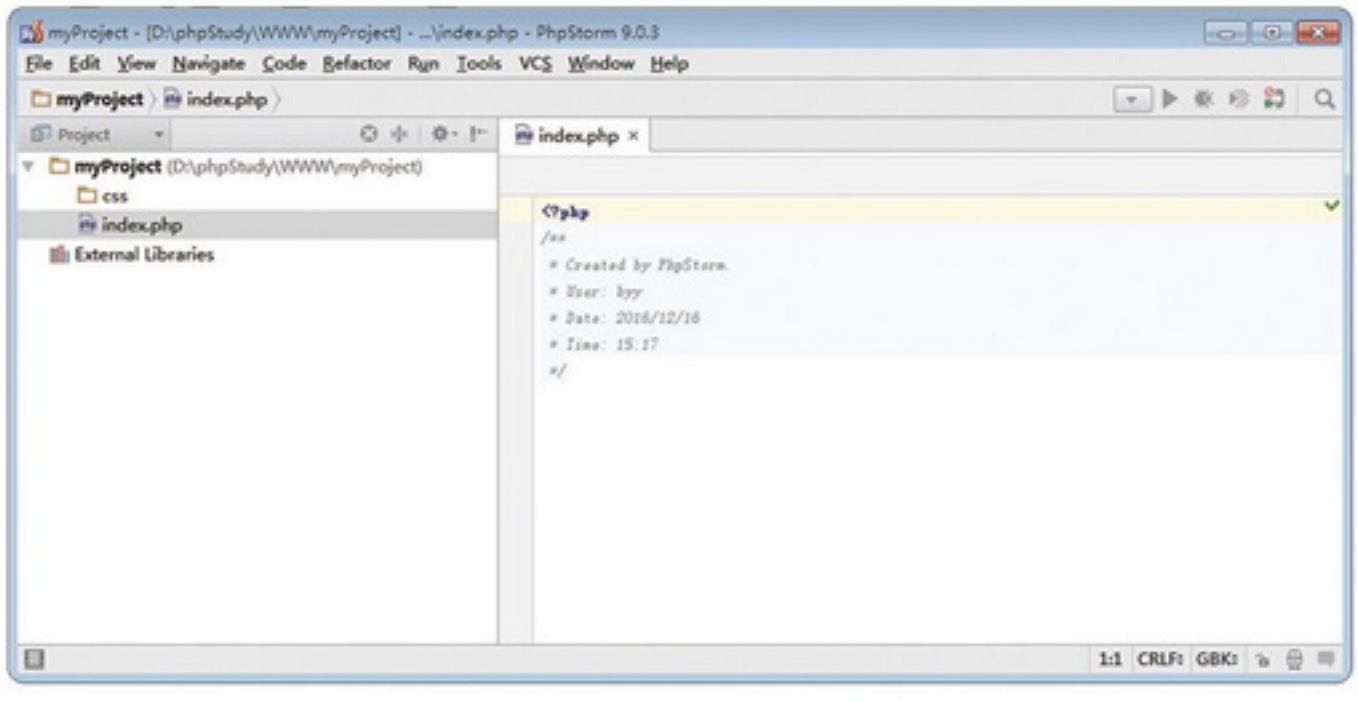


图 1.46　自动打开创建的文件

3. 运行第一个程序

下面来编写并运行第一个 PHP 程序。具体步骤如下：

（1）在 index.php 文件中编写代码，首先删除文件创建之后默认生成的代码，然后在页面中编写代码，输出字符串“Hello World”，如图 1.47 所示。

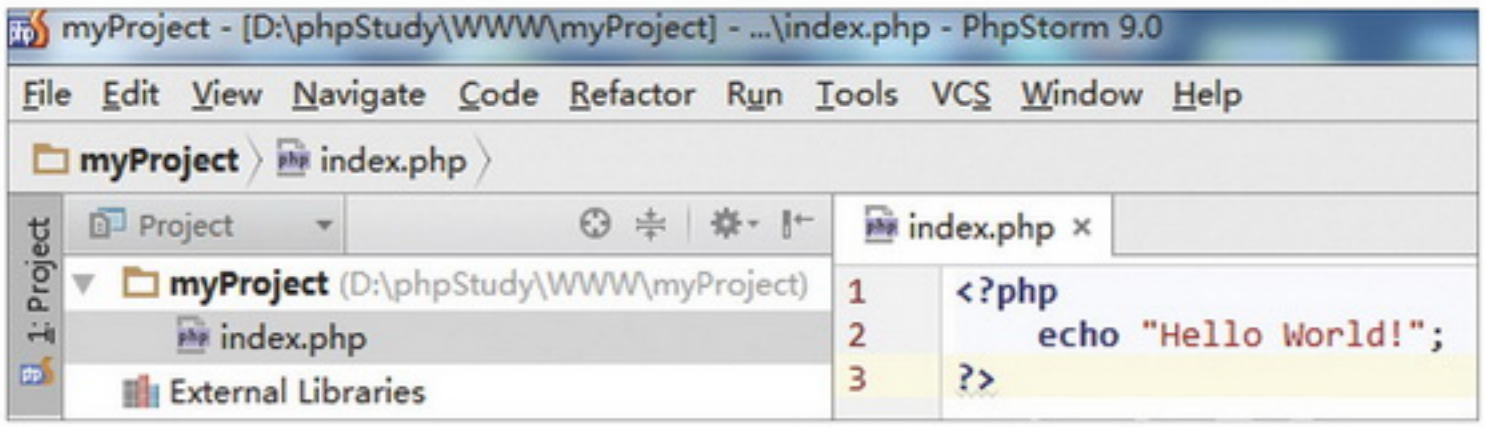


图 1.47　在文件中编写代码

（2）打开浏览器，在地址栏中输入“http://localhost/myProject/index.php”，按下 <Enter> 键后即可查看 index.php 页面的运行结果，如图 1.48 所示。

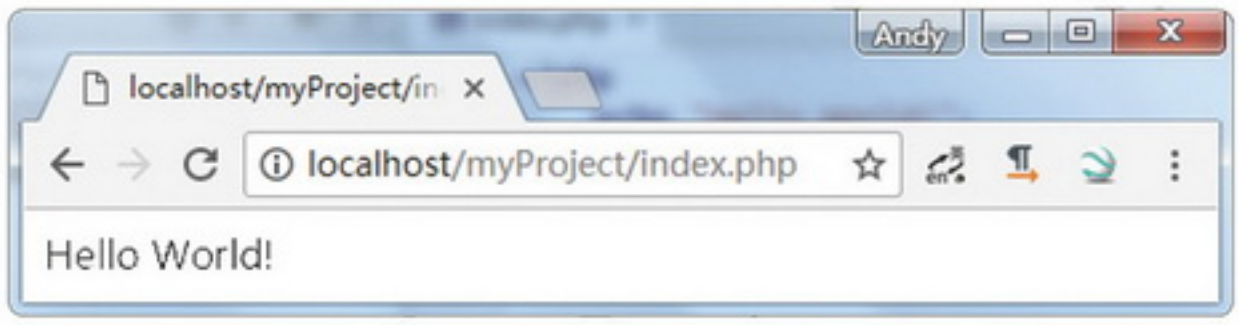


图 1.48　运行第一个 PHP 程序

1.6 PhpStorm 常用设置

PhpStorm 的功能十分强大，它可以快速有效地完成项目的创建，并为用户操作提供了很多方便之处。下面介绍一下在程序开发过程中 PhpStorm 的一些常用设置。

1.6.1 设置文件编码格式

视频讲解：光盘\Video\01\1.6.1 设置文件编码格式.mp4

现代 PHP 标准要求 PHP 文件的编码格式为 UTF-8，下面介绍两种方法设置文件编码格式。

1. 设置项目的编码格式

为保证整个项目的编码格式为 UTF-8，在创建完项目前先设置项目的编码格式。单击 PhpStorm 左上的 File，选择 Settings，在弹出的对话框左侧输入“encodings”，然后将“Project Encoding”设置为“UTF-8”。具体操作如图 1.49 所示。

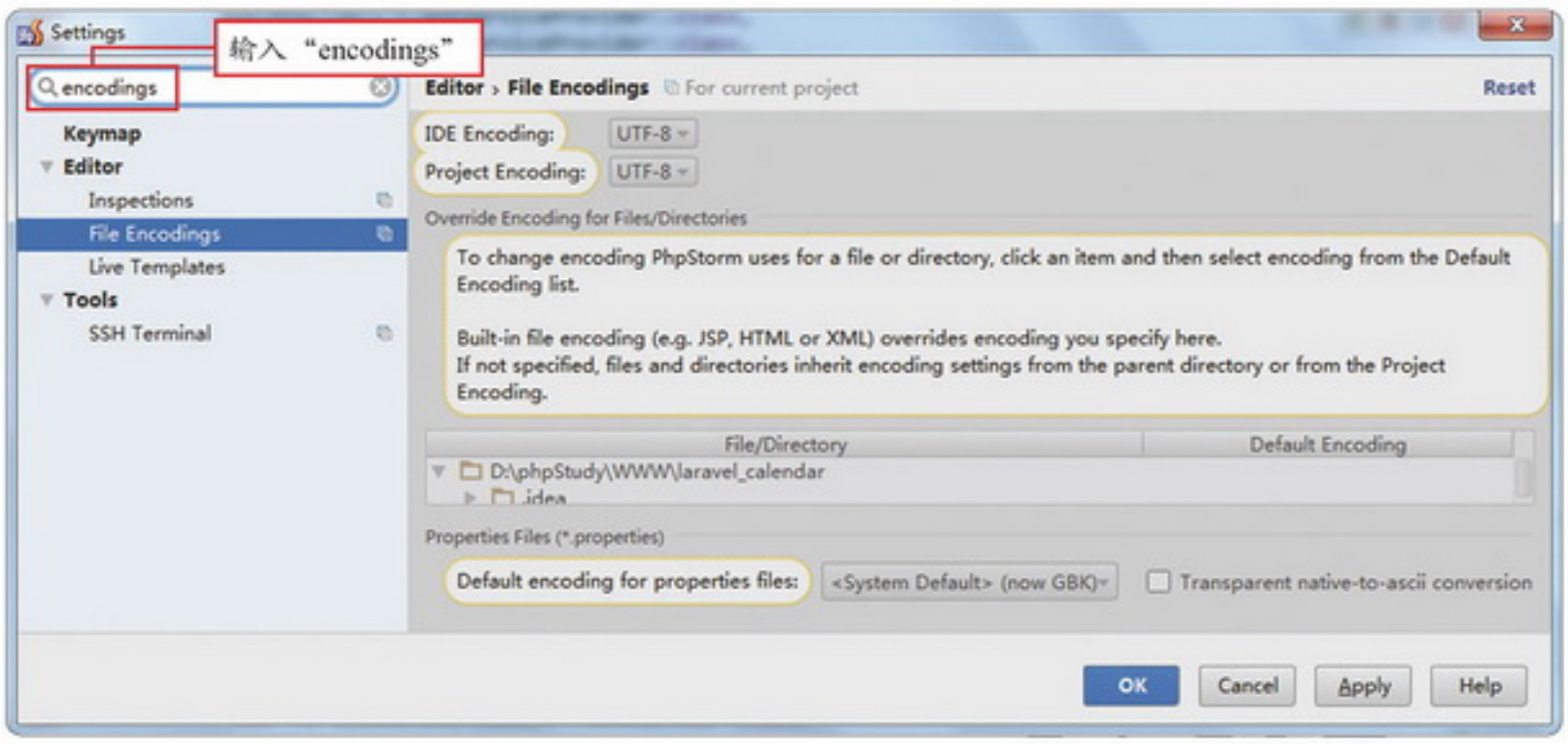


图 1.49　设置 PhpStorm 编码格式

2. 更改单个文件的编码格式

当要将一个文件复制到项目中时，如果该文件的编码格式为“GBK”，则需要将其更改为“UTF-8”。此时，可以使用 PhpStorm 更改单个文件的编码格式。首先使用 PhpStorm 打开该文件，然后单击 PhpStorm 右下角的文件编码（如“GBK”），将弹出所有编码的对话框，选择“UTF-8”，最后单击弹出框中的 Convert 按钮，具体操作如图 1.50 所示。

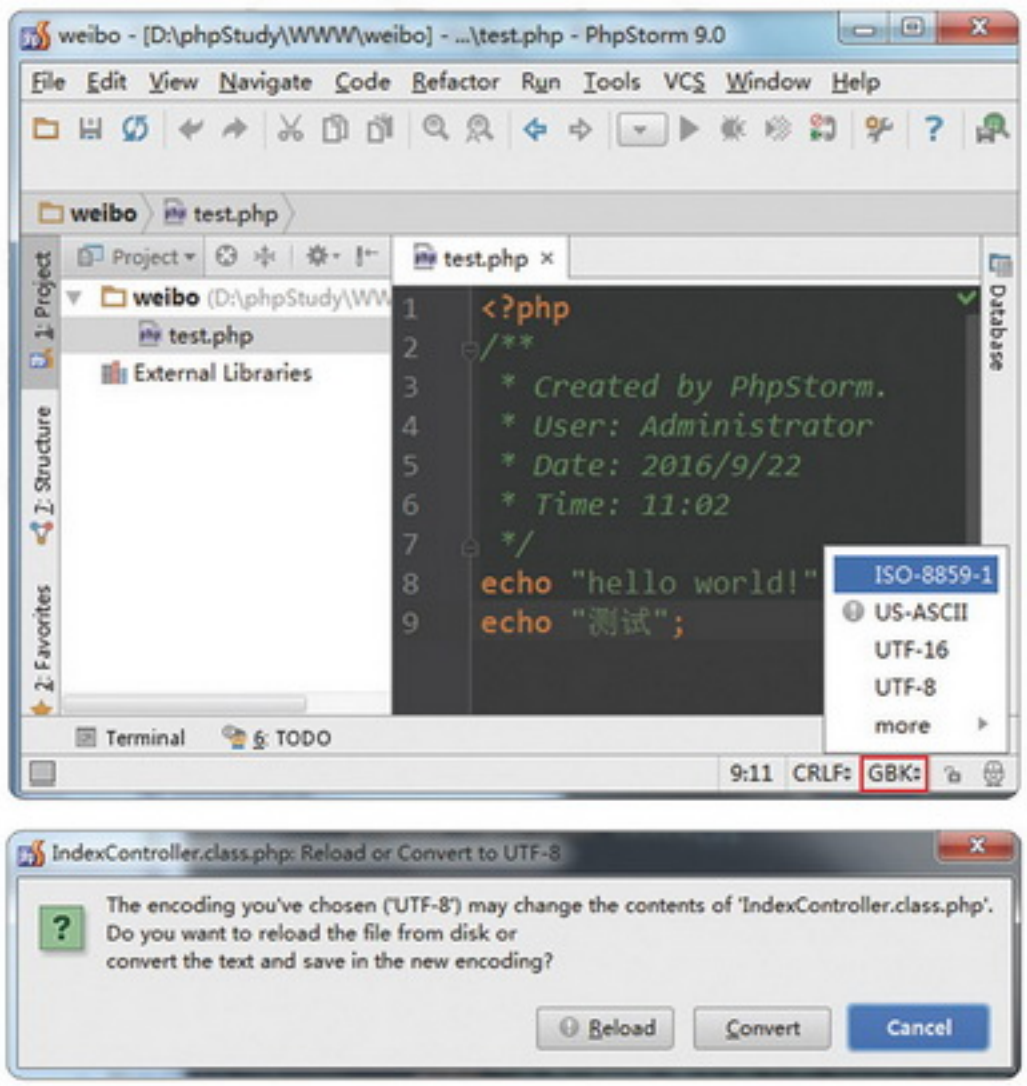


图 1.50　更改编码格式为“UTF-8”

1.6.2 其他常用设置

视频讲解：光盘\Video\01\1.6.1 其他常用设置.mp4

在 PhpStorm 的 Settings 选项中，还可以设置 PhpStorm 的主题、字体、颜色等。此外，还可以为 PhpStorm 添加实用插件，更多功能请查阅官方网站。

1.7 难点解答

1.7.1 为什么要设置文件编码格式为 UTF-8

UTF-8 是 Unicode 的一种变长字符编码，简单地说，该字符集可以解决多种语言文本显示问题，如网站中可以同时显示中文、英文或者日文等，从而实现应用国际化和本地化。此外，UTF-8 还能够兼容 ASCII 码、前缀码。

1.7.2 运行 PHP 程序前，先开启 phpStudy

很多初学者在运行 PHP 程序时，会遇到浏览器提示“无法访问此网站”的情况，这个问题很有可能是由于没有开启 phpStudy 导致的。所以，在运行 PHP 程序时，请确保先开启 phpStudy。

1.8 小　结

本章先介绍了什么是 PHP、PHP 语言的优势、PHP 的发展趋势等，然后讲解了在 Windows 下如何搭建 PHP 环境，包括 phpStudy 集成环境的下载、安装和使用等知识。接着，介绍了 PhpStorm 开发工具的下载、安装及设置。此外，还编写了第一个 PHP 程序：输出“Hello Word!”。希望读者通过本章的学习，对 PHP 能有一个初步的了解，并能够配置好开发环境，为接下来的开发之旅做好准备。

第 2 章

PHP 语言基础

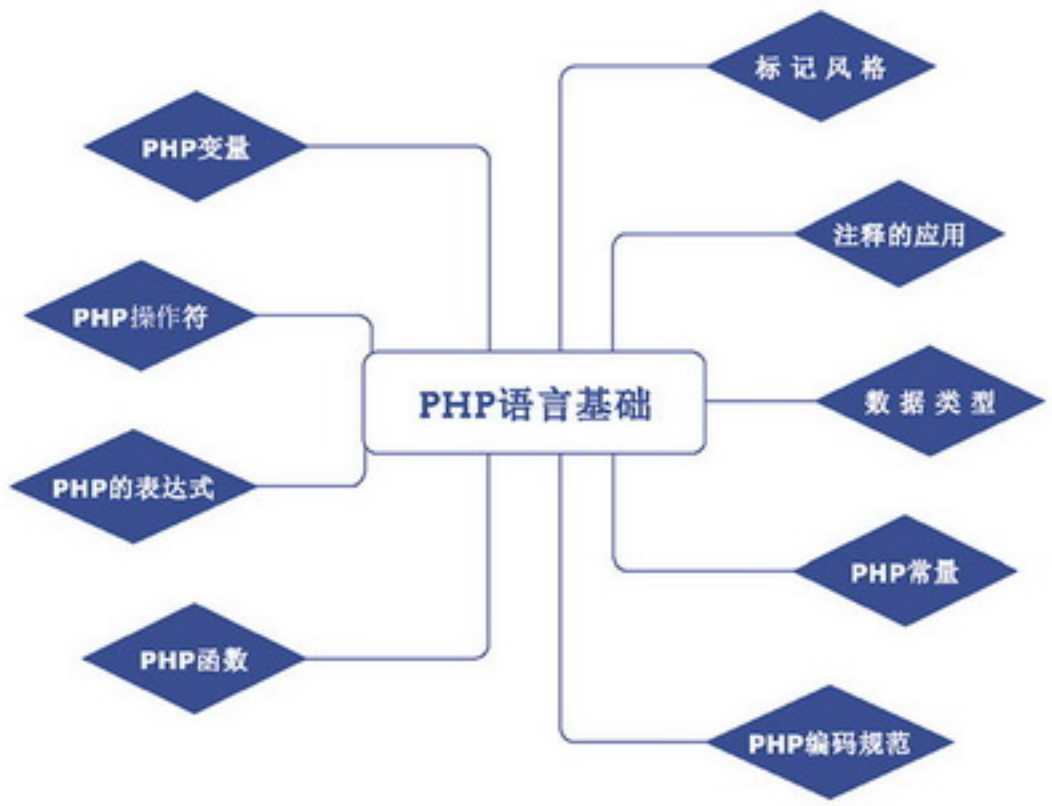
（视频讲解：2 小时 15 分）

本章概览

PHP 程序开发快、运行快，PHP 语言学起来也比其他语言快，它的特点就是易学、易用。

本章不仅对 PHP 标记风格、数据类型、PHP 常量、PHP 变量、PHP 运算符等基础内容进行了详细讲解，还介绍了编写 PHP 程序时应遵循的编码规范。通俗易懂的知识和实例将带领读者逐步走进 PHP 编程世界。每个实例后设置两个练习题目，让读者亲自动手，在掌握基础知识的前提下体验 PHP 程序开发的乐趣。

知识框架



2.1 PHP 标记风格

视频讲解：光盘\Video\02\2.1 PHP标记风格.mp4

PHP 和其他几种 Web 语言一样，都是使用一对标记对将 PHP 代码部分包含起来，以便和 HTML 代码相区分。PHP 一共支持 4 种标记风格，下面分别进行介绍。

◆ XML 风格

```
01  <?php
02      echo "这是XML风格的标记";
03  ?>
```

XML 风格的标记是本书所使用的标记，也是推荐使用的标记，服务器不能禁用。该风格的标记在 XML、XHTML 中都可以使用。

◆ 脚本风格

```
01  <script language="php">
02      echo '这是脚本风格的标记';
03  </script>
```

◆ 简短风格

```
<? echo '这是简短风格的标记'; ?>
```

◆ ASP 风格

```
01  <%
02      echo '这是ASP风格的标记';
03  %>
```

说明：如果要使用简短风格和 ASP 风格，需要在 php.ini 中对其进行配置，打开 php.ini 文件，将 short_open_tag 和 asp_tags 都设置为 On，重启 Apache 服务器即可。

注意：这里推荐使用 XML 风格的标记，原因可以参考本章 2.9 节的 PHP 编码规范。

2.2 PHP 注释的应用

视频讲解：光盘\Video\02\2.2 PHP注释的应用.mp4

注释即代码的解释和说明，一般放在代码的上方或代码的尾部，用来说明代码或函数的编写人、用途、时间等。注释不会影响到程序的执行，因为在执行时，注释部分会被解释器忽略不计。

PHP 支持 3 种风格的程序注释。

◆ 单行注释（//）

这是一种来源于 C++ 语法的注释模式，可以写在 PHP 语句的上方。

```
<?php
    //这是写在PHP语句上方的单行注释
    echo '使用C++风格的注释';
?>
```

也可以写在 PHP 语句的后方。

```
<?php
    echo '使用C++风格的注释';        //这是写在PHP语句后方的单行注释
?>
```

◆ 多行注释（/*…*/）

这是一种来源于 C 语言语法的注释模式，可以分为块注释和文档注释。

块注释：

```
<?php
    /*
    $a = 1;
    $b = 2;
    echo ($a + $b);
    */
    echo 'PHP的多行注释';
?>
```

文档注释：

```
<?php
/* 说明: 项目工具类
 * 作者: mrsoft
 * E-mail:mingrisoft@mingrisoft.com
 */
class Util
{
    /**
     * 方法说明: 给字符串加前缀
     * 参数: String $str
     * 返回值: String
     */
    function addPrefix ($str)
    {
        $str.= 'mingri';
        return $str;
    }
```

```
18  }
19  ?>
```

注意：多行注释是不允许进行嵌套操作的。

◆ # 号风格的注释（#）

```
01  <?php
02      echo '这是#号风格的注释';                    #这是Unix风格的单行注释
03  ?>
```

注意：在单行注释中的内容不要出现“?>”标志，因为解释器会认为 PHP 脚本结束，而不去执行“?>”后面的代码。例如：

```
01  <?php
02      echo '这样会出错的！！！！！'              //不会看到?>
03  ?>
```

运行结果为：

```
这样会出错的！！！！！不会看到 ?>
```

2.3 PHP 的数据类型

2.3.1 数据类型

视频讲解：光盘\Video\02\2.3.1 数据类型.mp4

PHP 一共支持 8 种原始数据类型，包括 4 种标量类型，即 integer（整型）、float/double（浮点型）、string（字符串型）和 boolean（布尔型）；两种复合类型，即 array（数组）和 object（对象）；两种特殊类型，即 resource（资源）与 NULL（空）。PHP 支持的数据类型及说明如表 2.1 所示。

表 2.1 PHP 支持的数据类型及说明

类　型	说　明
integer（整型）	整型数据类型只能包含整数，可以是正数或负数
float（浮点型）	浮点数据类型用于存储数字，和整型不同的是它有小数位
string（字符串型）	字符串就是连续的字符序列，可以是计算机所能表示的一切字符的集合
boolean（布尔型）	这是最简单的类型。只有两个值，真（true）和假（false）
array（数组）	用来保存具有相同类型的多个数据项
object（对象）	用来保存类的实例

续表

类　型	说　明
resource（资源）	资源是一种特殊的变量类型，保存了到外部资源的一个引用：如打开文件、数据库连接、图形画布区域等
NULL（空）	没有被赋值、已经被重置或者被赋值为特殊值 NULL 的变量

实例 01　输出个人信息

实例位置：光盘\Code\SL\02\01
视频位置：光盘\Video\02\

本实例将使用 echo 语句输出个人信息，包括“姓名”“性别”“年龄”“身高”和“体重”，代码如下：

实例01-1

```
<?php
    $name = "明日科技小助手";
    $gender = "女";
    $age  = 18;
    $height = 170;
    $weight = 45.5;
    echo "姓名:".$name."<br>";
    echo "性别:".$gender."<br>";
    echo "年龄:".$age."岁<br>";
    echo "身高:".$height."厘米<br>";
    echo "体重:".$weight."公斤<br>";
?>
```

上述代码中，包含的数据类型有字符串型、整型和浮点型，运行结果如图 2.1 所示。

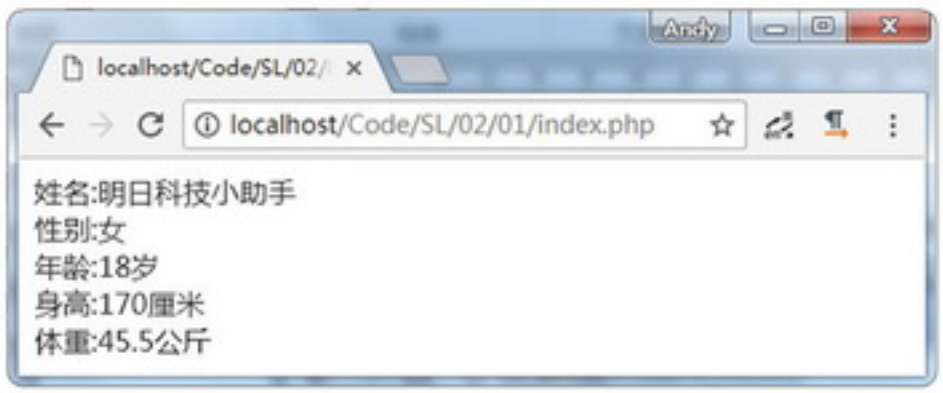


图 2.1　个人信息输出结果

多学两招：上述代码中，“.”是字符串连接符，“
”是换行标签，“echo”是 PHP 的输出语句，将文本内容显示在浏览器上。常用的输出语句还有 var_dump() 函数和 print_r() 函数。

练一练：

（1）编写一个简短的小程序，输出 3 行内容：你的名字、出生日期，还有你最喜欢的颜色。（光盘\Code\Try\02\01）

（2）使用 3 种书写方法（圆周率函数、传统书写格式和科学计数法）输出圆周率的近似值，运行效果如图 2.2 所示。（光盘\Code\Try\02\02）

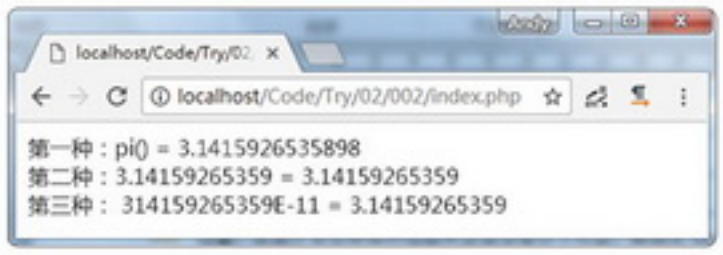


图 2.2　3 种方法输出圆周率的近似值

2.3.2 数据类型转换

视频讲解：光盘\Video\02\2.3.2 数据类型转换.mp4

PHP 是弱类型语言（或动态语言），不需要像 C 语言一样在使用变量前必须先声明变量的类型。在 PHP 中，变量的类型是由赋给它的值确定的。例如：

```
<?php
    $var1 = 'Hello World';                    //给变量var1赋值
    $var2 = 521;                              //给变量var2赋值
?>
```

说明：代码中的“=”不是数学中的“等于”，它是赋值操作符，表示将“=”右边的值赋给“=”左边的变量。

上述代码中，变量 var1 为字符串类型，变量 var2 为整型。虽然 PHP 不需要先声明变量的类型，但有时仍然需要用到数据类型转换。PHP 中的数据类型转换非常简单，只需在变量前加上用括号括起来的类型名称即可。允许转换的数据类型如表 2.2 所示。

表 2.2　PHP 中允许转换的数据类型

转换操作符	转 换 类 型	举　　例
(int)，(integer)	转换成整型	(int)$boo、(integer)$str
(bool)，(boolean)	转换成布尔型	(bool)$num、(boolean)$str
(string)	转换成字符串型	(string)$boo
(array)	转换成数组	(array)$str
(float)，(double)，(real)	转换成浮点型	(float)$str、(double)$str
(object)	转换成对象	(object)$str
(unset)	转换为 NULL	(unset)$str

注意：在进行数据类型转换的过程中应该注意以下内容：转换成 boolean 型时，null、0 和未赋值的变量或数组会被转换为 false，其他的为真；转换成整型时，布尔型的 false 转换为 0，true 转换为 1，浮点型的小数部分被舍去，字符串型如果以数字开头就截取到非数字位，否则输出 0。

数据类型转换还可以通过 settype() 函数来完成，该函数可以将指定的变量转换成指定的数据类型，语法如下：

```
bool settype ( mixed $var, string $type )
```

参数 var 为指定的变量，参数 type 为指定的类型，参数 type 有 7 个可选值，即 boolean、float、integer、array、null、object 和 string。如果转换成功则返回 true，否则返回 false。

当字符串转换为整型或浮点型时，如果字符串是以数字开头的，就会先把数字部分转换为整型，再舍去后面的字符串；如果数字中含有小数点，则会取到小数点前一位。

实例 02　将指定的字符串进行类型转换

实例位置：光盘\Code\SL\02\02

视频位置：光盘\Video\02\

本实例使用上面的两种方法将指定的字符串进行类型转换，并比较两种方法之间的不同。代码如下：

实例02-1

```
<?php
    $num = '3.1415926r*r';                          //定义一个字符串变量
    echo '将字符串型数据转化为整型的结果是：';
    echo (int)$num;                                 //使用integer转换类型
    echo '<br>';
    $result = settype($num,'integer');              //使用settype()函数转换类型
    echo '使用settype函数转换变量$num类型,函数的返回值为：'.$result;
    echo '<br>';
    echo '输出转化后$num的值：'.$num;                //输出原始变量$num
?>
```

运行结果如图 2.3 所示。

图 2.3　将指定字符串进行类型转换

从运行结果可以看出，使用 (int) 能直接输出转换后的变量类型，并且原变量不发生任何变化。而使用 settype() 函数返回的是布尔值，也就是 true 或 false，而原变量被改变了。在实际应用中，可根据情况自行选择转换方式。

练一练：

（1）使用 var_dump() 函数输出“1.0+2”的结果，查看结果的数据类型，并将其转化为整型。（光盘\Code\Try\02\03）

（2）试着使用 (int) 将一个小数（ 56.78 ）转换成一个整数，并查看是上取整还是下取整。（光盘\Code\Try\02\04）

2.3.3 检测数据类型

视频讲解：光盘\Video\02\2.3.3 检测数据类型.mp4

PHP 还内置了检测数据类型的系列函数，可以对不同类型的数据进行检测，判断其是否属于某个类型，如果符合则返回 true，否则返回 false。检测数据类型的函数如表 2.3 所示。

表 2.3　检测数据类型的函数

函　数	检 测 类 型	举　例
is_bool	检查变量是否是布尔类型	is_bool(true)、is_bool(false)
is_string	检查变量是否是字符串类型	is_string('string')、is_string(1234)
is_float/is_double	检查变量是否为浮点类型	is_float(2.1415)、is_float('2.1415')
is_integer/is_int	检查变量是否为整型	is_integer(34)、is_integer('34')
is_null	检查变量是否为 null	is_null(null)
is_array	检查变量是否为数组类型	is_array($arr)
is_object	检查变量是否是一个对象类型	is_object($obj)
is_numeric	检查变量是否为数字或由数字组成的字符串	is_numeric('5')、is_numeric('bccd110')

由于检测数据类型的函数功能和用法都是相同的，下面使用 is_numeric() 函数检测变量中的数据是否是数字，从而掌握 is 系列函数的用法。代码如下：

```
<?php
    $boo = "043112345678";                          //定义一个全由数字组成的字符串变量
    if(is_numeric($boo)){                           //判断该变量是否由数字组成
        echo "<p>$boo is a number</p>";             //如果是，输出该变量
    }else{
        echo "<p>$boo is not a number</p>";         //否则，输出错误语句
    }
    if(is_null($boo)){
        echo "<p>$boo is null</p>";                 //判断变量是否为null
    }else{
        echo "<p>$boo is not null</p>";
    }
?>
```

输出结果为：

```
043112345678 is a number
043112345678 is not null
```

2.4 PHP 常量

常量是一个简单值的标识符（名字）。如同其名称所暗示的，在脚本执行期间该值不能改变，常量默认为大小写敏感。一个常量由英文字母、下划线和数字组成，但数字不能作为首字符出现。传统上常量标识符总是大写的。

2.4.1 定义常量

视频讲解：光盘\Video\02\2.4.1 定义常量.mp4

在 PHP 中使用 define() 函数来定义常量，该函数的语法格式为：

```
define(string $constant_name,$mixed value,$case_sensitive=false)
```

该函数有 3 个参数：

- constant_name：必选参数，常量名称，即标识符。
- value：必选参数，常量的值。
- case_sensitive：可选参数，指定是否大小写敏感，设定为 true，表示不敏感。

定义完常量后，使用常量名可以直接获取常量值。例如：

```
<?php
    define ("MESSAGE","我是一名PHP程序员");
    echo "MESSAGE is:".MESSAGE."<br>";          //输出常量MESSAGE
    echo "Message is:".Message."<br>";          //输出错误提示，因为常量区分大小写
?>
```

运行结果如下：

```
MESSAGE is:我是一名PHP程序员
Notice: Use of undefined constant Message
```

2.4.2 预定义常量

视频讲解：光盘\Video\02\2.4.2 预定义常量.mp4

在 PHP 开发过程中，开发者们经常会使用一些通用的信息，PHP 已经将这些信息定义为常量，而不需要开发者重新定义，这就是预定义常量。常用的预定义常量如表 2.4 所示。

表 2.4　PHP 常用的预定义常量

常 量 名	功　能
__FILE__	默认常量，PHP 程序文件名
__LINE__	默认常量，PHP 程序行数
PHP_VERSION	内建常量，PHP 程序的版本，如 php6.0.0-dev
PHP_OS	内建常量，执行 PHP 解析器的操作系统名称，如 Windows
TRUE	该常量是一个真值（true）
FALSE	该常量是一个假值（false）
NULL	一个 null 值
E_ERROR	该常量指到最近的错误处

续表

常 量 名	功 能
E_WARNING	该常量指到最近的警告处
E_PARSE	该常量指到解析语法有潜在问题处
E_NOTICE	该常量为发生不寻常处的提示，但不一定是错误处

注意：__FILE__ 和 __LINE__ 中的“__”是两条下划线，而不是一条下划线。

说明：表中以 E_ 开头的预定义常量是 PHP 的错误调试部分。如果想要详细了解，请参考 error_reporting() 函数。

预定义常量与用户自定义常量在使用上没什么差别，都能直接获取常量值。例如，使用预定义常量输出 PHP 中的信息，代码如下：

```
<?php
    echo "当前文件路径：  ".__FILE__;                    //输出__FILE__常量
    echo "<br>当前行数：".__LINE__;                      //输出__LINE__常量
    echo "<br>当前PHP版本信息：".PHP_VERSION;            //输出PHP版本信息
    echo "<br> 当前操作系统：".PHP_OS ;                  //输出系统信息
?>
```

运行结果如下：

```
当前文件路径：  D:\phpStudy\WWW\Code\test.php
当前行数：3
当前PHP版本信息：5.5.30
当前操作系统：WINNT
```

说明：因为每个用户操作系统和软件版本的不同，所得的结果也不一定相同。

2.5 PHP 变量

把一个值赋给一个名字时，如把值“明日科技小助手”赋给“$name”，“$name”就称为变量。在大多数编程语言中，都把这称为“把值存储在变量中”。在计算机内存中的某个位置，字符串序列“"明日科技小助手""”已经存在，用户不需要准确地知道它到底在哪里，只需要告诉 PHP 这个字符串序列的名字是“$name”，就可以通过这个名字来引用这个字符串序列。这个过程就像上门取快递一样，内存就像一个巨大的货物架，在 PHP 中使用变量就像是给快递盒子加标签，如图 2.4 所示。

顾客的快递存放在货物架上，上面附着写有名字的标签。当顾客来取快递时，并不需要知道它们存放在这个大型货架的哪个具体位置。只需要提供名字，快递员就会把快递交送到顾客手上。实际上，顾客的快递可能并不在原先所放的位置，不过快递员会记录快递的位置，要取回顾客的快递，只需要提供顾客的名字。变量也一样，用户不需要准确地知道信息存储在内存中的哪个位置，只需要记住存储变量时所用的名字，直接使用这个名字就可以了。

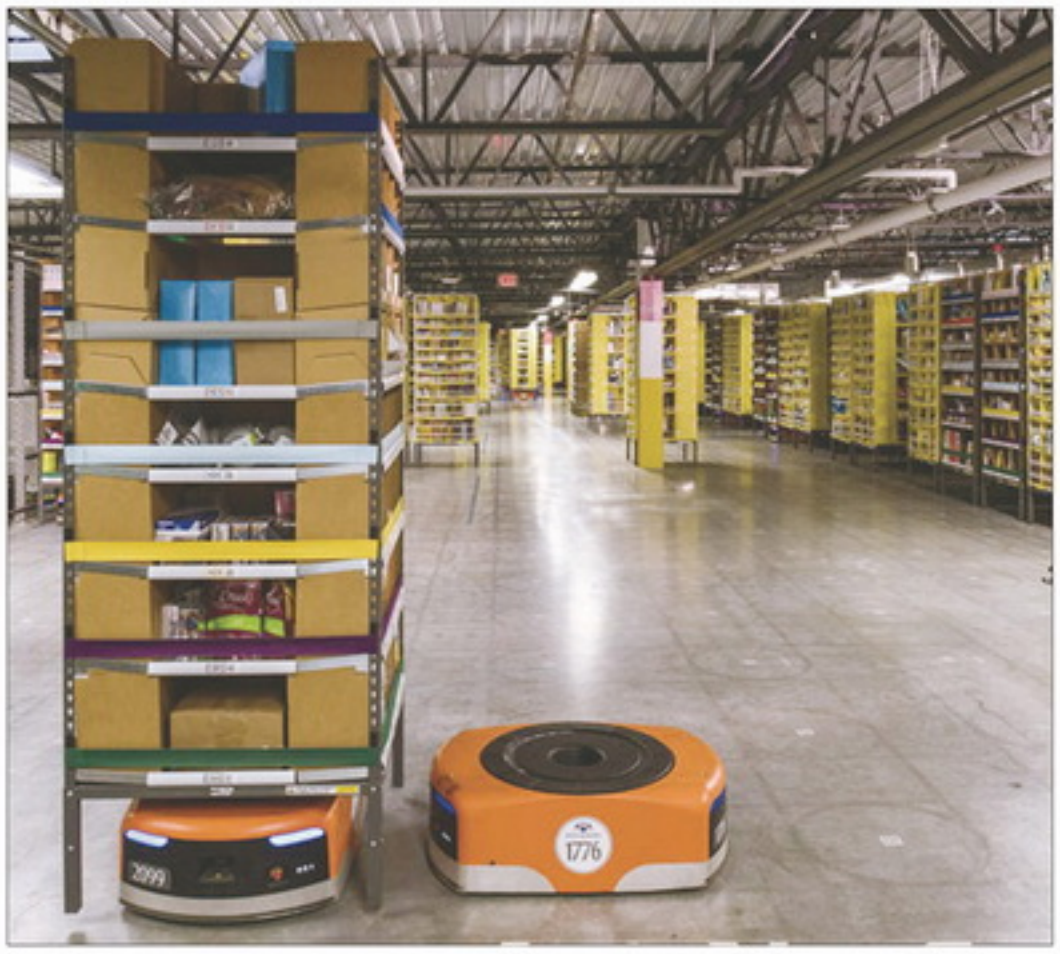


图 2.4　货物架中贴着标签的快递盒子

2.5.1 变量赋值及使用

视频讲解：光盘\Video\02\2.5.1 变量赋值及使用.mp4

和很多语言不同，在 PHP 中使用变量之前不需要声明变量（PHP 4 之前需要声明变量），只需为变量赋值即可。PHP 中的变量名称用“$ 符号 + 标识符”表示。标识符是由字母、数字或下划线组成，并且不能以数字开头。另外，变量名是区分大小写的。

为变量赋值，是指给变量一个具体的数据值，对于字符串和数字类型的变量，可以通过“=”来实现。语法格式为：

```
$name = value;
```

对变量命名时，要遵循变量命名规则。如下面的变量命名是合法的：

```
<?php
    $thisCup="oink";
    $_Class="roof ";
?>
```

下面的变量命名则是非法的：

```
<?php
    $11112_var=11112;                    //变量名不能以数字字符开头
    $@spcn = "spcn";                     //变量名不能以字母或下划线以外的其他字符开头
?>
```

除了直接赋值外，还有两种方式可以为变量赋值，一种是变量间的赋值。变量间的赋值是指赋值后两个变量使用各自的内存，互不干扰，代码如下：

```
<?php
    $string1 = "mingribook";                    //为变量$string1赋值
    $string2 = $string1;                        //使用$string1初始化$string2
    $string1 = "mrbccd";                        //改变变量$string1的值
    echo $string2;                              //输出变量$string2的值
?>
```

运行结果为：

```
mingribook
```

变量间的赋值就像在网上买了一个商品后，一天后又下单买了相同的商品。这样在快递点就有两个一样的快递，但这两个商品占用两个不同的货架位置，互不干扰。

另一种是引用赋值。从 PHP 4 开始，PHP 引入了“引用赋值”的概念。引用赋值是指用不同的名字访问同一个变量内容，当改变其中一个变量的值时，另一个也跟着发生变化。使用“&”符号来表示引用。例如，变量 $j 是变量 $i 的引用，当给变量 $i 赋值后，$j 的值也会跟着发生变化。代码如下：

```
<?php
    $i = "mingribook";                          //为变量$i赋值
    $j = &$i;                                   //使用引用赋值，这时$j已经赋值为mingribook
    $i = "mrbccd";                              //重新给$i赋值
    echo $j;                                    //输出变量$j
    echo "<br>";
    echo $i;                                    //输出变量$i
?>
```

运行结果为：

```
mrbccd
```

引用赋值就像在填写快递信息时，为避免和别人重名被人误取，于是在“收货人”位置上写了两个名字，一个是真名，一个是笔名。尽管是两个名字，但却是同一个商品，占用同一个货架。

注意：引用和复制的区别在于：复制是将原变量内容复制下来，开辟一个新的内存空间来保存，而引用则是给变量的内容再起一个名字。

2.5.2 PHP 预定义变量

视频讲解：光盘\Video\02\2.5.2 PHP预定义变量.mp4

PHP 还提供了很多非常实用的预定义变量，通过这些预定义变量可以获取到用户会话、用户操作系统的环境和本地操作系统的环境等信息。常用的预定义变量如表 2.5 所示。

表 2.5　常用的预定义变量

变量的名称	说　明
$_SERVER['SERVER_ADDR']	当前运行脚本所在的服务器的 IP 地址
$_SERVER['SERVER_NAME']	当前运行脚本所在服务器主机的名称。如果该脚本运行在一个虚拟主机上，则该名称是由虚拟主机所设置的值决定
$_SERVER['REQUEST_METHOD']	访问页面时的请求方法。如 GET、HEAD、POST、PUT 等，如果请求的方式是 HEAD，PHP 脚本将在送出头信息后中止（这意味着在产生任何输出后，不再有输出缓冲）
$_SERVER['REMOTE_ADDR']	正在浏览当前页面用户的 IP 地址
$_SERVER['REMOTE_HOST']	正在浏览当前页面用户的主机名。反向域名解析基于该用户的 REMOTE_ADDR
$_SERVER['REMOTE_PORT']	用户连接到服务器时所使用的端口
$_SERVER['SCRIPT_FILENAME']	当前执行脚本的绝对路径名。注意：如果脚本在 CLI 中被执行，作为相对路径，如“file.php”或者“.../file.php，$_SERVER['SCRIPT_FILENAME']”将包含用户指定的相对路径
$_SERVER['SERVER_PORT']	服务器所使用的端口，默认为 80。如果使用 SSL 安全连接，则这个值为用户设置的 HTTP 端口
$_SERVER['SERVER_SIGNATURE']	包含服务器版本和虚拟主机名的字符串
$_SERVER['DOCUMENT_ROOT']	当前运行脚本所在的文档根目录。在服务器配置文件中定义
$_COOKIE	通过 HTTPCookie 传递到脚本的信息。这些 cookie 多数是由执行 PHP 脚本时通过 setcookie() 函数设置的
$_SESSION	包含与所有会话变量有关的信息。$_SESSION 变量主要应用于会话控制和页面之间值的传递
$_POST	包含通过 POST 方法传递的参数的相关信息。主要用于获取通过 POST 方法提交的数据
$_GET	包含通过 GET 方法传递的参数的相关信息。用于获取通过 GET 方法提交的数据
$GLOBALS	由所有已定义全局变量组成的数组。变量名就是该数组的索引。它可以称得上是所有超级变量的超级集合

2.6 PHP 操作符

操作符就是会对它两边的东西有影响或者有“操作”的符号，这种影响可能是赋值、检查或者改

变一个或多个这样的东西。例如完成算术运算的“+”“-”“*”“/”都是操作符。“=”也是一种操作符，称为赋值操作符，可以用来为变量赋值。PHP 的操作符主要包括算术操作符、字符串操作符、赋值操作符、位操作符、逻辑操作符、比较操作符、递增或递减操作符和条件操作符，这里只介绍一些常用的操作符。

2.6.1 算术操作符

视频讲解：光盘\Video\02\2.6.1 算术操作符.mp4

算术操作符是处理四则运算的符号，在对数字的处理中应用得最多。常用的算术操作符如表 2.6 所示。

表 2.6 常用的算术操作符

名　称	操 作 符	举　例
加法运算	+	$a + $b
减法运算	-	$a-$b
乘法运算	*	$a * $b
除法运算	/	$a / $b
取余数运算	%	$a % $b

说明：在算术操作符中使用“%”求余，如果被除数（$a）是负数，那么所得的结果也是一个负值。

实例 03　计算汽车行驶一段距离所需的时间

实例位置：光盘\Code\SL\02\03

视频位置：光盘\Video\02\

本实例将编写一个程序，计算汽车以 80 千米/小时的速度行驶 200 千米需要多长时间，答案为“×小时×分”的格式。相应的公式（用文字表述）是“时间等于距离除以速度”。代码如下：

实例03-1

```
<?php
    $s = 200;                                   //距离
    $v = 80;                                    //速度
    $h = $s/$v;                                 //时间
    echo '需要花费'.$h.'小时';
    echo '<br>';
    /** 转化为"×小时×分"格式 **/
    $h1 = (int)$h;                              //时间取整
    $m  = ($h - $h1)*60;                        //将小数部分转化为分钟
    echo '转化为时/分格式后为: '.$h1.'小时'.$m.'分钟';
?>
```

运行结果如图 2.5 示。

图 2.5　显示计算得出的时间

练一练：

（1）时间对每个人来说都是平等的，一天都有 24 个小时，请试着使用程序计算并输出一周有多少分。（光盘\Code\Try\02\05）

（2）美国洛杉矶当前温度为 72 华氏度，请尝试将其转换为摄氏度。转换公式是 $C=5/9*(F-32)$。（光盘\Code\Try\02\06）

2.6.2 字符串操作符

视频讲解：光盘\Video\02\2.6.2 字符串操作符.mp4

字符串操作符只有一个，即英文的句号“.”。它将两个字符串连接起来，结合成一个新的字符串。例如，将“明日科技”和“有限公司”连接起来。代码如下：

```
<?php
    $str1 = "明日科技";                    //定义一个字符串变量
    $str2 = "有限公司";                    //定义另一个字符串变量
    $str = $str1.$str2;                    //使用“.”操作符将两个变量连接
    echo $str;
?>
```

结果为：

```
明日科技有限公司
```

多学两招：对于字符串型数据，既可以用单引号，也可以用双引号。分别应用单引号和双引号来输出同一个变量，其输出结果完全不同，双引号输出的是变量的值，而单引号输出的是字符串。例如：

```
<?php
    $i = '明日科技';                       //定义一个字符串变量
    echo "$i";                             //用双引号输出，结果为：明日科技
    echo '$i';                             //用单引号输出，结果为：$i
?>
```

2.6.3 赋值操作符

视频讲解：光盘\Video\02\2.6.3 赋值操作符.mp4

赋值操作符是把基本赋值操作符“=”右边的值赋给左边的变量或者常量。PHP 中的赋值操作符如表 2.7 所示。

表 2.7 PHP 中的赋值操作符

操 作	符 号	举 例	展 开 形 式	意 义
赋值	=	$a=3	$a=3	将右边的值赋给左边
加等于	+=	$a+= 2	$a=$a+2	将右边的值加到左边
减等于	-=	$a-= 3	$a=$a-3	将右边的值减到左边
乘等于	*=	$a*=4	$a=$a * 4	将左边的值乘以右边
除等于	/=	$a/= 5	$a=$a / 5	将左边的值除以右边
连接字符	.=	$a.= 'b'	$a=$a.'b'	将右边的字符加到左边
取余数	%=	$a%= 5	$a=$a % 5	将左边的值对右边取余数

注意：混淆“=”和“==”是编程中最常见的错误之一。

2.6.4 递增或递减操作符

视频讲解：光盘\Video\02\2.6.4 递增或递减操作符.mp4

两个加号“++”连接在一起，称为递增操作符。两个减号“--”连接在一起，称为递减操作符。递增或递减操作符有两种使用方法，一种是将操作符放在变量前面，即先将变量作加 1 或减 1 的运算后再将值赋给原变量，叫作前置递增或递减操作符，如图 2.6 所示。

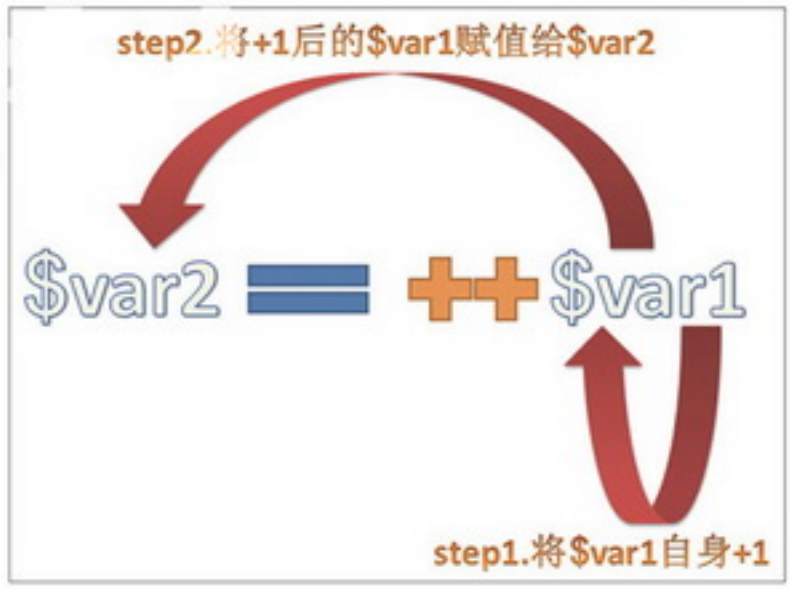


图 2.6 前置递增操作符执行顺序

另一种是将操作符放在变量后面，即先返回变量的当前值，然后变量的当前值作加 1 或减 1 的运算，叫作后置递增或递减操作符。如图 2.7 所示。

图 2.7　后置递增操作符执行顺序

例如，定义两个变量，将这两个变量分别利用递增和递减操作符进行操作，并输出结果。代码如下：

```
<?php
    //前置递增
    $a = 3;
    $b = ++$a;
    echo "前置递增运算后a值为:".$a;
    echo "<br>";
    echo "b值为:".$b;
    echo "<br>";
    //后置递增
    $c = 3;
    $d = $c++;
    echo "后置递增运算后c值为:".$c;
    echo "<br>";
    echo "d值为:".$d;
?>
```

运行结果如下：

```
前置递增运算后a值为:4
b值为:4
后置递增运算后c值为:4
d值为:3
```

2.6.5　逻辑操作符

视频讲解：光盘\Video\02\2.6.5 逻辑操作符.mp4

逻辑操作符用来组合逻辑运算的结果，是程序设计中一组非常重要的操作符。PHP 的逻辑操作符如表 2.8 所示。

表 2.8　PHP 的逻辑操作符

操　作　符	举　　例	结 果 为 真
&& 或 and（逻辑与）	$m && $n	当 $m 和 $n 都为真时
|| 或 or（逻辑或）	$m || $n	当 $m 为真或者 $n 为真时
xor（逻辑异或）	$m xor $n	当 $m、$n 一真一假时
!（逻辑非）	!$m	当 $m 为假时

在进行逻辑判断时，经常要使用逻辑操作符，在后续章节中会进行详细讲解。

2.6.6 比较操作符

视频讲解：光盘\Video\02\2.6.6 比较操作符.mp4

比较操作符就是对变量或表达式的结果进行大小、真假等比较，如果比较结果为真，则返回 true，如果为假，则返回 false。PHP 中的比较操作符如表 2.9 所示。

表 2.9　PHP 的比较操作符

操　作　符	说　　明	举　　例
<	小于	$m<$n
>	大于	$m>$n
<=	小于或等于	$m<=$n
>=	大于或等于	$m>=$n
==	相等	$m= =$n
!=	不等	$m!=$n
===	恒等	$m= = = $n
!==	非恒等	$m!==$n

其中，不太常见的就是 === 和 !==。例如 $a === $b，说明 $a 和 $b 不只是数值上相等，而且两者的类型也一样。例如 false 和 0，在判断时，它们的关系是相等（==），但不是恒等（===）。

2.6.7 条件操作符（或三元操作符）

视频讲解：光盘\Video\02\2.6.7 条件操作符（或三元操作符）.mp4

条件操作符（?:），也称为三元操作符，用于根据一个表达式在另两个表达式中选择一个，而不是用来在两个语句或者程序中选择。条件操作符最好放在括号里使用。

例如，应用条件操作符实现一个简单的判断功能，如果正确则输出“条件运算”，否则输出“没有该值”，代码如下：

```php
<?php
    $value=100;                                          //定义一个整型变量
    echo ($value==true)?"条件运算": "没有该值";           //对整型变量进行判断
?>
```

运行结果为：

```
条件运算
```

2.6.8 操作符的优先级

视频讲解：光盘\Video\02\2.6.8 操作符的优先级.mp4

所谓操作符的优先级，是指在应用中哪一个操作符先计算，哪一个后计算，与数学的四则运算遵循的“先乘除，后加减”是一个道理。

PHP 的操作符在运算中遵循的规则是：优先级高的运算先执行，优先级低的操作后执行，同一优先级的操作按照从左到右的顺序进行；可以像四则运算那样使用小括号，括号内的运算最先进行。同一行中的操作符具有相同优先级，此时它们的结合方向决定求值顺序。操作符的优先级按从高到低的顺序排列如表 2.10 所示。

表 2.10　操作符的优先级

类　型	说　明
clone new	clone 和 new
[	array()
++--	递增 / 递减操作符
~ - (int) (float) (string) (array) (object) (bool) @	类型
instanceof	类型
!	逻辑操作符
* / %	算术操作符
+ - .	算术操作符和字符串操作符
<< >>	位操作符
< <= > >= <>	比较操作符
== != === !==	比较操作符
&	位操作符和引用
^	位操作符

续表

类　　型	说　　明
|	位操作符
&&	逻辑操作符
||	逻辑操作符
?:	条件操作符
= += -= *= /= = %= &= |= ^= <<= >>=	赋值操作符
and	逻辑操作符
xor	逻辑操作符
or	逻辑操作符

这么多的级别，如果想都记住是不太现实的，也没有必要。如果写的表达式真的很复杂，而且包含了较多的操作符，不妨多使用括号，例如：

```
<?php
    $a and (($b != $c) or (5 * (50 - $d)));
?>
```

这样就会减少出现逻辑错误的可能。

2.7 PHP 的表达式

视频讲解：光盘\Video\02\2.7 PHP的表达式.mp4

表达式是构成 PHP 程序语言的基本元素，也是 PHP 最重要的组成元素。最基本的表达式形式是常量和变量。如 $m=20，即表示将值 20 赋给变量 $m。简单的表达式如以下代码所示：

```
<?PHP
    $num = 12;
    $a = "word" ;
?>
```

上述代码是由两个表达式组成的脚本，即 12 和 $a="word"。此外，还可以进行连续赋值，例如：

```
<?php
    $b = $a = 5;
?>
```

因为 PHP 赋值操作的顺序是由右到左的，所以变量 $b 和 $a 都被赋值 5。

在 PHP 的代码中，使用分号“;”来区分表达式，表达式也可以包含在括号内。可以这样理解：一

个表达式再加上一个分号，就是一条 PHP 语句。

注意：在编写程序时，应该注意表达式后面的分号“;”不要漏写。

2.8 PHP 函数

函数就是可以完成某个工作的代码块，它就像是小朋友搭房子用的积木一样，可以反复使用。在使用的时候，拿来即用，而不用考虑它的内部组成。PHP 函数可以分为两类，一类是内置函数，即 PHP 自身的函数，只需要根据函数名调用即可。PHP 备受欢迎的一个原因就是拥有大量的内置函数，包括字符串操作函数和数组操作函数等等。例如 var_dump() 函数就是输出变量的函数；另一类是自定义函数，就是由用户自己定义的、用来实现特定功能的函数。内置函数可以通过查阅《PHP 手册》来学习，下面讲解自定义函数。

2.8.1 定义和调用函数

视频讲解：光盘\Video\02\2.8.1 定义和调用函数.mp4

创建函数的基本语法格式为：

```
<?php
    function fun_name($str1,$str2,$strn){
        fun_body;
    }
?>
```

参数说明如下：

- function：为声明自定义函数时必须使用到的关键字。
- fun_name：为自定义函数的名称。
- $str1…$strn：为函数的参数。
- fun_body：为自定义函数的主体，是功能实现部分。

当函数被定义好后，所要做的就是调用这个函数。调用函数的操作十分简单，只需要引用函数名并赋予正确的参数即可完成函数的调用。

例如，自定义一个函数 example()，计算传入参数的平方，然后连同表达式和结果全部输出。代码如下：

```
01  <?php
02      /* 声明自定义函数 */
03      function example($num){
04          echo "$num * $num = ".$num * $num;          //输出计算后的结果
05      }
06      example(10);                                    //调用函数
07  ?>
```

运行结果为：

```
10 * 10 = 100
```

注意：如果定义了一个函数，但是从未调用这个函数，那么，这些代码将永远也不会执行。

2.8.2 在函数间传递参数

视频讲解：光盘\Video\02\2.8.2 在函数间传递参数.mp4

在调用函数时，有时需要向函数传递参数，如图 2.8 所示。

图 2.8 向函数传递参数

参数传递的方式有按值传递、按引用传递和默认参数 3 种。

1. 按值传递方式

按值传递方式是最常用的参数传递方式，即将调用者括号内的值依次传递给函数括号内的值。下面的例子可以验证函数接收参数的顺序。代码如下：

```
<?php
    function test($parameter1,$parameter2,$parameter3){
        echo '$parameter1 是: '.$parameter1."<br>";
        echo '$parameter2 是: '.$parameter2."<br>";
        echo '$parameter3 是: '.$parameter3;
    }
    test(1,2,3);
?>
```

运行结果如下：

```
$parameter1 是: 1
$parameter2 是: 2
$parameter3 是: 3
```

2. 按引用传递方式

按引用传递就是将参数的内存地址传递到函数中。这时，在函数内部的所有操作都会影响到调用者参数的值。使用引用传递方式传值时只需在原基础上加“&”即可。

按值传递和按引用传递的区别如下：

◆ 按值传递：张三和李四是同事，张三有一间独立的办公室，张三给李四建筑材料，李四也建造了一个跟张三一模一样的办公室，他们俩在各自的办公室办公，彼此独立。

◆ 按引用传递：由于公司经费紧张，将李四安排到张三的办公室。二人各有一把钥匙，共用办公室的资源，张三和李四就会相互影响。

例如，下面的代码中，在第一个参数前添加一个“&”。代码如下：

```php
<?php
    function test(&$parameter1,$parameter2,$parameter3){
        echo '$parameter1 是: '.$parameter1."<br>";
        $parameter1++;
        echo '$parameter2 是: '.$parameter2."<br>";
        echo '$parameter3 是: '.$parameter3."<br>";
    }

    $number1 = 1;
    $number2 = 2;
    $number3 = 3;
    test($number1,$number2,$number3);
    echo "<br>";
    echo '$number1 是: '.$number1."<br>";
    echo '$number2 是: '.$number2."<br>";
    echo '$number3 是: '.$number3."<br>";
?>
```

运行结果如下所示：

```
$parameter1 是: 1
$parameter2 是: 2
$parameter3 是: 3

$number1 是: 2
$number2 是: 2
$number3 是: 3
```

从运行结果中可以看出：第一个参数 &$parameter1 使用引用后，函数体内改变 $parameter1 的值，调用者的参数 $number1 也相应改变，而 $number2 和 $number3 的值则没有改变。

3. 默认参数（可选参数）

还有一种设置参数的方式，即可选参数。可以指定某个参数为可选参数，将可选参数放在参数列表末尾，并且给它指定一个默认值。

例如，使用可选参数实现一个简单的价格计算功能，设置自定义函数 values() 的参数 $tax 为可选参数，其默认值为空。第一次调用该函数，并且给参数 $tax 赋值 0.25，输出价格；第二次调用该函数，不给参数 $tax 赋值，输出价格。代码如下：

```php
<?php
    function values($price,$tax=0){          //定义一个函数，其中的一个参数初始值为0
        $price=$price+($price*$tax);         //声明一个变量$price，等于两个参数的运算结果
        echo "价格:$price<br>";              //输出价格
    }
    values(100,0.25);                        //为可选参数赋值0.25
    values(100);                             //没有给可选参数赋值
?>
```

运行结果为：

```
价格:125
价格:100
```

注意：当使用默认参数时，默认参数必须放在非默认参数的右侧，否则函数可能出错。

2.8.3 从函数中返回值

视频讲解：光盘\Video\02\2.8.3 从函数中返回值.mp4

调用函数时可以向函数发送信息（参数），函数也可以向调用者发回信息（返回值）。从函数返回的值称为结果（result）或返回值（return value）。函数将返回值传递给调用者的方式是使用关键字 return。return 将函数的值返回给函数的调用者，即将程序控制权返回到调用者的作用域。该过程如图 2.9 所示。

图 2.9　向函数传递参数及函数返回值

实例 04　计算购物车中商品的总价

实例位置：光盘\Code\SL\02\04
视频位置：光盘\Video\02\

本实例将模拟淘宝购物车功能，并计算购物车中商品的总价。购物车中有如下商品信息：

苹果手机单价 5000 元，购买数量 2 台；联想笔记本电脑单机 8000 元，购买数量 10 台。

操作步骤：先定义一个函数，命名“total”，该函数的作用是输入物品的单价和数量，然后计算总金额，最后返回商品金额。代码如下：

实例04-1

```php
<?php
```

```
02      //定义total()函数，计算商品总价
03      function total($price,$number){
04          $total = $price * $number;
05          return $total;
06      }
07      $sum = 0;
08      $phone     = total(5000,2);              //调用函数，计算手机价格
09      $computer  = total(8000,10);             //调用函数，计算电脑价格
10      $sum = $phone + $computer;
11      echo "合计".$sum."元";
12  ?>
```

结果为：

```
合计90000元
```

return 语句只能返回一个参数，即只能返回一个值，不能一次返回多个值。如果要返回多个结果，就要在函数中定义一个数组，将返回值存储在数组中返回。

练一练：

（1）淘宝的收货地址栏由省、市、区和详细地址组成，试着定义一个 address() 函数，并且包含省、市、区和详细地址 4 个参数，该函数用于拼接一个完整的收货地址。（光盘\Code\Try\02\07）

（2）周恩来总理智力超人，一次，有位外国首脑问周总理："贵国共有多少现金？"周总理轻松地回答："有 18 元 8 角 8 分。"周总理为何这么说呢？原来当时我国流通使用的第三套人民币共有 10 种面额，即 10 元、5 元、2 元、1 元、5 角、2 角、1 角、5 分、2 分、1 分，累计是 18 元 8 角 8 分。而现在流通使用的第五套人民币增加了 20 元、50 元和 100 元面额的人民币。试着定义一个计算面额总值的函数，该函数包含元、角、分三个参数。调用该函数，分别输出第三套人民币和第五套人民币的面额总值。（光盘\Code\Try\02\08）

2.8.4 变量作用域

视频讲解：光盘\Video\02\2.8.4 变量作用域.mp4

在编写代码时，虽然有些变量在函数之外，有些在函数之内，但它们必须在有效范围内使用，如果变量超出有效范围，变量也就失去其意义。变量的作用域如表 2.11 所示。

表 2.11　变量的作用域

作　用　域	说　　明
局部变量	在函数的内部定义的变量，其作用域是所在函数
全局变量	被定义在所有函数以外的变量，其作用域是整个 PHP 文件，但在用户自定义函数内部是不可用的。如果希望在用户自定义函数内部使用全局变量，则要使用 global 关键字声明

续表

作　用　域	说　　明
静态变量	能够在函数调用结束后仍保留变量值，当再次回到其作用域时，又可以继续使用原来的值。而一般变量在函数调用结束后，其存储的数据值将被清除，所占的内存空间被释放。使用静态变量时，先要用关键字 static 来声明变量，把关键字 static 放在要定义的变量之前

在函数内部定义的变量，其作用域为所在函数，如果在函数外赋值，将被认为是完全不同的另一个变量。在退出声明变量的函数时，该变量及相应的值就会被清除。

实例 05　比较局部变量和全局变量

实例位置：光盘\Code\SL\02\05
视频位置：光盘\Video\02\

比较在函数内赋值的变量（局部变量）和在函数外赋值的变量（全局变量），代码如下：

实例05-1

```
<?php
    $example="在……函数外";                          //声明全局变量
    function example(){
        $example="……在函数内……";                    //声明局部变量
        echo "在函数内输出的内容是：$example.<br>";    //输出局部变量
    }
    example();                                       //调用函数，输出变量值
    echo "在函数外输出的内容是：$example.<br>";       //输出全局变量
?>
```

运行结果如图 2.10 所示。

图 2.10　输出局部变量和全局变量

练一练：

（1）试着使用静态变量和普通变量同时输出一个数据，查看一下两者的功能有什么不同。（光盘\Code\Try\02\09）

（2）试着在函数内部使用 global 关键字，并查看其作用域范围。（光盘\Code\Try\02\10）

2.9 PHP 编码规范

如今的 Web 开发，不再是一个人就可以全部完成的，尤其是一些大型的项目，要十几人，甚至几十人共同来完成。在开发过程中，可能会有新的开发人员参与进来，那么这个新的开发人员在阅读前

人留下的代码时，就会有问题了——这个变量起到什么作用？那个函数实现什么功能？ TmpClass 类在哪里被使用到了？……诸如此类。这时，编码规范的重要性就体现出来了。

以 PHP 开发为例，编码规范就是融合开发人员长时间积累下来的经验，形成一种良好统一的编程风格，这种良好统一的编程风格会在团队开发或二次开发时达到事半功倍的效果。编码规范是一种总结性的说明和介绍，并不是强制性的规则。

PSR 是 PHP Standard Recommendations 的简写，由 PHP FIG 组织制定的 PHP 规范，是 PHP 开发的实践标准。PHP 标准组提出并发布了一系列的风格建议。其中有部分是关于代码风格的，即 PSR-0、PSR-1、PSR-2 和 PSR-4，但这些推荐只是一些被其他项目所遵循的规则，如 Drupal、Zend、Symfony、CakePHP、phpBB、AWS SDK、FuelPHP、Lithium 等。

2.9.1 PSR-1 基础编码规范

视频讲解：光盘\Video\02\2.9.1 PSR-1基础编码规范.mp4

本篇规范制定了代码基本元素的相关标准，以确保共享的 PHP 代码间具有较高程度的技术互通性。

◆ PHP 代码文件必须以“<?php”或“<?=”标签开始；

◆ PHP 代码文件必须以不带 BOM 的 UTF-8 编码；

◆ PHP 代码中应该只定义类、函数、常量等，或其他会产生副作用的操作（如：生成文件输出以及修改 .ini 配置文件等），二者只能选其一。

◆ 命名空间以及类必须符合 PSR 的自动加载规范：PSR-4 中的一个。根据规范，每个类都独立为一个文件，且命名空间至少有一个层次：顶级的组织名称（vendor name）。类的命名必须遵循 StudlyCaps 大写开头的驼峰命名规范。PHP 5.3 及以后版本的代码必须使用正式的命名空间。例如：

```
01  <?php
02  //PHP 5.3及以后版本的写法
03  namespace Vendor\Model;
04
05  class Foo
06  {
07  }
```

PHP 5.2.x 及之前的版本应该使用伪命名空间的写法，约定俗成使用顶级的组织名称（vendor name），例如：

```
01  <?php
02  //PHP5.2.x及之前版本的写法
03  class Vendor_Model_Foo
04  {
05  }
```

◆ 类中的常量所有字母都必须大写，单词间用下划线分隔，例如：

```
01  <?php
```

```
02  namespace Vendor\Model;
03
04  class Foo
05  {
06      const VERSION = '1.0';
07      const DATE_APPROVED = '2012-06-01';
08  }
```

方法名称必须符合 camelCase 式的小写开头驼峰命名规范。

2.9.2 PSR-2 编码风格规范

视频讲解：光盘\Video\02\2.9.2 PSR-2编码风格规范.mp4

本篇规范是 [PSR-1] 基本代码规范的继承与扩展。

◆ 代码必须遵循 PSR-1 中的编码规范。

◆ 代码必须使用 4 个空格符而不是 <Tab> 键进行缩进。

◆ 每行的字符数应该软性保持在 80 个之内，理论上一定不可多于120 个，但一定不可有硬性限制。

◆ 每个 namespace 命名空间声明语句和 use 声明语句块后面，必须插入一个空白行。

◆ 类的开始大括号“{”必须写在函数声明后自成一行，结束大括号“}”也必须写在函数主体后自成一行。

◆ 方法的开始大括号“{”必须写在函数声明后自成一行，结束大括号“}”也必须写在函数主体后自成一行。

◆ 类的属性和方法必须添加访问修饰符（private、protected 以及 public），abstract 以及 final 必须声明在访问修饰符之前，而 static 必须声明在访问修饰符之后。

◆ 控制结构的关键字后必须要有一个空格符，而调用方法或函数时则一定不可有。

◆ 控制结构的开始大括号“{”必须写在声明的同一行，而结束大括号“}”必须写在主体后自成一行。

◆ 控制结构的开始大括号“{”后和结束大括号“}”前，都不可以有空格符。

以下代码简单地展示了上述大部分规范：

```
01  <?php
02  namespace Vendor\Package;
03
04  use FooInterface;
05  use BarClass as Bar;
06  use OtherVendor\OtherPackage\BazClass;
07
08  class Foo extends Bar implements FooInterface
09  {
10      public function sampleFunction($a, $b = null)
11      {
12          if ($a === $b) {
```

```
13              bar();
14          } elseif ($a > $b) {
15              $foo->bar($arg1);
16          } else {
17              BazClass::bar($arg2, $arg3);
18          }
19      }
20
21      final public static function bar()
22      {
23          //方法的内容
24      }
25  }
```


2.10 难点解答

2.10.1 类型转换异常

值类型变量直接存储其数据值，主要包含整数类型、浮点类型以及布尔类型等。值类型变量在栈中进行分配，因此效率很高，使用值类型变量主要目的是为了提高性能。在类型转换时，注意不要将未知的小数强制转换为 integer，这样有时会导致不可预料的结果，例如：

```
01  <?php
02  echo (int) ( (0.1+0.7) * 10 );          //输出7
03  ?>
```

2.10.2 什么函数需要使用默认参数

参数个数不确定的情况下，如果调用函数时，缺少参数，PHP 会提示“参数缺失”。例如，定义一个函数 address($province,$city,$district,$detail)，这 4 四个参数分别代表省、市、区和具体住址。但是对于直辖市而言，如北京，不是以省份开头。所以，就没有相应的第 4 个参数。这时，就可以使用默认参数来解决该问题，可以这样定义函数：address($province,$city,$district,$detail='')。使第 4 个参数默认为空字符串。

2.11 小　结

本章主要介绍了 PHP 语言的基础知识，包括数据类型、常量、变量、操作符、表达式和自定义函数，并详细介绍了各种类型之间的转换、系统预定义的常量、变量、操作符的优先级、如何使用函数

和 PHP 编码规范。基础知识是一门语言的核心，希望初学者能静下心来，牢牢掌握这些知识，这对以后的学习和发展能起到事半功倍的效果。

2.12 动手纠错

1．运行“光盘\Code\Debug\02\01”文件夹下的程序，没有输出内容，请根据注释将变量类型改为整型，改正后运行程序。

2．运行“光盘\Code\Debug\02\02”文件夹下的程序，出现没有定义变量 a 和变量 b 的错误提示，请根据注释改正程序。

3．运行“光盘\Code\Debug\02\03”文件夹下的程序，出现结果为浮点型数值，请根据注释改正程序，使输出结果为整型。

4．运行“光盘\Code\Debug\02\04”文件夹下的程序，运行结果与预期不一致，请根据注释改正程序。

5．运行“光盘\Code\Debug\02\05”文件夹下的程序，运行结果与预期不一致，请根据注释改正程序。

第3章 流程控制语句

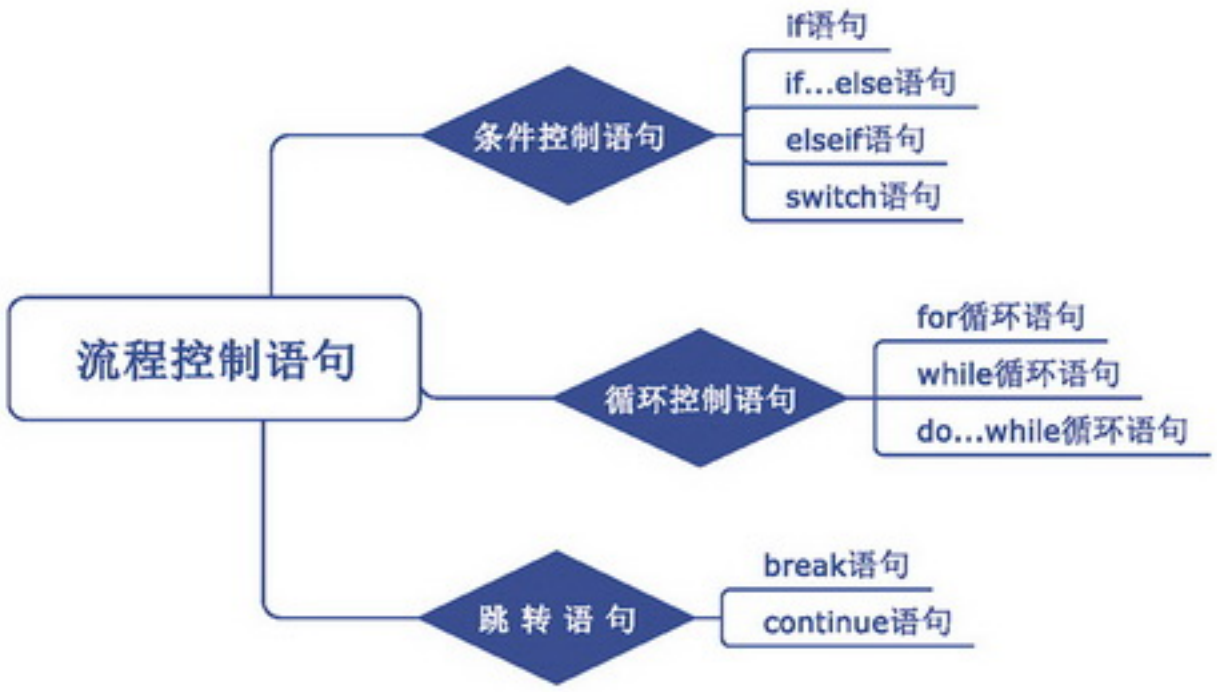
（视频讲解：1小时22分）

本章概览

流程控制对于任何一门编程语言来说都是至关重要的，它能够控制程序的执行顺序。合理使用这些控制结构可以使程序流程清晰、可读性强，从而提高开发效率。例如输出10以内的偶数、计算100的阶乘、列举1000以内的所有素数，使用PHP中的流程控制语句可以很快地解决上述问题。

本章将详细讲解PHP中的流程控制语句，并以示意图的方式展现各种流程控制语句的执行过程，让读者真正理解，从而在开发中灵活运用。

知识框架



3.1 条件控制语句

在生活中，总会遇到需要做判断和决策的情况，程序也是一样。下面给出几个常见的例子：

◆ 如果购买成功，用户余额减少，用户积分增多。

◆ 如果输入的用户名和密码正确，提示登录成功，进入网站，否则，提示登录失败。

◆ 如果用户使用微信登录，则使用微信扫一扫；如果使用 QQ 登录，则输入 QQ 号和密码；如果使用微博登录，则输入微博号和密码；如果使用手机号登录，则输入手机号和密码。

以上例子中的判断，就是程序中的条件控制语句。按照条件选择执行不同的代码片段。条件控制语句主要有 if、if...else、elseif 和 switch4 种。下面分别进行讲解。

3.1.1 if 语句

视频讲解：光盘\Video\03\3.1.1 if语句.mp4

PHP 的 if 语句的格式为：

```
<?php
    if (表达式)
        语句 ;
?>
```

如果表达式的值为真，那么就顺序执行语句；否则，就会跳过该条语句，再往下执行。如果需要执行的语句不止一条，那么可以使用“{}”，在“{}”中的语句被称为语句组，其格式为：

```
<?php
    if(表达式){
        语句1;
        语句2;
        …
    }
?>
```

if 语句的流程就像一辆运行的火车，从 A 站出发，可以直接到达 C 站，也可以经过 B 站，然后再到达 C 站。如图 3.1 所示。

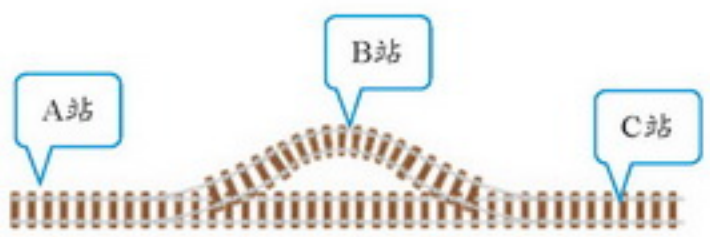


图 3.1　模拟 if 语句流程控制图

实例 01　判断随机数是不是偶数

实例位置：光盘\Code\SL\03\01
视频位置：光盘\Video\03\

本实例将使用 rand() 函数生成一个随机数 $num，然后判断这个随机数是不是偶数，如果是，则输出结果。代码如下：

实例01-1

```
<?php
    $num = rand(1,20);                          //使用rand()函数生成一个随机数
    echo '$num = '.$num;                        //输出随机数
    if ($num % 2 == 0){                         //判断变量$num是否为偶数
        echo "<br>$num 是偶数。";
    }
?>
```

运行结果如图 3.2 所示。

图 3.2　判断随机数是否为偶数

说明：rand() 函数的作用是获取一个随机的整数，每次刷新页面后，会生成一个新的随机数，可能与图 3.2 所示不同。

练一练：

（1）已知英语四级考试及格线为 425 分，一位大二学生英语四级分数为 424 分，判断并输出该学生是否通过了英语四级考试。（光盘\Code\Try\03\01）

（2）公司年会抽奖，中奖号码及奖品设置如下：

① “1” 代表一等奖，奖品是 42 寸彩电。

② “2” 代表二等奖，奖品是光波炉。

③ “3” 代表三等奖，奖品是加湿器。

④ “4” 代表安慰奖，奖品是 16G-U 盘。

使用 rand() 函数生成随机数，根据随机的奖号，输出与该奖号对应的奖品。（光盘\Code\Try\03\02）

3.1.2 if...else 语句

视频讲解

视频讲解：光盘\Video\03\3.1.2 if...else语句.mp4

有时程序需要在满足某个条件时执行一条语句，而在不满足该条件时执行其他语句。这时可以使用 if...else 语句，其语法格式为：

```
<?php
    if(表达式){
```

```
        语句1;
    }else{
        语句2;
    }
?>
```

该语句的含义为：当表达式的值为真时，执行语句 1；如果表达式的值为假，则执行语句 2。就像一辆运行的火车，只有两条轨道可以选择，如图 3.3 所示。

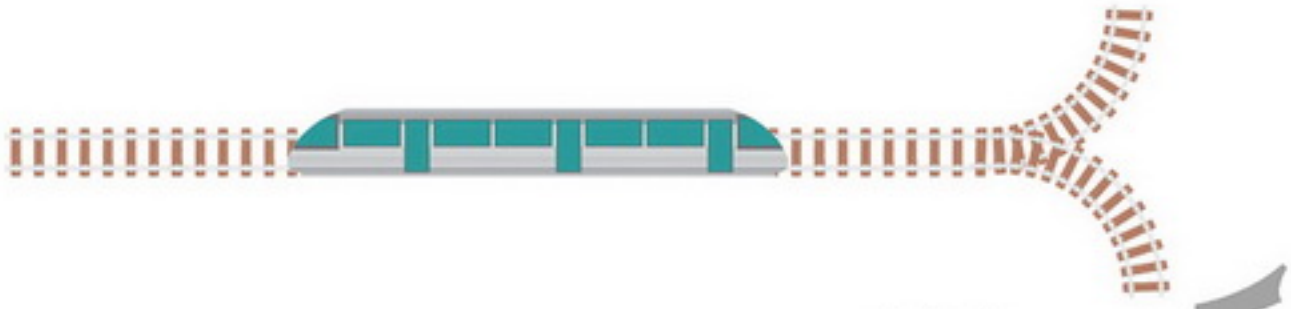


图 3.3　模拟 if...else 语句流程控制图

如，以实例 01 为基础，首先使用 rand() 函数生成一个随机数 $num，然后判断这个随机数是偶数还是奇数，再根据不同结果显示不同的字符串。代码如下：

```
<?php
    $num = rand(1,20);                          //使用rand()函数生成一个随机数
    if ($num % 2 == 0){                         //判断变量$num是否为偶数
            echo '变量'.$num.'是偶数。';          //如果为偶数
        }else {
            echo '变量'.$num.'是奇数。';          //如果为奇数
    }
?>
```

3.1.3 elseif 语句

视频讲解：光盘\Video\03\3.1.3 elseif语句.mp4

if...else 语句只能选择两种结果：要么执行语句 1，要么执行语句 2。但有时会出现两种以上的选择，例如：一个班的考试成绩，90 分以上的为“优秀”，60 ～ 90 分之间的为“良好”，低于 60 分的为“不及格”。这时可以使用 elseif 语句来执行，语法格式为：

```
<?php
    if(表达式1){
        语句1;
    }elseif(表达式2){
        语句2;
    }…
    else{
        语句n;
```

```
    }
?>
```

elseif 语句的流程就像一辆运行的火车，从 A 站出发到达 B 站，有多条线路可以选择，根据铁路局的不同指示，选择相应的路线。如图 3.4 所示。

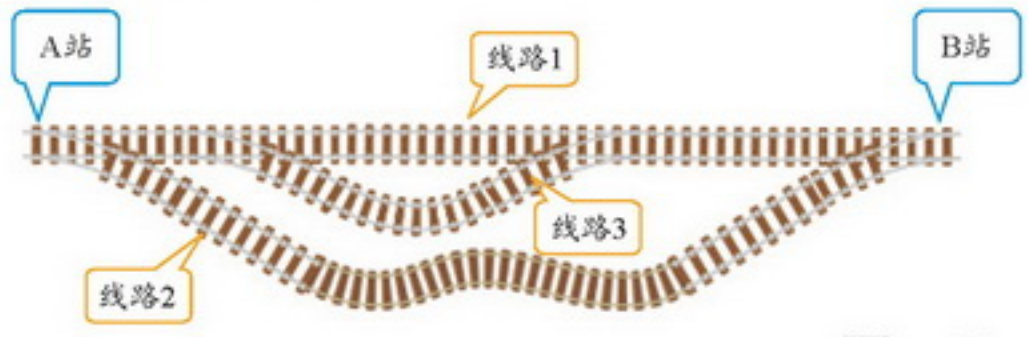


图 3.4　模拟 elseif 语句的流程控制图

实例 02　判断某个日期是该月的哪一旬

实例位置：光盘\Code\SL\03\02
视频位置：光盘\Video\03\

本实例将通过 elseif 语句，判断某个日期是该月的上旬、中旬、下旬中的哪一旬。代码如下：

实例02-1

```
<?php
    date_default_timezone_set('Asia/Shanghai');          //设置时区
    $month = date("n");                                  //设置月份变量$month
    $today = date("j");                                  //设置日期变量$today
    if ($today >= 1 and $today <= 10){                   //判断日期变量是否在1～10之间
        echo "今天是".$month."月".$today."日，是本月上旬"; //如果是，说明是上旬
    }elseif($today > 10 and $today <= 20){               //否则判断日期变量是否在11～20之间
        echo "今天是".$month."月".$today."日，是本月中旬"; //如果是，说明是中旬
    }else{                                               //如果上面两个判断都不符合要求，则输出默认值
        echo "今天是".$month."月".$today."日，是本月下旬"; //说明是本月的下旬
    }
?>
```

运行结果如图 3.5 所示。

图 3.5　判断某个日期是该月的哪一旬

练一练：

（1）将学生成绩转化为不同等级，划分标准如下：

① “优秀”，大于或等于 90 分。

② “良好”，大于或等于 80 分，小于 90 分。

③ “合格”，大于或等于 60 分。

④ “不合格”，小于 60 分。

使用 rand() 函数随机生成成绩，输出与该成绩对应的等级。（光盘\Code\Try\03\03）

（2）BMI 身体质量指数的等级划分标准如下：

① “偏轻”，BMI 小于 18.5；

② “正常”，BMI 大于或等于 18.5，小于 25；

③ “偏重”，BMI 大于或等于 25，小于 30；

④ “肥胖”，BMI 大于或等于 30。

使用 rand() 函数随机输出 BMI 指数以及与该 BMI 指数对应的等级。（光盘\Code\Try\03\04）

3.1.4 switch 语句

视频讲解：光盘\Video\03\3.1.4 switch语句.mp4

虽然 elseif 语句可以进行多种选择，但如果条件较多时，就会变得十分烦琐。为了避免 if 语句过于冗长，提高程序的可读性，可以使用 switch 分支控制语句。switch 语句的语法格式如下：

```
<?php
    switch(变量或表达式){
        case 常量表达式1:
            语句1;
            break;
        case 常量表达式2:
            …
        case 常量表达式n:
            语句n;
            break;
        default:
            语句n+1;
    }
?>
```

switch 语句根据变量或表达式的值，依次与 case 中的常量表达式的值相比较，如果不相等，继续查找下一个 case；如果相等，就执行对应的语句，直到 switch 语句结束或遇到 break 为止。一般来说，switch 语句最终都有一个默认值 default，如果在前面的 case 中没有找到相符的条件，则输出默认语句，和 else 语句类似。

实例 03	选择第三方登录接口	实例位置：光盘\Code\SL\03\03
		视频位置：光盘\Video\03\

明日学院网站支持第三方登录，第三方登录包括 QQ 登录、微信登录、微博登录等。根据不同的登录方式，需要调用相应的第三方接口，这时，可以根据网址中传递的值不同，使用 switch 语句判断用户选择了哪一个第三方应用，然后调用该应用的接口。代码如下：

实例03-1

```php
<?php
    //接收传递的参数，并使用三元运算符判断赋值
    $type = isset($_GET['type']) ? $_GET['type'] : '';
    //根据参数值，执行不同的操作
    switch($type)
    {
        case 'qq':
            echo "执行qq登录流程";
            break;
        case 'wechat':
            echo "执行微信登录流程";
            break;
        case 'weibo':
            echo "执行微博登录流程";
            break;
        default:
            echo "执行普通登录流程";
    }
?>
```

运行结果如图 3.6 所示。

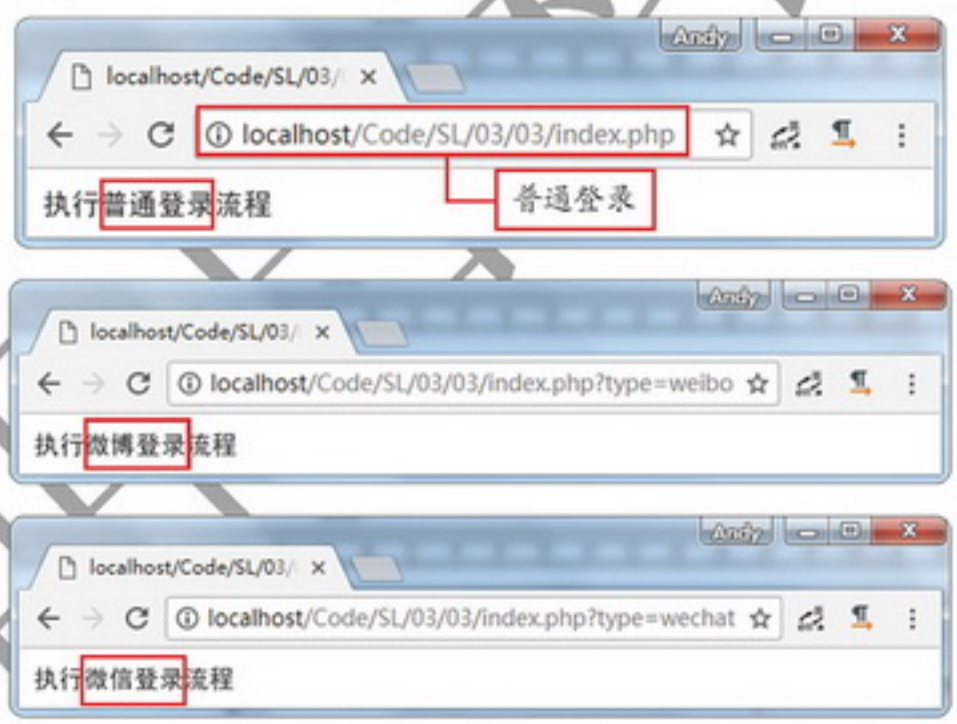


图 3.6　switch 多重判断语句

注意： switch 语句在执行时，即使遇到符合要求的 case 语句段，也会继续往下执行，直到 switch 语句结束。为了避免这种浪费时间和资源的行为，一定要在每个 case 语句段后加上 break 语句。这里 break 语句的意思是跳出当前 case 语句，在 3.3.1 小节中将详细介绍 break 语句。

练一练：

（1）使用 switch 语句判断输入的某个月份属于哪个季节。（光盘\Code\Try\03\05）

（2）某大型商超为答谢新老顾客，当累计消费金额达到一定数额时，顾客可享受不同的折扣：

①尚未超过 200 元，顾客须按照小票价格支付全款。

②不少于 200 元但尚未超过 600 元，顾客全部的消费金额可享 8.5 折优惠。
③不少于 600 元但尚未超过 1000 元，顾客全部的消费金额可享 7 折优惠。
④不少于 1000 元，顾客全部的消费金额可享 6 折优惠。
随机生成顾客消费金额，输出该顾客将享受的折扣与打折后需支付的金额。（光盘\Code\Try\03\06）

3.2 循环控制语句

对于大多数人来说，反复地做同样的事情会让人厌烦，但计算机却非常擅长完成重复的任务。计算机程序通常会周而复始地重复同样的步骤，这称为循环。循环主要有两种类型：

重复一定次数的循环，称为计数循环，如 for 循环；重复直至发生某种情况时结束的循环，称为条件循环（conditional loop），只要条件为真，这种循环会一直持续下去，如 while 循环和 do...while 循环。

3.2.1 for 循环语句

视频讲解：光盘\Video\03\3.2.1 for循环语句.mp4

for 循环是 PHP 的计数循环结构，它的语法格式为：

```
<?php
    for (初始化表达式; 条件表达式; 迭代表达式){
        语句;
    }
?>
```

其中，初始化表达式在第一次循环时无条件取一次值；条件表达式在每次循环开始前求值，如果值为真，则执行循环体里面的语句，否则跳出循环，继续往下执行；迭代表达式在每次循环后被执行。for 循环语句的流程控制图如图 3.7 所示。

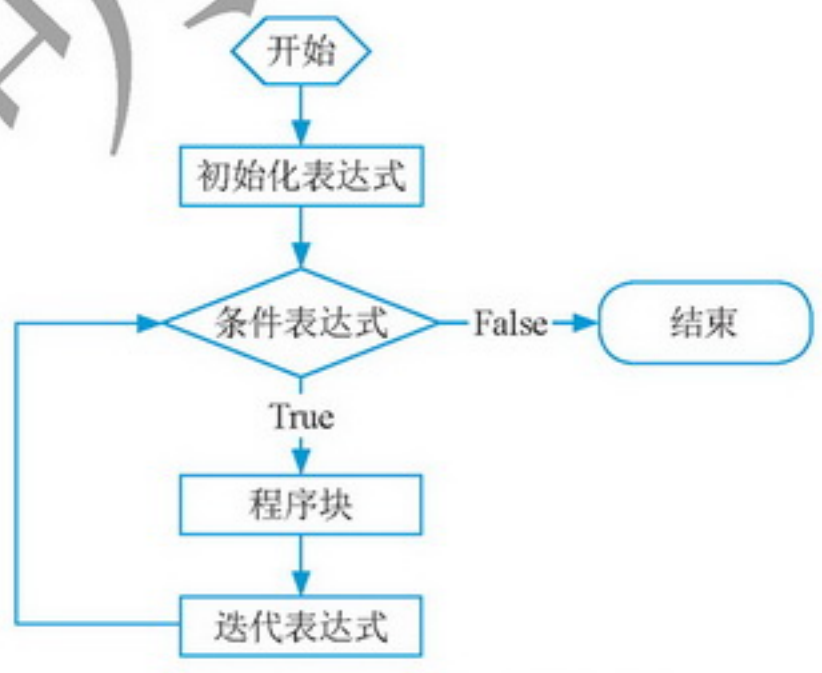


图 3.7　for 循环语句流程控制图

可以以现实生活中的例子来理解 for 循环的执行流程：在体育课上，体育老师要求同学们沿着环形操场跑步 3 圈。老师从 0 开始计数，每次跑完 1 圈，将数量加 1。当完成第 3 圈时，同学会停下来，即循环结束。

实例 04　使用 for 循环来计算 100 的阶乘

实例位置：光盘\Code\SL\03\04
视频位置：光盘\Video\03\

本实例将通过 for 循环来计算 100 的阶乘，即 1×2×3×4×……×100。具体代码如下：

实例04-1

```
<?php
    $sum = 1;                                    //声明整型变量$sum
    for ($i = 1;$i <=100;$i++){
        $sum *= $i;                              //当$i小于或等于100时，计算阶乘
    }
    echo "100的阶乘是".$sum;
?>
```

上述代码中，第一步，执行 for 循环的初始表达式，即为 $i 赋值为 1。第二步，判断条件表达式，即 $i 是否小于或等于 100，如果判断的结果为 true，则执行下面的程序块，将 $sum 乘以当前的 $i。否则跳出循环，不再继续执行。第三步，执行迭代表达式，即将 $i 加 1。此时，第一次循环结束，$i 的值为 2。然后判断 $i 是否小于或等于 100，重复第一次的操作。当 $i 为 100 时，执行第 100 次程序块代码。然后 $i 继续迭代，值为 101。此时，判断表达式的结果为 false，循环结束，不再执行。运行结果如图 3.8 所示。

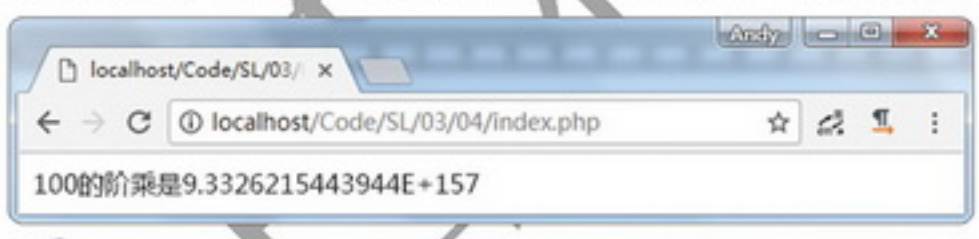


图 3.8　使用 for 循环计算 100 的阶乘

注意：在 for 语句中，无论采用循环变量递增还是递减的方式，前提是一定要保证循环能够结束，无期限的循环（死循环）将导致程序崩溃。

练一练：

（1）试着用 for 循环求 1+2+…+100 的和。（光盘\Code\Try\03\07）

（2）试着用 for 循环打印乘法口诀表。（光盘\Code\Try\03\08）

3.2.2 while 循环语句

视频讲解：光盘\Video\03\3.2.2 while循环语句.mp4

while 循环是 PHP 中条件循环语句的一种，它的语法格式为：

```
<?php
    while (expr)
        statement
?>
```

当表达式的值为真时，将执行循环体内的 PHP 语句。执行结束后，再返回到表达式继续进行判断。直到表达式的值为假，才跳出循环。

while 循环语句的流程控制如图 3.9 所示。

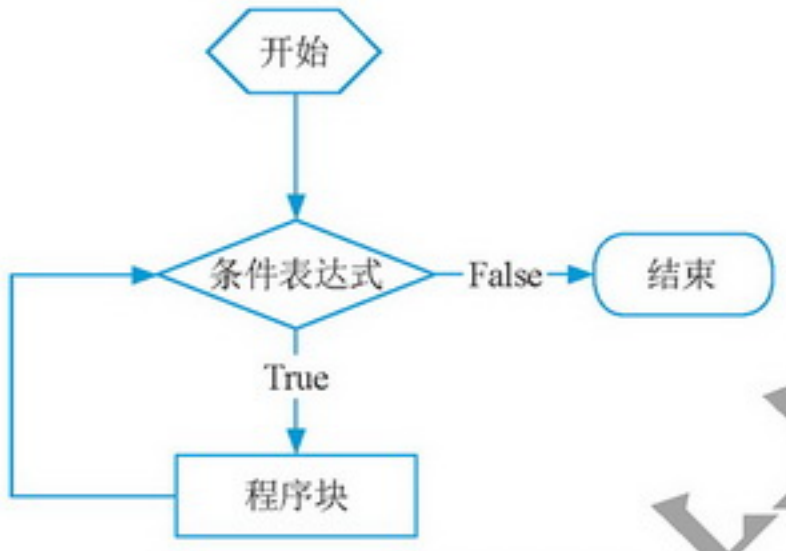


图 3.9　while 语句流程控制图

同样以沿着操场跑步的例子来理解 while 循环。这一次，老师没有要求同学们跑几圈，而是要求当听到老师吹的哨子声时就停下来。同学们每跑一圈，可能会请求一次老师吹哨子。如果老师吹哨子，则停下来，即循环结束；否则，继续跑步，即执行循环。

实例 05　使用 while 循环输出 10 以内的偶数

实例位置：光盘\Code\SL\03\05

视频位置：光盘\Video\03\

本实例将依次判断 1 ～ 10 以内的数是否为偶数，如果是，则输出；如果不是，则继续下一次循环。代码如下：

实例05-1

```
<?php
    $num = 1;                          //定义一个整型变量$num
    $str = "10以内的偶数为：";          //定义一个字符变量$str
    while($num <= 10){                 //判断变量$num是否小于或等于10
        if($num % 2 == 0){             //如果小于或等于10，则判断$num是否为偶数
            $str .= $num." ";          //如果当前变量为偶数，则添加到字符变量$str的后面
        }
        $num++;                        //变量$num加1
    }
    echo $str;                         //循环结束后，输出字符串$str
?>
```

运行结果如图 3.10 所示。

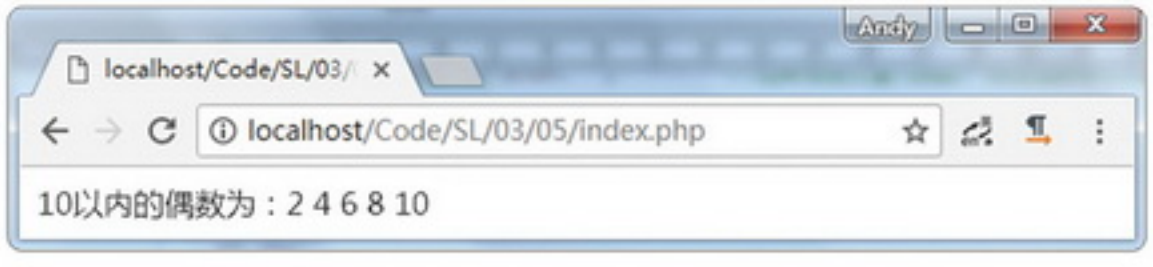


图 3.10　使用 while 循环输出 10 以内的偶数

练一练：

（1）猜数字游戏：假设目标数字为 23，使用 while 循环实现 1~50 随机猜数，猜对终止程序。（光盘\Code\Try\03\09）

（2）输出一个形状呈三角形的金字塔。该金字塔共 5 行，第 1 行 1 颗星，第 2 行 3 颗星，第 3 行 5 颗星，第 4 行 7 颗星，第 5 行 9 颗星。（光盘\Code\Try\03\10）

3.2.3 do...while 循环语句

视频讲解：光盘\Video\03\3.2.3 do...while循环语句

while 语句还有另一种形式的表示，即 do...while。两者的区别在于，do...while 要比 while 语句多循环一次。当 while 表达式的值为假时，while 循环直接跳出当前循环；而 do...while 语句则是先执行一遍程序块，然后再对表达式进行判断。do...while 语句的流程控制如图 3.11 所示。

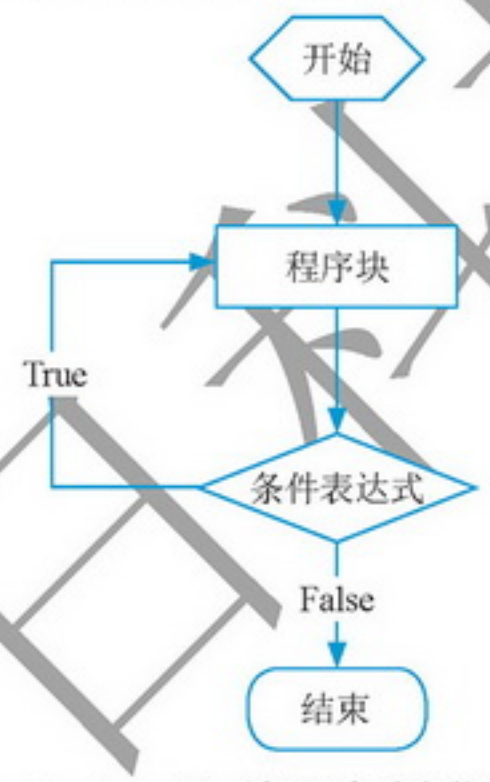


图 3.11　do...while 循环语句流程控制图

依然以沿着操场跑步的例子来理解 do...while 循环。这一次，老师要求同学们先跑 1 圈，然后当听到老师吹的哨子声时再停下来。

实例 06　对比 while 语句和 do...while 语句

实例位置：光盘\Code\SL\03\06

视频位置：光盘\Video\03\

分别使用 while 语句和 do...while 语句执行相同的代码块，即使用 echo 语句输出一段内容，并对比这两个语句的区别，代码如下：

实例06-1

```
<?php
    $num = 1;                                   //定义一个整型变量$num
    while($num != 1){                           //使用while循环输出
        echo "执行while循环";                    //这句话不会输出
    }
    do{                                         //使用do…while循环输出
```

```
07            echo "执行do…while循环";                    //这句话会输出
08        }while($num != 1);
09    ?>
```

结果如图 3.12 所示。

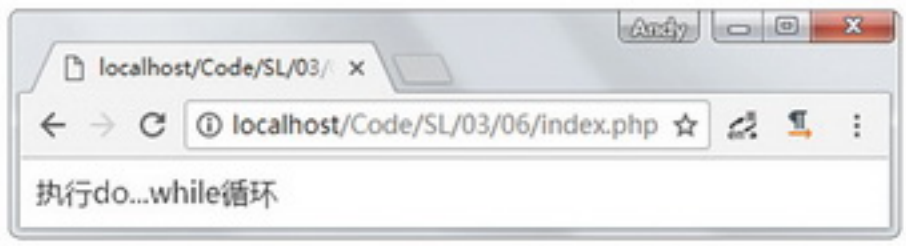


图 3.12　while 和 do...while 的区别

练一练：

（1）用户输入一个值，从这个值开始，依次与这个值之后的连续 n 个自然数相加，当和超过 100 时结束，输出此时的和与自然数的值。（光盘\Code\Try\03\11）

（2）自动售卖机有三种饮料，价格分别为 3 元、5 元、7 元。自动售卖机仅支持 1 元硬币支付，请编写该售卖机自动收费系统。（光盘\Code\Try\03\12）

3.3 跳转语句

当循环条件一直满足时，程序将会一直执行下去，就像一辆迷路的车，在某个地方不停地转圆圈。如果希望在中间离开循环，也就是 for 循环结束计数之前，或者 while 循环找到结束条件之前。有两种方法来做到：

- break：完全中止循环。
- continue：直接跳到循环的下一次迭代。

3.3.1 break 语句

视频讲解：光盘\Video\03\3.3.1 break语句.mp4

break 关键字可以终止当前的循环，包括 while、do...while、for 和 switch 在内的所有控制语句。以独自一人沿着操场跑步为例，计划跑步 10 圈。可是在跑到第 2 圈的时候，遇到自己喜欢的人，于是果断停下来，终止跑步，这样就提前终止循环。

实例 07　使用 break 语句终止循环

实例位置：光盘\Code\SL\03\07
视频位置：光盘\Video\03\

本实例先使用 while 循环，while 后面的表达式的值为 true，即为一个无限循环。在 while 程序块中将声明一个随机数变量 $tmp，当生成的随机数等于 10 时，使用 break 语句跳出循环。代码如下：

实例07-1

```
01    <?php
```

```
02      while(true){                                    //使用while循环
03          $tmp = rand(1,20);                          //声明一个随机数变量$tmp
04          echo $tmp." ";                              //输出随机数，使用空格分隔
05          if($tmp == 10){                             //判断随机数是否等于10
06              echo "<p>变量等于10，终止循环</p>";
07              break;                                  //如果等于10，使用break语句跳出循环
08          }
09      }
10  ?>
```

运行结果如图 3.13 所示。

图 3.13　使用 break 语句退出循环

练一练：

（1）地铁 1 号线共有 18 个地铁站，某人乘坐 1 号线从始发站前往第 4 站，请输出此人经过哪些地铁站。（地铁站名采用数字编号，例如第 4 站）。（光盘\Code\Try\03\13）

（2）有一口井深 10 米，一只蜗牛从井底向井口爬，白天向上爬 2 米，晚上向下滑 1 米，问多少天可以爬到井口？（光盘\Code\Try\03\14）

3.3.2 continue 语句

视频讲解：光盘\Video\03\3.3.2 continue语句.mp4

continue 关键字的作用没有 break 关键字强大，它只能终止本次循环而进入到下一次循环中，continue 也可以指定跳出几重循环。

仍然以独自一人沿着操场跑步为例，计划跑步 10 圈。当跑到第 2 圈一半的时候，看到自己喜欢的人也在跑步，于是果断停下来，跑回起点等待，制造一次完美邂逅，然后从第 3 圈继续开始跑步。

实例 08　使用 continue 语句跳出循环

实例位置：光盘\Code\SL\03\08

视频位置：光盘\Video\03\

本实例使用 for 循环输出 0 到 4，当 $i 等于 2 时，执行 continue 语句，此时不执行下面的 echo 语句，跳出该循环，继续执行 $i 等于 3 的语句。代码如下：

实例08-1

```
01  <?php
02      for ($i = 0; $i < 5; ++$i) {
03          if ($i == 2){
04              continue;
```

```
05            }
06            echo "$i\n";
07        }
08    ?>
```

运行结果如图 3.14 所示：

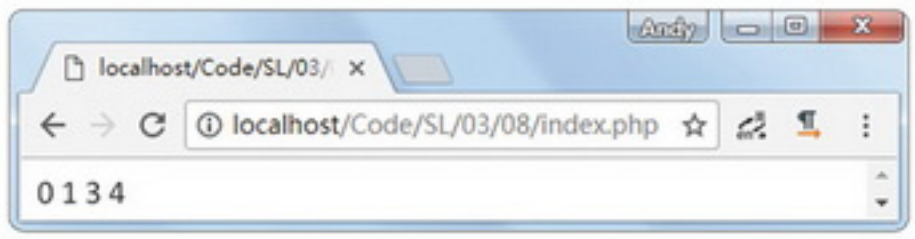


图 3.14　使用 continue 语句跳出循环

练一练：

（1）某剧院发售演出门票，演播厅观众席有 4 行，每行有 10 个座位。为了不影响观众视角，使用 continue 语句在发售门票时屏蔽掉最左一列和最右一列的座位。（光盘\Code\Try\03\15）

（2）某公司新建 4×4 个办公卡位，现只有第 1 排第 3 个和第 3 排第 2 个卡位被使用，使用 continue 语句输出尚未使用的新卡位。（光盘\Code\Try\03\16）

3.4 难点解答

3.4.1 if...else 执行顺序

当判断条件满足 if 条件又满足 else 条件时，程序该如何执行呢？程序运行时，会遵循由上至下的顺序。当遇到第一个满足的条件时，会选择第一个 if 条件，执行内部的代码块，跳过其余的代码块。

3.4.2 while 和 do...while 的区别

while 语句先判断循环条件，条件为真的时候，执行循环体，完成操作，一直循环，直到为 false 时，退出循环。

do...while 循环和 while 循环非常相似，区别在于表达式的值是在每次循环结束时检查而不是开始时。和一般的 while 循环主要的区别是 do...while 的循环语句保证会执行一次（表达式的真值在每次循环结束后检查），然而在一般的 while 循环中就不一定了（表达式真值在循环开始时检查，如果一开始就为 false，则整个循环立即终止）。

3.5 小　结

本章通过几个简单的数学题学习了 PHP 的流程控制语句。流程控制语句是程序中必不可少的，也

是变化最多样的技术。无论是入门的数学公式，还是高级的复杂算法，都是通过这几个简单的语句来实现的。相信读者学习完本章之后，通过不断地练习和总结，能够掌握一套自己的方法和技巧。

3.6 动手纠错

1．运行“光盘\Code\Debug\03\01”文件夹下的程序，出现“syntax error, unexpected '='”错误提示，如图 3.15 所示。请根据注释改正程序 "。

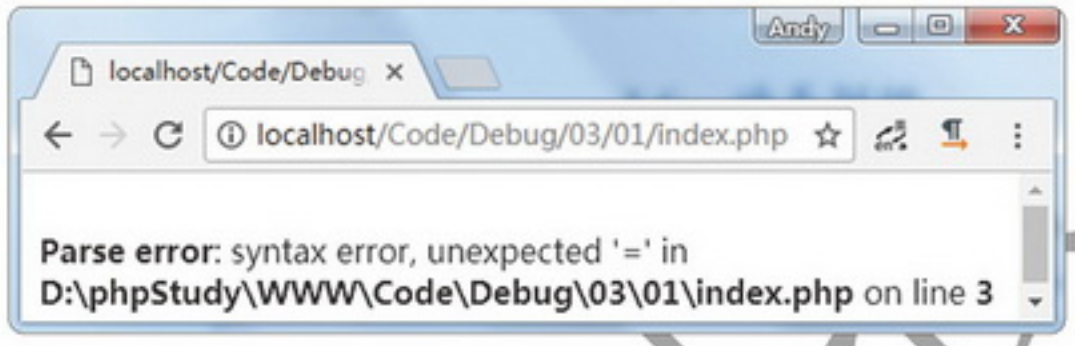


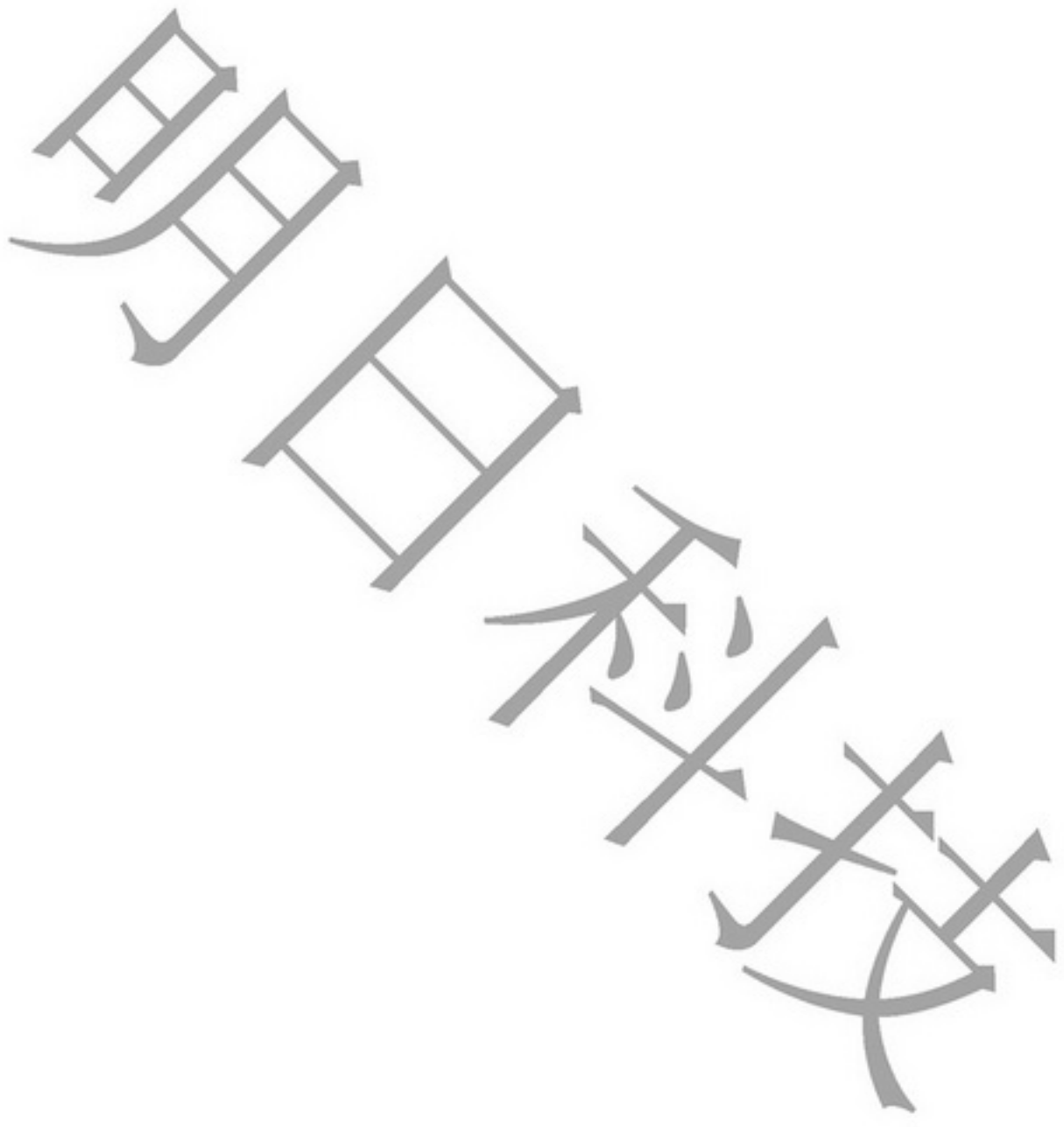
图 3.15　运行程序时出现的语法错误提示

2．运行“光盘\Code\Debug\03\02”文件夹下的程序，判断“等于”条件时出错，请根据注释改正程序。

3．运行“光盘\Code\Debug\03\03”文件夹下的程序，运行结果中输出数组数量不匹配，请根据注释改正程序。

4．运行“光盘\Code\Debug\03\04”文件夹下的程序，运行结果没有按预期退出程序，请根据注释改正程序。

5．运行“光盘\Code\Debug\03\05”文件夹下的程序，出现死循环，请根据注释改正程序。



第 4 章

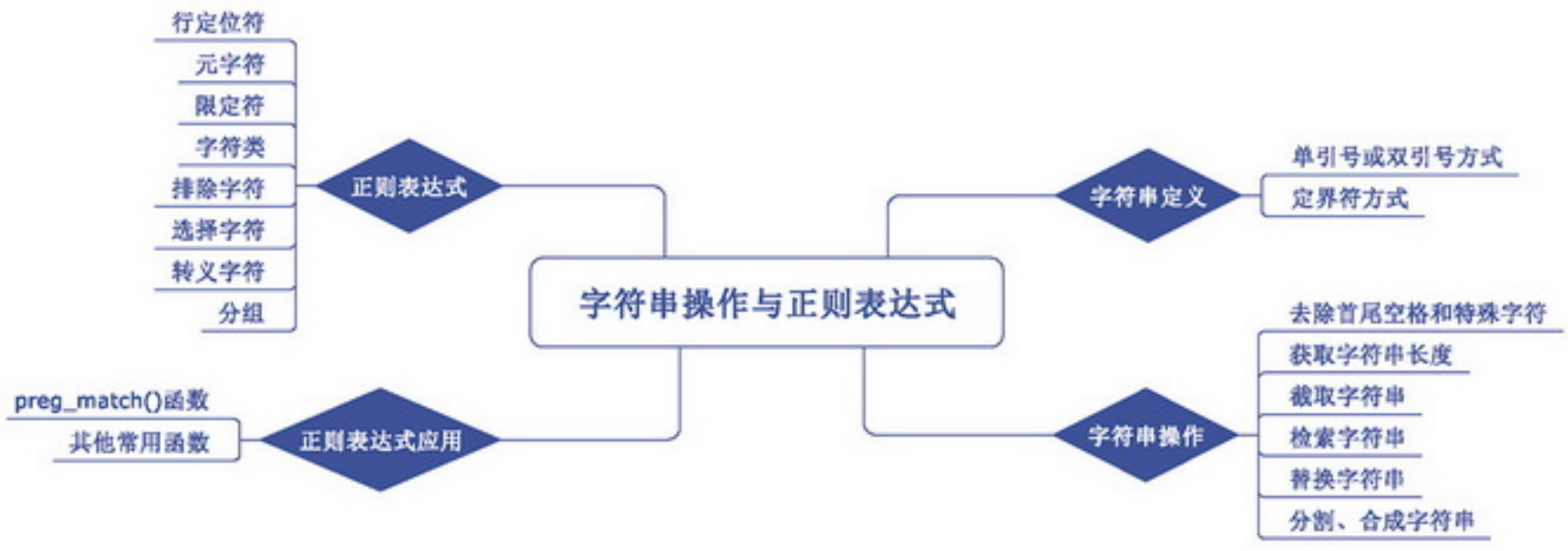
字符串操作与正则表达式

（视频讲解：2 小时 28 分）

本章概览

在 Web 编程中，字符串总是会被大量地生成和处理。正确地使用和处理字符串，对于 PHP 程序员来说越来越重要。本章从最简单的字符串定义一直引导读者到复杂的正则表达式，希望广大读者能够通过本章的学习，了解和掌握 PHP 字符串，达到举一反三的目的，为了解和学习其他的字符串处理技术奠定良好的基础。

知识框架



4.1 字符串的定义方法

字符串，顾名思义，就是将字符连在一起。字符串最简单的定义方法是使用英文单引号（''）或双引号（""）包含字符。另外，还可以使用定界符指定字符串。

4.1.1 使用单引号或双引号定义字符串

视频讲解：光盘\Video\04\4.1.1 使用单引号或双引号定义字符串.mp4

字符串通常整体作为操作对象，一般用双引号或者单引号标识一个字符串。单引号和双引号在使用上有一定区别。

下面分别使用双引号和单引号来定义一个普通字符串：

```
<?php
    $str1 = "I Like PHP";                    //使用双引号定义一个字符串
    $str2 = 'I Like PHP';                    //使用单引号定义一个字符串
    echo $str1;                              //输出双引号中的字符串
    echo $str2;                              //输出单引号中的字符串
?>
```

运行结果为：

```
I Like PHP
I Like PHP
```

从上面的结果中可以看出，对于定义的普通字符串看不出两者之间的区别。而通过对变量的处理，即可轻松地理解两者之间的区别：

```
<?php
    $test = "PHP";
    $str = "I Like $test";
    $str1 = 'I Like $test';
    echo $str;                               //输出双引号中的字符串
    echo $str1;                              //输出单引号中的字符串
?>
```

运行结果为：

```
I Like PHP
I Like $test
```

从以上代码中可以看出，双引号中的内容是经过 PHP 的语法分析器解析过的，任何变量在双引号中都会被转换为它的值进行输出显示；而单引号中的内容是“所见即所得”的，无论有无变量，都被当作普通字符串原样输出。

📖 **说明：**单引号定义的字符串和双引号定义的字符串在 PHP 中的处理是不相同的。双引号中的内容可以被解释并且被替换，而单引号中的内容则被作为普通字符进行处理。

4.1.2 使用定界符定义字符串

视频讲解：光盘\Video\04\4.1.2 使用定界符定义字符串.mp4

定界符（<<<）是从 PHP 4.0 开始支持的。定界符用于定义格式化的大文本，格式化是指文本中的格式将被保留，所以文本中不需要使用转义字符。在使用时后接一个标识符，然后是字符串，最后是同样的标识符结束字符串。定界符的格式如下：

```
$string = <<< str
        要输出的字符串。
str
```

其中 str 为指定的标识符，标识符读者可以自己设定，切记要前后保持一致。

使用定界符和双引号没什么区别，包含的变量也被替换成实际数值，代码如下：

```
01  <?php
02  $i = '显示该行内容';                          //声明变量$i
03  echo <<<EOT
04  这和双引号没有什么区别，\$i同样可以被输出出来。<p>
05  \$i的内容为：$i
06  EOT;
07  ?>
```

运行结果如下：

```
这和双引号没有什么区别，$i同样可以被输出出来。
$i的内容为：显示该行内容
```

⚡ **注意：**结束标识符必须单独另起一行，并且不允许有空格。在标识符前后有其他符号或字符也会发生错误。

4.2 字符串操作

字符串的操作在 PHP 编程中占有重要的地位，几乎所有的输入与输出都会用到字符串。尤其是在 PHP 项目开发过程中，为了实现某项功能，经常需要对某些字符串进行特殊处理，如获取字符串的长度、截取字符串、替换字符串等。在本节中将对 PHP 常用的字符串操作技术进行详细讲解，并通过具体的实例加深对字符串函数的理解。

4.2.1 去除字符串首尾空格和特殊字符

视频讲解：光盘\Video\04\4.2.1 去除字符串首尾空格和特殊字符.mp4

用户在输入数据时，可能会无意中输入多余的空格，在一些情况下，字符串前后不允许出现空格和特殊字符，此时就需要去除字符串中的空格和特殊字符。如图 4.1 所示的“HELLO”这个字符串前后都有一个空格。可以使用 PHP 中提供的 trim() 函数去除字符串左右两边的空格和特殊字符，也可以使用 ltrim() 函数去除字符串左边的空格和特殊字符，或使用 rtrim() 函数去除字符串中右边的空格和特殊字符。

图 4.1 前后包含空格的字符串

1. trim() 函数

trim() 函数用于去除字符串首尾处的空白字符（或者其他字符）。语法格式如下：

```
string trim(string $str [,string $charlist]);
```

trim() 函数的参数说明如下：

◆ str：操作的字符串。

◆ charlist：为可选参数，一般要列出所有希望过滤的字符，也可以使用“..”列出一个字符范围。如果不设置该参数，则所有的可选字符都将被删除。如果不指定 charlist 参数，trim() 函数将去除如表 4.1 所示的字符。

◆ 返回值：过滤后的字符串。

表 4.1 不指定 charlist 参数 trim() 函数去除的字符

字　符	说　明
\0	NULL，空值
\t	tab，制表符
\n	换行符
\x0B	垂直制表符
\r	回车符
" "	空格

注意：除了以上默认的过滤字符列表外，也可以在 charlist 参数中提供要过滤的特殊字符。

实例 01　去除搜索框中字符串左右两边的空格

实例位置：光盘\Code\SL\04\01

视频位置：光盘\Video\04\

明日学院网站中有搜索课程和社区的功能，当在输入框中输入关键词并单击“搜索”按钮时，程序会先处理用户输入的关键词，将其左右两边的空格去除。可以使用 trim() 函数实现该功能。具体代码如下：

实例01-1

```php
<?php
    $keyword = '  PHP开发  ';
    echo "用户输入的关键字是：".$keyword;
    $keyword = trim($keyword);
    echo "<br>";
    echo "使用trim函数处理后关键字是：".$keyword;
?>
```

运行结果如图 4.2 所示。

图 4.2　使用 trim() 函数去除字符串左右两边的空格

练一练：

（1）模拟用户注册过程，用户输入账户名时，忽略所有空格；用户输入密码时，不忽略任何字符。用户注册完之后，输出用户数据。（光盘\Code\Try\04\01）

（2）使用 trim() 函数删除任意代码中的所有缩进格式。（光盘\Code\Try\04\02）

2. ltrim() 函数

ltrim() 函数用于去除字符串左边的空格或者指定字符串。ltrim() 函数参数与 trim() 函数相同。语法格式如下：

```
string ltrim( string $str [,string $charlist]);
```

例如，使用 ltrim() 函数去除字符串左边的空格及特殊字符“(:@_@”，代码如下：

```php
<?php
    $str="  (:@_@  创图书编撰伟业  @_@:)   ";
    echo ltrim($str);                          //去除字符串左边的空格
    echo "<br>";                               //执行换行
    echo ltrim($str," (:@_@ ");                //去除字符串左边的特殊字符(:@_@
?>
```

运行结果为：

```
(:@_@ 创图书编撰伟业 @_@:)
创图书编撰伟业 @_@:)
```

3. rtrim() 函数

rtrim() 函数用于去除字符串右边的空格或指定字符。语法格式如下：

```
string rtrim(string $str [,string $charlist]);
```

例如，使用 rtrim() 函数去除字符串右边的空格及特殊字符“@_@:)”，代码如下：

```
<?php
    $str="  (:@_@  展软件开发雄风  @_@:)    ";
    echo rtrim($str);                              //去除字符串右边的空格
    echo "<br>";                                   //执行换行
    echo rtrim($str," @_@:)");                     //去除字符串右边的特殊字符@_@:)
?>
```

结果为：

```
(:@_@展软件开发雄风 @_@:)
(:@_@展软件开发雄风
```

4.2.2 获取字符串的长度

视频讲解：光盘\Video\04\4.2.2 获取字符串的长度.mp4

在 PHP 中，常见的计算字符串长度的函数有：strlen() 和 mb_strlen()。当字符全是英文字符的时候，两者功能是一样的。但是，当字符串中包含中文时，所占字节就会不同。先来了解一下英文和中文所占字节情况。

数字、英文、小数点、下划线和空格占一个字节，一个汉字可能会占 2~4 个字节，具体占几个字节取决于采用的是什么编码。汉字在 GBK/GB2312 编码中占 2 个字节，在 UTF-8/unicode 中一般占用 3 个字节 (或 2 ～ 4 字节)。由于本书中所有文件均使用 UTF-8 编码，即一个汉字占 3 个字节，如图 4.3 所示。

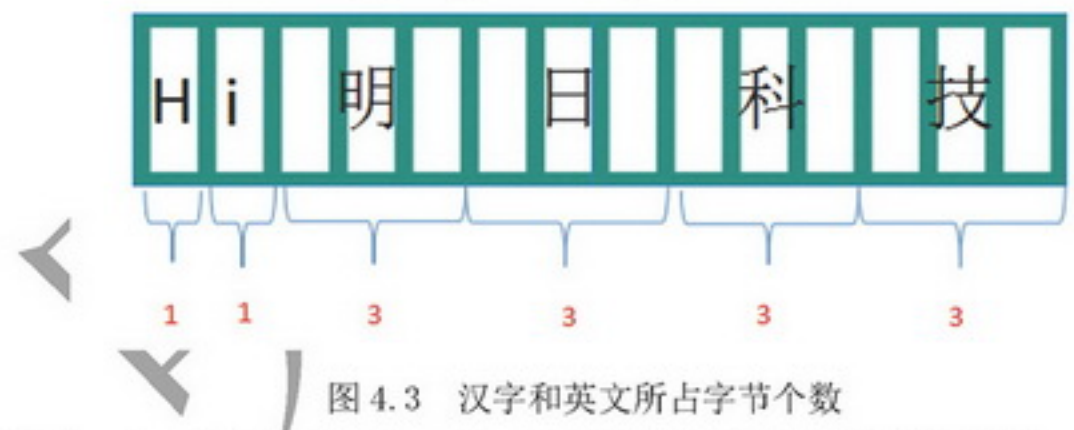


图 4.3　汉字和英文所占字节个数

下面讲解如何使用 strlen() 函数和 mb_strlen() 函数获取指定字符串的长度。

1. strlen() 函数

strlen() 函数主要用于获取指定字符串的长度。语法格式如下：

```
int strlen(string $str)
```

参数和返回值如下：

◆ str：需要计算长度的字符串。

◆ 返回值：成功则返回字符串 str 的长度。如果 str 为空，则返回 0。

例如，使用 strlen() 函数来获取指定字符串的长度，代码如下：

```php
<?php
    $str = "明日学院官方网站：www.mingrisoft.com";
    echo "字符串长度为：".strlen($str);
?>
```

在上述代码中“明日学院官方网站：”均为中文字符，每个占 3 字节，共占用 27 字节。“www.mingrisoft.com”均为英文字符，每个占 1 字节，共占用 18 字节。运行结果如下：

```
字符串长度为：45
```

2. mb_strlen() 函数

由于 strlen() 函数无法正确处理中文字符串，它得到的只是字符串所占的字节数，采用 mb_strlen() 函数较好地解决这个问题。

mb_strlen() 函数主要用于获取指定字符串的长度。语法格式如下：

```
mixed mb_strlen ( string $str , string $encoding = mb_internal_encoding()  )
```

参数和返回值如下：

◆ str：需要计算长度的字符串。

◆ encoding：字符编码。如果省略，则使用内部字符编码。

◆ 返回值：返回具有 encoding 编码的字符串 $str 包含的字符数。多字节的字符被计为 1。如果给定的 encoding 无效则返回 false。

mb_strlen() 函数的用法和 strlen() 函数类似，只不过它有第二个可选参数用于指定字符编码。例如，得到 UTF-8 的字符串 $str 长度，可以用 mb_strlen($str,'UTF-8')。如果省略第二个参数，则会使用 PHP 的内部编码。内部编码可以通过 mb_internal_encoding() 函数得到。

注意： mb_strlen() 函数并不是 PHP 核心函数，使用前需要确保在 php.ini 中加载了 php_mbstring.dll，即确保“extension=php_mbstring.dll”这一行存在并且没有被注释掉，否则会出现未定义函数的问题。

实例 02　判断注册的用户名是否为 3 ~ 18 位

实例位置：光盘\Code\SL\04\02

视频位置：光盘\Video\04\

明日学院注册页面中，用户注册时输入的用户名必须为 3~18 位中英文字符，既可以是全中文，也可以全英文或者中英文混合。可以使用 mr_strlen() 函数实现该功能，代码如下：

实例02-1

```php
<?php
    /** 定义checkUsername()函数 **/
    function checkUsername($username){
        $length = mb_strlen($username,'UTF-8'); //使用mb_strlen()函数获取字符串长度
        //判断字符串长度是否满足3~18
        if($length < 3 or $length > 18){
            $message = "不满足注册条件，用户名应该为3~18位";
        }else{
            $message = "满足注册条件，可以注册";
```

```
10          }
11          return $message;
12      }
13      $username1 = '明日';                          //定义变量
14      $username2 = '明日MR';                        //定义变量
15      $result1 =  checkUsername($username1);       //调用checkUsername()函数，传递$username1
16      $result2 =  checkUsername($username2);       //调用checkUsername()函数，传递$username2
17      echo '$username1'.$result1;                  //输出结果
18      echo "<br>";
19      echo '$username2'.$result2;                  //输出结果
20  ?>
```

运行结果如图 4.4 所示。

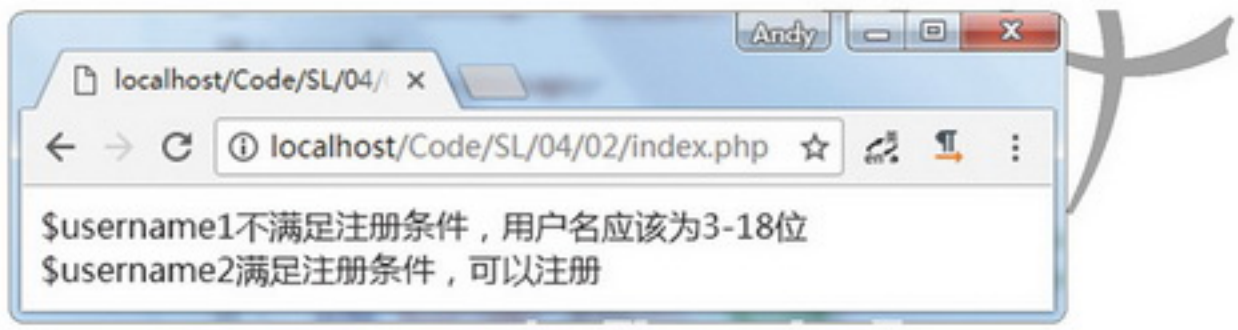


图 4.4　判断用户名是否满足条件

练一练：

（1）分别使用 strlen() 函数和 mb_strlen() 函数统计字符串“你好 ABC”的长度，并对比两个函数的差别。（光盘\Code\Try\04\03）

（2）模拟用户注册过程，用户名不能超过 10 位，如果超过 10 位，提示错误信息。（光盘\Code\Try\04\04）

4.2.3 截取字符串

视频讲解：光盘\Video\04\4.2.3 截取字符串.mp4

PHP 对字符串截取可以采用内置函数 substr() 和 mb_substr() 实现。通常使用 substr() 函数截取英文字符，mb_substr() 函数截取中文或中英文混合字符。

1. substr() 函数

语法格式如下：

```
string substr ( string $str, int $start [, int $length])
```

◆ str：指定字符串对象。

◆ start：指定开始截取字符串的位置。如果参数 start 为负数，则从字符串的末尾开始截取。

◆ length：可选参数，指定截取字符的个数，如果 length 为负数，则表示取到倒数第 length 个字符。

返回值：返回提取的子字符串，或者在失败时返回 false。

注意：在 substr() 函数中，参数 start 的指定位置是从 0 开始计算的，即字符串中的第一个字符表示为 0。如图 4.5 所示。

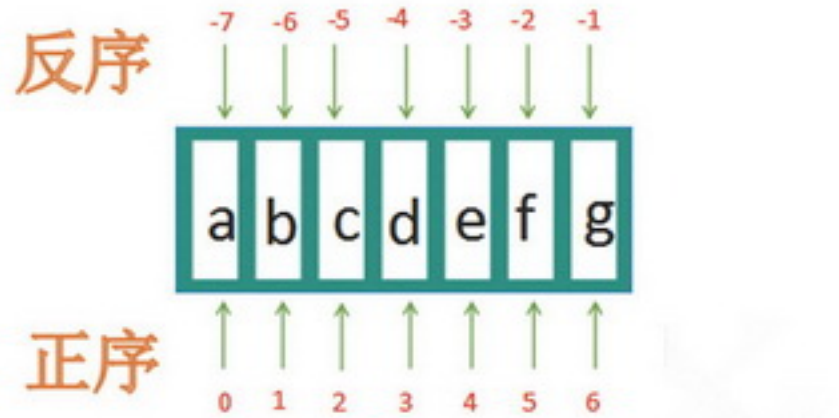


图 4.5　参数 start 的开始位置

使用 substr() 函数截取字符串中指定长度的字符，代码如下：

```
<?php
    $str = "She is a well-read girl";
    echo substr($str,0);                    //从第1个字符开始截取
    echo "<br>";                            //执行换行
    echo substr($str,4,14);                 //从第5个字符开始连续截取14个字符
    echo "<br>";                            //执行换行
    echo substr($str,-4,4);                 //从倒数第4个开始截取4个字符
    echo "<br>";                            //执行换行
    echo substr($str,0,-4);                 //从第一个字符开始截取，到倒数第4个字符为止
?>
```

运行结果如下：

```
She is a well-read girl
is a well-read
girl
She is a well-read
```

由于在 UTF-8 编码下，一个汉字占 3 个字节，所以在使用 substr() 函数时，可能出现截取汉字不完整的情况。例如，使用 substr() 函数截取字符串“Hi 明日科技”，代码如下：

```
<?php
    $string  = "Hi明日科技";
    echo substr($string,0,7);
?>
```

上述代码中，start 为 0，length 为 7，即从第一个位置开始，截取 7 个字节，如图 4.6 所示。由于在第 7 个字符位置的汉字“日”没有被截取完成，将会出现汉字乱码，运行结果如图 4.7 所示。

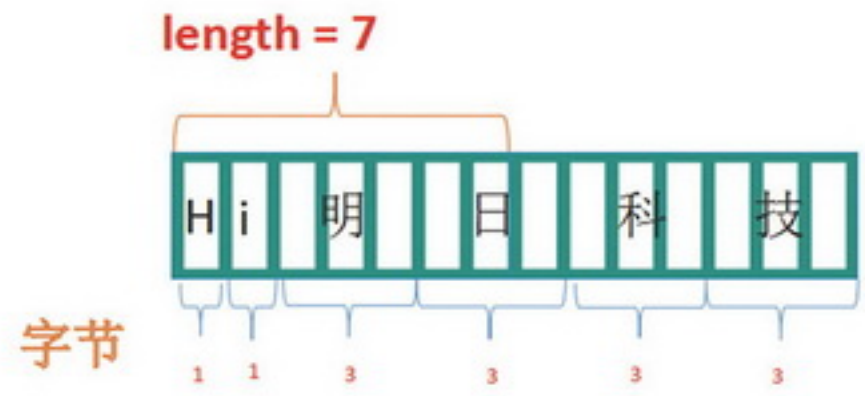


图 4.6　使用 substr() 函数截取字符串

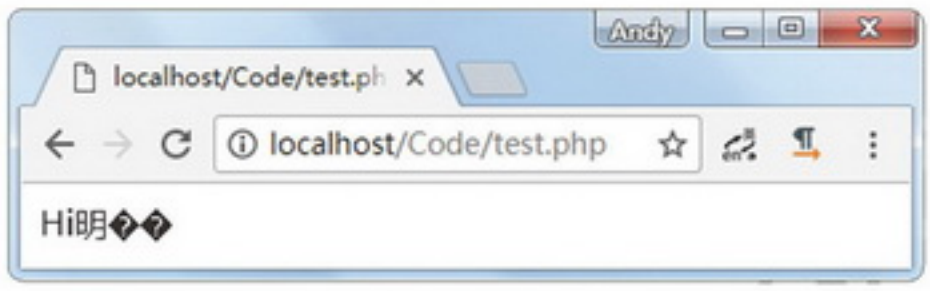


图 4.7　substr() 函数截取字符串出现汉字乱码

2. mb_substr () 函数

针对 substr() 函数截取字符串时出现汉字乱码问题，可以使用 mb_substr() 函数来解决。mb_substr() 函数语法格式如下：

```
string mb_substr ( string $str , int $start [, int $length = NULL [, string $encoding = mb_
internal_encoding() ]] )
```

substr() 函数参数如下：

- ◆ str：从该 string 中提取子字符串。
- ◆ start：str 中要截取的第一个字符的位置。
- ◆ length：可选参数，指定截取字符的个数，如果 length 为负数，则表示取到倒数第 length 个字符。
- ◆ encoding：字符编码。如果省略，则使用内部字符编码。

返回值：根据参数 start 和 length 返回 str 中指定的部分。

实例 03　截取列表页中过长的标题

实例位置：光盘\Code\SL\04\03
视频位置：光盘\Video\04\

在明日学院网站“最新动态”专栏中，可以显示所有最新课程标题的列表。为了保持整个页面的合理布局，需要对一些超长标题进行部分显示。可以使用 substr() 函数截取超长文本的部分字符串，剩余的部分用“…”代替。具体代码如下：

实例03-1

```
<?php
    /** 列表页内容 **/
    $row1 = "11.5 继承泛型类与实现泛型接口（下）";
    $row2 = "11.4 继承泛型类与实现泛型接口（上）";
    $row3 = "11.3 限制泛型：泛型通配符";
    $row4 = "11.2 限制泛型：泛型继承类和接口";
```

```
07      $row5 = "11.1 泛型类";
08      $row6 = "10.7 枚举实现接口";
09      $row7 = "10.6 枚举的类成员";
10      /** 定义字符串截取函数 **/
11      function getSubstr($string){
12          if(mb_strlen($string,"UTF-8")>15){              //如果文本的字符串长度大于15个字符
13              echo mb_substr($string,0,15,"UTF-8")."…";   //输出文本的前15个字符，然后输出"…"
14          }else{
15              echo $string;                               //直接输出文本
16          }
17          echo "<br>";
18      }
19      /** 调用getSubstr()函数，输出截取后的结果 **/
20      getSubstr($row1);
21      getSubstr($row2);
22      getSubstr($row3);
23      getSubstr($row4);
24      getSubstr($row5);
25      getSubstr($row6);
26      getSubstr($row7);
27  ?>
```

运行结果如图 4.8 所示。

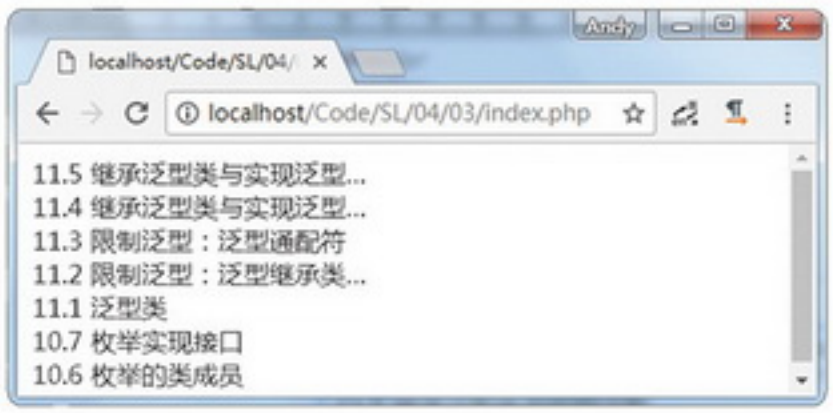


图 4.8　使用 mb_substr() 函数截取字符串

练一练：

（1）模拟淘宝详情页，当商品名称多于 30 字时，截取前 20 个字符，剩余用“...”代替。（光盘\Code\Try\04\05）

（2）使用 mb_substr() 函数截取任意 QQ 邮箱地址中的 QQ 号。（光盘\Code\Try\04\06）

4.2.4 检索字符串

视频讲解：光盘\Video\04\4.2.4 检索字符串.mp4

在 PHP 中，提供了很多用于检索字符串的函数，最常用的有 strstr() 函数和 strpos() 函数。

1. strstr() 函数

检索一个指定字符串在另一个字符串中首次出现的位置到后者末尾的子字符串。语法格式如下：

```
string strstr ( string $haystack , mixed $needle [, bool $before_needle = false ] )
```

参数如下：

◆ haystack：指定从该字符串中进行搜索。

◆ needle：指定搜索的对象。如果 needle 不是一个字符串，那么它将被转化为整型并且作为字符的序号来使用。

◆ before_needle：可选参数，默认为 false。若为 true，strstr() 函数将返回 needle 在 haystack 中的位置之前的部分。

返回值：返回 haystack 字符串从 needle 第一次出现的位置开始到 haystack 结尾的字符串。

例如，检索“Hi 明日科技”字符串中“明日”以后的内容。代码如下：

```
<?php
    $string  = "Hi明日科技";
    echo strstr($string,"明日");
?>
```

strstr() 函数实现方式如图 4.9 所示，运行结果如下：

```
明日科技
```

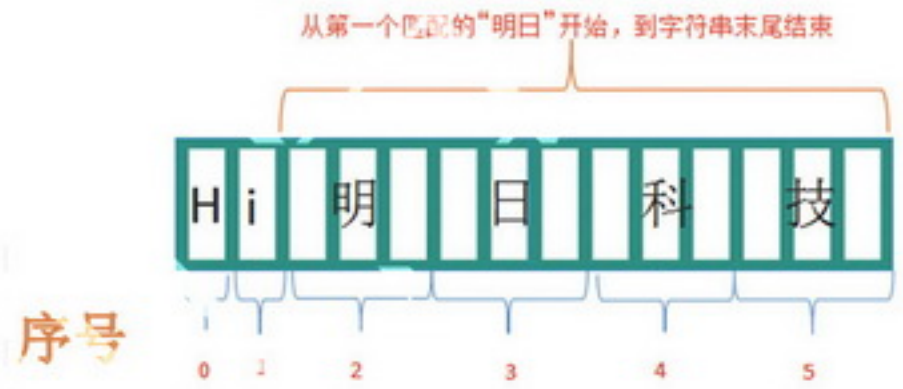


图 4.9　strstr() 函数实现方式

注意：strstr() 函数区分字母的大小写，如不区分大小写，可以使用 stristr() 函数。

实例 04　使用 strstr() 函数获取用户名和服务器名

实例位置：光盘\Code\SL\04\04
视频位置：光盘\Video\04\

本实例将使用 strstr() 函数，根据邮箱地址获取邮箱用户名和服务器名。代码如下：

实例04-1

```
<?php
    $email  = 'mingrisoft@163.com';
    $domain = strstr($email, '@');
    echo '邮箱服务器是:'.$domain. '<br>';          //输出: @163.com
    $user = strstr($email, '@', true);            //使用strstr()函数
```

```
06        echo '用户名 是:'.$user;                    //输出: mingrisoft
07    ?>
```

运行结果如图 4.10 所示。

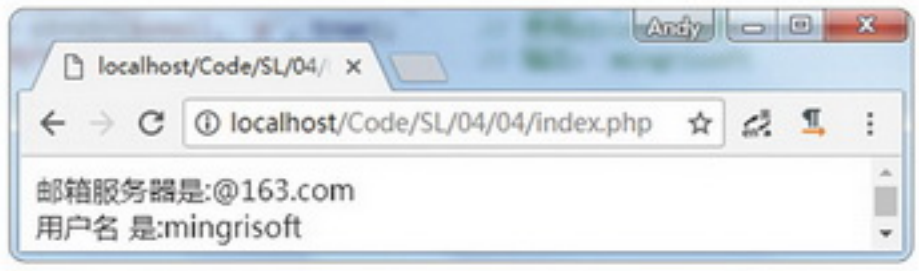


图 4.10　使用 strstr() 函数获取用户名和服务器名

📖 **说明：** strrchr() 函数与其正好相反，该函数是从字符串倒序的位置开始检索子字符串的。

练一练：

（1）将“津 A · 12345”“沪 A · 23456”“京 A · 34567”这三张号牌放到数组中，然后在遍历数组的过程中完成对每张号牌归属地的判断。（光盘\Code\Try\04\07）

（2）明日学院登录页面的用户名输入框中，允许用户输入手机号或邮箱登录，使用 strstr() 函数判断用户输入的是邮箱还是手机号。（光盘\Code\Try\04\08）

2. strpos() 函数

strpos() 函数可以检索字符串首次出现的位置，返回首次出现的数字位置。语法格式如下：

```
mixed strpos ( string $haystack , mixed $needle [, int $offset = 0 ] )
```

参数如下：

◆ haystack：必要参数，指定从该字符串中进行搜索。

◆ needle：必要参数，指定搜索的对象。如果 needle 不是一个字符串，那么它将被转化为整型并且作为字符的序号来使用。

◆ offset：可选参数，默认为 0。如果提供了此参数，搜索会从字符串中该字符数的起始位置开始统计。

返回值：返回 needle 在 haystack 字符串中起始的位置。同时注意字符串位置是从 0 开始，而不是从 1 开始的。如果没找到 needle，将返回 false。

注意： strpos() 函数区分字母的大小写，如不区分大小写，可以使用 stripos() 函数。

例如，检索“Hi 明日科技”字符串中“明日”以后的内容。代码如下：

```
01  <?php
02      $string  = "Hi明日科技";
03      echo strpos($string,"明日");
04  ?>
```

strpos() 函数实现方式如图 4.11 所示，运行结果如下：

```
明日科技
```

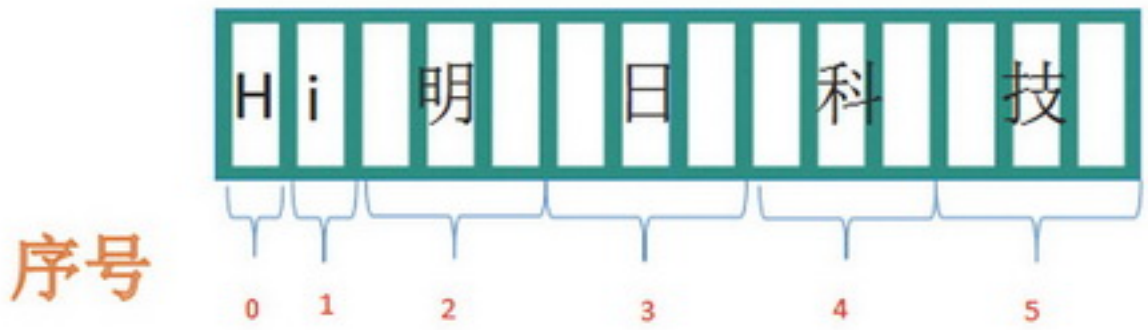


图 4.11　strpos() 函数实现方式

📖 **说明：** strrpos() 函数与其正好相反，该函数是计算指定字符串在目标字符串中最后一次出现的位置。strrpos() 函数也区分大小写，如不区分大小写，可以使用 strripos() 函数。

4.2.5 替换字符串

视频讲解：光盘\Video\04\4.2.5 替换字符串.mp4

通过字符串的替换技术可以实现对指定字符串中的指定字符进行替换。字符串的替换技术可以通过以下两个函数实现：str_replace() 函数和 substr_replace() 函数。

1. str_replace() 函数

str_replace() 函数可以使用新的子字符串，替换原始字符串中被指定要替换的字符串。语法格式如下：

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

将所有在参数 subject 中出现的参数 search 以参数 replace 取代，参数 &$count 表示取代字符串执行的次数。本函数区分大小写。

- ◆ search：必要参数，要搜索的值，可以使用 array 来提供多个值。
- ◆ replace：必要参数，指定替换的值。
- ◆ subject：必要参数，要被搜索和替换的字符串或数组。
- ◆ count：可选参数，如果被指定，它的值将被设置为替换发生的次数。
- ◆ 返回值：替换后的字符串或者数组。

例如，将文本中的指定字符串“某某”替换为“**”，并且输出替换后的结果，代码如下：

```
<?php
    $str2="某某";                                          //定义字符串常量
    $str1="**";                                            //定义字符串常量
    $str="某某公司是一家以计算机软件技术为核心的高科技企业，涉及生产、管理、控制、仓储、物流、
          营销、服务等行业";                               //定义字符串常量
    echo str_replace($str2,$str1,$str,$count);             //输出替换后的字符串
    echo "<br>";
    echo "替换数量：".$count."个";
?>
```

运行结果如下：

```
**公司是一家以计算机软件技术为核心的高科技企业，涉及生产、管理、控制、仓储、物流、营销、服务等行业
替换数量：2个
```

注意：str_replace() 函数在执行替换操作时区分大小写，如果不需要对大小写加以区分，可以使用 str_ireplace() 函数。

2. substr_replace() 函数

substr_replace() 函数可对指定字符串中的部分字符串进行替换。语法格式如下：

```
mixed substr_replace ( mixed $string , mixed $replacement , mixed $start [, mixed $length ] )
```

◆ string：指定要操作的原始字符串，可以是字符串或数组。

◆ replacement：指定替换后的新字符串。

◆ start：指定替换字符串开始的位置。正数表示替换从字符串的第 start 位置开始；负数表示替换从字符串的倒数第 start 位置开始；0 表示替换从字符串中的第一个字符开始。

◆ length：可选参数，指定返回的字符串长度。默认值是整个字符串。正数表示被替换的子字符串的长度；负数表示待替换的子字符串结尾处距离字符串末端的字符个数；0 表示将 replacement 插入到 string 的 start 位置处。

◆返回值：返回结果字符串。如果 string 是个数组，那么也将返回一个数组。

注意：如果参数 start 设置为负数，而参数 length 数值小于或等于 start 数值，那么 length 的值自动为 0。

实例 05　将手机号中间 4 位数字用“**”替换**

实例位置：光盘\Code\SL\04\05

视频位置：光盘\Video\04\

明日学院网站举办抽奖活动，活动结束后将获奖用户姓名和手机号公布在网站上，为保护用户隐私，将获奖用户的手机号中间 4 位用“****”替换，实现该功能的代码如下：

实例05-1

```
<?php
    /** 定义3个用户 **/
    $username1 = '张三';
    $username1_phone = '40084978981';
    $username2 = '李四';
    $username2_phone = '40084988981';
    $username3 = '王五';
    $username3_phone = '40084998981';
    $replace = '****';                          //替换的字符串
    /** 输出替换后的结果 **/
    echo '姓名：'.$username1." 手机号：".substr_replace($username1_phone,$replace,3,4);
    echo "<br>";
    echo '姓名：'.$username2." 手机号：".substr_replace($username2_phone,$replace,3,4);
    echo "<br>";
    echo '姓名：'.$username3." 手机号：".substr_replace($username3_phone,$replace,3,4);
?>
```

运行结果如图 4.12 所示。

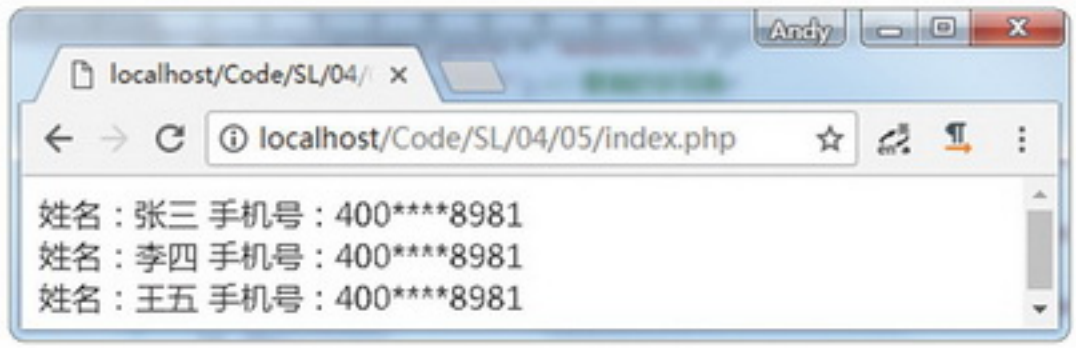


图 4.12　substr_replace() 函数替换手机号

练一练：

（1）使用 str_replace() 函数替换查询关键字，当显示所查询的相关信息时，将输出的关键字的字体替换为红色。（光盘\Code\Try\04\09）

（2）模拟 Word 软件的替换功能，将“mr”全部替换成“mrsoft”。（光盘\Code\Try\04\10）

4.2.6 分割、合成字符串

视频讲解：光盘\Video\04\4.2.6 分割、合成字符串.mp4

在 PHP 中，提供了分割和合成字符串的函数，它们都与数组相关。数组就是一组数据的集合，把一系列数据组织起来，形成一个可操作的整体。数组的知识会在第 5 章讲解，现在先来了解一下如何分割和合成字符串。

1. 分割字符串

explode() 函数按照指定的规则对一个字符串进行分割，返回值为数组。语法格式如下：

```
array explode ( string $delimiter , string $string [, int $limit ] )
```

参数及返回值如下：

◆ delimiter：边界上的分隔字符。

◆ string：指定将要被进行分割的字符串。

◆ limit：可选参数，如果设置了 limit，则返回的数组包含最多 limit 个元素，而最后的元素将包含 $string 的剩余部分。

◆ 返回值：此函数返回由字符串组成的 array，每个元素都是 string 的一个子串，它们被字符串 delimiter 作为边界点分割出来。

实例 06　输出被 @ 的好友名称

实例位置：光盘\Code\SL\04\06

视频位置：光盘\Video\04\

在微博 @ 好友时，输入“@mr @mrsoft @ 明日科技”（好友名称之间用一个空格区分），即可同时 @ 三个好友，使用 explode() 函数，输出被 @ 的好友名称。实现该功能的代码如下：

实例06-1

```
<?php
    $string = "@mr @mrsoft @明日科技";                    //定义字符串
```

```
03      $array  = explode(' ',$string);                    //根据空格拆分字符串
04      echo "您@的好友有：<br>";
05      /** 遍历数组 **/
06      for($i=0;$i<3;$i++){
07          echo trim($array[$i],'@')."<br>";              //$i为数组下标
08      }
09  ?>
```

运行结果如图 4.13 所示。

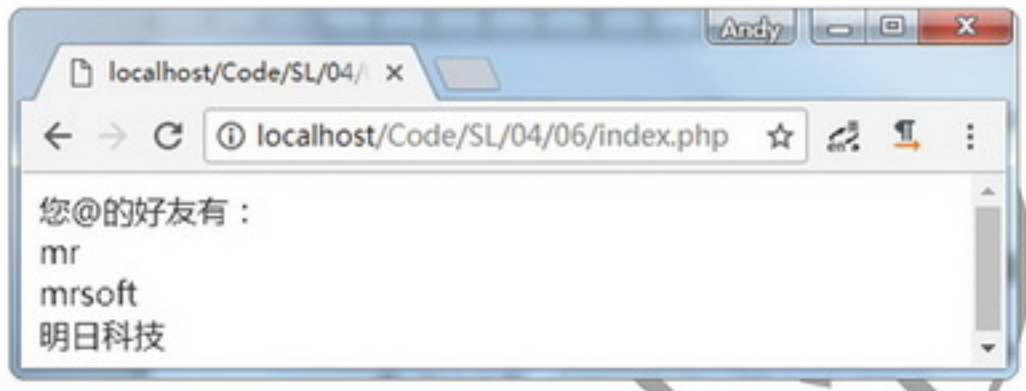


图 4.13　输出被 @ 的好友名称

注意：在默认情况下，数组第一个元素的索引为 0。关于数组的相关知识将在第 5 章中进行详细讲解。

练一练：

（1）试着结合 trim() 函数和 explode() 函数，从字符串“|1|2|3|4|”中，提取出“1,2,3,4”。（光盘\Code\Try\04\11）

（2）明日学院网站包含上传附件功能，文件格式字符串如下：

```
'zip,rar,doc,docx,zip,pdf,txt,ppt,pptx,xls,xlsx'
```

使用 explode() 函数将文件格式字符串拆分为数组。（光盘\Code\Try\04\12）

2. 合成字符串

implode() 函数可以将数组的内容组合成一个新字符串。语法格式如下：

```
string implode(string $glue, array $pieces)
```

参数及返回值如下：

- glue：指定分隔符。
- pieces：指定要被合并的数组。
- 返回值：返回一个字符串，其内容为由 glue 分割开的数组的值。

例如，使用 implode() 函数将数组中的内容以 @ 为分隔符进行连接，从而组合成一个新的字符串，代码如下：

```
01  <?php
02      $str="PHP编程词典@NET编程词典@ASP编程词典@JSP编程词典";     //定义字符串常量
```

```
03      $str_arr = explode("@",$str);                    //应用@分割字符串
04      $string = implode("@",$str_arr);                 //将数组合成字符串
05      echo $string;                                    //输出字符串
06  ?>
```

结果为：

```
PHP编程词典@NET编程词典@ASP编程词典@JSP编程词典
```

说明：implode() 函数和 explode() 函数是两个相对的函数，一个用于合成，一个用于分割。

4.3 正则表达式

4.3.1 正则表达式简介

视频讲解：光盘\Video\04\4.3.1 正则表达式简介.mp4

在编写处理字符串的程序或网页时，经常会有查找符合某些复杂规则的字符串的需要，正则表达式就是用于描述这些规则的工具。换句话说，正则表达式就是记录文本规则的代码。对于接触过 DOS 用户来说，如果想匹配当前文件夹下所有的文本文件，可以输入“dir *.txt”命令，按下 <Enter> 键后所有“.txt”文件将会被列出来。这里的“*.txt”即可理解为一个简单的正则表达式。

4.3.2 行定位符

视频讲解：光盘\Video\04\4.3.2 行定位符.mp4

行定位符就是用来描述字符串的边界。“^”表示行的开始；“$”表示行的结尾。如：

```
^tm
```

该表达式表示要匹配字符串 tm 的开始位置是行头，如 tm equal Tomorrow Moon 就可以匹配，而 Tomorrow Moon equal tm 则不匹配。但如果使用：

```
tm$
```

则后者可以匹配而前者不能匹配。如果要匹配的字符串可以出现在字符串的任意部分，那么可以直接写成：

```
tm
```

这样两个字符串就都可以匹配了。

4.3.3 元字符

视频讲解：光盘\Video\04\4.3.3 元字符.mp4

前面已经讲述了几个很有用的元字符，如“^”“$”。正则表达式里还有更多的元字符，例如下面的表达式：

```
\bmr\w*\b
```

匹配以字母“mr”开头的单词，先是从某个单词开始处 (\b)，然后匹配字母“mr”，接着是任意数量的字母或数字 (\w*)，最后单词结束处 (\b)。该表达式可以匹配“mrsoft”“mrbook”“mr123456”等。更多常用元字符如表 4.2 所示。

表 4.2　常用元字符

元　字　符	说　　明
.	匹配除换行符以外的任意字符
\w	匹配字母或数字或下划线或汉字
\s	匹配任意的空白符
\d	匹配数字
\b	匹配单词的开始或结束
^	匹配字符串的开始
$	匹配字符串的结束

4.3.4 限定符

视频讲解：光盘\Video\04\4.3.4 限定符.mp4

前面的表达式中使用 (\w*) 匹配任意数量的字母或数字。如果想匹配特定数量的数字，该如何表示呢？正则表达式提供了限定符（指定数量的字符）来实现该功能。如匹配 8 位 QQ 号可用如下表示式：

```
^\d{8}$
```

常用的限定符如表 4.3 所示。

表 4.3　常用限定符

限　定　符	说　　明	举　　例
?	匹配前面的字符零次或一次	colou?r，该表达式可以匹配 colour 和 color
+	匹配前面的字符一次或多次	go+gle，该表达式可以匹配的范围从 gogle 到 goo…gle
*	匹配前面的字符零次或多次	go*gle，该表达式可以匹配的范围从 ggle 到 goo…gle
{n}	匹配前面的字符 n 次	go{2}gle，该表达式只匹配 google

续表

限 定 符	说 明	举 例
{n,}	匹配前面的字符最少 n 次	go{2,}gle，该表达式可以匹配的范围从 google 到 goo…gle
{n,m}	匹配前面的字符最少 n 次，最多 m 次	employe{0,2}，该表达式可以匹配 employ、employe 和 employee 3 种情况

4.3.5 字符类

视频讲解：光盘\Video\04\4.3.5 字符类.mp4

正则表达式查找数字和字母是很简单的，因为已经有了对应这些字符集合的元字符（如 \d，\w），但是如果要匹配没有预定义元字符的字符集合（比如元音字母 a,e,i,o,u），应该怎么办？

很简单，只需要在方括号里列出它们就行了，像 [aeiou] 就匹配任何一个英文元音字母，[.?!] 匹配标点符号（. 或？或！）。也可以轻松地指定一个字符范围，像 [0-9] 代表的含义与 \d 就是完全一致的，都表示一位数字；同理 [a-z0-9A-Z_] 也完全等同于 \w（如果只考虑英文的话）。

4.3.6 排除字符

视频讲解：光盘\Video\04\4.3.6 排除字符.mp4

前面介绍了匹配符合命名规则的变量。要匹配不符合命名规则的变量，正则表达式提供了“^”字符。这个元字符在 4.3.1 小节中出现过，表示行的开始。而这里将会放到方括号中，表示排除的意思。例如：

```
[^a-zA-Z]
```

该表达式匹配的就是不以字母开头的变量名。

4.3.7 选择字符

视频讲解：光盘\Video\04\4.3.7 选择字符.mp4

试想一下，如何匹配身份证号码？首先需要了解一下身份证号码的规则。一代身份证号码长度为 15 位且全为数字；二代身份证为 18 位，前 17 位为数字，最后一位是校验位，可能为数字或字符 X。

在上面的描述中，包含着条件选择的逻辑，这就需要使用选择字符（|）来实现。该字符可以理解为“或”，匹配身份证的表达式可以写成如下格式：

```
^\d{15}$)|(^\d{18}$)|(^\d{17}(\d|X|x)$
```

该表达式的意思是以匹配 15 位数字，或者 18 位数字，或者 17 位数字和最后一位。最后一位可以是数字、X，或者是 x。

4.3.8 转义字符

视频讲解：光盘\Video\04\4.3.8 转义字符.mp4

正则表达式中的转义字符（\）和 PHP 中的大同小异，都是将特殊字符（如“.”“?”“\”等）变为普通的字符。以 IP 地址为例，用正则表达式匹配诸如 127.0.0.1 这样格式的 IP 地址，如果直接使用点字符，格式为：

```
[0-9]{1,3}(.[0-9]{1,3}){3}
```

这显然不对，因为“.”可以匹配一个任意字符。这时，不仅可以匹配 127.0.0.1 这样的 IP 地址，连 127101011 这样的字符串也会被匹配出来。所以在使用“.”时，需要使用转义字符（\）。修改后的正则表达式格式为：

```
[0-9]{1,3}(\.[0-9]{1,3}){3}
```

说明：括号在正则表达式中也是一个元字符。

4.3.9 分组

视频讲解：光盘\Video\04\4.3.9 分组.mp4

小括号字符的第一个作用就是可以改变限定符的作用范围，如“|”“*”“^”等。来看下面的一个表达式：

```
(six|four)th
```

这个表达式的意思是匹配单词 sixth 或 fourth，如果不使用小括号，那么就变成了匹配单词 sixth 和 fourth 了。

小括号的第二个作用是分组，也就是子表达式。如 (\.[0-9]{1,3}){3}，就是对 (\.[0-9]{1,3}) 进行重复操作。

4.4 正则表达式在 PHP 中的应用

视频讲解：光盘\Video\04\4.4. 正则表达式在PHP中的应用.mp4

PHP 中提供了两套支持正则表达式的函数库，PCRE 函数库和 POSIX 函数库。PCRE 函数库在执行效率上要略优于 POSIX 函数库，所以这里只讲解 PCRE 函数库中的函数。PCRE 函数库中常用函数如表 4.4 所示。

表 4.4　PCRE 函数库中常用函数

函　数	说　明
preg_filter	执行一个正则表达式搜索和替换

续表

函　数	说　明
preg_grep	返回匹配模式的数组条目
preg_last_error	返回最后一个 PCRE 正则表达式执行产生的错误代码
preg_match_all	执行一个全局正则表达式匹配
preg_match	执行匹配正则表达式
preg_quote	转义正则表达式字符
preg_replace_callback	执行一个正则表达式搜索并且使用一个回调进行替换
preg_replace	执行一个正则表达式的搜索和替换
preg_split	通过一个正则表达式分割字符串

下面讲解如何使用 PHP 中最常用的 preg_match() 函数。

preg_match() 函数用于执行匹配正则表达式，函数语法如下：

```
int preg_match ( string $pattern , string $subject [, array &$matches] )
```

pattern：要搜索的模式，字符串类型。

subject：输入字符串。

matches：可选参数，如果提供了参数 matches，它将被填充为搜索结果。$matches[0] 将包含完整模式匹配到的文本，$matches[1] 将包含第一个捕获子组匹配到的文本，依此类推。

返回值：返回 pattern 的匹配次数。它的值将是 0 次（不匹配）或 1 次，因为 preg_match() 函数在第一次匹配后将会停止搜索。如果发生错误则返回 false。

实例 07　使用 preg_match() 函数检测手机号码格式

实例位置：光盘\Code\SL\04\07

视频位置：光盘\Video\04\

在明日学院注册页面中，需要对用户输入的手机号码格式进行检测，以避免用户手误导致注册失败。使用 preg_match() 函数实现该功能，具体代码如下：

实例07-1

```
<?php
    $mobile1 = '12888888888';                          //手机号码1
    $mobile2 = '13578982158';                          //手机号码2
    /** 定义检测手机号码格式的函数 **/
    function checkMobile($mobile){
        if(preg_match('/1[34578]\d{9}$/',$mobile)){    //判断格式是否正确
            echo $mobile."手机号格式正确";              //输出正确的信息
        }else{
            echo $mobile."手机号格式错误";              //输出错误的信息
        }
```

```
11      }
12
13      checkMobile($mobile1);                    //调用检测方法
14      echo "<br>";
15      checkMobile($mobile2);                    //调用检测方法
16  ?>
```

运行结果如图 4.14 所示。

图 4.14　使用 preg_match() 函数检测手机号码格式

多学两招：preg_match_all() 函数用于执行一个全局正则表达式匹配，它会一直搜索 subject 直到结尾。

练一练：

（1）模拟明日学院登录页面，实现检测邮箱格式的功能。（光盘\Code\Try\04\13）

（2）定义一个判断用户密码强度的函数，如果用户的密码包含大小写字母和数字的组合，不能使用特殊字符，并且长度在 8~10 之间，判断该用户密码强度高，否则判断密码强度低。（光盘\Code\Try\04\14）

4.5 难点解答

4.5.1 慎用 strlen() 函数处理中文字符

在使用 strlen() 函数计算时，对待一个 UTF-8 的中文字符是 3 个长度，所以“中文 a 字 1 符”长度是 3*4+2=14，在 mb_strlen() 函数计算时，选定内码为 UTF-8，则会将一个中文字符当作长度 1 来计算，所以“中文 a 字 1 符”长度是 6。

4.5.2 strstr() 函数和 strpos() 函数的区别

两个函数都用于查找字符串首次出现的位置，并且都区分大小写。不同的是，strstr() 函数返回的是一个字符串，即从首次出现位置到输入的字符串结束；而 strpos() 函数返回的是一个数字，即字符串首次出现的数字位置，注意从 0 开始计数。

4.6 小　结

本章主要对常用的字符串操作技术进行了详细的讲解，其中去除字符串首尾空格、获取字符串的长度、截取字符串和字符串的查找与替换等都是需要重点掌握的技术。此外，还介绍了正则表达式的基础知识。这些内容也是作为一个 PHP 程序员必须熟悉和掌握的知识。相信通过本章的学习，读者能够举一反三，对所学知识灵活运用，从而开发实用的 PHP 程序。

4.7 动手纠错

1．运行“光盘\Code\Debug\04\01”文件夹下的程序，使用 explode() 函数拆分的字符串包含空格如图 4.15 所示。请根据注释改正程序。

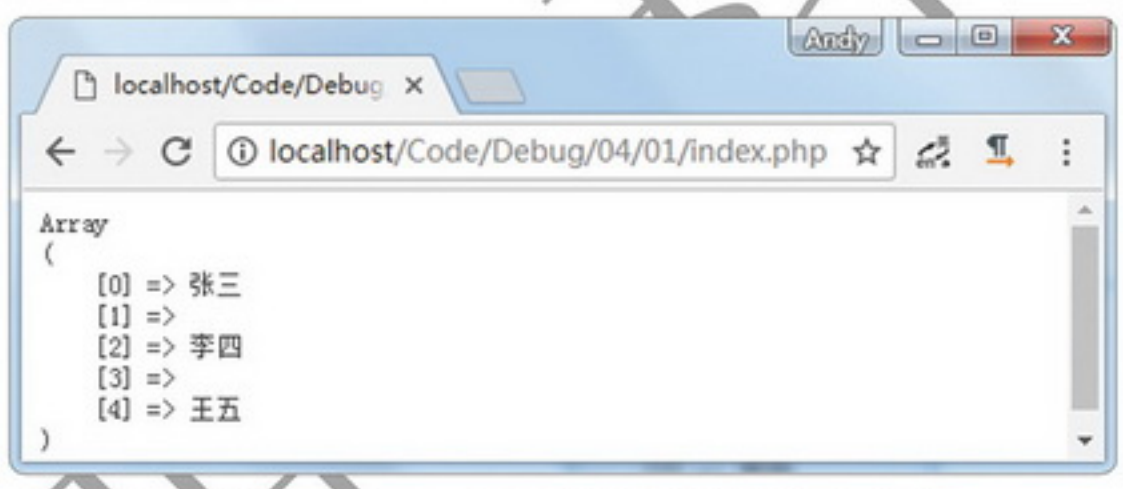


图 4.15　explode() 函数拆分的字符串包含空格

2．运行“光盘\Code\Debug\04\02”文件夹下的程序，输出警告信息，如图 4.16 所示。请根据注释改正程序。

图 4.16　警告信息

3．运行“光盘\Code\Debug\04\03”文件夹下的程序，当用户名为“mr”时，应输出“用户名不满足条件”，但是运行结果却如图 4.17 所示，请根据注释改正程序。

图 4.17　输出错误信息

4．运行“光盘\Code\Debug\04\04”文件夹下的程序，出现中文乱码，如图 4.18 所示。请根据注释改正程序。

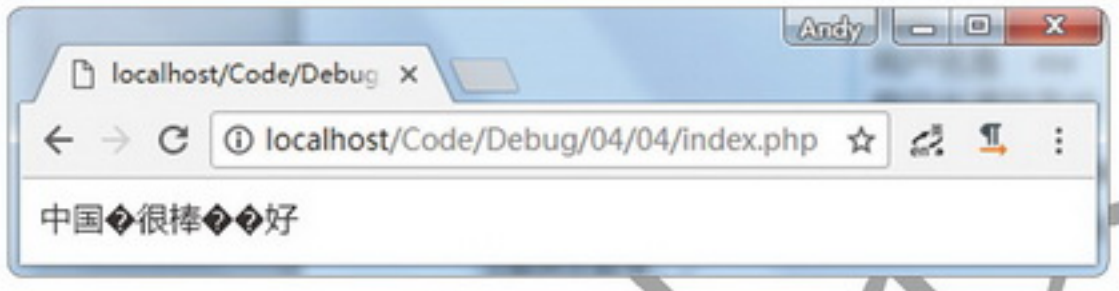
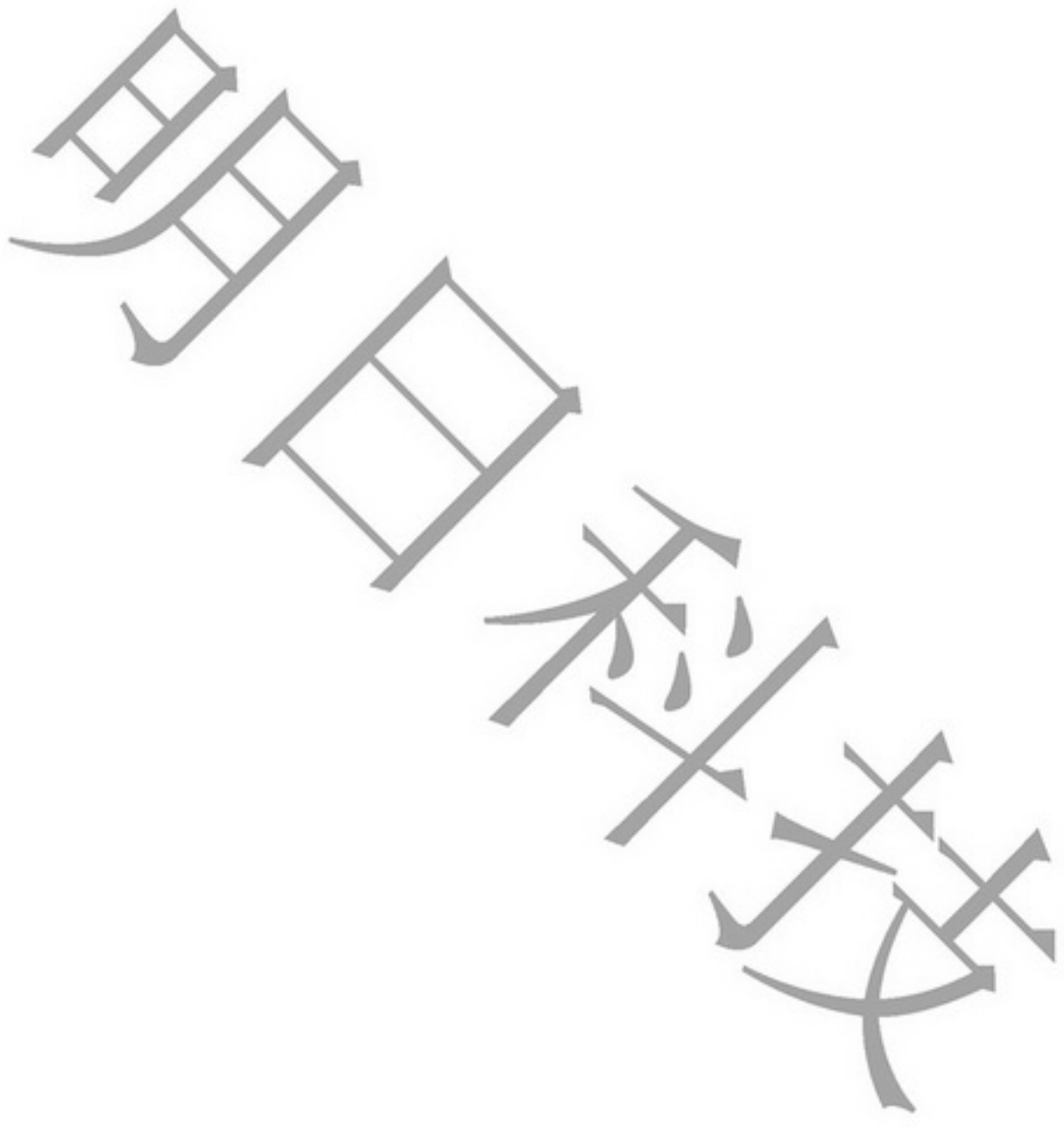


图 4.18　中文乱码

5．运行“光盘\Code\Debug\04\05”文件夹下的程序，输出 preg_match() 函数语法错误提示，如图 4.19 所示。请根据注释改正程序。

图 4.19　preg_match() 函数语法错误提示



第 5 章

PHP 数组

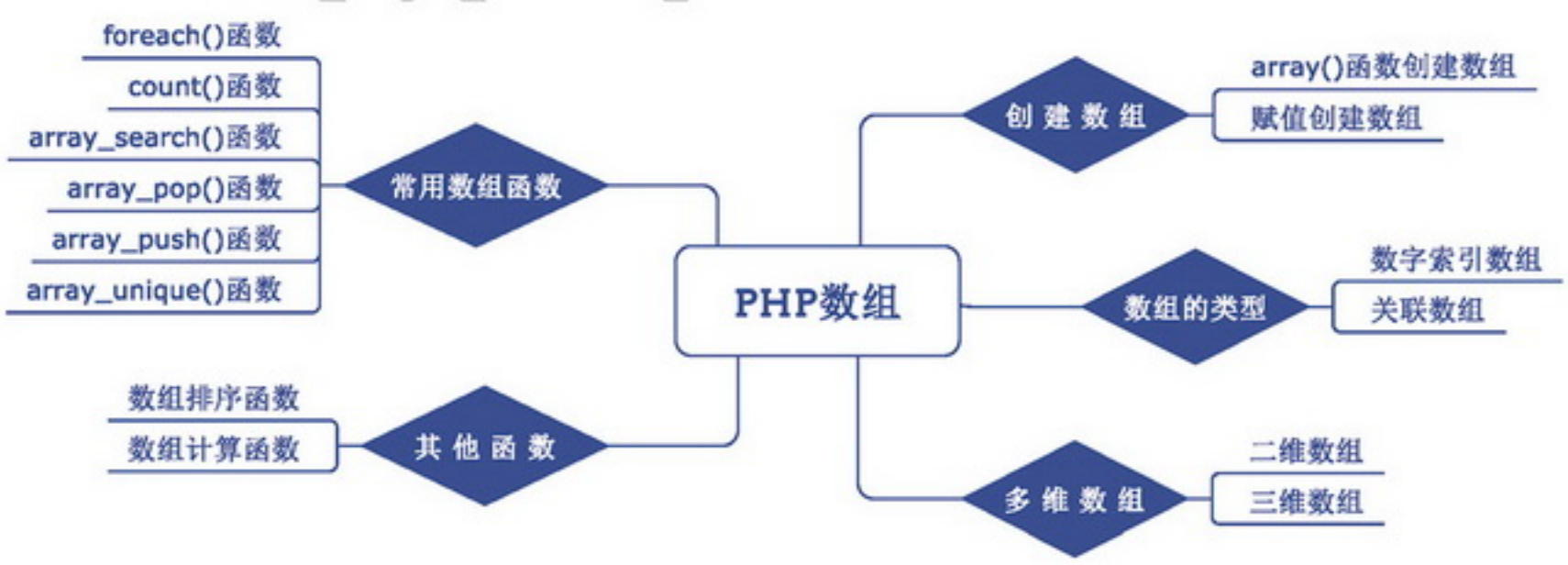
（视频讲解：1 小时 50 分）

本章概览

数组本质上就是一系列数据的组合，可分为一维数组、二维数组以及多维数组，在程序设计中引入数组可以更有效地管理和处理数据。

PHP 对数组的操作能力非常强大，尤其是 PHP 为程序开发人员提供了大量方便、易懂的数组操作函数。本章不仅对数组的创建和类型进行了详细讲解，同时还介绍了统计数组元素个数、查询数组中指定元素等基本操作。通过学习本章的知识和实例讲解，读者可以掌握数组的使用方法，提高编程技能。

知识框架



5.1 什么是数组

视频讲解：光盘\Video\05\5.1 什么是数组.mp4

数组，顾名思义，本质上就是一系列数据的组合。在这个组合中，每个数据都是独立的，可以对每个单独的数据进行分配和读取。在程序设计中引入数组可以更有效地管理和处理数据。我们可以单独定义 a、b、c、d、e 这 5 个变量，也可以定义一个数组包含这 5 个变量，如图 5.1 所示。

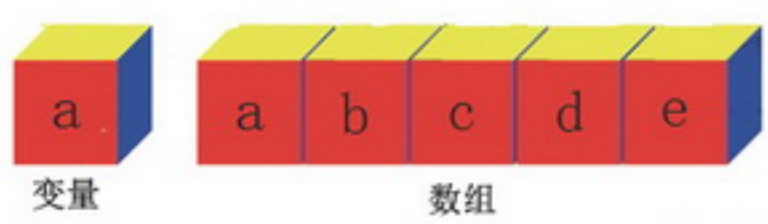


图 5.1 变量和一维数组的概念图

数组中的每个实体都包含两项：键（也称为下标）和值。可以通过键值来获取相应数组元素。这就像篮球球员和球衣号码一样，如 NBA 芝加哥公牛队乔丹球衣号码是 23 号，假如公牛队是一个数组，那么，23 就是数组的一个键，乔丹就是该键对应的值。我们可以通过球衣号码对应找到球员。例如，2017 年 NBA 全明星西部首发阵容可以用数组表示，如图 5.2 所示。

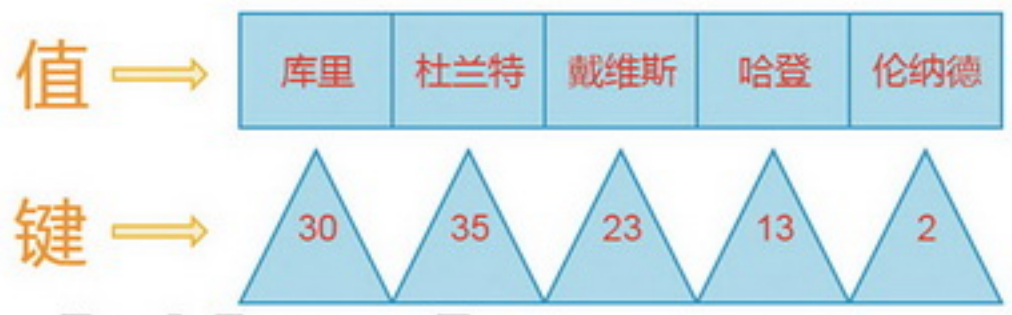


图 5.2 NBA 全明星西部首发数组键值对

5.2 创建数组

在 PHP 中创建数组的方式主要有两种：一种是应用 array() 函数创建数组，另一种是直接通过为数组元素赋值的方式创建数组。

5.2.1 使用 array() 函数创建数组

视频讲解：光盘\Video\05\5.2.1 使用array()函数创建数组.mp4

可以用 array() 语言结构来新建一个数组，该数组接受任意数量用逗号分隔的键（key）=> 值（value）对，格式如下：

```
array(  key =>  value,
...
     )
```

📖 **说明：**键（key）可是一个整数 integer 或字符串 string，如果省略了索引，则会自动产生从 0 开始的整数索引。如果索引是整数，则下一个产生的索引将是目前最大的整数索引 +1。如果定义了两个完全一样的索引，则后面一个会覆盖前一个。值（value）可以是任意类型的值，如果是数组类型时，就是二维数组。

应用 array() 函数声明数组时，数组下标既可以是数值索引也可以是关联索引。下标与数组元素值之间用“=>”进行连接，不同数组元素之间用逗号进行分隔。

应用 array() 函数定义数组比较灵活，可以在函数体中只给出数组元素值，而不必给出键值。例如：

```
01  <?php
02      $array = array ("asp", "php", "jsp");                    //定义数组
03      echo "<pre>";
04      print_r($array);                                         //输出数组元素
05  ?>
```

结果为：

```
Array
(
    [0] => asp
    [1] => php
    [2] => jsp
)
```

⚡ **注意：**自 PHP 5.4 起可以使用短数组定义语法，用 [] 替代 array()，如 $array = ["asp", "php", "jsp"];。

PHP 提供了创建数组的 array() 语言结构。在使用其中的数据时，可以直接利用它们在数组中的排列顺序取值，这个顺序称为数组的下标。例如：

```
01  <?php
02      $array = array ("asp", "php", "jsp");                    //定义数组
03      echo $array[ 1 ];                                        //输出数组元素
04  ?>
```

运行结果为：

```
php
```

⚡ **注意：**使用这种方式定义数组时，下标默认从 0 开始，而不是 1，然后依次增加 1。所以下标为 2 的元素是指数组的第 3 个元素。

下面将通过 array() 函数创建数组，代码如下：

```php
<?php
    $array=array("1"=>"编","2"=>"程","3"=>"词","4"=>"典");  //声明数组
    print_r($array);                                          //输出数组元素
    echo "<br>";
    echo $array[1];                                           //输出数组元素的值
    echo $array[2];                                           //输出数组元素的值
    echo $array[3];                                           //输出数组元素的值
    echo $array[4];                                           //输出数组元素的值
?>
```

运行结果为：

```
Array ( [1] => 编 [2] => 程 [3] => 词 [4] => 典 )
编程词典
```

5.2.2 通过赋值方式创建数组

视频讲解：光盘\Video\05\5.2.2 通过赋值方式创建数组.mp4

PHP 中另一种比较灵活的数组创建方式是直接为数组元素赋值。如果在创建数组时不知道所创建数组的大小，或在实际编写程序时数组的大小可能发生改变，采用这种数组创建的方法较好。

下面通过具体的例子对该种数组声明方式进行讲解，代码如下：

```php
<?php
    $array[1]="编";
    $array[2]="程";
    $array[3]="词";
    $array[4]="典";
    print_r($array);                                          //输出所创建数组的结构
?>
```

运行结果为：

```
Array ([1] => 编 [2] => 程 [3] => 词 [4] => 典)
```

5.3 数组的类型

PHP 支持两种数组：数字索引数组（indexed array）和关联数组（associative array），前者使用数字作为键，后者使用字符串作为键。

5.3.1 数字索引数组

视频讲解：光盘\Video\05\5.3.1 数字索引数组.mp4

PHP 数字索引一般表示数组元素在数组中的位置，它由数字组成，数字索引数组默认索引值从数字 0 开始，不需要特别指定，PHP 会自动为索引数组的键名赋一个整数值，然后从这个值开始自动增量，当然，也可以指定从某个位置开始保存数据。我们可以使用数字索引定义 5.1 节中的 NBA 全明星数组，如图 5.3 所示。

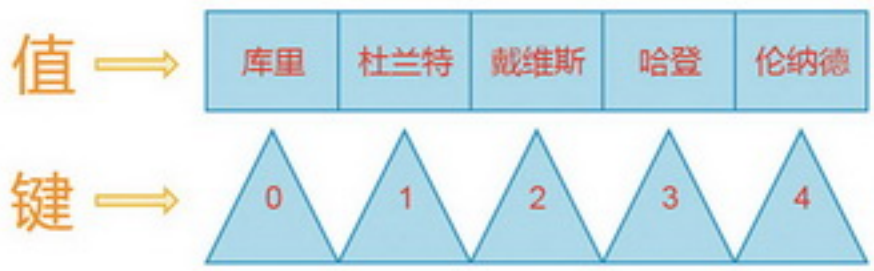


图 5.3　NBA 全明星数组数字索引

例如，创建两个数组 $project1 和 $project2，具体代码如下：

```
<?PHP
    $project1 = array('明日科技','明日学院','明日图书','明日论坛');          //不用下标
    $project2 = array(1=>'明日科技','明日学院','明日图书','明日论坛');        //下标从1开始，递增
    print_r($project);                                                    //输出数组
    echo "<br>";
    print_r($project1);                                                   //输出数组
?>
```

运行结果如下所示：

```
Array ( [0] => 明日科技 [1] => 明日学院 [2] => 明日图书 [3] => 明日论坛 )
Array ( [1] => 明日科技 [2] => 明日学院 [3] => 明日图书 [4] => 明日论坛 )
```

5.3.2 关联数组

视频讲解：光盘\Video\05\5.3.2 关联数组.mp4

关联数组（associative array）的键名可以是数值和字符串混合的形式，而不像数字索引数组的键名只能为数字，在一个数组中，只要键名中有一个不是数字，那么这个数组就称为关联数组。以水果名称和价钱的数组为例，键为水果名称，值为水果价格，如图 5.4 所示。

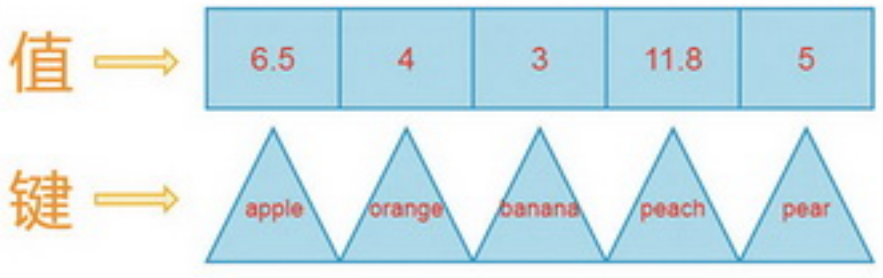


图 5.4　关联数组示意图

创建一个关联数组，代码如下：

```
<?php
    $newarray = array("first"=>1,"second"=>2,"third"=>3);
    echo $newarray["second"];
    echo "<br>";
    $newarray["third"]=8;
    echo $newarray["third"];
?>
```

运行结果为：

```
2
8
```

多学两招：关联数组的键名可以是任何一个整数或字符串。如果键名是一个字符串，不要忘了给这个键名或索引加上一个定界修饰符——单引号（''）或双引号（""）。

5.4 多维数组

视频讲解：光盘\Video\05\5.4 多维数组.mp4

数组不一定就是一个键名和值的简单列表，数组中的每个位置还可以保存另一个数组。使用这种方法，可以创建一个二维数组。以某酒店的楼层和房间号为例，如图 5.5 所示，每一个楼层都是一个一维数组，楼层数本身又构成了一个数组，这样一间酒店就构成了一个二维数组。

楼层	房间号						
一楼	1101	1102	1103	1104	1105	1106	1107
二楼	2101	2102	2103	2104	2105	2106	2107
三楼	3101	3102	3103	3104	3105	3106	3107
四楼	4101	4102	4103	4104	4105	4106	4107
五楼	5101	5102	5103	5104	5105	5106	5107
六楼	6101	6102	6103	6104	6105	6106	6107
七楼	7101	7102	7103	7104	7105	7106	7107

图 5.5　二维表结构的楼层房间号

二维数组常用于表示表，表中的信息以行和列的形式表示，第一个下标代表元素所在的行，第二个下标代表元素所在的列。下面创建一个二维数组，代码如下：

```
<?php
    $str = array (
        "书籍"=>array ("文学","历史","地理"),
        "体育用品"=>array ("m"=>"足球","n"=>"篮球"),
```

```
05          "水果"=>array ("橙子",8=>"葡萄","苹果") );          //声明数组
06      echo "<pre>";
07      print_r ( $str) ;                                      //输出数组元素
08  ?>
```

运行结果如图 5.6 所示。

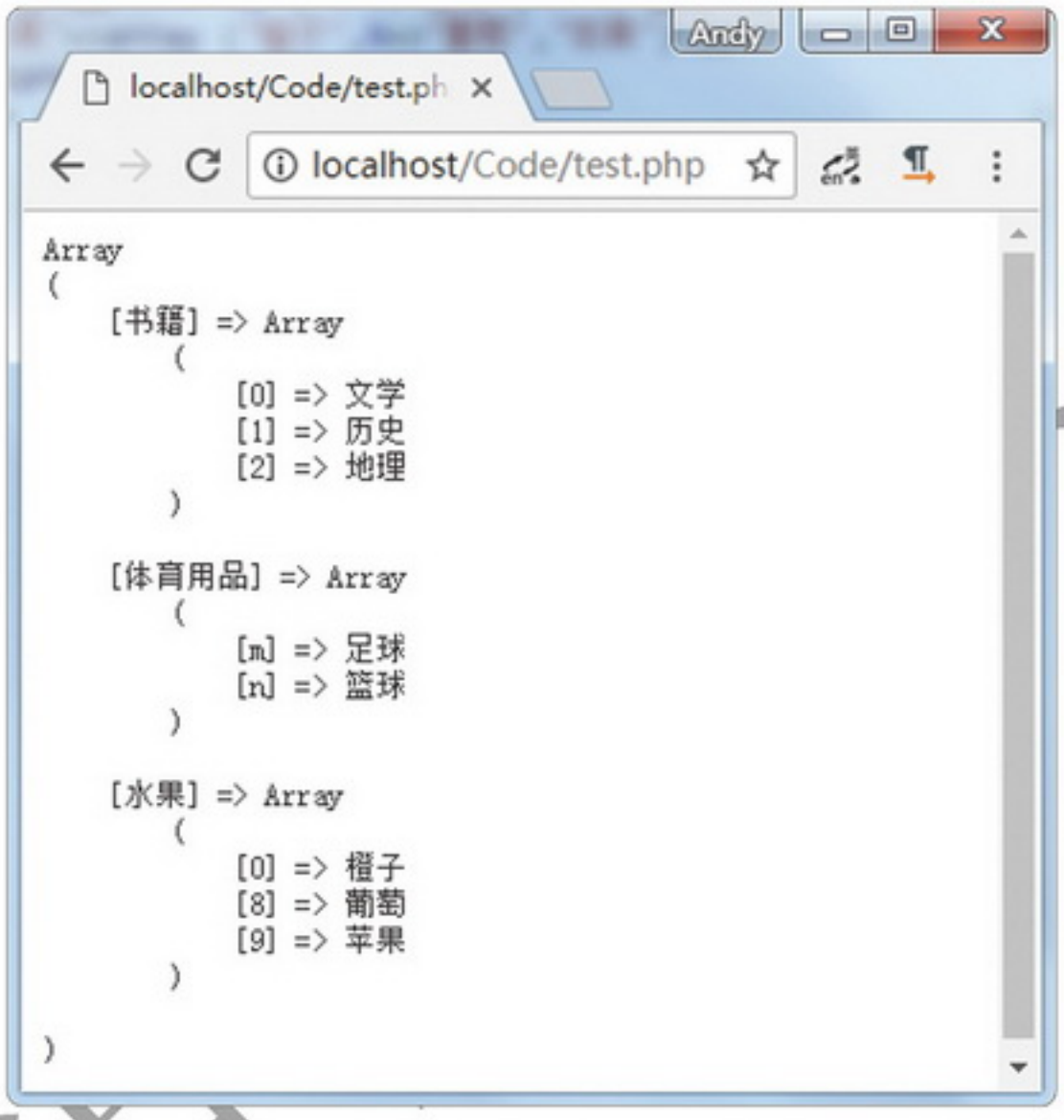


图 5.6　输出二维数组运行结果

按照同样的思路，将前面二维数组中最底层元素替换成数组，就可以创建一个三维数组。下面创建一个三维数组，代码如下：

```
01  <?php
02      //创建数组
03      $str = array (
04          "书籍"=>array ("文学"=>array('红楼梦','西游记'),"历史"=>array('上下五千年')),
05          "体育用品"=>array ("m"=>"足球","n"=>"篮球"),
06          "水果"=>array ("橙子",8=>"葡萄","苹果")
07      );
08      echo "<pre>";
09      print_r ( $str) ;                                      //输出数组元素
10  ?>
```

运行结果如图 5.7 所示。

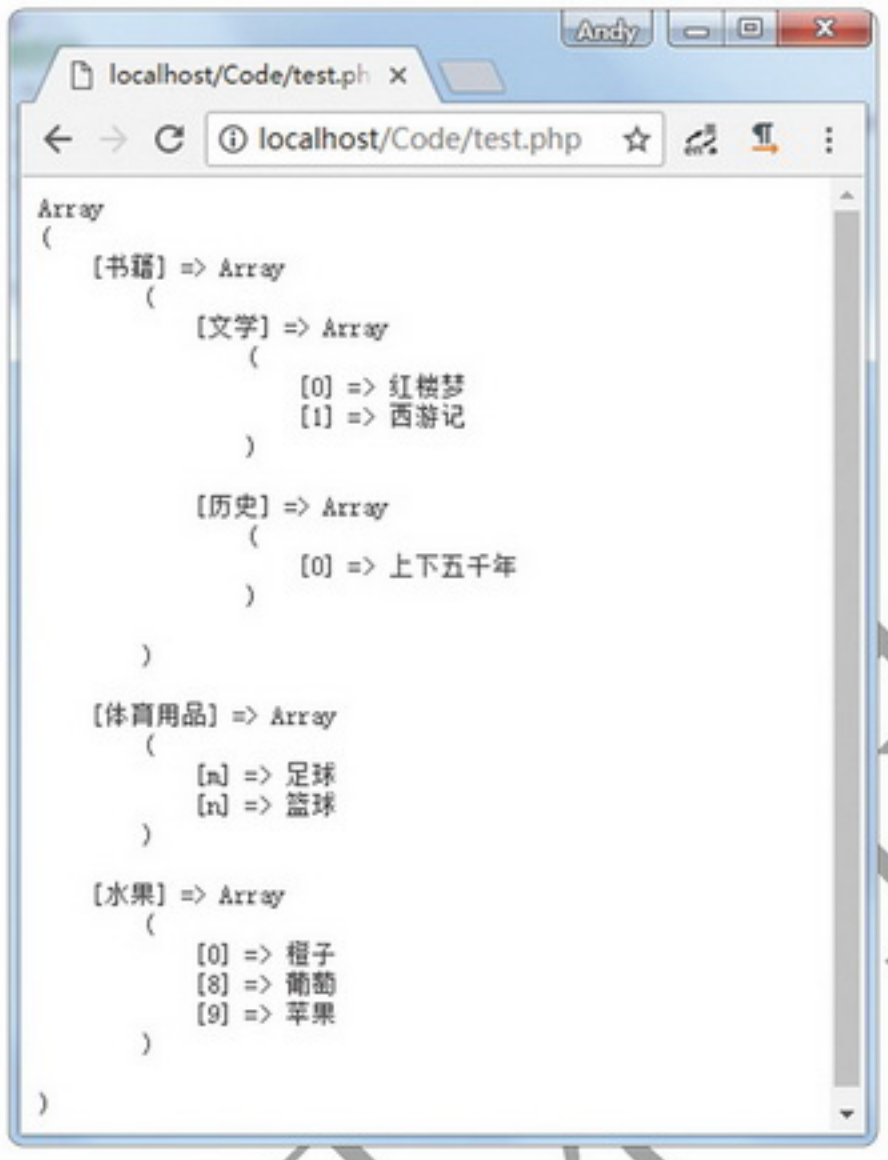


图 5.7　输出三维数组运行结果

5.5 遍历数组

视频讲解

视频讲解：光盘\Video\05\5.5 遍历数组.mp4

遍历数组中的所有元素是常用的一种操作，在遍历的过程中可以完成查询等功能。在生活中，如果想要去商场买一件衣服，就需要在商场中逛一遍，看是否有想要的衣服，逛商场的过程就相当于遍历数组的操作。在 PHP 中遍历数组的方法有多种，下面介绍最常用的 foreach() 函数遍历数组。

实例 01　使用 foreach() 函数遍历数组

实例位置：光盘\Code\SL\05\01
视频位置：光盘\Video\05\

对于一个存有大量网址的数组变量 $url，如果应用 echo 语句一个个地输出，将相当烦琐，而通过 foreach() 函数遍历数组则可轻松获取数据信息，代码如下：

实例01-1

```
<?php
    //定义数组
    $url = array('明日学院'=>'www.mingrisoft.com',
                 'PHP官网'=>'www.PHP.net',
```

```
05                  'PHP之道'=>'https://laravel-china.github.io/php-the-right-way'
06      );
07      //遍历数组
08      foreach ( $url as $key=>$link ) {
09          echo $key.":".$link.'<br>';
10      }
11  ?>
```

在上面的代码中，PHP 为 $url 的每个元素依次执行循环体（echo 语句）一次，将 $link 赋值给当前元素的值，其中 $key 为数组的键值。各元素按数组内部顺序进行处理。

运行结果如图 5.8 所示。

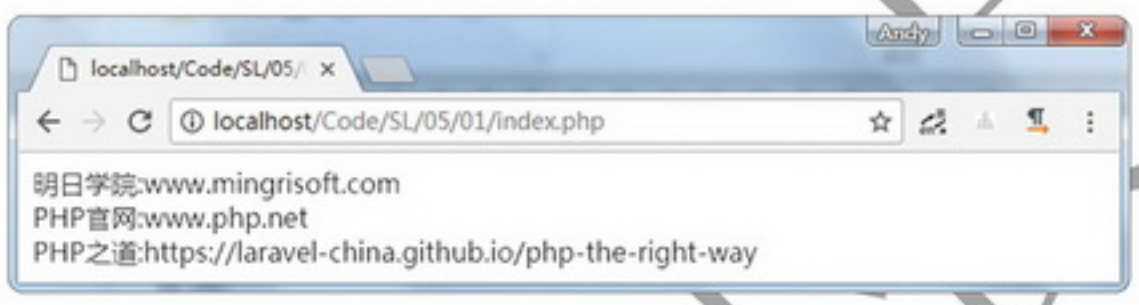


图 5.8　foreach() 函数遍历数组运行结果图

练一练：

（1）在明日学院网站的课程分类中，有如下数组：

```
$category=array( '后端开发'=>['PHP','Java','C++'],
                 '前端开发'=>['HTML','CSS','JavaScript'],
                 '数据库开发'=>['Mysql','Oracle']);
```

使用 foreach() 函数嵌套 foreach() 函数，输出该课程列表，运行结果如图 5.9 所示。（光盘\Code\Try\05\01）

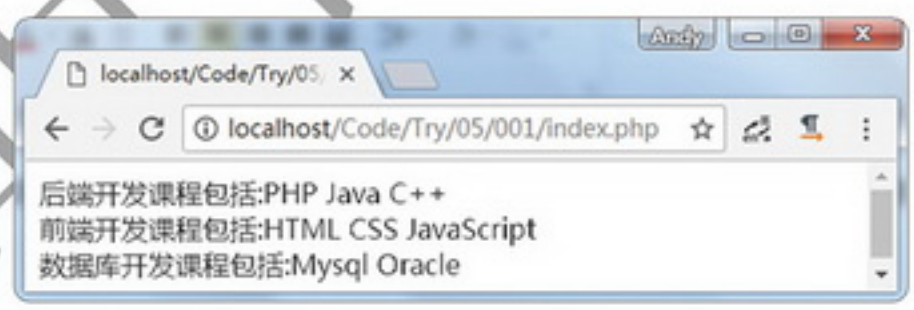


图 5.9　foreach() 函数循环嵌套

（2）在博客首页中，左侧通常会有一个标签栏。当用户单击相应的标签后，页面即跳转到该标签下文章列表页。根据标签数组，使用 foreach() 函数生成标签链接。如图 5.10 所示。（光盘\Code\Try\05\02）

图 5.10　使用 foreach() 函数生成标签链接

5.6 统计数组元素个数

视频讲解：光盘\Video\05\5.6 统计数组元素个数.mp4

在 PHP 中，使用 count() 函数对数组中的元素个数进行统计。语法格式如下：

```
int count ( mixed $array [, int $mode])
```

参数如下：

◆ array：必要参数，为输入的数组。

◆ mode：可选参数，参数值为 COUNT_RECURSIVE（或 1），如选中此参数，本函数将递归地对数组计数，对计算多维数组的所有单元尤其有用。此参数的默认值为 0。

返回值：返回 array 中的单元数量。

例如，使用 count() 函数统计数组元素的个数，代码如下：

```
<?php
    $array = array("《PHP函数参考大全》","《PHP程序开发范例宝典》",
                   "《PHP网络编程自学手册》","《PHPS从入门到精通》");
    echo count($array);                          //统计数组元素的个数，输出结果为4
?>
```

运行结果如下：

```
4
```

例如，使用 count() 函数递归地统计数组中图书数量并输出，代码如下：

```
<?php
//声明一个二维数组
$array = array("php" => array("《PHP函数参考大全》",
                              "《PHP程序开发范例宝典》",
                              "《PHP数据库系统开发完全手册》"),
               "asp" => array("《ASP经验技巧宝典》")
);
echo count($array,COUNT_RECURSIVE);          //递归统计数组元素的个数，2+4=6
?>
```

运行结果为：

```
6
```

注意：在统计二维数组时，如果直接使用 count() 函数只会显示到一维数组的个数，所以参数设为 COUNT_RECURSIVE（或 1），对计算多维数组的所有单元尤其有用。

5.7 查询数组中指定元素

视频讲解：光盘\Video\05\5.7 查询数组中指定元素.mp4

array_search() 函数可以在数组中查询给定的值，找到后返回键名，否则返回 false。

语法格式如下：

```
mixed array_search ( mixed $needle, array $haystack [, bool $strict])
```

参数如下：

◆ needle：指定在数组中搜索的值。

◆ haystack：指定被搜索的数组。

◆ strict：为可选参数，默认值为 false。如果值为 true，还将在数组中检查给定值的类型。

返回值：如果找到了 needle 则返回它的键，否则返回 false。

实例 02　查询数组中指定的元素的值

实例位置：光盘\Code\SL\05\02

视频位置：光盘\Video\05\

明日学院图书销量排行榜中，排名前四位的 PHP 书籍分别是《零基础学 PHP》《PHP 项目开发实战入门》《PHP 从入门到精通》《PHP 开发实战》，其对应的价格依次是 69.80 元、69.80 元、62.90 元、55.90 元。使用 array_search() 函数查询图书《PHP 从入门到精通》的价格。代码如下：

实例02-1

```
<?PHP
    $book_name  = '《PHP从入门到精通》';
    $books = ['《零基础学PHP》','《PHP项目开发实战入门》','《PHP从入门到精通》','《PHP开发实战》'];
    $price = [69.80,69.80,62.90,55.90];
    $key = array_search($book_name,$books);
    if($key){
        echo $book_name."价格：￥".$price[$key];
    }else{
        echo $book_name."价格："."未知";
    }
?>
```

上述代码中，使用 array_search() 函数查询 $book_name 变量在 $book 数组中的下标，根据该下标获取 $price 价格数组中对应的值。运行结果如图 5.11 所示。

图 5.11　查询数组中指定元素的值

练一练：

（1）明日学院社区中，每个版块下都有相应的版主，只有版主才能够编辑该版块。已知 PHP 答疑区的版主 id 是一个数组，如 $moderator=[1,3,5,7]，使用 rand() 函数随机生成一个 id，使用 array_search() 函数判断该 id 的用户是否是版主。（光盘\Code\Try\05\03）

（2）明日学院社区中，有一个版块排名功能，排名结果以数组形式展示，键为“名次”，值为“版块名称”，如 $rank=array(1=>' Java',2=>'PHP',3=>'C++',4=>'C#',5=>'JS')。试着使用 array_search() 函数获取 PHP 版块的当前排名。（光盘\Code\Try\05\04）

5.8 获取数组中最后一个元素

视频讲解：光盘\Video\05\5.8 获取数组中最后一个元素.mp4

在 PHP 中，可以通过函数 array_pop() 获取数组中的最后一个元素。语法格式如下：

```
mixed array_pop ( array $array)
```

参数及返回值如下：

◆ array：输入的数组。

◆ 返回值：返回数组的最后一个单元，并将原数组的长度减 1，如果数组为空（或者不是数组）将返回 null。

例如，应用 array_pop() 函数获取数组中的最后一个元素，代码如下：

```
<?php
    $arr = array ("ASP", "Java", "Java Web", "PHP", "VB");          //定义数组
    $array = array_pop ($arr);                                      //获取数组中最后一个元素
    echo "被弹出的单元是: $array <br />";                           //输出最后一个元素值
    print_r($arr);                                                  //输出数组结构
?>
```

运行结果为：

```
被弹出的单元是: VB
Array ( [0] => ASP [1] => Java [2] => Java Web [3] => PHP )
```

5.9 向数组中添加元素

视频讲解：光盘\Video\05\5.9 向数组中添加元素.mp4

可以通过 array_push() 函数向数组中添加元素。array_push() 函数将数组当成一个栈，将传入的变量压入该数组的末尾，该数组的长度将增加入栈变量的数目，返回数组新的元素总数。语法格式如下：

```
int array_push ( array $array, mixed $var [, mixed ...])
```

参数及返回值如下：

◆ array：指定的数组。

◆ var：压入数组中的值。

◆ 返回值：数组新的单元总数。

例如，应用 array_push() 函数向数组中添加元素，代码如下：

```
<?php
    $array_push = array ("《PHP从入门到精通》", "《PHP范例手册》");          //定义数组
    array_push ($array_push, "《PHP开发典型模块大全》","《PHP网络编程自学手册》"); //添加元素
    print_r($array_push);                                                  //输出数组结果
?>
```

运行结果如下：

```
Array ( [0] => 《PHP从入门到精通》 [1] => 《PHP范例手册》 [2] => 《PHP开发典型模块大全》 [3] =>
《PHP网络编程自学手册》 )
```

5.10 删除数组中重复元素

视频讲解：光盘\Video\05\5.10 删除数组中重复元素.mp4

通过 array_unique() 函数可以删除数组中重复的元素。array_unique() 函数将值作为字符串排序，然后对每个值只保留第一个键名，忽略后面的所有键名，即删除数组中重复的元素。语法格式如下：

```
array array_unique ( array $array)
```

参数及返回值如下：

◆ array：输入的数组。

◆ 返回值：过滤后的数组。

实例 03　使用 array_unique() 函数删除重复图书

实例位置：光盘\Code\SL\05\03

视频位置：光盘\Video\05\

本实例将模拟明日图书系统添加图书的操作，如果添加的某本图书已经存在，则删除重复图书。使用 array_push() 函数向数组中添加数据，应用 array_unique() 函数删除数组中重复的元素，代码如下：

实例03-1

```
<?php
    $array = array ("PHP从入门到精通", "PHP范例手册",
                    "PHP范例手册","PHP网络编程自学手册");                    //定义数组
    array_push ($array, "PHP开发典型模块大全","PHP网络编程自学手册");
```

```
05      print_r($array);                                    //输出数组
06      echo "<br>";
07      $result = array_unique($array);                     //删除数组中重复的元素
08      print_r($result);                                   //输出删除后的数组
09  ?>
```

运行结果如图 5.12 所示。

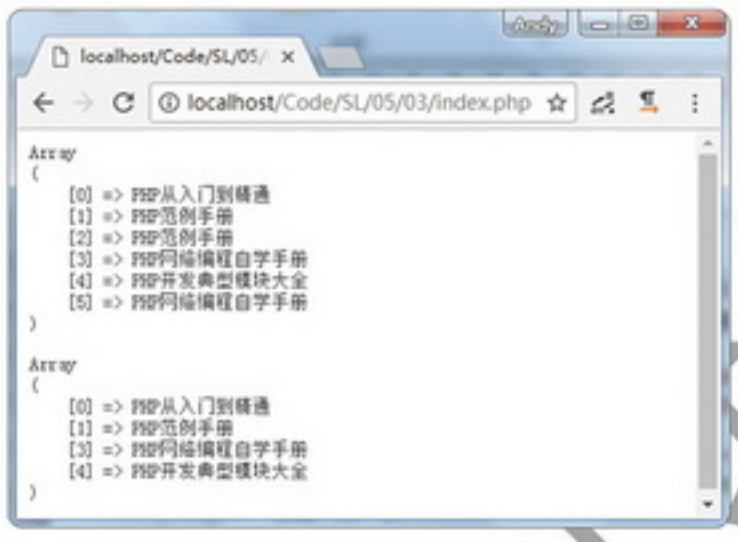


图 5.12　array_unique() 函数删除重复图书

练一练：

（1）在微博@好友功能中，尽管可以重复@好友，但好友只能接收到一次@信息。试着使用 array_unique() 函数去除重复的好友信息。（光盘\Code\Try\05\05）

（2）在明日学院后台，可以添加相应版块的版主功能。每个版块可以有多个版主，用“,”分隔版主名称，如 PHP 版块的版主有“张三,李四,王五,赵六,张三”。试着使用 array_unique() 函数去除重复的版主名称。运行结果如图 5.13 所示。（光盘\Code\Try\05\06）

图 5.13　使用 array_unique() 函数去除重复数据

5.11　其他常用数组函数

由于篇幅有限，本章不能将数组函数逐一介绍，下面将简单介绍一些其他常用数组函数。在遇到问题需要使用时，可自行查找《PHP 手册》，查找相应函数的用法，实现自己所设计的功能。

5.11.1　数组排序函数

视频讲解：光盘\Video\05\5.11.1 数组排序函数.mp4

常用的数组排序函数如表 5.1 所示：

表 5.1　常用数组排序函数

函 数 名 称	描　　述
sort()	本函数对数组进行排序。当本函数结束时数组元素将被从最低到最高重新排序，不保持索引关系
rsort()	对数组逆向排序
asort()	对数组进行排序并保持索引关系
arsort()	对数组进行逆向排序并保持索引关系
ksort()	对数组按照键名排序
krsort()	对数组按照键名逆向排序
natsort()	用“自然排序”算法对数组排序
natcasesort()	用“自然排序”算法对数组进行不区分大小写字母的排序

实例 04　根据帖子的回复数量排序

实例位置：光盘\Code\SL\05\04
视频位置：光盘\Video\05\

明日学院社区中有一个热帖功能，即根据帖子的回复数量由多到少作为热帖的排名顺序。帖子数组如下所示：

```php
$data = array(
    array('post_id'=>1,'title'=>'如何学好PHP','reply_num'=>582),
    array('post_id'=>2,'title'=>'PHP数组常用函数汇总','reply_num'=>182),
    array('post_id'=>3,'title'=>'PHP字符串常用函数汇总','reply_num'=>982)
);
```

实现根据“reply_num”由多到少进行排序的功能，代码如下：

实例04-1

```php
<?php
    /**
     * 根据数组中的某个键值大小进行排序，仅支持二维数组
     *
     * @param array $array 排序数组
     * @param string $key 键值
     * @param bool $asc 默认正序,false为降序
     * @return array 排序后数组
     */
    function arraySortByKey($array=array(), $key="", $asc = true){
        $result = array();
        /** 整理出准备排序的数组 **/
        foreach ( $array as $k => $v ) {
            $values[$k] = isset($v[$key]) ? $v[$key] :"";
        }
```

```
16          unset($v);                                          //销毁变量
17          $asc ? asort($values) : arsort($values);            //对需要排序的键值进行排序
18          /** 重新排列原有数组 **/
19          foreach ( $values as $k => $v ) {
20              $result[$k] = $array[$k];
21          }
22          return $result;
23      }
24      /** 定义数组 **/
25      $data = array(
26                  array('post_id'=>1,'title'=>'如何学好PHP','reply_num'=>582),
27                  array('post_id'=>2,'title'=>'PHP数组常用函数汇总','reply_num'=>182),
28                  array('post_id'=>3,'title'=>'PHP字符串常用函数汇总','reply_num'=>982)
29      );
30      $new_arrray = arraySortByKey($data,'reply_num',false); //调用arrySortByKey()函数
31      echo "<pre>";                                           //指定输出格式
32      print_r($new_arrray);                                   //输出数组
33  ?>
```

运行结果如图 5.14 所示。

图 5.14　帖子排序运行结果

练一练：

（1）某班级的数学成绩是一个数组，键为“学生姓名”，值为“数学成绩”，将班级数学成绩由高到底排序。（光盘\Code\Try\05\07）

（2）在调用微信支付接口生成签名时，需要对回调字符串参数按音序排序，请试着实现该排序功能。（光盘\Code\Try\05\08）

5.11.2 数组计算函数

视频讲解：光盘\Video\05\5.11.2 数组计算函数.mp4

常用的数组计算函数如表 5.2 所示。

表 5.2　常用数组计算函数

函 数 名 称	描　　述
array_sum()	计算数组中所有值的和
array_merge()	合并一个或多个数组
array_diff()	计算数组的差集
array_diff_assoc()	带索引检查计算数组的差集
array_intersect()	计算数组的交集
array_intersect_assoc()	带索引检查计算数组的交集

实例 05　多条件筛选商城商品

实例位置：光盘\Code\SL\05\05

视频位置：光盘\Video\05\

本实例将模拟淘宝多条件筛选商品的功能，根据手机品牌筛选出商品数组 $brand，根据手机颜色筛选出的商品数组 $color。现选择品牌为“iPhone”，颜色为“土豪金”的手机。使用 array_intersect() 函数实现该功能。代码如下：

实例05-1

```
<?php
    $brand = array('iPhone7土豪金','华为P10宝石蓝','小米6玫瑰红');
    $color = array('iPhone7土豪金','华为土豪金','小米土豪金');
    $result = array_intersect($brand,$color);
    print_r($result);
?>
```

运行结果如图 5.15 所示。

图 5.15　使用 array_intersect() 函数获取交集

练一练：

（1）模拟购物车列表功能，购物车内商品以二维数组形式显示，包含“商品名称”“商品价格”“商品数量”，数据如下所示：

```
$data = array(
            array('iphone7',5700,1),
            array('kindle',521,1),
            array('手机壳',9.9,2)
);
```

试着计算所有商品的总价。（光盘\Code\Try\05\09）

（2）明日学院会员中心有浏览历史记录功能，用户每浏览一次网页，会将浏览的网址加入到浏览列表。使用 array_merge() 函数实现该功能。（光盘\Code\Try\05\10）

5.12 难点解答

5.12.1 数组的索引

为什么索引是从 0 开始的，而不是从 1 开始呢？这是继承了汇编语言的传统，此外，从 0 开始也更利于计算机做二进制的运算和查找。

5.12.2 使用 count() 函数计算二维数组长度

count() 函数有两个参数，当第二个参数设为 COUNT_RECURSIVE（或 1），count() 函数将对数组进行递归计数。请计算如下二维数组的长度，代码如下：

```
01  <?php
02      $numb = array(
03          array(10,15,30),array(10,15,30),array(10,15,30)
04      );
05      echo count($numb,1);
```

输出结果为：

```
12
```

首先遍历的是外面的数组，得出有 3 个元素，再遍历里面的数组，得出的是 9 个元素，结果就是 3+9=12。

5.13 小　结

本章重点讲解了数组的常用操作，这些操作在实际应用中会经常使用。另外，PHP 提供了大量的

数组函数，可以在开发任务中轻松实现所需要的功能。希望通过本章的学习，读者能够举一反三，对所学知识进行灵活运用，开发实用的 PHP 程序。

5.14 动手纠错

1．运行“光盘 \Code\Debug\05\01”文件夹下的程序，使用 array_push() 函数出现“syntax error, unexpected '=>' (T_DOUBLE_ARROW), expecting ',' or ')'”的错误提示，如图 5.16 所示。请根据注释改正程序。

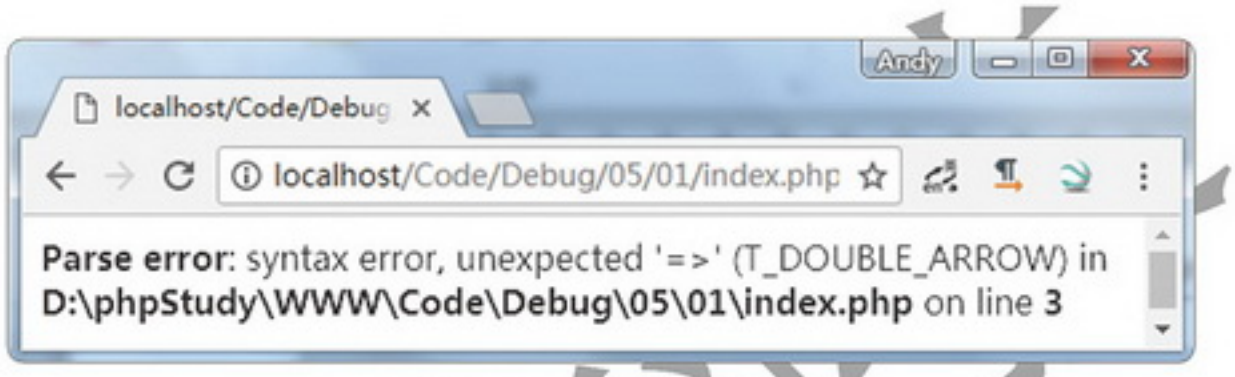


图 5.16　语法错误提示

2．运行“光盘\Code\Debug\05\02”文件夹下的程序，键为 3 的数值为空，如图 5.17 所示。与预期不符，请根据注释改正程序。

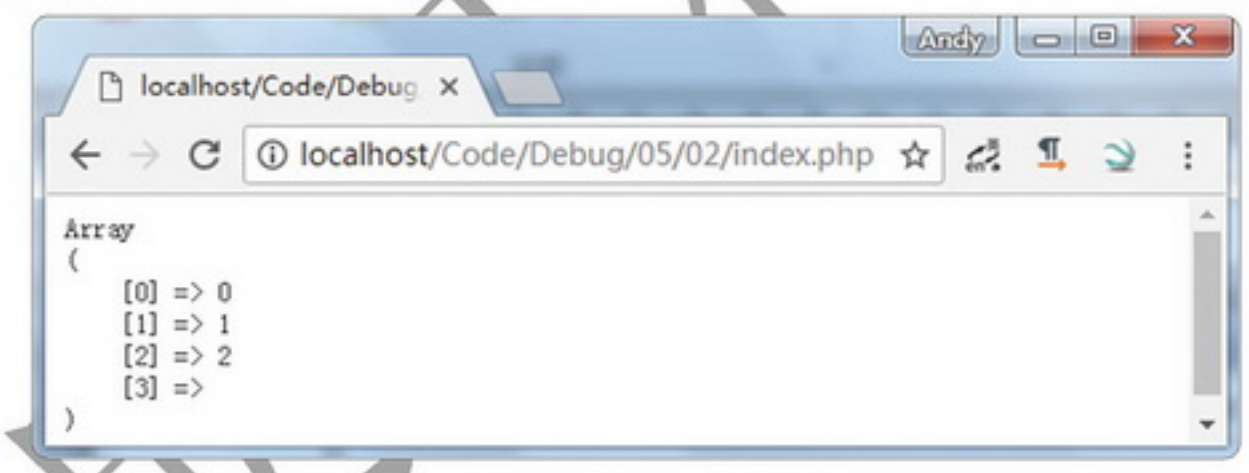


图 5.17　键为 3 的数值为空

3．运行“光盘\Code\Debug\05\03”文件夹下的程序，使用 array_merge() 函数合并数组后，键为 firstname 的数值为空，如图 5.18 所示。请根据注释改正程序。

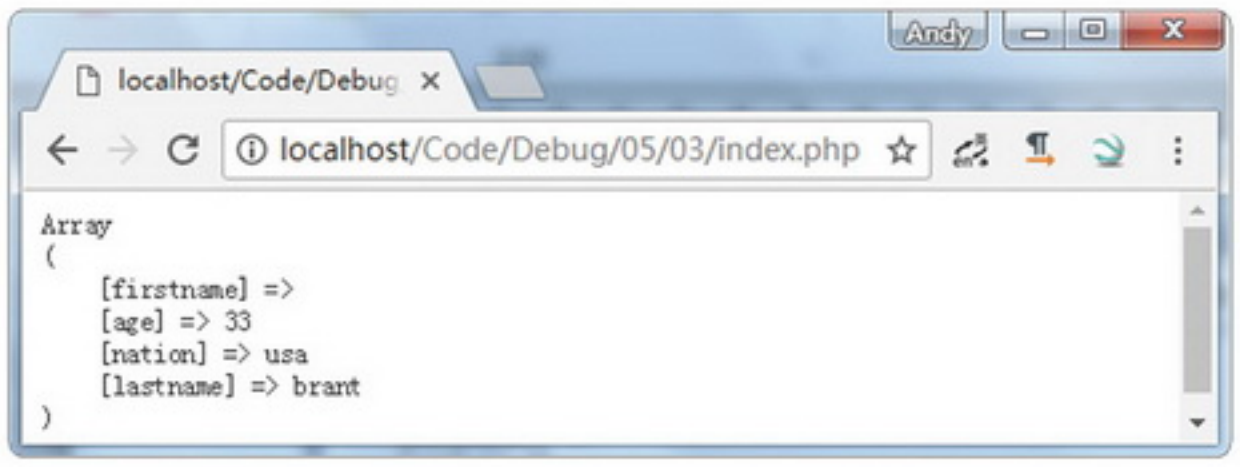


图 5.18　键为 firstname 的数值为空

4．运行“光盘\Code\Debug\05\04”文件夹下的程序，使用 array_unique() 函数去除重复数据。但是输出结果却如图 5.19 所示，与预期不符，请根据注释改正程序。

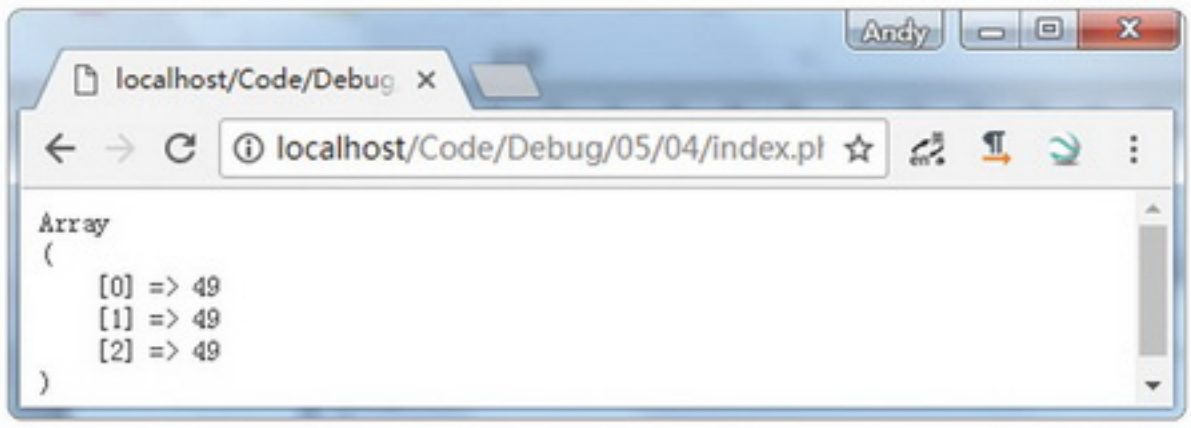


图 5.19　结果数组包含重复数据

5．运行“光盘\Code\Debug\05\05”文件夹下的程序，使用 sort() 函数排序，按照数字从小到大的顺序排序，且字母在数字后面。但是输出结果如图 5.20 所示，没有按照预期将数字排在前面，请根据注释改正程序。

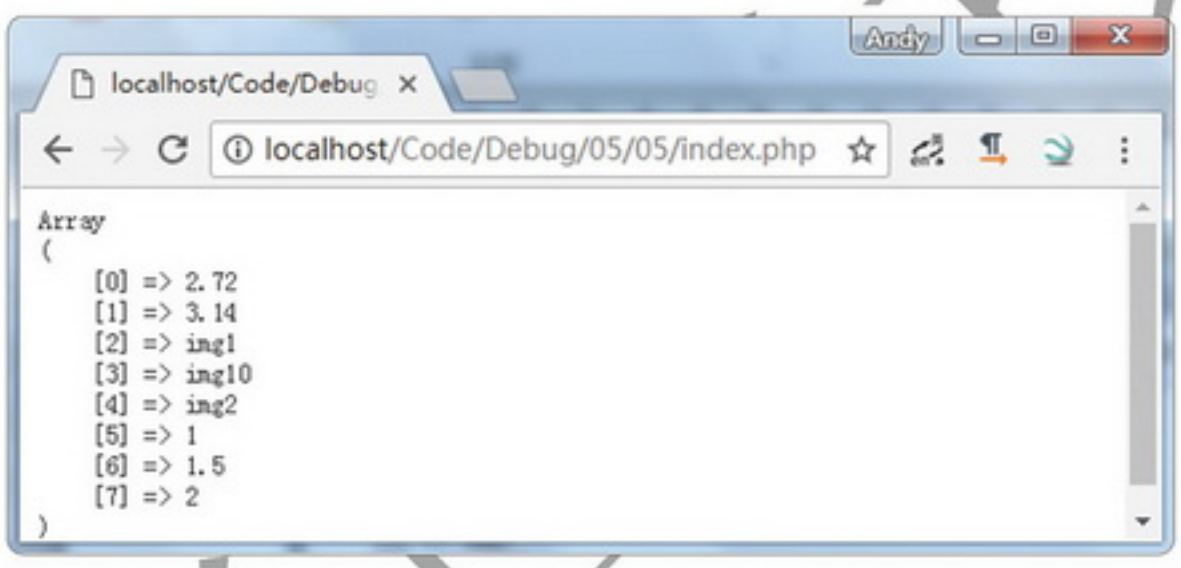


图 5.20　数据没有按预期排序

第 2 篇

核心技术

- 第 6 章　面向对象
- 第 7 章　PHP 与 Web 页面交互
- 第 8 章　MySQL 数据库基础
- 第 9 章　PHP 操作 MySQL 数据库
- 第 10 章　PDO 数据库抽象层

第 6 章

面向对象

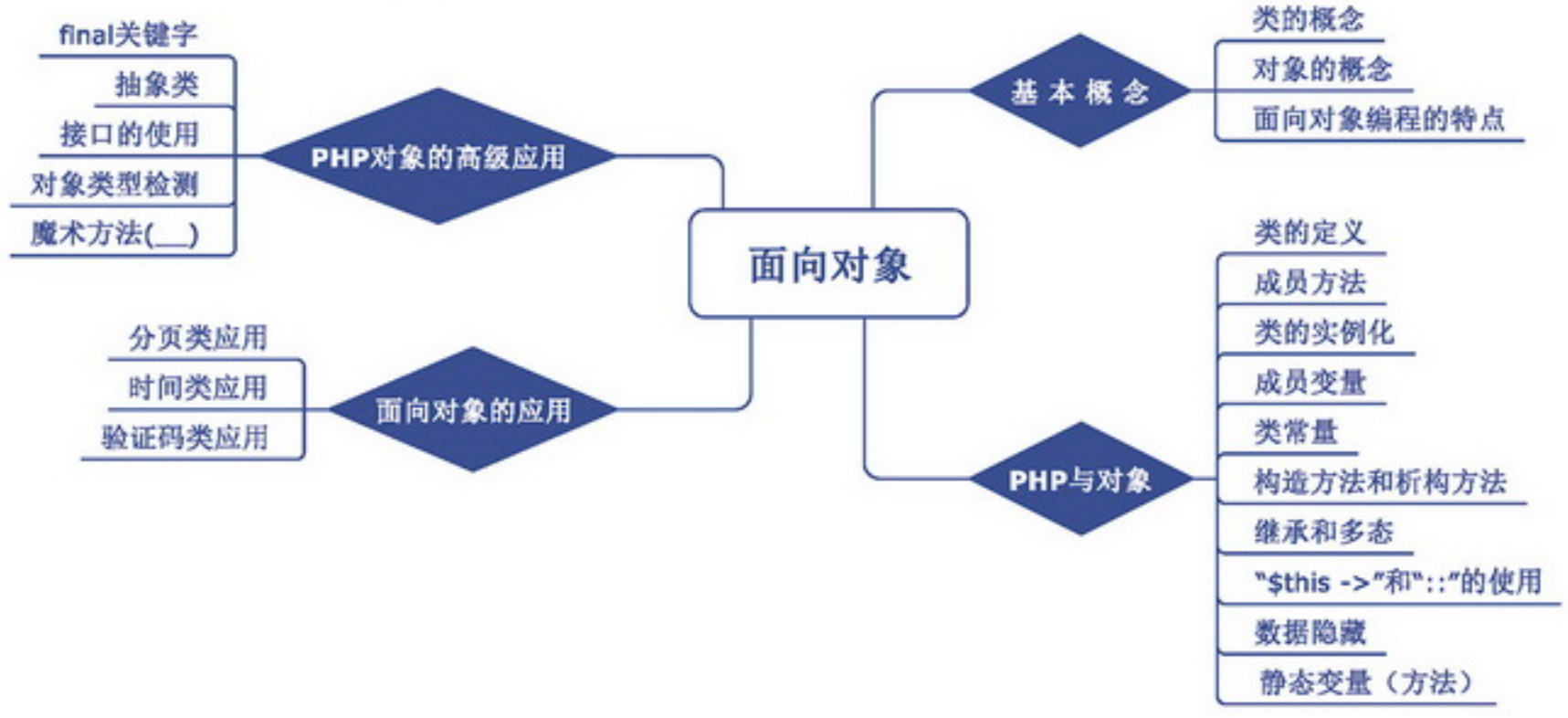
（视频讲解：2 小时 57 分）

本章概览

面向对象是一种计算机编程架构，比面向过程编程具有更强的灵活性和扩展性。类与对象是面向对象编程的重要概念，想要学好 PHP 语言，就一定要掌握面向对象编程技术。

本章将全面讲解面向对象的概念、PHP 与对象及 PHP 对象的高级应用等，为了使初学者更容易入门，在讲解过程中列举了大量实例，并配合生动的图片，让读者更好地了解面向对象的编程思想，并将其灵活运用在程序开发中。

知识框架



6.1 面向对象的基本概念

前几章已经学习了使用字符串和数组组织数据，也学习了使用函数把一些代码收集到能够反复使用的单元中。本章介绍的对象（object）则让这种收集的思想更向前迈进一步。使用对象可以把函数和数据收集在一起。下面来了解一下面向对象的基本概念。

6.1.1 类的概念

视频讲解：光盘\Video\06\6.1.1 类的概念.mp4

世间万物都具有其自身的属性和方法，通过这些属性和方法可以将不同物质区分开来。例如，人具有身高、体重和肤色等属性，还可以进行吃饭、学习、走路等能动活动，这些活动可以说是人具有的功能。可以把人看作程序中的一个类，那么人的身高可以看作类中的属性，走路可以看作类中的方法。也就是说，类是属性和方法的集合，这是面向对象编程方式的核心和基础。通过类可以将零散的用于实现某项功能的代码进行有效管理。例如，创建一个运动类，包括 5 个属性：姓名、身高、体重、年龄和性别，定义 4 个方法：踢足球、打篮球、举重和跳高，如图 6.1 所示。

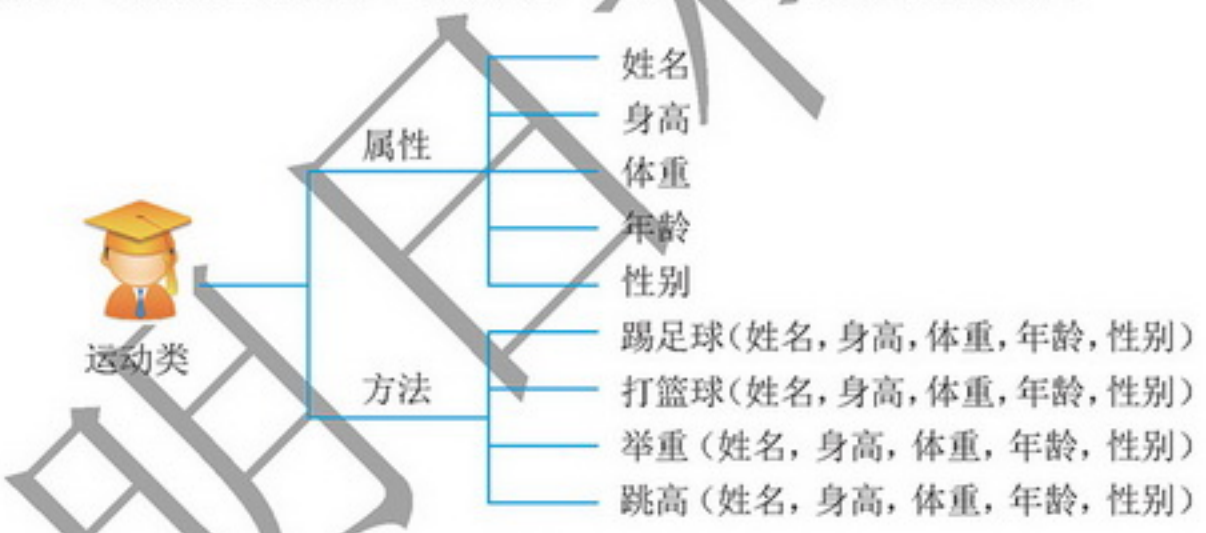


图 6.1 运动类

6.1.2 对象的概念

视频讲解：光盘\Video\06\6.1.2 对象的概念.mp4

类只是具备某项功能的抽象模型，实际应用中还需要对类进行实例化，这样就引入了对象的概念。对象是类进行实例化后的产物，是一个实体。仍然以人为例，“黄种人是人”这句话没有错误，但反过来说“人是黄种人”这句话是错误的。因为除了有黄种人，还有黑种人、白种人等。那么“黄种人”就是“人”这个类的一个实例对象。可以这样理解对象和类的关系：对象实际上就是“有血有肉的、能摸得到看得到的”一个类。

实例化 6.1.1 小节中创建的运动类，如图 6.2 所示。

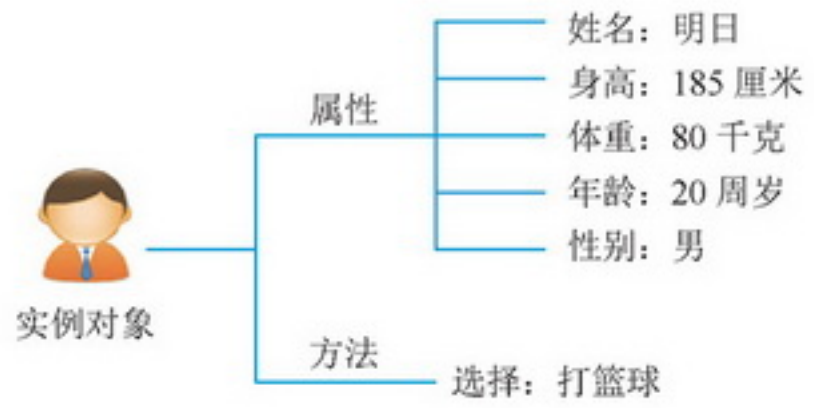


图 6.2　实例化对象

6.1.3 面向对象编程的三大特点

视频讲解：光盘\Video\06\6.1.3 面向对象编程的三大特点.mp4

面向对象编程的三大特点是封装性、继承性和多态性。

1. 封装性

封装性，也可以称为信息隐藏。就是将一个类的使用和实现分开，只保留有限的接口（方法）与外部联系。对于用到该类的开发人员，只要知道这个类该如何使用即可，而不用去关心这个类是如何实现的。这样做可以让开发人员更多地把精力集中起来专注别的事情，同时也避免了程序之间相互依赖带来的不便。这就像普通用户购买汽车，我们只需要知道如何驾驶汽车，并不需要去了解汽车的内部构造。

2. 继承性

在现实生活中，人们可以从他们的父母或者其他直系亲戚那里继承一些东西。比如，在图 6.3 中，“我”可以继承爸爸和妈妈的财产。同理，爸爸也可以继承祖父和祖母的财产。继承性就是派生类（子类）自动继承一个或多个基类（父类）中的属性与方法，并可以重写或添加新的属性或方法。继承这个特性简化了对象和类的创建，增强了代码的可重用性。

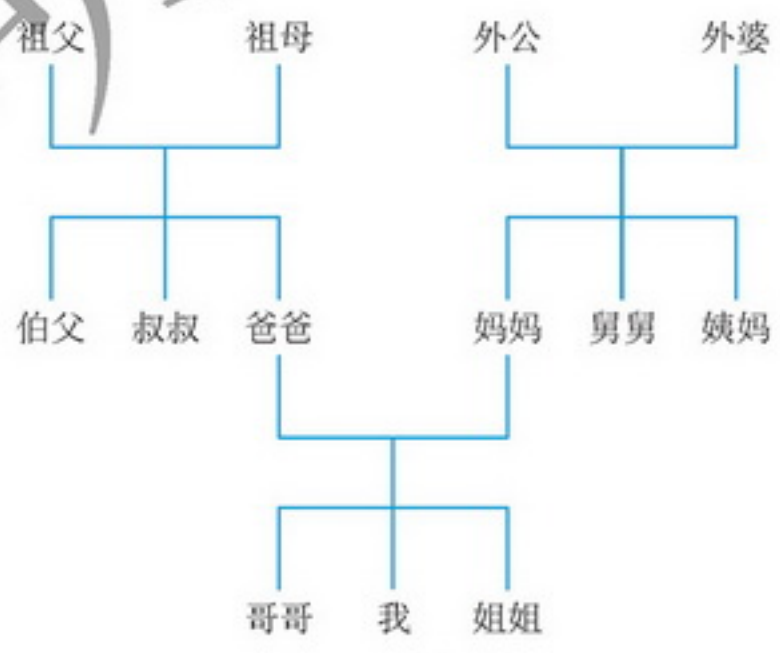


图 6.3　家族图谱

3. 多态性

多态性是指对于不同的类，可以有同名的两个（或多个）方法。例如，定义一个汽车类和一个自行车类，二者都可以具有不同的“移动”操作。多态性增强了软件的灵活性和重用性。

6.2 PHP 与对象

6.2.1 类的定义

视频讲解：光盘\Video\06\6.2.1 类的定义.mp4

和很多面向对象的语言一样，PHP 也是通过 class 关键字加类名来定义类的。类的定义格式如下：

```
<?php
class SportObject{                    //定义运动类
     //…
}
?>
```

上面代码中大括号中间的部分是类的全部内容，如 SportObject 就是一个最简单的类。SportObject 类仅有一个类的骨架，什么功能都没有实现，但这并不影响它的存在。

注意：一个类，即一对大括号之间的全部内容都要在一段代码段中，即一个“<?php … ?>”之间不能分割成多块，例如下面的格式是不允许的：

```
<?php
class SportObject{                    //定义运动类
     //…
?>
<?php
     //…
}
?>
```

6.2.2 成员方法

视频讲解：光盘\Video\06\6.2.2 成员方法.mp4

类中的函数被称为成员方法。函数和成员方法唯一的区别就是，函数实现的是某个独立的功能，而成员方法是实现类中的一个行为，是类的一部分。

下面将创建在图 6.1 中编写的运动类，并添加成员方法。将类命名为 SportObject，并添加打篮球的成员方法 beatBasketball()。代码如下：

```php
<?php
    class SportObject{
        function beatBasketball($name,$height,$weight,$age,$sex){    //声明成员方法
            echo "姓名: ".$name;                                     //方法实现的功能
            echo "身高: ".$height;                                   //方法实现的功能
            echo "年龄: ".$age;                                      //方法实现的功能
        }
    }
?>
```

该方法的作用是输出申请打篮球人的基本信息，包括姓名、身高和年龄。这些信息是通过方法的参数传进来的。

6.2.3 类的实例化

视频讲解：光盘\Video\06\6.2.3 类的实例化.mp4

定义完对象和方法后，并不会真正创建一个对象。这类似汽车的设计图，设计图可以告诉你汽车看上去长什么样，但设计图本身不是一个汽车。你不能开走它，只能用来制造真正的汽车，而且可以使用它制造很多汽车。那么如何创建对象呢？

首先要对类进行实例化，实例化是通过关键字 new 来声明一个对象。然后使用如下格式来调用要使用的方法：

```
对象名 -> 成员方法
```

在 6.1 节中已经讲过，类是一个抽象的描述，是功能相似的一组对象的集合。如果想用类中的方法或变量，首先就要把它具体落实到一个实体，也就是对象上。

以 SportObject 类为例，实例化一个对象并调用 playBasketball() 方法。代码如下：

```php
<?php
    class SportObject{
        function playBasketball($name,$height,$weight,$age,$sex){    //声明成员方法
            if($height>180 and $weight<=100){
                return $name.", 符合打篮球的要求!";                   //方法实现的功能
            }else{
                return $name.", 不符合打篮球的要求!";                 //方法实现的功能
            }
        }
    }
    $sport=new SportObject();
    echo $sport->playBasketball('小明','185','80','20周岁','男');
?>
```

运行结果为：

```
小明, 符合打篮球的要求!
```

6.2.4 成员变量

视频讲解：光盘\Video\06\6.2.4 成员变量.mp4

类中的变量，也称为成员变量（也有称为属性或字段的）。成员变量用来保存信息数据，或与成员方法进行交互来实现某项功能。例如，在 SportObject 类中定义一个 name（运动员姓名）成员变量，接下来就可以在 playBasketball() 方法中使用该变量完成某个功能。

定义成员变量的格式为：

```
关键字 成员变量名
```

说明：关键字可以使用 public、private、protected、static 和 final 中的任意一个。在 6.2.9 小节之前，所有的实例都使用 public 来修饰。对于关键字的使用，将在 6.2.9 小节中进行介绍。

访问成员变量和访问成员方法是一样的。只要把成员方法换成成员变量即可，格式为：

```
对象名 -> 成员变量
```

实例 01 定义并实例化运动类 SportObject

实例位置：光盘\Code\SL\06\01
视频位置：光盘\Video\06\

本实例以图 6.1 和图 6.2 中描述的类和类的实例化为例，将其通过代码实现。首先定义运动类 SportObject，声明 5 个成员变量 $name、$height、$weight、$age 和 $sex。然后定义一个成员方法 playFootball()，用于判断申请的运动员是否适合这个运动项目。最后实例化类，通过实例化返回对象调用指定的方法，根据运动员填写的参数，判断申请的运动员是否符合要求。代码如下：

实例01-1

```
<?php
class SportObject{
    public $name;                                          //定义成员变量
    public $height;                                        //定义成员变量
    public $weight;                                        //定义成员变量

    public function playFootball($name,$height,$weight){   //定义成员方法
        $this->name=$name;
        $this->height=$height;
        $this->weight=$weight;
        if($this->height<185 and $this->weight<85){
            return $this->name.", 符合踢足球的要求!";      //方法实现的功能
        }else{
            return $this->name.", 不符合踢足球的要求!";    //方法实现的功能
        }
    }
}
$sport=new SportObject();                                  //实例化类
echo $sport->playFootball('明日','185','80');              //执行类中的方法
?>
```

运行结果如图 6.4 所示。

图 6.4　实例化类运行结果

📖 **说明：** “$this ->”作用是调用本类中的成员变量或成员方法，这里只要知道含义即可。在 6.2.8 小节中将介绍相关的知识。

⚡ **注意：** 无论是使用“$this->”还是使用“对象名 ->”的格式，后面的变量是没有 $ 符号的，如 $this-> beatBasketBall、$sport-> beatBasketBall。

📊 **练一练：**

（1）试着创建一个商品类 Goods，声明 1 个成员变量 $ids（商品 id 数组）。然后定义一个成员方法 searchGoods()，用于查找某个商品 id 是否存在于商品数组 $id 中。（光盘\Code\Try\06\01）

（2）试着创建一个订单类 Order，定义一个成员方法 total()，用于计算商品总价。（光盘\Code\Try\06\02）

6.2.5 类常量

🎬 **视频讲解：** 光盘\Video\06\6.2.5 类常量.mp4

既然有变量，当然也会有常量。常量就是不会改变的量，是一个恒值。如圆周率就是一个众所周知的常量。定义常量使用关键字 const，如：

```
const PI= 3.14159;
```

例如，先定义一个常量，再定义一个变量，实例化对象后分别输出两个值。代码如下：

```
<?php
class SportObject{
    const BOOK_TYPE = '计算机图书';                    //定义常量BOOK_TYPE
    public $object_name;                               //定义变量，用来存放商品名称
    function setObjectName($name){                     //定义方法setObjectName()
        $this -> object_name = $name;                  //设置成员变量值
    }
    function getObjectName(){                          //定义方法getObjectName()
        return $this -> object_name;
    }
}
$book = new SportObject();                             //实例化对象
$book->setObjectName("PHP类");                         //调用方法setObjectName()
echo SportObject::BOOK_TYPE." -> ";                    //输出常量BOOK_TYPE
echo $book->getObjectName();                           //调用方法getObjectName()
```

```
16  ?>
```

运行结果为：

```
计算机图书 -> PHP类
```

可以发现，常量的输出和变量的输出是不一样的。常量不需要实例化对象，直接由“类名＋常量名”调用即可。常量输出的格式为：

```
类名::常量名
```

📖 **说明**：类名和常量名之间的两个冒号“::”称为作用域操作符，使用这个操作符可以在不创建对象的情况下调用类中的常量、变量和方法。关于作用域操作符，将在6.2.8小节中进行介绍。

6.2.6 构造方法和析构方法

视频讲解：光盘\Video\06\6.2.6 构造方法和析构方法.mp4

1. 构造方法

当一个类实例化一个对象时，可能会随着对象初始化一些成员变量。

📖 **说明**：初始化表示“开始时做好准备”。在软件开发中对某个东西初始化时，就是把它设置成一种我们期望的状态或条件，以备使用。

如实例01中的SportObject类，现在再添加一些成员变量，类的形式如下：

```
01  class SportObject{
02        public $name;                    //定义姓名成员变量
03        public $height;                  //定义身高成员变量
04        public $weight;                  //定义体重成员变量
05        public $age;                     //定义年龄成员变量
06        public $sex;                     //定义性别成员变量
07  }
```

实例化一个SportObject类的对象，并对这个类的一些成员变量赋初值。代码如下：

```
01  $sport=new SportObject('明日','185','80','20','男');    //实例化类，并传递参数
02  $sport ->name="明日 ";                                   //为成员变量赋值
03  $sport ->height=185;                                     //为成员变量赋值
04  $sport ->weight=80;                                      //为成员变量赋值
05  $sport ->age=20;                                         //为成员变量赋值
06  $sport ->sex="男";                                       //为成员变量赋值
07  echo $sport->playFootball();                             //执行方法
```

可以看到，如果赋初值比较多，写起来就比较麻烦。为此，PHP引入了构造方法。构造方法是生成对象时自动执行的成员方法，作用就是初始化对象。该方法可以没有参数，也可以有多个参数。构造方法的格式如下：

```
void __construct([mixed args [,…]])
```

注意：函数中的“__”是两条下划线。

例如，重写 SportObject 类和 playFootBall() 方法，下面通过具体例子查看重写后的对象在使用上有哪些不一样。代码如下：

```
<?php
class SportObject{
    public $name;                                          //定义成员变量
    public $height;                                        //定义成员变量
    public $weight;                                        //定义成员变量
    public $age;                                           //定义成员变量
    public $sex;                                           //定义成员变量
    public function __construct($name,$height,$weight,$age,$sex){ //定义构造方法
        $this->name=$name;                                 //为成员变量赋值
        $this->height=$height;                             //为成员变量赋值
        $this->weight=$weight;                             //为成员变量赋值
        $this->age=$age;                                   //为成员变量赋值
        $this->sex=$sex;                                   //为成员变量赋值
    }
    public function playFootBall(){                        //声明成员方法
        if($this->height<185 and $this->weight<85){
            return $this->name."，符合踢足球的要求!";       //方法实现的功能
        }else{
            return $this->name."，不符合踢足球的要求!";     //方法实现的功能
        }
    }
}
$sport=new SportObject('小明','185','80','20','男');       //实例化类，并传递参数
echo $sport->playFootball();                               //执行类中的方法
```

运行结果为：

```
小明，不符合踢足球的要求!
```

可以看到，重写后的类，在实例化对象时只需一条语句即可完成赋值。

说明：构造方法是初始化对象时使用的。如果类中没有构造方法，那么 PHP 会自动生成。自动生成的构造方法没有任何参数，也没有任何操作。

2. 析构方法

析构方法的作用和构造方法正好相反，是在对象被销毁时调用，作用是释放内存。析构方法的格式为：

```
void __destruct ( void )
```

例如，首先声明一个对象 $Sport，然后再销毁对象。可以看出，使用析构方法十分简单。代码如下：

```
<?php
class SportObject{
    public $name;                                                    //定义姓名成员变量
    public $height;                                                  //定义身高成员变量
    public $weight;                                                  //定义体重成员变量
    public $age;                                                     //定义年龄成员变量
    public $sex;                                                     //定义性别成员变量
    public function __construct($name,$height,$weight,$age,$sex){ //定义构造方法
        $this->name=$name;                                           //为成员变量赋值
        $this->height=$height;                                       //为成员变量赋值
        $this->weight=$weight;                                       //为成员变量赋值
        $this->age=$age;                                             //为成员变量赋值
        $this->sex=$sex;                                             //为成员变量赋值
    }
    public function playFootball(){                                  //声明成员方法
        if($this->height<185 and $this->weight<85){
            return $this->name."，符合踢足球的要求!";                //方法实现的功能
        }else{
            return $this->name."，不符合踢足球的要求!";              //方法实现的功能
        }
    }
    function __destruct(){                                           //析构方法
        echo "<p><b>对象被销毁，调用析构函数。</b></p>";
    }
}
$sport=new SportObject('明日','185','80','20','男');                 //实例化类，并传递参数
?>
```

运行结果为：

```
对象被销毁，调用析构函数。
```

说明：PHP 使用的是一种“垃圾回收”机制，自动清除不再使用的对象，释放内存。也就是说即使不使用 unset() 函数，析构方法也会自动被调用，这里只是明确一下析构方法在何时被调用。一般情况下是不需要手动创建析构方法的。

6.2.7 继承和多态

视频讲解：光盘\Video\06\6.2.7 继承和多态.mp4

继承和多态最根本的作用就是完成代码的重用。下面就来介绍 PHP 的继承和多态。

1. 继承

子类可以继承父类的所有成员变量和方法，包括构造方法。当子类被创建时，PHP 会先在子类中

查找构造方法。如果子类有自己的构造方法，PHP 会先调用子类中的方法。当子类中没有时，PHP 则去调用父类中的构造方法，这就是继承。

例如，在 6.1 节中通过图片展示了一个运动类，在这个运动类中包含很多个方法，代表不同的体育项目，各种体育项目的方法中有公共的属性。例如，姓名、性别、年龄……但还会有许多不同之处，例如，篮球对身高的要求、举重对体重的要求……如果都由一个 SportObject 类来生成各个对象，除了那些公共属性外，其他属性和方法则需自己手动来写，工作效率得不到提高。这时，可以使用面向对象中的继承来解决这个难题。

下面来看如何通过 PHP 中的继承来解决上述问题。继承是通过关键字 extends 来声明的，继承的格式如下：

```
class subClass extends superClass{
…
}
```

说明：subClass 为子类名称，superClass 为父类名称。

实例 02 使用类的继承实现父类方法

实例位置：光盘\Code\SL\06\02
视频位置：光盘\Video\06\

本实例使用 SportObject 类生成了两个子类：PlayBasketBall 和 WeightLifting，两个子类使用不同的构造方法实例化了两个对象 playbasketball 和 weightlifting，并输出信息。代码如下：

实例02-1

```
<?php

/**
 * Class SportObject 运动类（父类）
 */
class SportObject{
    public $name;                                          //定义姓名成员变量
    public $age;                                           //定义年龄成员变量
    public $weight;                                        //定义体重成员变量
    public $sex;                                           //定义性别成员变量
    public function __construct($name,$age,$weight,$sex){ //定义构造方法
        $this->name=$name;                                 //为成员变量赋值
        $this->age=$age;                                   //为成员变量赋值
        $this->weight=$weight;                             //为成员变量赋值
        $this->sex=$sex;                                   //为成员变量赋值
    }
    function showMe(){                                     //在父类中定义方法
        echo '这句话不会显示。';
    }
}

/**
 * Class PlayBasketBall 篮球类（子类）
```

```
24  */
25 class PlayBasketBall extends SportObject{                    //定义子类，继承父类
26     public $height;                                          //定义身高成员变量
27     function __construct($name,$height){                     //定义构造方法
28         $this -> height = $height;                           //为成员变量赋值
29         $this -> name   = $name;                             //为成员变量赋值
30     }
31     function showMe(){                                       //定义方法
32         if($this->height>185){
33             return $this->name."，符合打篮球的要求!";          //方法实现的功能
34         }else{
35             return $this->name."，不符合打篮球的要求!";        //方法实现的功能
36         }
37     }
38 }
39
40 /**
41  * Class WeightLifting 举重类（子类）
42  */
43 class WeightLifting extends SportObject{                     //继承父类
44     function showMe(){                                       //定义方法
45         if($this->weight<85){
46             return $this->name."，符合举重的要求!";           //方法实现的功能
47         }else{
48             return $this->name."，不符合举重的要求!";         //方法实现的功能
49         }
50     }
51 }
52 //实例化对象
53 $Playbasketball = new PlayBasketBall('明日','190');          //实例化子类
54 $weightlifting  = new WeightLifting('科技','185','80','20','男');
55 echo $Playbasketball->showMe()."<br>";                       //输出结果
56 echo $weightlifting->showMe()."<br>";
57 ?>
```

运行结果如图6.5所示。

图6.5 继承父类方法运行结果

练一练：

（1）访问明日学院后台需要具有管理员权限，只有当管理员登录后才允许访问后台管理页面，否

则跳转到后台登录页。创建一个子类 Index，继承父类 Base。在 Base 类中判断用户是否登录。（光盘\Code\Try\06\03）

（2）在网站布局时，通常将头部和底部这些固定的信息写入父类 Base 中，通过继承父类的方式，在父类构造方法中，输出网站的头部和底部。创建一个子类 Index，继承父类 Base，实现该功能。（光盘\Code\Try\06\04）

2. 多态

多态好比有一个成员方法让大家去游泳，这个时候有的人带游泳圈，有的人拿浮板，有的人什么也不带。虽是同一种方法，却产生了不同的形态，这就是多态。

例如，定义一个汽车抽象类 Car，它有一个获取速度的成员方法 getSpeed()。现在有 3 个汽车品牌的子类，分别继承 Car 父类，并且都有一个获取速度的成员方法 getSpeed()。三个不同子类，调用同一个方法，将产生 3 种不同的形态，代码如下：

```php
<?php
/**
 * Class Car 定义抽象类car
 */
abstract class Car {
    abstract function getSpeed();                    //定义抽象方法
}

/**
 * Class Toyota 定义Toyota类，继承Car类
 */
class Toyota extends Car {
    function getSpeed() {
        return "Toyota's speed";
    }
}
/**
 * Class Nissan 定义Nissan类，继承Car类
 */
class Nissan extends Car {
    function getSpeed() {
        return "Nissan's speed";
    }
}
/**
 * Class Tesla 定义Tesla类，继承Car类
 */
class Tesla extends Car {
    function getSpeed() {
        return "Tesla's speed";
    }
```

```
32  }
33
34  $car = new Toyota();                                    //实例化Toyota类
35  $speed = $car->getSpeed();                              //调用getSpeed()方法
36  echo $speed;
37
38  ?>
```

运行结果如下：

```
Toyota'speed
```

6.2.8 “$this ->”和“::”的使用

视频讲解：光盘\Video\06\6.2.8 “$this ->”和“::”的使用.mp4

通过实例02可以发现，子类不仅可以调用自己的变量和方法，也可以调用父类中的变量和方法。

PHP是通过伪变量“$this ->”和作用域操作符“::”来实现这些功能的，这两个符号在前面的学习中都有过简单的介绍。本节将详细讲解两者的使用方法。

1. $this->

在6.2.3小节中，对如何调用成员方法有了简单的介绍，即使用对象名加方法名，格式为“对象名 -> 方法名”。但在定义类时（如SportObject类），根本无法得知对象的名称是什么。这时如果想调用类中的方法，就要用伪变量“$this ->”。“$this”的意思就是本身，所以“$this->”只可以在类的内部使用。

例如，当类被实例化后，“$this”同时被实例化为本类的对象，这时，对“$this”使用get_class()函数将返回本类的类名。代码如下：

```
01  <?php
02  class example{                                          //创建类example
03      function exam(){                                    //创建成员方法
04          if(isset($this)){                               //判断变量$this是否存在
05              echo '$this的值为: '.get_class($this);      //如果存在，输出$this所属类的名字
06          }else{
07              echo '$this未定义';
08          }
09      }
10  }
11  $class_name = new example();                            //实例化对象$class_name
12  $class_name->exam();                                    //调用方法exam()
13  ?>
```

运行结果如下：

```
$this的值为: example
```

📖 **说明：** get_class() 函数返回对象所属类的名字，如果不是对象，则返回 false。

2. 操作符“::”

相比伪变量“$this”只能在类的内部使用，操作符“::”更为强大。操作符“::”可以在没有声明任何实例的情况下访问类中的成员方法或成员变量。使用“::”操作符的通用格式为：

```
关键字::变量名/常量名/方法名
```

这里的关键字分为以下 3 种情况：

- ◆ parent 关键字：可以调用父类中的成员变量、成员方法和常量。
- ◆ self 关键字：可以调用当前类中的静态成员和常量。
- ◆ 类名：可以调用本类中的变量、常量和方法。

例如，依次使用了类名、parent 关键字和 self 关键字来调用变量和方法。读者可以观察输出的结果。代码如下：

```php
<?php
class Book{
    const NAME = 'computer';                          //常量NAME
    function __construct(){                           //构造方法
        echo '本年度图书类冠军为: '.Book::NAME.'<br>';    //输出默认值
    }
}
class BookRank extends Book{                          //Book类的子类
    const NAME = 'foreign language';                  //声明常量
    function __construct(){                           //子类的构造方法
        parent::__construct();                        //调用父类的构造方法
        echo '本月图书类冠军为: '.self::NAME.' ';        //输出本类中的默认值
    }
}
$obj = new BookRank();                                //实例化对象
?>
```

运行结果如下：

```
本年度图书类冠军为: computer
本月图书类冠军为: foreign language
```

📖 **说明：** 关于静态方法（变量）的声明及使用可参考 6.2.10 小节相关内容。

6.2.9 数据隐藏

视频讲解：光盘\Video\06\6.2.9 数据隐藏.mp4

细心的读者看到这里，一定会有一个疑问：面向对象编程的特点之一是封装性，即数据隐藏。但

是在前面的学习中并没有突出这一点。对象中的所有变量和方法可以随意调用，甚至不用实例化也可以使用类中的方法、变量。这就是面向对象吗？

这当然不算是真正的面向对象。如果读者是从本章第一节来开始学习的，一定还会记得在6.2.4小节讲成员变量时所提到的那几个关键字：public、private、protected、static和final，它们是用来限定类成员（包括变量和方法）的访问权限的。本节先来学习前3个。

📖 **说明**：成员变量和成员方法在关键字的使用上都是一样的。这里只以成员变量为例说明几种关键字的不同用法。对于成员方法同样适用。

1. public（公共成员）

public，顾名思义，就是可以公开的、没有必要隐藏的数据信息。可以在程序中的任何位置（类内、类外）被其他的类和对象调用。子类可以继承和使用父类中所有的公共成员。

在本章的前半部分，所有的变量都被声明为public，而所有的方法在默认状态下也是public。所以对变量和方法的调用显得十分混乱。为了解决这个问题，就需要使用第二个关键字private。

2. private（私有成员）

被private关键字修饰的变量和方法，只能在所属类的内部被调用和修改，不可以在类外被访问。在子类中也不可以。

例如，对私有变量$name的修改与访问，只能通过调用成员方法来实现。如果直接调用私有变量，将会发生错误。代码如下：

```php
<?php
class Book{
    private $name = 'computer';                    //声明私有变量$name
    public function setName($name){                //设置公有变量方法
        $this -> name = $name;
    }
    public function getName(){                     //读取私有变量方法
        return $this -> name;
    }
}
class LBook extends Book{                          //Book类的子类
}
$lbook = new LBook();                              //实例化对象
echo '正确操作私有变量的方法：';                    //正确操作私有变量
$lbook -> setName("PHP5从入门到应用开发");
echo $lbook -> getName();
echo '<br>直接操作私有变量的结果：';                //错误操作私有变量
echo Book::$name;
?>
```

运行结果如图6.6所示。

图 6.6　private 关键字的使用

📖 **说明**：对于成员方法，如果没有写关键字，那么默认就是 public。从本节开始，以后所有的方法及变量都会带上关键字，这是一种良好的代码编写习惯。

3. protected（保护成员）

private 关键字可以将数据完全隐藏起来，除了在本类外，其他地方都不可以调用，子类也不可以。有些变量希望子类能够调用，但对另外的类来说，还要做到封装。这时，就可以使用 protected 关键字。

📖 **说明**：被 protected 修饰的类成员，可以在本类和子类中被调用，其他地方则不可以被调用。

例如，声明一个 protected 变量，然后使用子类中的方法调用一次，最后在类外直接调用一次，观察一下运行结果。代码如下：

```
<?php
class Book{
    protected $name = 'computer';                          //声明保护变量$name
}
class LBook extends Book{                                  //Book类的子类
    public function showMe(){
        echo '对于protected修饰的变量，在子类中是可以直接调用的。如：$name = '.$this -> name;
    }
}
$lbook = new LBook();                                      //实例化对象
$lbook->showMe();
echo '<p>但在其他的地方是不可以调用的，否则：';              //对私有变量进行操作
$lbook->name = 'history';
?>
```

运行结果如图 6.7 所示。

图 6.7　protected 关键字运行结果

📖 **说明**：虽然 PHP 中没有对修饰变量的关键字做强制性的规定和要求，但从面向对象的特征和设计方面考虑，一般使用 private 或 protected 关键字来修饰变量，以防止变量在类外被直接修改和调用。

6.2.10 静态变量（方法）

视频讲解：光盘\Video\06\6.2.10 静态变量（方法）.mp4

不是所有的变量（方法）都需要通过创建对象来调用。可以通过给变量（方法）加上 static 关键字来直接调用。调用静态成员的格式为：

```
关键字::静态成员
```

关键字可以是：

◆ self，在类内部调用静态成员时所使用。

◆ 静态成员所在的类名，在类外调用类内部的静态成员时所用。

注意：在静态方法中，只能调用静态变量，而不能调用普通变量，普通方法则可以调用静态变量。

使用静态成员，除了可以不需要实例化对象，另一个作用就是在对象被销毁后，仍然可以保存被修改的静态数据，以便下次继续使用。这个概念比较抽象，下面结合一个例子进行说明。

首先声明一个静态变量 $num，声明一个方法，在方法的内部调用静态变量，然后给变量加 1。依次实例化这个类的两个对象，并输出方法。可以发现两个对象中的方法返回的结果有了一些联系。直接使用类名输出静态变量，看有什么效果。代码如下：

```
<?php
class Book{                                          //Book类
    static $num = 0;                                 //声明一个静态变量$num，初值为0
    public function showMe(){                        //声明一个方法
        echo '您是第'.self::$num.'位访客';            //输出静态变量
        self::$num++;                                //将静态变量加1
    }
}
$book1 = new Book();                                 //实例化对象$book1
$book1->showMe();                                    //调用对象$book1的showMe()方法
echo "<br>";
$book2 = new Book();                                 //实例化对象$book2;
$book2->showMe();                                    //调用对象$book2的showMe()方法
echo "<br>";
echo '您是第'.Book::$num.'为访客';                    //直接使用类名调用静态变量
?>
```

运行结果如下：

```
您是第0位访客
您是第1位访客
您是第2为访客
```

如果将程序代码中的静态变量改为普通变量，如“private $num = 0;”，那么结果就不一样了。读者可以动手试一试。

📖 **说明：**静态成员不用实例化对象，当类第一次被加载时就已经分配了内存空间，所以直接调用静态成员的速度要快一些。但如果静态成员声明得过多，空间一直被占用，反而会影响系统的功能。这个尺度只能通过实践积累，才能真正地掌握。

6.3 PHP 对象的高级应用

通过 6.2 节的学习，相信读者已经初步了解了 PHP 对象。下面来学习一些面向对象的高级应用。

6.3.1 final 关键字

视频讲解：光盘\Video\06\6.3.1 final关键字.mp4

final，中文含义是最终的、最后的。被 final 修饰过的类和方法就是“最终的版本”。

如果有一个类的格式为：

```
final class class_name{
//…
}
```

说明该类不可以再被继承，也不能再有子类。

如果有一个方法的格式为：

```
final function method_name()
```

说明该方法在子类中不可以进行重写，也不可以被覆盖。

例如，为 SportObject 类设置关键字 final，并生成一个子类 MyBook。运行后，可以看到程序报错，无法执行。代码如下：

```
<?php
final class SportObject{                          //final类SportObject
    function __construct(){                       //构造方法
        echo 'initialize object';
    }
}
class MyBook extends SportObject{                 //创建SportObject的子类Mybook
    static function exam(){                       //子类中的方法
        echo "You can't see me";
    }
}
MyBook::exam();                                   //调用子类方法
?>
```

结果如图 6.8 所示。

图 6.8 继承 final 类的错误提示

6.3.2 抽象类

视频讲解：光盘\Video\06\6.3.2 抽象类.mp4

抽象类是一种不能被实例化的类，只能作为其他类的父类来使用。抽象类使用 abstract 关键字来声明，格式为：

```
abstract class AbstractName{
…
}
```

抽象类和普通类相似，包含成员变量、成员方法。两者的区别在于，抽象类至少要包含一个抽象方法。抽象方法没有方法体，其功能的实现只能在子类中完成。抽象方法也是使用 abstract 关键字来修饰的，它的格式为：

```
abstract function abstractName();
```

注意： *在抽象方法后面要有分号“;”。*

抽象类和抽象方法主要应用于复杂的层次关系中，这种层次关系要求每一个子类都包含并重写某些特定的方法。举一个例子，中国的美食是多种多样的，有鲁菜、川菜、粤菜等。每种菜系使用的都是煎、炒、烹、炸等手法，只是在具体的步骤上各有各的不同。如果把中国美食当作一个大类 Cate，下面的各大菜系就是 Cate 的子类，而煎、炒、烹、炸则是每个类中都有的方法。每个方法在子类中的实现都是不同的，在父类中无法规定。为了统一规范，不同子类的方法要有一个相同的方法名：decoct（煎）、stir_fry（炒）、cook（烹）、fry（炸）。

下面实现一个商品抽象类 CommodityObject，该抽象类包含一个抽象方法 service()。为抽象类生成两个子类 MyBook 和 MyComputer，分别在两个子类中实现抽象方法。最后实例化两个对象，调用实现后的抽象方法，输出结果。代码如下：

```
<?php
abstract class CommodityObject{                                    //定义抽象类
    abstract function service($getName,$price,$num);               //定义抽象方法
}
class MyBook extends CommodityObject{                             //定义子类，继承抽象类
    function service($getName,$price,$num){                        //定义方法
        echo '您购买的商品是'.$getName.'，该商品的价格是：'.$price.'元。';
        echo '您购买的数量为：'.$num.'本。';
```

```
09            echo '如发现缺页损坏，请在3日内更换。';
10        }
11    }
12    class MyComputer extends CommodityObject{                    //定义子类继承父类
13        function service($getName,$price,$num){                  //定义方法
14            echo '您购买的商品是'.$getName.'，该商品的价格是：'.$price.' 元。';
15            echo '您购买的数量为：'.$num.' 台。';
16            echo '如发生非人为质量问题，请在3个月内更换。';
17        }
18    }
19    $book = new MyBook();                                        //实例化子类
20    $computer = new MyComputer();                                //实例化子类
21    $book->service('《零基础学PHP》',85,3);                       //调用方法
22    echo '<br>';
23    $computer->service('苹果笔记本',8500,1);                      //调用方法
24    ?>
```

运行结果如下：

```
您购买的商品是《零基础学PHP》，该商品的价格是：85 元。您购买的数量为：3 本。如发现缺页损坏，请在3日内更换。
您购买的商品是苹果笔记本，该商品的价格是：8500 元。您购买的数量为：1 台。如发生非人为质量问题，请在3
个月内更换。
```

6.3.3 接口的使用

视频讲解：光盘\Video\06\6.3.3 接口的使用.mp4

继承特性简化了对象、类的创建，增强了代码的可重用性。但 PHP 只支持单继承。如果想实现多重继承，就要使用接口类。

接口类通过 interface 关键字来声明，并且类中只能包含未实现的方法和一些成员变量，格式如下：

```
interface InterfaceName{
    function interfaceName1();
    function interfaceName2();
    …
}
```

注意：不要用 public 以外的关键字来修饰接口中的类成员，对于方法，不写关键字也可以。这是由接口类自身的属性决定的。

子类是通过 implements 关键字来实现接口的，如果要实现多个接口，那么每个接口之间应使用逗号“,”连接。而且所有未实现的方法需要在子类中全部实现，否则将会出现错误。格式如下：

```
class SubClass implements InterfaceName1, InterfaceName2{
    function interfaceName1(){
        //功能实现
```

```
}
function interfaceName2(){
        //功能实现
}
        …
}
```

例如，先声明了两个接口 MPopedom 和 MPurview，接着声明了两个类 Member 和 Manager，其中 Member 类继承了 MPopedom 接口；Manager 继承了 MPopedom 和 MPurview 接口。分别实现各自的成员方法后，实例化两个对象 $member 和 $manager。最后调用实现后的方法。实例代码如下：

```
<?php
/**
 * 声明接口MPopedom
 */
interface MPopedom{
    function popedom();
}
/**
 * 声明接口 MPurview
 */
interface MPurview{
    function purview();
}

/**
 * 创建子类Member，实现一个接口MPurview
 */
class Member implements MPurview{
    function purview(){
        echo '会员拥有的权限。';
    }
}
/**
 * 创建子类Manager，实现多个接口MPurview和MPopedom
 */
class Manager implements MPurview,MPopedom{
    function purview(){
        echo '管理员拥有会员的全部权限。';
    }
    function popedom(){
        echo '管理员还有会员没有的权限。';
    }
}
$member = new Member();                    //类Member实例化
```

```
35  $manager = new Manager();                    //类Manager实例化
36  $member->purview();                          //调用$member对象的purview()方法
37  echo '<p>';
38  $manager->purview();                         //调用$manager对象的purview()方法
39  $manager->popedom();                         //调用$manager对象的popedom()方法
40  ?>
```

运行结果如下：

```
会员拥有的权限。
管理员拥有会员的全部权限。管理员还有会员没有的权限。
```

通过上面的例子可以发现，抽象类和接口实现的功能十分相似。抽象类的优点是可以在抽象类中实现公共的方法，而接口则可以实现多重继承。至于何时使用抽象类和接口要看具体情况。

6.3.4 对象类型检测

视频讲解：光盘\Video\06\6.3.4 对象类型检测.mp4

instanceof 操作符可以检测当前对象是属于哪个类。一般格式为：

```
ObjectName instanceof ClassName
```

例如，首先创建两个类，一个基类（SportObject）与一个子类（MyBook）。实例化一个子类对象，判断对象是否属于该子类，再判断对象是否属于基类。代码如下：

```
01  <?php
02  class SportObject{}                          //创建空类SportObject
03  class MyBook extends SportObject{            //创建子类MyBook
04      private $type;
05  }
06  $cBook = new MyBook();                       //实例化对象$cBook
07  if($cBook instanceof MyBook)                 //判断对象是否属于类MyBook
08      echo '对象$cBook属于MyBook类<br>';
09  if($cBook instanceof SportObject)            //判断对象是否属于类SportObject
10      echo '对象$cBook属于SportObject类<br>';
11  ?>
```

运行结果如下：

```
对象$cBook属于MyBook类
对象$cBook属于SportObject类
```

6.3.5 魔术方法 (__)

视频讲解：光盘\Video\06\6.3.5 魔术方法(__).mp4

PHP 中有很多以两个下划线开头的方法，如已经介绍过的 __construct() 方法和 __destruct()__clone() 方法，

这些方法被称为魔术方法。当然不是它们真的会魔术，而是指在创建类时 PHP 自动包含的一些方法。本节中将会学习到一些其他魔术方法。

注意： PHP 中保留了所有以“__”开头的方法，所以只能使用 PHP 文档中已有的方法，不要自己创建。

1. __set() 和 __get() 方法

这两个魔术方法的作用分别为：

◆ 当程序试图写入一个未定义或不可见的成员变量时，PHP 就会执行 __set() 方法。__set() 方法包含两个参数，分别表示变量名称和变量值，两个参数不可省略。

◆ 当程序调用一个未定义或不可见的成员变量时，PHP 就会执行 __get() 方法来读取变量值。__get() 方法有一个参数，表示要调用的变量名。

例如，声明一个类 Student，在类中创建私有属性和公共属性以及两个魔术方法 __set()、__get()，然后实例化一个对象 $s，分别调用私有属性和公有属性，最后分别赋值，对比输出结果。代码如下：

```php
<?php
class Student{
    private $a;                          //定义私有属性$a
    private $b = 0;                      //定义私有属性$b
    public $c;                           //定义公有属性$c
    public $d = 0;                       //定义公有属性$d

    public function __get($name) {
        return 123;
    }

    public function __set($name, $value) {
        echo "This is set function";
    }
}

$s = new Student();
var_dump($s->a);                         //输出: 123
var_dump($s->b);                         //输出: 123
var_dump($s->c);                         //输出: null
var_dump($s->d);                         //输出: 0
var_dump($s->e);                         //输出: 123
$s->a = 3;                               //输出: This is set function
$s->c = 3;                               //没有输出
$s->f = 3;                               //输出: This is set function
?>
```

上述代码中，公有变量 $c 和 $d 可以直接调用和赋值。而私有变量 $a 和 $b 只能在 Student 类内部使用。当在类外部调用时，程序会执行 __get() 魔术方法，即返回“123”。当在类外部为私有变量 $a、

$b 赋值时，调用 __set() 魔术方法，即输出“This is set function”。对于未定义的属性 $e 和 $f，与私有变量方式相同。运行结果如下：

```
int(123)
int(123)
NULL
int(0)
int(123)
This is set functionThis is set function
```

注意：魔术方法均用 public 关键字修饰。

2. __call() 方法

__call() 方法的作用是：当程序试图调用不存在或不可见的成员方法时，PHP 会先调用 __call() 方法来存储方法名及其参数。__call() 方法包含两个参数，即方法名和方法参数。其中，方法参数是以数组形式存在的。

例如，声明一个类 SportObject，类中包含两个方法，即 myDream() 和 __call()。实例化对象 $exem 需调用两个方法，一个是类中存在的 myDream() 方法，一个是不存在的 mDream() 方法。代码如下：

```
<?php
class SportObject{
    public function myDream(){                          //myDream()方法
        echo '调用的方法存在，直接执行此方法。<br>';
    }
    public function __call($method, $parameter) {       //__call()方法
        echo '方法不存在，则执行__call()方法。<br>';
        echo '方法名为：'.$method.'<br>';                //输出第一个参数，即方法名
        echo '参数有：';
        echo "<pre>";
        print_r($parameter);                            //输出第二个参数，是一个参数数组
    }
}
$exam = new SportObject();                              //实例化对象$exam
$exam->myDream();                                       //调用存在的方法myDream()
$exam->mDream('how','what','why');                      //调用不存在的方法mDream()
?>
```

运行结果如下：

```
调用的方法存在，直接执行此方法。
方法不存在，则执行__call()方法。
方法名为：mDream
参数有：
Array
```

```
(
    [0] => how
    [1] => what
    [2] => why
)
```


3. __toString() 方法

魔术方法 __toString() 的作用是：当使用 echo 或 print 输出对象时，将对象转化为字符串。

例如，输出类 SportObject 的对象 $myComputer，输出的内容为 __toString() 方法返回的内容。代码如下：

```
01 <?php
02 class SportObject{                          //类SportObject
03     private $type = 'DIY';                   //声明私有变量$type
04     public function __toString(){           //声明__toString()方法
05         return $this -> type;                //方法返回私有变量$type的值
06     }
07 }
08 $myComputer = new SportObject();            //实例化对象$myComputer
09 echo '对象$myComputer的值为: ';
10 echo $myComputer;                           //输出对象$myComputer
11 ?>
```

运行结果如下：

```
对象$myComputer的值为: DIY
```

注意：如果没有 __toString() 方法，直接输出对象将会发生“致命错误”（fatal error）。输出对象时应注意，echo 或 print 函数后面直接跟要输出的对象，中间不要加多余的字符，否则 __toString() 方法不会被执行。如 echo '字串 '.$myComputer、echo ''.$myComputer 等都不可以，一定要注意。

4. __autoload() 方法

通常使用 include() 函数或 require() 函数在一个 PHP 文件中引入类文件。如在 index.php 文件中引入类 A，代码如下：

```
01 <?php
02         require('A.php');                  //引入类A
03         $a = new A();                      //实例化类A
04 ?>
```

但是，多数情况下程序中需要引进很多的类，难道需要一个个的引入吗？

PHP 5 解决了这个问题，__autoload() 方法可以自动实例化需要使用的类。当程序要用到一个类，但该类还没有被实例化时，PHP 将使用 __autoload() 方法，在指定的路径下自动查找和该类名称相同的文件。如果找到，程序则继续执行；否则，报告错误。

实例 03 使用 _autoload() 方法实现自动加载

实例位置：光盘\Code\SL\06\03
视频位置：光盘\Video\06\

本实例将创建 2 个类文件 StudyObject.php 和 SportObject.php，以及 1 个 index.php 文件，然后使用 __autoload() 方法实现自动加载。StudyObject.php 文件具体代码如下：

实例03-1

```
<?php
class StudyObject{                          //声明类StudyObject
    private $cont;                          //声明私有变量$cont
    public function __get($name){           //创建__get()方法
        return "江山代有才人出，各领风骚数百年";
    }
}
?>
```

SportObject.php 文件具体代码如下：

实例03-2

```
<?php
class SportObject{                          //声明类SportObject
    private $cont;                          //声明私有变量$cont
    public function __construct($cont){     //创建构造方法
        $this -> cont = $cont;
    }
    public function __toString(){           //创建__toString()方法
        return $this -> cont;
    }
}
?>
```

在 index.php 文件中使用 __autoload() 方法加载 StudyObject.php 和 SportObject.php，具体代码如下：

实例03-3

```
<?php
function __autoload($class_name){           //创建__autoload()方法
    $class_path = $class_name.'.php';       //类文件路径
    if(file_exists($class_path)){           //判断类文件是否存在
        include_once($class_path);          //动态包含类文件
    }else
        echo '类路径错误。';
}

$myBook = new StudyObject();                //实例化对象
echo $myBook->cont;                         //输出类内容
echo "<br>";
$string  = "德智体美劳，全面发展";
$mySport = new SportObject($string);        //实例化对象
echo $mySport;                              //输出类内容
?>
```

运行 index.php 文件，输出结果如图 6.9 所示。

图 6.9 __autoload() 方法自动加载运行结果

练一练：

（1）PHP 5.1.2 版本引入了 spl_autoload_register() 函数，其作用是注册给定的函数作为 __autoload() 方法的实现。试着创建一个 Autoload.php 文件，使用 version_compare() 函数来判断 PHP 版本，如果版本号大于 5.1.2，则使用 spl_autoload_register() 函数自动加载，否则使用 __autoload() 魔术方法加载。（光盘\Code\Try\06\05）

（2）模拟明日学院短信发送系统，创建 autoload.php 文件，自动加载 MrSoft\Notifier\SMS 类。运行 autoload.php 文件，效果如下。（光盘\Code\Try\06\06）

```
Message: 明日学院开课了 to 小明
```

6.4 面向对象的应用

视频讲解：光盘\Video\06\6.4 面向对象的应用.mp4

本节将实现理论与实践的结合，将面向对象技术应用到实际的程序开发中。

在网站开发过程中，经常会使用分页功能。分页包括设置每页显示的数量、总页数、当前所在页码、支持前一页、后一页跳转等功能。由于分页功能是常用功能，可以将其封装为一个类，方便在任何时候使用该功能。

实例 04 使用分页类实现分页功能

实例位置：光盘\Code\SL\06\04

视频位置：光盘\Video\06\

使用开源分页类 php-paginator（项目地址：https://github.com/jasongrimes/php-paginator），调用分页类方法，实现分页功能。代码如下：

实例04-1

```
<?php
//定义数组数据
$data = array(
    array('id'=>1,'title'=>'西班牙留学面签常见问答11 ') ,
    array('id'=>2,'title'=>'法国综合性大学怎么样'),
    array('id'=>3,'title'=>'香港研究生留学申请时间表'),
    array('id'=>4,'title'=>'英国留学找工作：职位获得途径有哪些？'),
    array('id'=>5,'title'=>'加拿大国际教育委员会（CBIE）发布最新加拿大国际教育数据报告'),
```

```
09      array('id'=>6,'title'=>'日本留学名校申请需要满足哪些条件'),
10      array('id'=>7,'title'=>'韩国留学签证类型及其周期时间'),
11      array('id'=>8,'title'=>'西班牙留学不同阶段的费用'),
12      array('id'=>9,'title'=>'法国留学住宿、饮食、交通、保险费用明细'),
13      array('id'=>10,'title'=>'香港优才计划详解') ,
14      array('id'=>11,'title'=>'去德国留学语言障碍大吗？'),
15      array('id'=>12,'title'=>'美术类高中生如何选择意大利美院'),
16      array('id'=>13,'title'=>'2016英国旅游签证新政策'),
17      array('id'=>14,'title'=>'性价比皇冠大学－西门菲莎大学2016/2017年度风采依旧 '),
18      array('id'=>15,'title'=>'转学到美国什么时间最合适？'),
19      array('id'=>16,'title'=>'申请季，加拿大留学专业解析之【工程学科篇】 ') ,
20      array('id'=>17,'title'=>'日本高中留学的八大优势 ') ,
21      array('id'=>18,'title'=>'韩国留学申请如何做好准备工作 ') ,
22      array('id'=>19,'title'=>'法国留学拒签原因全面分析 ') ,
23      array('id'=>20,'title'=>'澳洲留学生打工攻略') ,
24      array('id'=>21,'title'=>'2016美国留学生打工攻略')
25  );
26
27  $totalItems = count($data);                                    //获取总数
28  $itemsPerPage = 5;                                             //每页显示条数
29  $currentPage = isset($_GET['page']) ? $_GET['page'] : 1;       //获取当前页码
30  $urlPattern  = 'index.php?page=(:num)';                        //拼接url
31  include("Paginator.php");                                      //引入分页类
32  //实例化分页类
33  $paginator = new Paginator($totalItems, $itemsPerPage, $currentPage, $urlPattern);
34  //使用array_splice()函数查找元素位置
35  $getData = array_splice($data,($currentPage-1)*$itemsPerPage,$itemsPerPage);
36  ?>
37
38  <html>
39  <head>
40      <!-- 引入Bootstrap 前端UI框架-->
41      <link rel="stylesheet" href="css/bootstrap.css">
42  </head>
43  <body>
44  <div class="container bg-info">
45      <h3>
46          留学新闻列表
47      </h3>
48      <ul>
49          <!—使用foreach遍历数组-->
50          <?php foreach($getData as $vo){ ?>
51              <li>
52                  <?php  echo $vo['title'] ?>
53              </li>
54          <?php } ?>
55      </ul>
```

```
56      <?php
57      //输出分页
58      echo $paginator;
59      ?>
60  </div>
61  </body>
62  </html>
```

上述代码中，首先定义了一个二维数组，然后使用 count() 函数获取数组总数，使用 array_splice() 函数从数组中筛选数据，注意数组小标从 0 开始计数，所以第二个参数为当前页数减 1。最后引入分页类，实例化该类，并传递相应的参数。运行效果如图 6.10 所示。

图 6.10 分页效果图

练一练：

（1）试着导入 Carbon 类（一个简单的日期时间 API 扩展），项目地址为“https://github.com/briannesbitt/Carbon”。使用该类输出“当前时间”“今天日期”“昨天日期”“明天日期”。（光盘\Code\Try\06\07）

（2）试着创建 ValidateCode 类，用于生成图片验证码，运行结果如图 6.11 所示。（光盘\Code\Try\06\08）

图 6.11 生成图片验证码效果图

6.5 难点解答

6.5.1 类和对象的关系

类的实例化结果就是对象，而对一类对象的抽象就是类。类描述了一组有相同特性（属性）和相同行为（方法）的对象。类和对象的关系就像模具和月饼的关系。用一个写着“五仁月饼”的模具，

能够做出一批五仁月饼，它们具有相同的属性，比如，月饼上都写着“五仁月饼”，这个模具就相当于类，月饼即是对象。

6.5.2 方法与函数的区别

方法就是包含在对象中的函数，函数能做到的，方法都能做到，包括传递参数和返回值。不同之处在于，方法是被对象调用，而函数可以在任何地方被调用。

6.6 小　结

本章主要介绍了面向对象的概念、特点和面向对象的应用。虽然本章关于 OOP（面向对象）概念介绍得很全面、很详细，但要想真正明白面向对象思想，必须要多动手实践、多动脑思考、注意平时积累等。希望读者通过自己的努力，能有所突破。

6.7 动手纠错

1．运行“光盘\Code\Debug\06\01”文件夹下的程序，出现“Fatal error”错误提示，如图 6.12 所示。请根据注释改正程序。

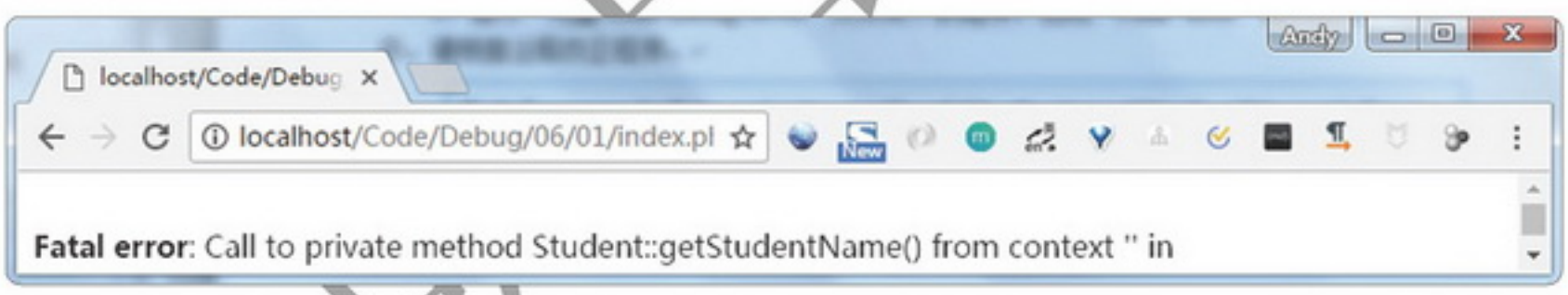


图 6.12　调用私有方法错误提示

2．运行“光盘\Code\Debug\06\02”文件夹下的程序，出现“Fatal error”错误提示，如图 6.13 所示。请根据注释改正程序。

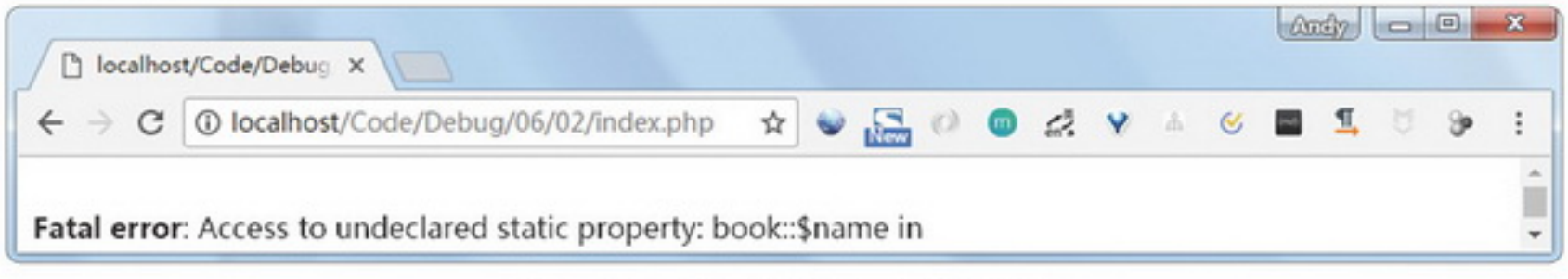


图 6.13　没有声明静态属性错误提示

3．运行“光盘\Code\Debug\06\03”文件夹下的程序，父类构造方法被子类覆盖，请根据注释改正程序，改正后运行结果如图 6.14 所示。

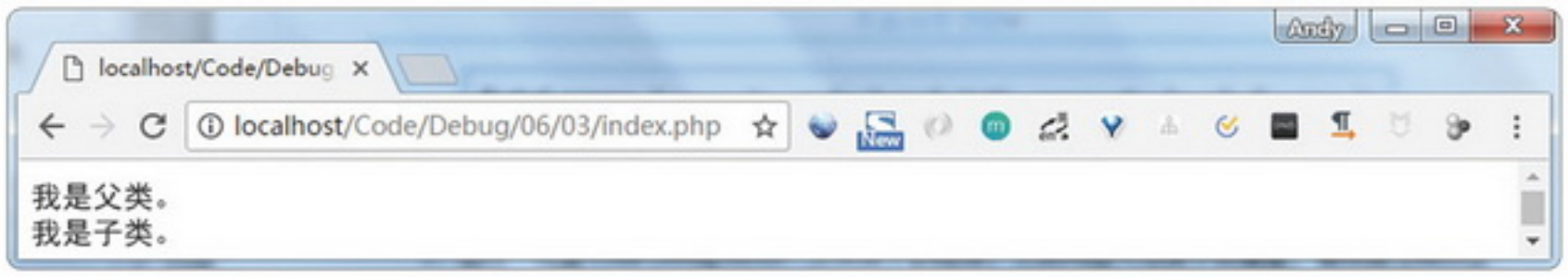


图 6.14 输出父类内容

4．运行“光盘\Code\Debug\06\04”文件夹下的程序，出现“Undefined variable: name”的错误提示，如图 6.15 所示。请根据注释改正程序。

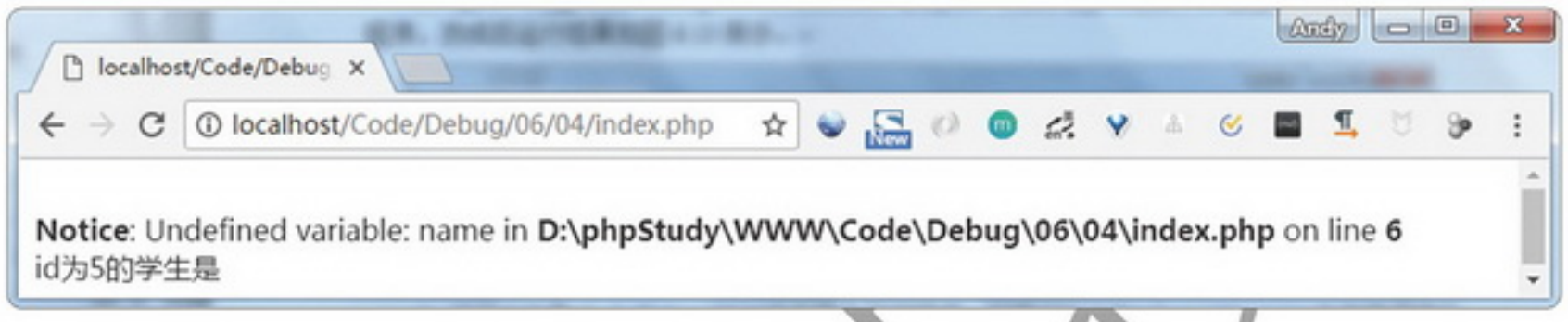


图 6.15 变量名不存在错误提示

5．运行“光盘\Code\Debug\06\05”文件夹下的程序，出现“Fatal error”的错误提示，如图 6.16 请根据注释改正程序。

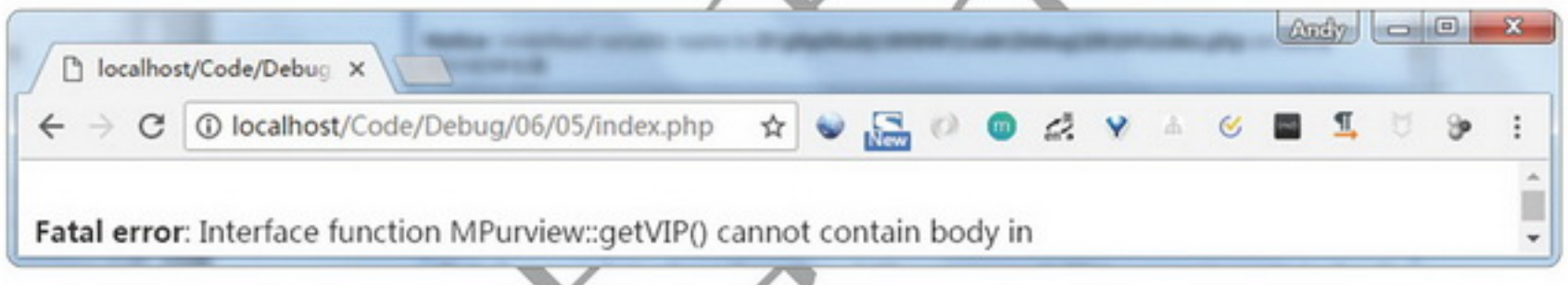
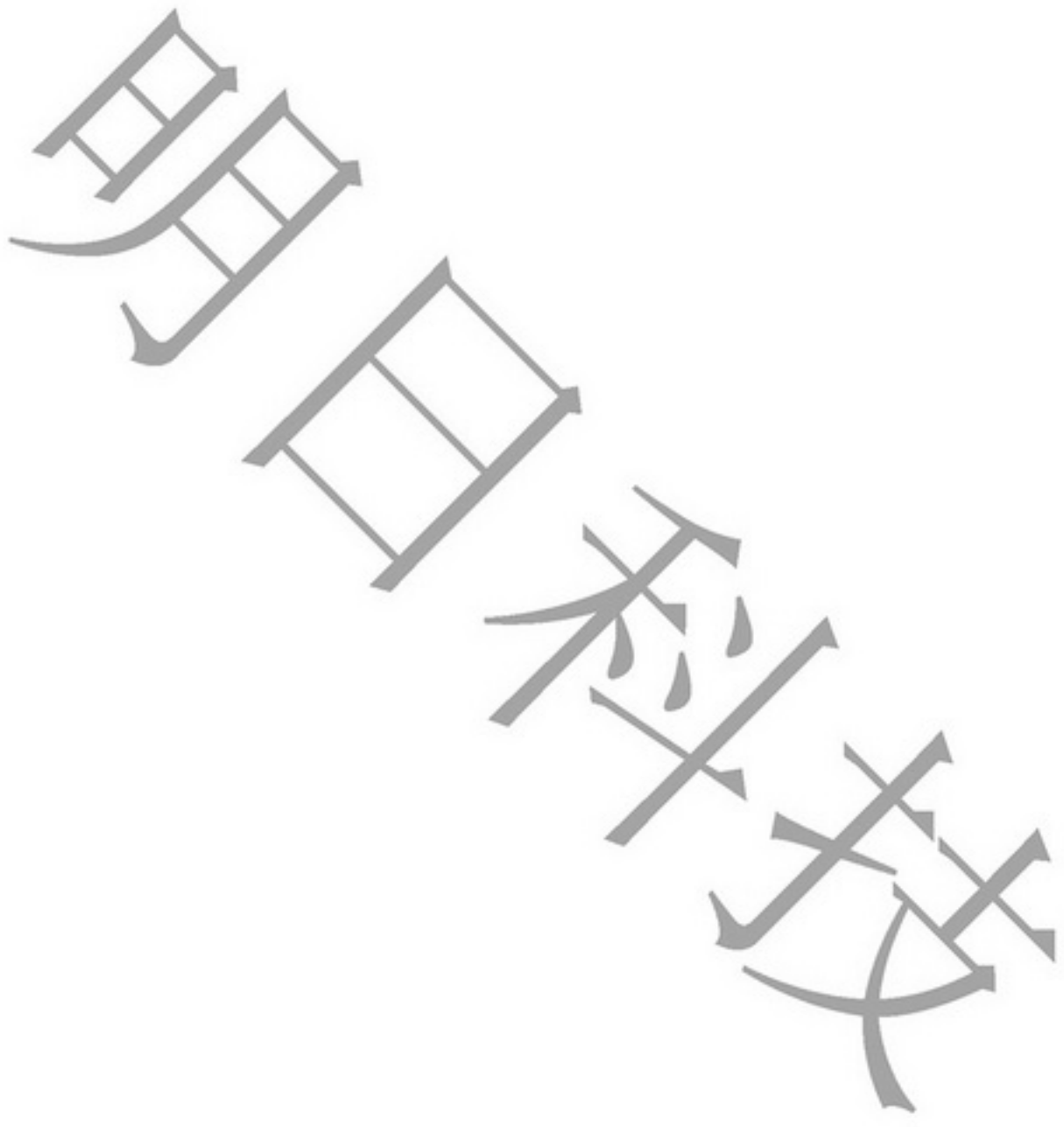


图 6.16 接口方法定义错误提示



第 7 章

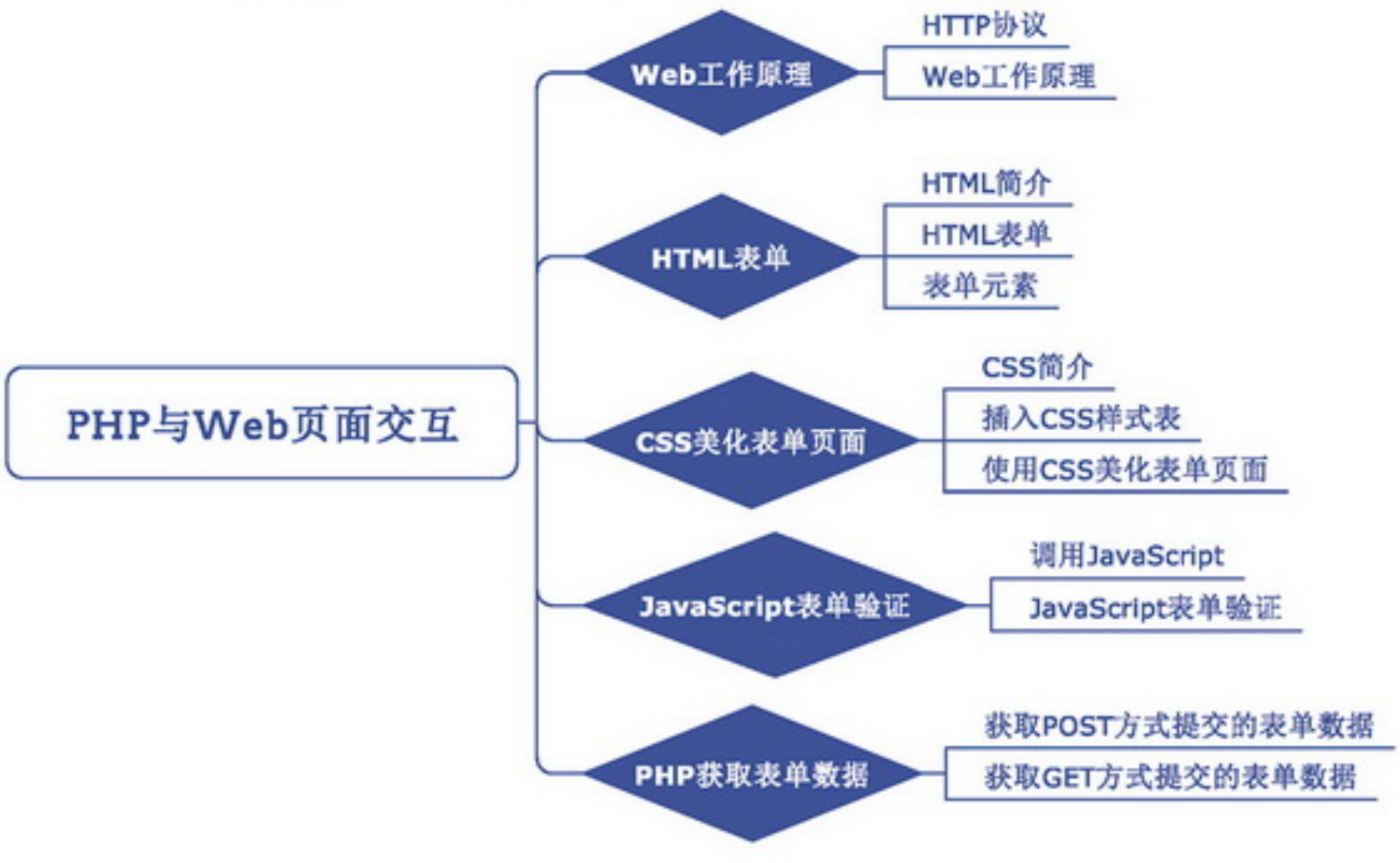
PHP 与 Web 页面交互

（视频讲解：1 小时）

本章概览

当用户浏览网页时，会打开浏览器，输入网址后按下 <Enter> 键，然后浏览器中就会显示出想要浏览的内容。那么，在 PHP 编程中是如何实现 PHP 与 Web 页面交互的呢？在 PHP 中提供了两种与 Web 页面交互的方法，一种是通过 Web 表单提交数据，另一种是通过 URL 参数传递。本章详细讲解了 HTML 表单、CSS 美化表单页面、JavaScript 表单验证等 PHP 与 Web 页面交互的相关知识，可以为以后学习 Web 开发做好铺垫。

知识框架



7.1 Web 工作原理

当用户浏览网页时，会打开浏览器，输入网址后按下回车键，然后浏览器中就会显示出想要浏览的内容。在这个看似简单的用户行为背后，到底隐藏了些什么呢？

7.1.1 HTTP 协议

超文本传输协议（HTTP，HyperText Transfer Protocol）是互联网上应用最为广泛的一种网络协议。所有的 WWW 文件都必须遵守这个标准。设计 HTTP 最初的目的是为了提供一种发布和接收 HTML 页面的方法。

HTTP 是一个客户端和服务器端请求和应答的标准（TCP）。客户端指的是终端用户，服务器端指的是服务器上的网站。通过使用 Web 浏览器、网络爬虫或者其他的工具，客户端发起一个到服务器上指定端口（默认端口为 80）的 HTTP 请求，基本原理如图 7.1 所示。

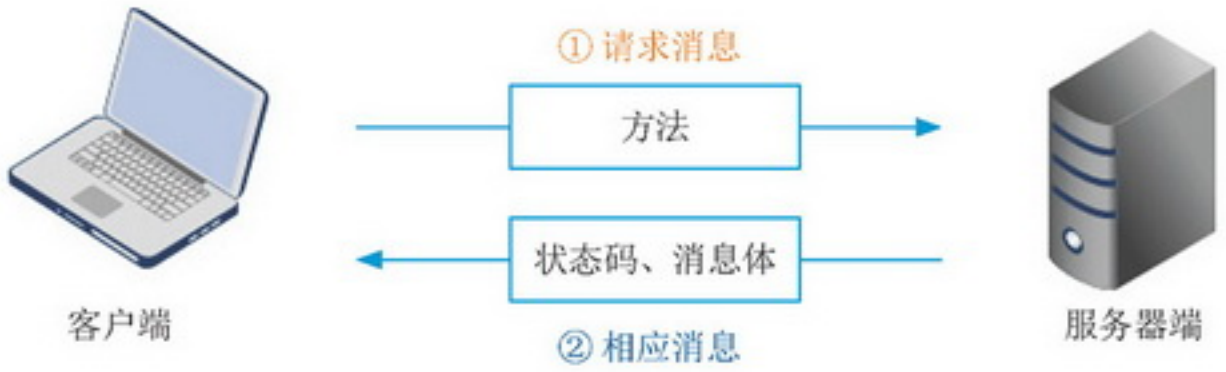


图 7.1 HTTP 基本原理

在客户端向服务器端发起请求时，常用的请求方法如表 7.1 所示。

表 7.1 HTTP 协议的常用请求方法

方　法	描　述
GET	请求指定的页面信息，并返回实体主体
POST	向指定资源提交数据进行处理请求（例如提交表单或者上传文件）。数据被包含在请求体中。POST 请求可能会导致新的资源的建立或已有资源的修改
HEAD	类似于 GET 请求，只不过返回的响应中没有具体的内容，用于获取报头
PUT	从客户端向服务器传送的数据取代指定的文档内容
DELETE	请求服务器删除指定的页面
OPTIONS	允许客户端查看服务器的性能

在 PHP 与 Web 页面交互时，最常用的就是 GET 和 POST 两种方式。

7.1.2 Web 工作原理

遵循 HTTP 协议，就可以向服务器发送请求，并接收消息。这中间又经历了哪些过程呢？ Web 工作原理可以简化为 8 个步骤，如图 7.2 所示。

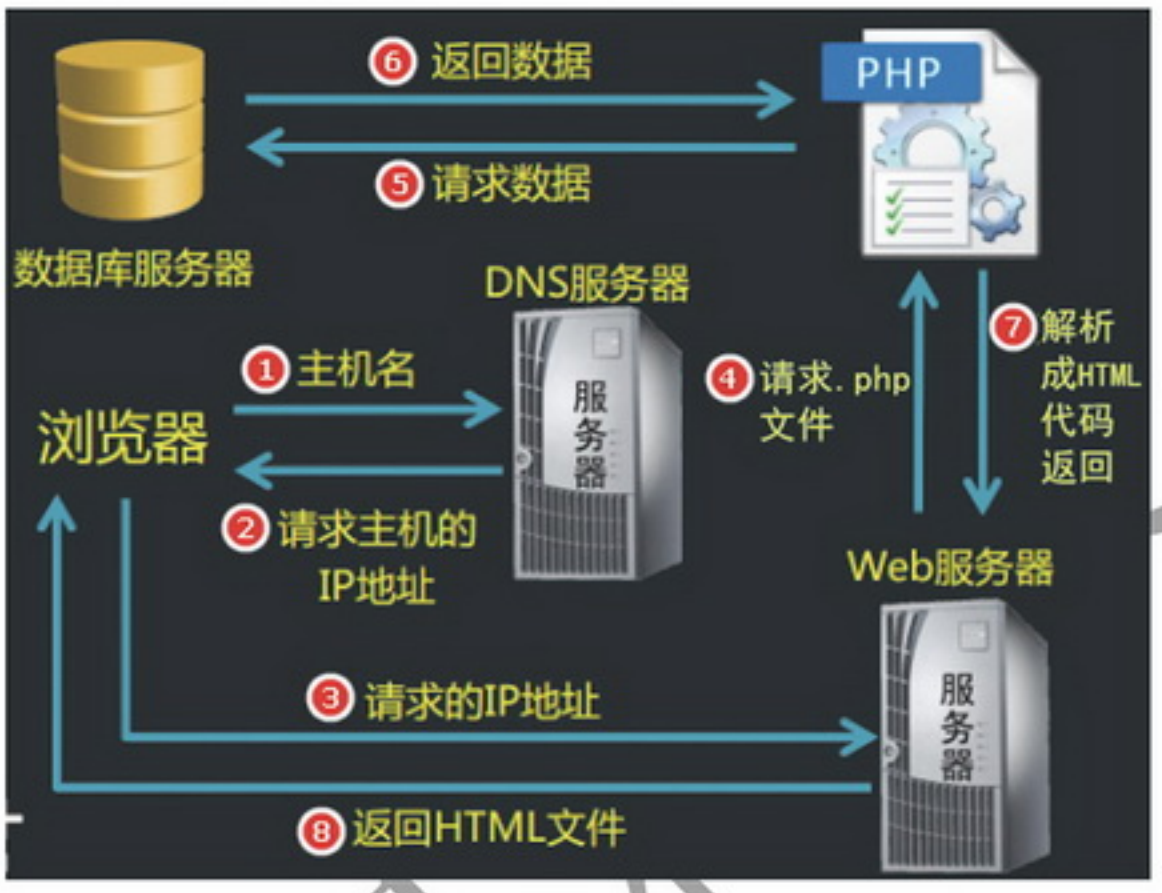


图 7.2　Web 工作原理图

下面就详细介绍每一个步骤。

（1）用户在浏览器中输入网址，如“www.mingrisoft.com”，浏览器会去请求 DNS 服务器，DNS（Domain Name System）是“域名系统”的英文缩写，是一种组织成域层次结构的计算机和网络服务命名系统，应用于 TCP/IP 网络，从事将主机名或域名转换为实际 IP 地址的工作。DNS 就是这样的一位“翻译官”，将“www.mingrisoft.com”翻译成 IP 地址“101.201.120.85”。

（2）DNS 服务器将翻译过来的 IP 地址“101.201.120.85”传递给浏览器。

（3）浏览器通过 IP 地址找到 IP 对应的 Web 服务器（通常是 Apache 或者是 Nginx），建立 TCP 连接，并向服务器发送 HTTP Request（请求）包。

（4）Web 服务器发现用户访问了后缀为“.php”的文件，如 index.php 文件，那么服务器就会访问 PHP 解析引擎。

（5）PHP 在解析时，发现需要使用数据库。于是，连接数据库，访问数据库服务器（可能是 MySQL、SQL Server、Oracle 等）。

（6）数据库根据查询条件，查找数据，并将数据返回给 PHP 引擎。

（7）PHP 引擎拼接数据，解析成 HTML，返回给 Web 服务器。

（8）Web 服务器将 HTML 文件返回给浏览器，浏览器开始解析 HTML 文件，此时，用户在浏览器中就能看到访问的网站内容。

注意：步骤（5）中的数据库内容将在第 8 章讲解，步骤（7）和步骤（8）中 HTML 内容会在接下来的 7.2 节中讲解。

7.2 HTML 表单

7.2.1 HTML 简介

HTML 是用来描述网页的一种语言。HTML 指的是超文本标记语言（Hyper Text Markup Language），它不是一种编程语言，而是一种标记语言。标记语言是一套标记标签，这种标记标签通常被称为 HTML 标签，它们是由尖括号包围的关键词，比如 <html>。HTML 标签通常是成对出现的，比如 <h1> 和 </h1>。标签对中的第一个标签是开始标签，第二个标签是结束标签。Web 浏览器的作用是读取 HTML 文档，并以网页的形式显示出它们。浏览器不会显示 HTML 标签，而是使用标签来解释页面的内容。如图 7.3 所示。

图 7.3　显示页面内容

在图 7.3 中，左侧是 HTML 代码，右侧是显示的页面内容。HTML 代码中，第一行的 <!DOCTYPE html> 表示使用的是 HTML 5（最新 HTML 版本），其余的标签都是成对出现，并且在右侧的页面中，只显示标签里的内容，不显示标签。

那么，该如何创建一个 HTML 文件呢？当然，你可以先创建一个文本文档，然后将后缀名“.txt”格式更改为“.html”。但是“.txt”文件默认编码格式为“ANSI”，而 PHP 编码规范要求使用“UTF-8”，这就需要更改文件编码格式。下面介绍如何使用 PhpStorm 创建 HTML 文件。

创建一个 HTML 文件，将其命名为 index.html。在 index.html 文件中，编写 HTML 代码。具体步骤如下：

（1）使用 PhpStorm 创建 index.html 文件。

在“D:\phpstudy\WWW”路径下创建 Code 文件夹，打开 PhpStorm，选择“Code”文件夹。在 Code 文件下创建 index.html 文件，步骤如图 7.4 和图 7.5 所示。

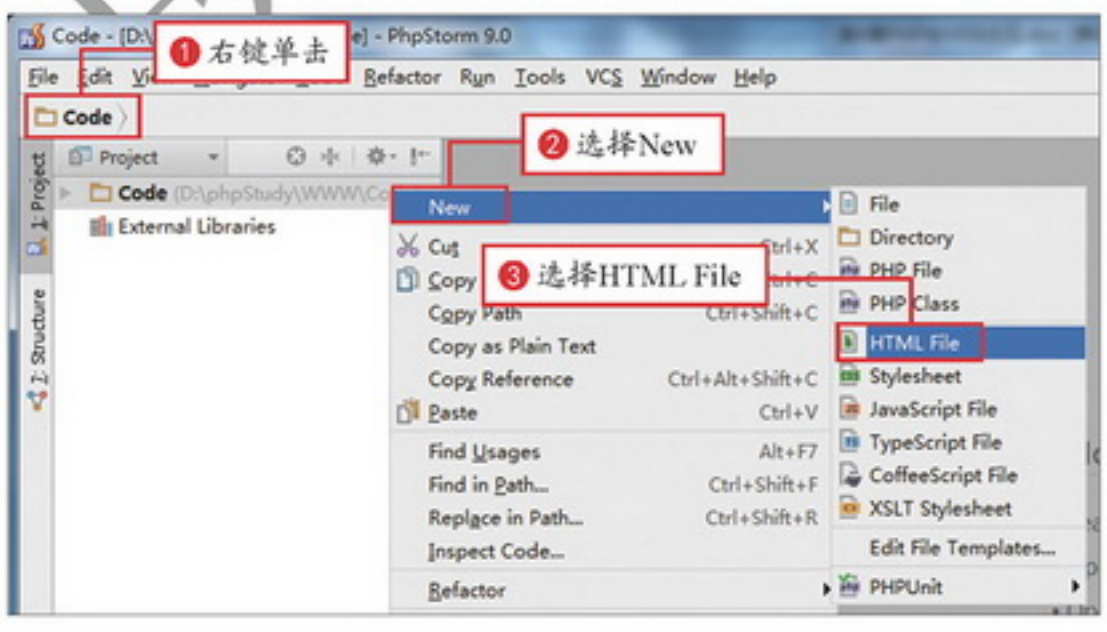


图 7.4　创建 HTML 文件

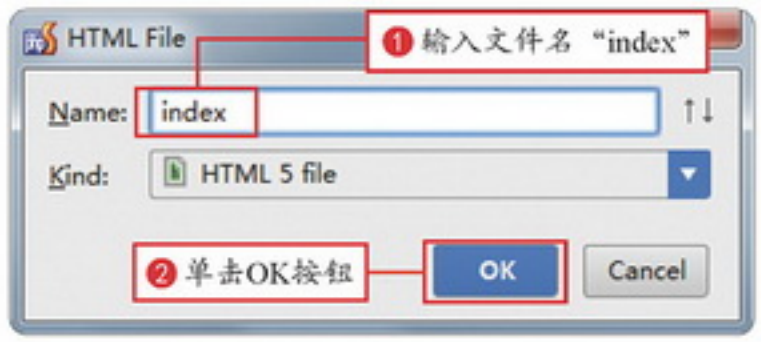


图 7.5　命名为 index.html

（2）编写 HTML 代码。

创建完成后，编辑器默认生成了基本的 HTML 5 代码结构。在 <body> 和 </body> 标签内编写 HTML 代码，具体代码如下：

```
01  <!DOCTYPE html>
02  <html lang="en">
03  <head>
04      <meta charset="UTF-8">
05      <title></title>
06  </head>
07  <body>
08      <h1> 明日学院 </h1>
09      <p>
10          明日学院，是吉林省明日科技有限公司倾力打造的在线实用技能学习平台，该平台于2016年正式
11          上线，主要为学习者提供海量、优质的课程，课程结构严谨，用户可以根据自身的学习程度，自主安
12          排学习进度。我们的宗旨是，为编程学习者提供一站式服务，培养用户的编程思维。
13      </p>
14  </body>
15  </html>
```

新增代码

（3）查看运行结果如图 7.6 所示。

图 7.6　index.html 运行结果

注意： index.html 是 HTML 文件，不是 PHP 文件，此处只涉及了 Web 工作原理中的步骤（1）、步骤（2）、步骤（3）、步骤（8）。

说明： 由于 HTML 内容广泛，本章不可能全部涵盖，作为 PHP 初学者，只要求掌握基本的 HTML 内容。

7.2.2 HTML 表单

为了实现浏览器和服务器的互动，可以使用 HTML 表单搜集不同类型的用户输入，将输入的内容从客户端的浏览器传送到服务器端，经过服务器上的 PHP 程序进行处理后，再将用户所需要的信息传递回客户端的浏览器上，从而获得用户信息，使 PHP 与 Web 页面实现交互。HTML 表单形式很多，比如用户注册、登录、个人中心设置等页面，常见的登录表单如图 7.7 所示。

图 7.7 HTML 表单

在 HTML 中，使用 <form> 标签，即可创建一个表单。表单结构如下：

```
<form name="form_name"  method="method" action="url"  enctype="value"  target="target_win">
…   //省略插入的表单元素
</form >
```

<form> 标签的属性如表 7.2 所示。

表 7.2 <form> 标签的属性

<form> 标签的属性	说　明
name	表单的名称
method	设置表单的提交方式，GET 或者 POST 方式
action	指向处理该表单页面的 URL（相对位置或者绝对位置）
enctype	设置表单内容的编码方式
target	设置返回信息的显示方式，target 的属性值包括“_blank”“_parent”“_self”“_top”

📖 **说明：** GET 方式是将表单内容附加在 URL 地址后面发送；POST 方式是将表单中的信息作为一个数据块发送到服务器上的处理程序中，在浏览器的地址栏不显示提交的信息。method 属性默认方法为 GET 方式。

7.2.3 表单元素

表单（form）由表单元素组成。常用的表单元素有以下几种标签：输入域标签 <input>、选择域标签 <select> 和 <option>、文字域标签 <textarea> 等。

1. 输入域标签 <input>

输入域标签 <input> 是表单中最常用的标签之一。常用的文本框、按钮、单选按钮、复选框等构成了一个完整的表单。

语法格式如下：

```
<form>
<input name="file_name"  type="type_name">
</form>
```

参数 name 是指输入域的名称，参数 type 是指输入域的类型。在 <input type=""> 标签中一共提供了 10 种类型的输入区域，用户所选择使用的类型由 type 属性决定。type 属性取值及举例如表 7.3 所示。

表 7.3　type 属性取值及举例

值	举　例	说　明	运 行 结 果
text	<input name="user" **type="text"** value= " 纯净水 " size="12" maxlength="1000">	name 为文本框的名称，value 是文本框的默认值，size 指文本框的宽度（以字符为单位），maxlength 指文本框的最大输入字符数	添加一个文本框：纯净水
password	<input name="pwd" **type="password"** value="666666" size="12" maxlength="20">	密码域，用户在该文本框中输入的字符将被替换显示为 *，以起到保密作用	添加一个密码域：******
file	<input name="file" **type="file" enctype="multipart/form-data"** size="16" maxlength="200">	文件域，当文件上传时，可用来打开一个模式窗口以选择文件。然后将文件通过表单上传到服务器，如上传 Word 文件等。必须注意的是，上传文件时需要指明表单的属性 enctype="multipart/form-data" 才可以实现上传功能	添加一个文件域：浏览...
image	<input name="imageField" **type="image"** src="images/banner.gif" width="120" height="24" border="0">	图像域是指可以用在提交按钮位置上的图片，这幅图片具有按钮的功能	添加一个图像域：

续表

值	举　例	说　明	运 行 结 果
radio	<input name="sex" **type="radio"** value= "1" **checked**> 男 <input name="sex" **type="radio"** value= "0"> 女	单选按钮，用于设置一组选择项，用户只能选择一项。checked 属性用来设置该单选按钮默认被选中	添加一组单选按钮（例如，您的性别为：） ◉ 男 ○ 女
checkbox	<input name="checkbox" **type="checkbox"** value="1" **checked**> 封面 <input name="checkbox" **type="checkbox"** value="1" **checked**> 正文内容 <input name="checkbox" **type="checkbox"** value="0"> 价 格	复选框，允许用户选择多个选择项。checked 属性用来设置该复选框默认被选中。例如，收集个人信息时，要求在个人爱好的选项中进行多项选择等	添加一组复选框，（如影响您购买本　） ☑ 封面 ☑ 正文内容 ☐ 价 格
submit	<input **type="submit"**name="Submit"value= "提交 ">	将表单的内容提交到服务器端	添加一个提交按钮： 提交
reset	<input **type="reset"** name="Submit" value= "重置 ">	清除与重置表单内容，用于清除表单中所有文本框的内容，并使选择菜单项恢复到初始值	添加一个重置按钮： 重置
button	<input **type="button"** name="Submit" value= "按钮 ">	按钮可以激发提交表单的动作，可以在用户需要修改表单时，将表单恢复到初始的状态，还可以依照程序的需要发挥其他作用。普通按钮一般是配合 JavaScript 脚本进行表单处理的	添加一个普通按钮： 按钮
hidden	<input **type="hidden"** name="bookid">	隐藏域，用于在表单中以隐含方式提交变量值。隐藏域在页面中对于用户是不可见的，添加隐藏域的目的在于通过隐藏的方式收集或者发送信息。浏览者单击“发送”按钮发送表单时，隐藏域的信息也被一起发送到 action 指定的处理页	添加一个隐藏域：

2. 选择域标签 <select> 和 <option>

通过选择域标签 <select> 和 <option> 可以建立一个列表或者菜单。菜单的使用是为了节省空间，正常状态下只能看到一个选项，单击右侧的倒三角按钮，打开菜单后才能看到全部的选项。列表可以显示一定数量的选项，如果超出了这个数量，会自动出现滚动条，浏览者可以通过拖动滚动条来查看各选项。

语法格式如下：

```
<select name="name" size="value" multiple>
<option value="value" selected>选项1</option>
<option value="value">选项2</option>
<option value="value">选项3</option>
…
</select>
```

参数 name 表示选择域的名称；参数 size 表示列表的行数；参数 value 表示菜单选项值；参数 multiple 表示以列表方式显示数据，省略则以菜单方式显示数据。

选择域标签 <select> 和 <option> 的显示方式及举例如表 7.4 所示。

表 7.4　选择域标签 <select> 和 <option> 的显示方式及举例

显 示 方 式	举　　例	说　　明	运 行 结 果
列表方式	<select name="spec" id="spec"> <option value="0" selected> 网络编程 </option> <option value="1"> 办公自动化 </option> <option value="2"> 网页设计 </option> <option value="3"> 网页美工 </option> </select>	下拉列表框，通过选择域标签 <select> 和 <option> 建立一个列表，列表可以显示一定数量的选项，如果超出了这个数量，会自动出现滚动条，浏览者可以通过拖动滚动条来查看各选项。selected 属性用来设置该菜单项默认被选中	 专业：
菜单方式	<select name="spec" id="spec" multiple > <option value="0" selected> 网络编程 </option> <option value="1"> 办公自动化 </option> <option value="2"> 网页设计 </option> <option value="3"> 网页美工 </option> </select>	multiple 属性用于下拉列表 <select> 标签中，指定该选项用户可以使用 <Shift+Ctrl> 进行多选	 专业：

📖 **说明**：表 7.4 中给出了静态菜单项的添加方法，而在 Web 程序开发过程中，也可以通过循环语句动态添加菜单项。

3. 文字域标签 <textarea>

文字域标签 <textarea> 用来制作多行的文字域，可以在其中输入更多的文本。

语法格式如下：

```
<textarea name="name" rows=value cols=value value="value" warp="value">
        …文本内容
</textarea>
```

参数 name 表示文字域的名称；rows 表示文字域的行数；cols 表示文字域的列数（这里的 rows 和 cols 以字符为单位）；value 表示文字域的默认值，warp 用于设定显示和送出时的换行方式，值为 off 表

示不自动换行，值为 hard 表示自动硬回车换行，换行标签一同被发送到服务器，输出时也会换行，值为 soft 表示自动软回车换行，换行标签不会被发送到服务器，输出时仍然为一列。

文字域标签 <textarea> 的值及举例如表 7.5 所示。

表 7.5　文字域标签 <textarea> 的值及举例

值	举　例	说　明	运 行 结 果
textarea	**<textarea** name="remark" cols="20" rows="4" id="remark"> 请输入您的建议！**</textarea>**	文本域，也称多行文本框，用于多行文本的编辑 warp 属性默认为自动换行方式	请发表您的建议： 

7.3 CSS 美化表单页面

7.3.1 CSS 简介

CSS 是 Cascading Style Sheets（层叠样式表）的缩写。它是一种标记语言，用于为 HTML 文档定义布局，涉及字体、颜色、边距、高度、宽度、背景图像、高级定位等方面。运用 CSS 样式可以让页面变得美观，就像化妆前和化妆后的效果一样。如图 7.8 所示。

图 7.8　使用 CSS 前后效果对比

📖 **说明**：作为 PHP 初学者，只要求掌握基本的 CSS 知识。更多 CSS 知识，请查阅相关教程。

7.3.2 插入 CSS 样式表

在 HTML 文件中插入 CSS 样式有三种方式：

1. 行内样式表

行内样式表就是使用 HTML 属性 style，在 style 属性内添加 CSS 样式。具体代码如下：

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title></title>
</head>
<body>
    <h1 style="color: blue">明日学院</h1>
    <p style="background: yellow">
        明日学院，是吉林省明日科技有限公司倾力打造的在线实用技能学习平台，该平台于2016年正式
        上线，主要为学习者提供海量、优质的课程，课程结构严谨，用户可以根据自身的学习程度，自主安
        排学习进度。我们的宗旨是，为编程学习者提供一站式服务，培养用户的编程思维。
    </p>
</body>
</html>
```

新增代码，设置<h1>标签内容为蓝色

新增代码，设置<p>标签背景色为黄色

运行效果如图 7.9 所示。

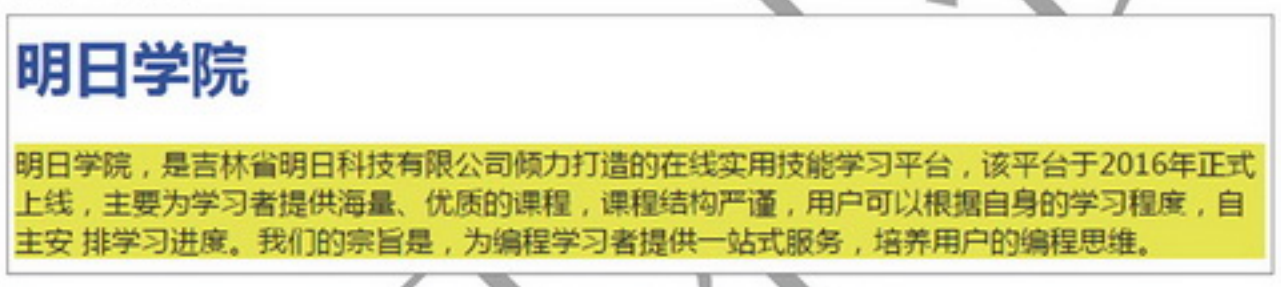


图 7.9　新增 CSS 行内样式表效果

2. 内部样式表

内部样式表即在 HTML 文件内使用 <style> 标签，在文档头部 <head> 标签内定义内部样式表，具体代码如下所示：

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title></title>
    <style>
        h1 {color: blue}
        p {background: yellow}
    </style>
</head>
<body>
    <h1>明日学院</h1>
    <p>
        明日学院，是吉林省明日科技有限公司倾力打造的在线实用技能学习平台，该平台于2016年正式
        上线，主要为学习者提供海量、优质的课程，课程结构严谨，用户可以根据自身的学习程度，自主安
        排学习进度。我们的宗旨是，为编程学习者提供一站式服务，培养用户的编程思维。
```

新增<style>标签，设置CSS样式

```
17        </p>
18    </body>
19    </html>
```

运行结果如图 7.9 所示。

3. 外部样式表

外部样式表是一个扩展名为 .css 的文本文件。跟其他文件一样，可以放在 Web 服务器上或者本地硬盘上。例如，在 test 文件目录下有两个文件 index.html 和 css 文件夹。创建一个 CSS 文件，命名为 default.css，存放于 css 的目录中。目录结构如图 7.10 所示。

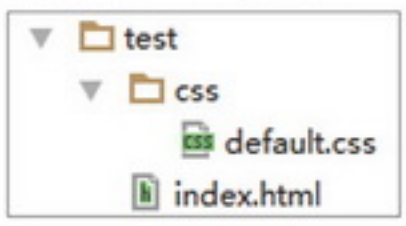


图 7.10　目录结构

那么，如何在一个 index.html 文档里引用一个外部样式表文件（default.css）呢？只需在 index.html 里创建一个指向外部样式表文件的链接（link）即可，语法格式如下：

```
<link rel="stylesheet" type="text/css" href="style/default.css" />
```

首先，编写 default.css 文件代码，即把原 index.html 内部的 CSS 代码单独写入 default.css 文件中，default.css 文件具体代码如下：

```
01  h1 {color: blue}
02  p {background: yellow}
```

然后，在 index.html 文件中使用 <link> 标签引入 default.css 外部 CSS 文件。注意要在 href 属性里给出样式表文件的地址。这行代码必须被插入 HTML 代码的头部（header），即放在标签 <head> 和标签 </head> 之间。index.html 文件完整代码如下：

```
01  <!DOCTYPE html>
02  <html lang="en">
03  <head>
04      <meta charset="UTF-8">
05      <title></title>
06      <link rel="stylesheet" type="text/css" href="css/default.css" />
07  </head>
08  <body>
09      <h1>明日学院</h1>
10      <p>
11          明日学院，是吉林省明日科技有限公司倾力打造的在线实用技能学习平台，该平台于2016年正式
12          上线，主要为学习者提供海量、优质的课程，课程结构严谨，用户可以根据自身的学习程度，自主安
13          排学习进度。我们的宗旨是，为编程学习者提供一站式服务，培养用户的编程思维。
14      </p>
```

新增<link>标签，引入外部CSS样式

```
15  </body>
16  </html>
```

运行结果如图 7.9 所示。

7.3.3 使用 CSS 美化表单页面

下面的实例应用 HTML 表单，并使用 CSS 美化表单，创建一个模拟京东的商城注册页面。

实例 01　创建商城注册页面　　实例位置：光盘\Code\SL\07\01

创建一个 HTML 文件，命名为“register.html”，该页面中包含一个注册表单。表单包含“邮箱”“密码”“确认密码”“手机号”和“是否同意服务协议”。在 register.html 文件中，使用 <link> 标签引入 2 个外部样式表文件 basic.css 和 login.css。注册页面 register.html 文件的具体代码如下：

实例01-1

```
<!DOCTYPE html>
<html>
<head lang="en">
    <meta charset="UTF-8">
    <title>注册</title>
    <!-- 引入外部CSS文件 -->
    <link rel="stylesheet" type="text/css"  href="css/basic.css" />
    <link rel="stylesheet" type="text/css"  href="css/login.css" />
</head>
<body>
<!-- 顶部 -->
<div class="login-boxtitle">
    <a href="index.html"><img alt="" src="images/logobig.png"/></a>
</div>
<!-- 主区域 -->
<div class="res-banner">
    <div class="res-main">
        <div class="login-banner-bg"><span></span><img src="images/big.png"/></div>
        <div class="login-box">
            <div class="mr-tabs" id="doc-my-tabs">
                <ul class="mr-tabs-nav mr-nav mr-nav-tabs mr-nav-justify">
                    <li class="mr-active"><a href="">注册</a></li>
                </ul>
                <div class="mr-tabs-bd">
                    <div class="mr-tab-panel mr-active">
                        <!-- 表单开始 -->
                        <form method="" action="">
                            <!-- 邮箱输入框 -->
                            <div class="user-email">
```

```
30                      <label for="email"><i class="mr-icon-envelope-o"></i></label>
31                      <input type="email" name="" id="email" placeholder="请输入邮箱账号">
32                      </div>
33                  <!-- 密码输入框 -->
34                  <div class="user-pass">
35                  <label for="password"><i class="mr-icon-lock"></i></label>
36                  <input type="password" name="" id="password" placeholder="设置密码">
37                  </div>
38                  <!-- 确认密码输入框 -->
39                  <div class="user-pass">
40                  <label for="passwordRepeat"><i class="mr-icon-lock"></i></label>
41                  <input type="password" name="" id="passwordRepeat"
42                         placeholder="确认密码">
43                  </div>
44                  <!-- 手机号输入框 -->
45                  <div class="user-pass">
46                  <label for="passwordRepeat">
47                      <i class="mr-icon-mobile"></i>
48                      <span style="color:red;margin-left:5px">*</span></label>
49                      <input type="text" name="" id="tel" placeholder="请输入手机号">
50                  </div>
51                  </form>
52                  <!-- 表单结束 -->
53                  <div class="login-links">
54                      <!-- 服务协议勾选框 -->
55                      <label for="reader-me">
56                          <input id="reader-me" type="checkbox">
57                          点击表示您同意商城《服务协议》
58                      </label>
59                      <a href="login.html" class="mr-fr">登录</a>
60                  </div>
61                  <div class="mr-cf">
62                      <input type="submit" value="注册"
63                      class="mr-btn mr-btn-primary mr-btn-sm mr-fl">
64                  </div>
65              </div>
66          </div>
67      </div>
68  </div>
69  <!-- 底部信息 -->
70  <div class="footer ">
71      <div class="footer-hd ">
72          <p>
73              <a href="http://www.mingrisoft.com/" target="_blank">明日科技</a>
74
```

```
                <b>|</b>
                <a href="#">商城首页</a>
                <b>|</b>
                <a href="#">支付宝</a>
                <b>|</b>
                <a href="#">物流</a>
            </p>
        </div>
        <div class="footer-bd ">
            <p>
                <a href="#">关于明日</a>
                <a href="#">合作伙伴</a>
                <a href="#">联系我们</a>
                <a href="#">网站地图</a>
                <em>© 2007-2017 mingrisoft.com 版权所有</em>
            </p>
        </div>
    </div>
</div>
</body>
</html>
```

运行结果如图 7.11 所示。

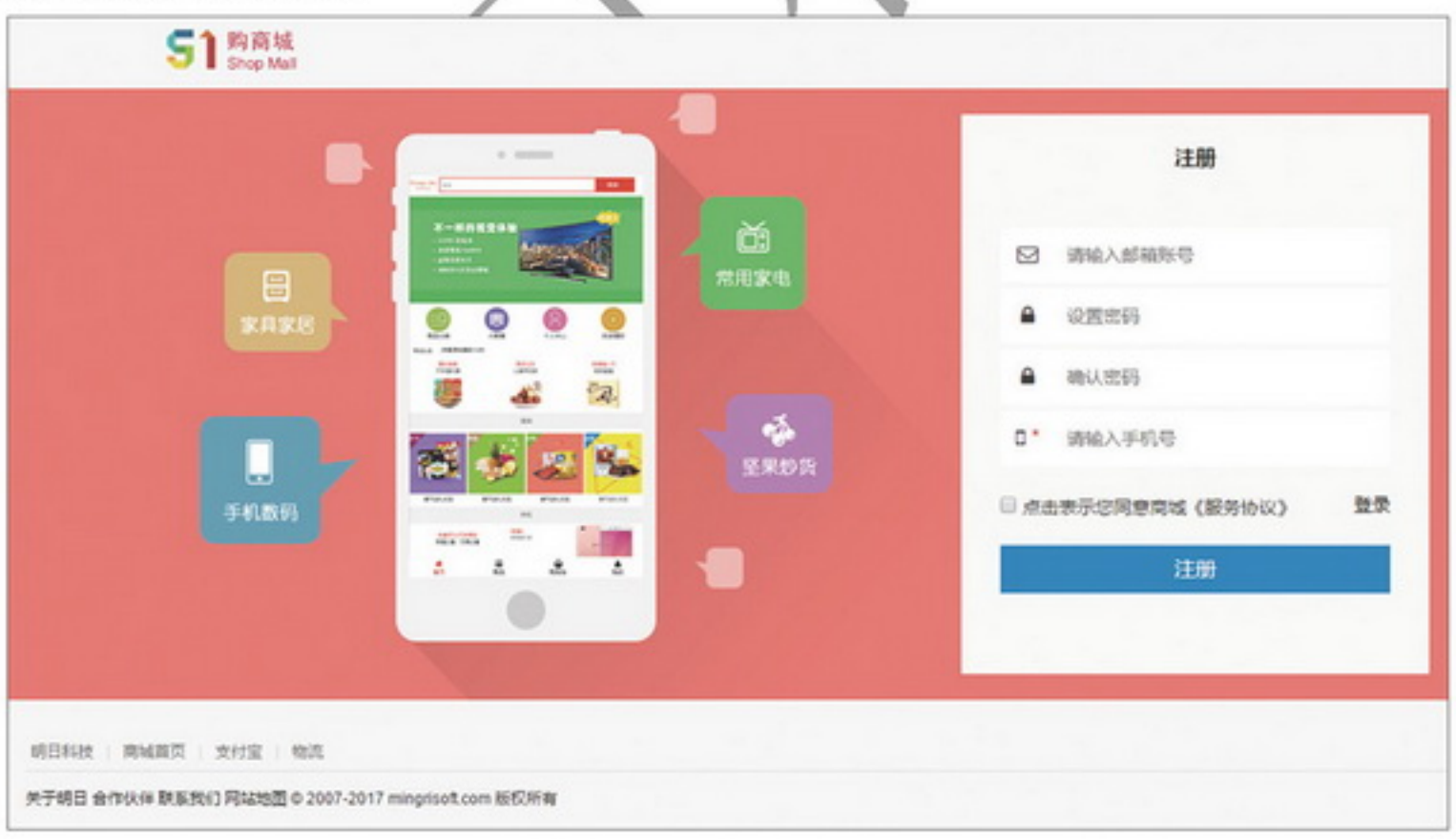


图 7.11　商城注册页面效果

练一练：

（1）修改实例 01，在商城注册页面中，新增一个输入框“手机验证码”，运行效果如图 7.12 所示。（光盘\Code\Try\07\01）

图 7.12　新增手机验证码输入框

（2）修改实例 01，创建一个商城登录页面，命名为“login.html”，如图 7.13 所示。（光盘\Code\Try\07\02）

图 7.13　商城登录页面效果

7.4 JavaScript 表单验证

7.4.1 JavaScript 简介

通常，我们所说的前端就是指 HTML、CSS 和 JavaScript 三项技术，它们的作用分别为：

- HTML：定义了网页的内容。
- CSS：描述了网页的布局。

◆ JavaScript：描述网页的行为。

JavaScript 是一种可以嵌入在 HTML 代码中由客户端浏览器运行的脚本语言。在网页中使用 JavaScript 代码，不仅可以实现网页特效，还可以响应用户请求实现动态交互的功能。例如，在用户注册页面中，需要对用户输入信息的合法性进行验证，包括是否填写了“邮箱”和“手机号”，填写的“邮箱”和“手机号”格式是否正确等等。JavaScript 验证邮箱是否为空的效果如图 7.14 所示。

图 7.14　JavaScript 验证为空

注意：由于 JavaScript 是客户端编程语言，根据用户使用的浏览器不同，JavaScript 的提示框出现的位置可能不同。本书所有实例均使用谷歌浏览器运行。

7.4.2　调用 JavaScript

1. 在 HTML 中嵌入 JavaScript 脚本

JavaScript 作为一种脚本语言，可以使用 <script> 嵌入到 HTML 文件中。

语法格式如下：

```
<script >
…
</script>
```

例如，在 HTML 文件中嵌入 JavaScript 脚本，可以直接在 <script> 和 </script> 标签中间写入 JavaScript 代码，用于弹出一个提示对话框，代码如下：

```
01  <!DOCTYPE html>
02  <html>
03  <head>
04      <title>在HTML中嵌入JavaScript脚本</title>
```

```
</head>
<body>
<script>
    alert("我很想学习PHP编程，请问如何才能学好这门语言!");
</script>
</body>
</html>
```

在上面的代码中，<script> 与 </script> 标签之间调用 JavaScript 脚本语言 window 对象的 alert() 方法，向客户端浏览器弹出一个提示对话框。运行结果如图 7.15 所示。

图 7.15　在 HTML 中嵌入 JavaScript 脚本

2. 应用 JavaScript 事件调用自定义函数

在 Web 程序开发过程中，经常需要在表单元素相应的事件下调用自定义函数。例如，在按钮的单击事件下调用自定义函数 check() 来验证表单元素是否为空，代码如下：

```
<input type="submit" name="Submit" value="检测" onClick="check();">
```

然后在该表单的当前页中编写一个 check() 自定义函数，在该函数内实现验证表单元素是否为空。

3. 引用外部 JavaScript 文件

在网页中，除了可在 <script> 与 </script> 标签之间编写 JavaScript 脚本代码，还可以通过 <script> 标签中的 src 属性指定外部的 JavaScript 文件（即 JS 文件，以 .js 为扩展名）的路径，从而引用对应的 JS 文件。该方式与引用外部 CSS 文件类似。

语法格式如下：

```
<script src=url></script>
```

其中，url 是 JS 文件的路径。使用外部 JS 文件的优点如下：

◆ 使用 JS 文件可以将 JavaScript 脚本代码从网页中独立出来，便于代码的阅读。

◆ 一个外部 JS 文件，可以同时被多个页面调用。当共用的 JavaScript 脚本代码需要修改时，只需要修改 JS 文件中的代码即可，便于代码的维护。

◆ 通过 <script> 标签中的 src 属性不但可以调用同一个服务器上的 JS 文件，还可以通过指定路径来调用其他服务器上的 JS 文件。

7.4.3 JavaScript 表单验证

下面的实例就应用了 JavaScript 事件调用自定义函数，检测商城注册页面的输入信息。

实例 02　检测商城注册页面输入信息　　实例位置：光盘\Code\SL\07\02

本实例将使用实例 01 中的代码，在 register.html 文件中添加 JavaScript 验证的代码。首先给 register.html 页面中的“注册”按钮添加 onclick 点击事件，调用自定义函数 mr_verify()。然后，在函数体内实现表单验证功能。register.html 关键代码如下：

实例02-1

```
<!DOCTYPE html>
<html>
<head lang="en">
    <!--省略部分代码 -->
</head>
<body>
<!-- 省略部分代码 -->
<!-- 主区域 -->
<div class="res-banner">
    <div class="res-main">
            <!-- 省略部分代码 -->
        <!-- 表单开始 -->
        <form method="" action="">
        <!--省略部分代码-->
        <div class="mr-cf">
         <input type="submit" onclick="mr_verify()" value="注册"
                class="mr-btn mr-btn-primary mr-btn-sm mr-fl">
        </div>
        <!-- 表单结束 -->
    </div>
    <!-- 底部信息 -->
    <div class="footer ">
        <!-- 省略部分代码 -->
    </div>
</div>
<script>
    //表单验证
    function mr_verify(){
        //获取表单对象
        var email = document.getElementById("email");
        var password = document.getElementById("password");
        var passwordRepeat = document.getElementById("passwordRepeat");
        var tel = document.getElementById("tel");
        var reader_me = document.getElementById("reader-me");
```

添加onclick点击事件

新增mr_verify()函数

```
35          //验证项目是否为空
36          if(email.value===" || email.value===null){
37              alert("邮箱不能为空！");
38              return false;    //终止程序，不再继续执行
39          }
40          if(password.value===" || password.value===null){
41              alert("密码不能为空！");
42              return false;
43          }
44          if(passwordRepeat.value===" || passwordRepeat.value===null){
45              alert("确认密码不能为空！");
46              return false;
47          }
48          if(tel.value===" || tel.value===null){
49              alert("手机号码不能为空！");
50              return false;
51          }
52          if(password.value!==passwordRepeat.value ){
53              alert("密码设置前后不一致！");
54              return false;
55          }
56          //验证邮件格式
57          apos = email.value.indexOf("@")
58          dotpos = email.value.lastIndexOf(".")
59          if (apos < 1 || dotpos - apos < 2) {
60              alert("邮箱格式错误！");
61          }
62          //验证手机号格式
63          if(isNaN(tel.value)){
64              alert("手机号请输入数字！");
65              return false;
66          }
67          if(tel.value.length!==11){
68              alert("手机号是11个数字！");
69              return false;
70          }
71          //验证是否选择服务协议
72          if(reader_me.checked == false){
73              alert("只有同意商城《服务协议》才能注册");
74              return false;
75          }
76          alert('注册成功！');
77      }
78  </script>
79  </body>
80  </html>
```

当输入错误的手机号格式，运行结果如图 7.16 所示。注册成功如图 7.17 所示。

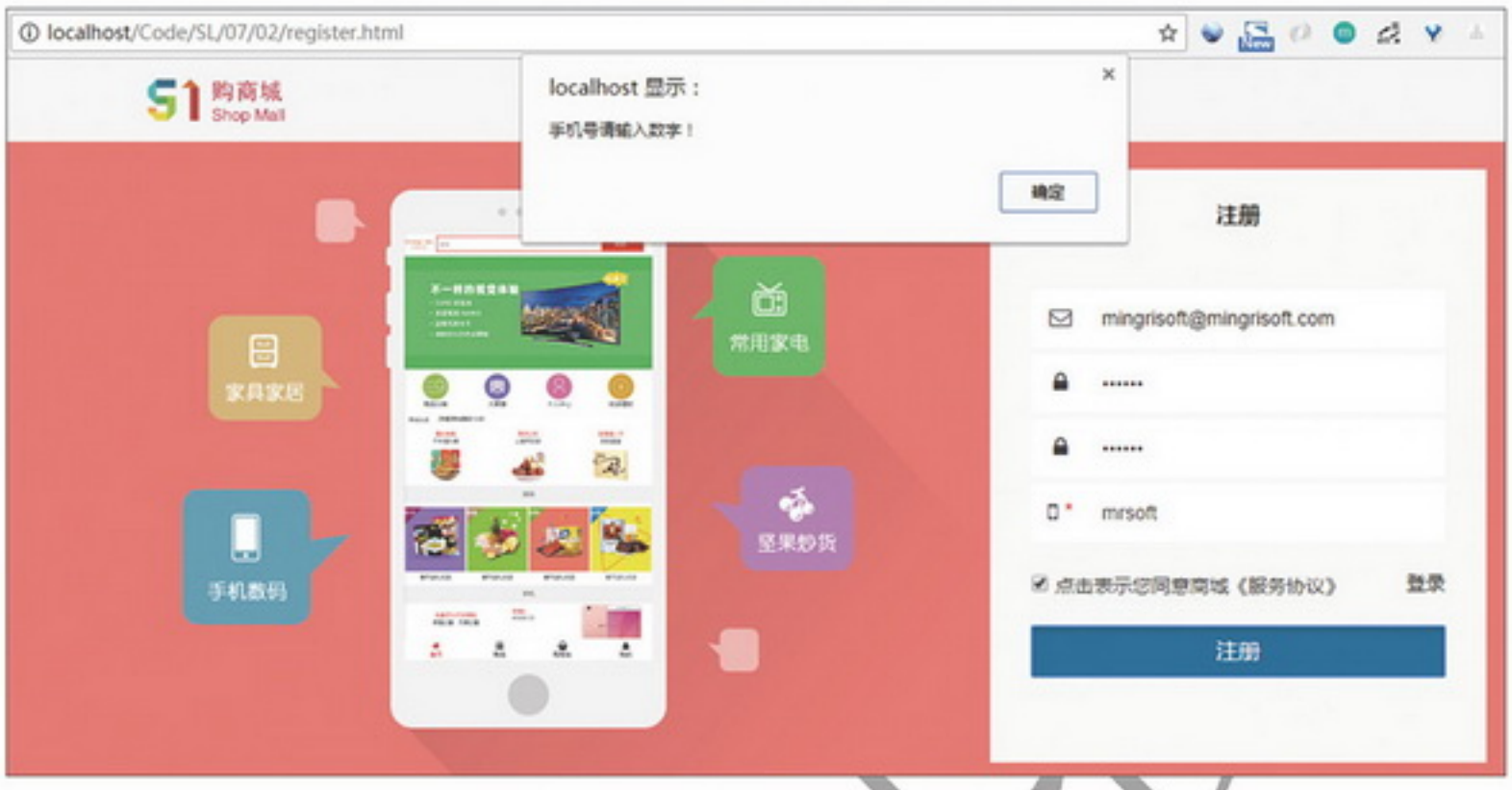


图 7.16　手机号格式错误提示

图 7.17　注册成功提示

练一练：

（1）实例 02 商城注册页面中，对手机号的验证只包括是否为空，是否为 11 位数字。如果用户在手机号输入框中输入“01234567891”，程序会检测该手机号满足注册条件。修改实例 02，在 JavaScript 中自定义函数 checkMobile()，使用正则表达式，实现对手机号格式的验证功能。（光盘\Code\Try\07\03）

（2）实例 02 商城注册页面中，对邮箱的验证使用了多个 if 语句实现，代码看起来比较烦琐。修改实例 02，在 JavaScript 中自定义函数 checkEmail()，使用正则表达式，实现对邮箱格式的验证功能。（光盘\Code\Try\07\04）

7.5 PHP 获取表单数据

HTML 表单已经准备就绪，接下来就要提交表单并获取表单数据了。提交表单，即是将表单信息从客户端提交到服务器端，这时需要使用 HTTP 协议的请求方法，本节中只讲解最常用的 POST 方法和 GET 方法，采用哪种方法是由 HTML 文件中 <form> 标签的 method 属性所指定的。服务器接收请求信息，这时服务器端语言 PHP 闪亮登场。

PHP 接收数据的方式非常简单，如果客户端使用 POST 方式提交，提交的表单域代码如下：

```
<form method="POST" action="register.php">
    <input name="username" value="张三" />
    <!-- 省略其余代码 -->
</form>
```

上述代码中，使用 $_POST['username'] 接收 <input> 标签中 name 属性为 username 的值，$_POST['username'] 的值为“张三”。

如果以 GET 方式提交，则使用 $_GET['username'] 接收。如图 7.18 所示。

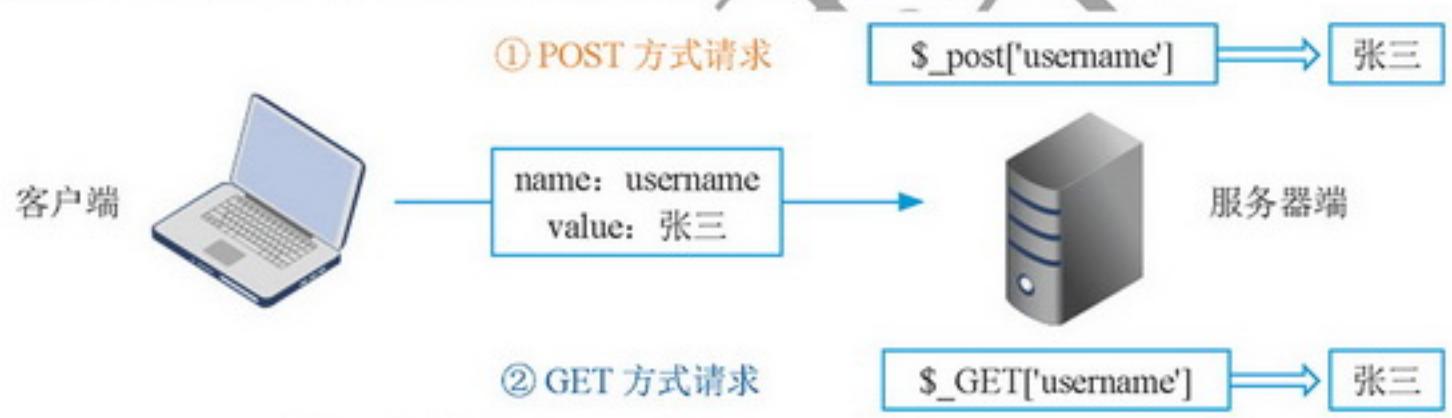


图 7.18　PHP 获取表单数据

7.5.1 获取 POST 方式提交的表单数据

应用 POST 方式时，只需将 <form> 表单中的属性 method 设置成“POST”即可。POST 方式不依赖于 URL，不会显示在地址栏。POST 方式可以没有限制地传递数据到服务器，所有提交的信息在后台传输，用户在浏览器端是看不到这一过程的，安全性高。所以，POST 方式比较适合用于发送一个保密的（如账号密码）或者容量较大的数据到服务器。

实例 03　获取商城注册页面的输入信息　　实例位置：光盘\Code\SL\07\03

本实例将在实例 02 商城注册页面中，使用 POST 方式将表单数据提交到 addUser.php 文件中，在该文件中接收表单数据并显示数据内容。具体步骤如下：

（1）修改 register.html 文件，使用 POST 方式提交表单数据，主要代码如下：

实例03-1

```
<!DOCTYPE html>
<html>
<head lang="en">
    <!-- 省略部分代码 -->
</head>
<body>
        <!-- 省略部分代码 -->
        <!-- 表单开始 -->
        <form method="POST" action="addUser.php">
        <!-- 省略部分代码 -->
</body>
</html>
```

修改<form>标签，使用POST方式提交

（2）创建 addUser.php 文件，接收表单数据，完整代码如下：

实例03-2

```
<?php
    $email      = $_POST['email'];                    //接收email
    $password   = $_POST['password'];                 //接收password
    $passwordRepeat = $_POST['passwordRepeat'];       //接收passwordRepeat
    $tel        = $_POST['tel'];                      //接收tel
    /** 输出接收的信息 **/
    echo "接收到的email是:".$email."<br>";
    echo "接收到的password是:".$password."<br>";
    echo "接收到的passwordRepeat是:".$passwordRepeat."<br>";
    echo "接收到的tel是:".$tel."<br>";
    $array = $_POST;                                  //接收全部信息
    echo "<pre>";
    print_r($_POST);                                  //打印信息
?>
```

在浏览器中输入“http://localhost/Code/SL/07/03/register.html”，按下 <Enter> 键，进入商城注册页面，在表单中输入相应信息，单击“注册”按钮，输出用户注册信息。如图 7.19 所示。

图 7.19　获取 POST 方式提交的表单数据

练一练：

（1）创建一个填写个人信息的页面“user.html”，使用 POST 方式提交到“update.php”，在 update.php 文件中，输出用户填写的信息。如图 7.20 所示。（光盘\Code\Try\07\05）

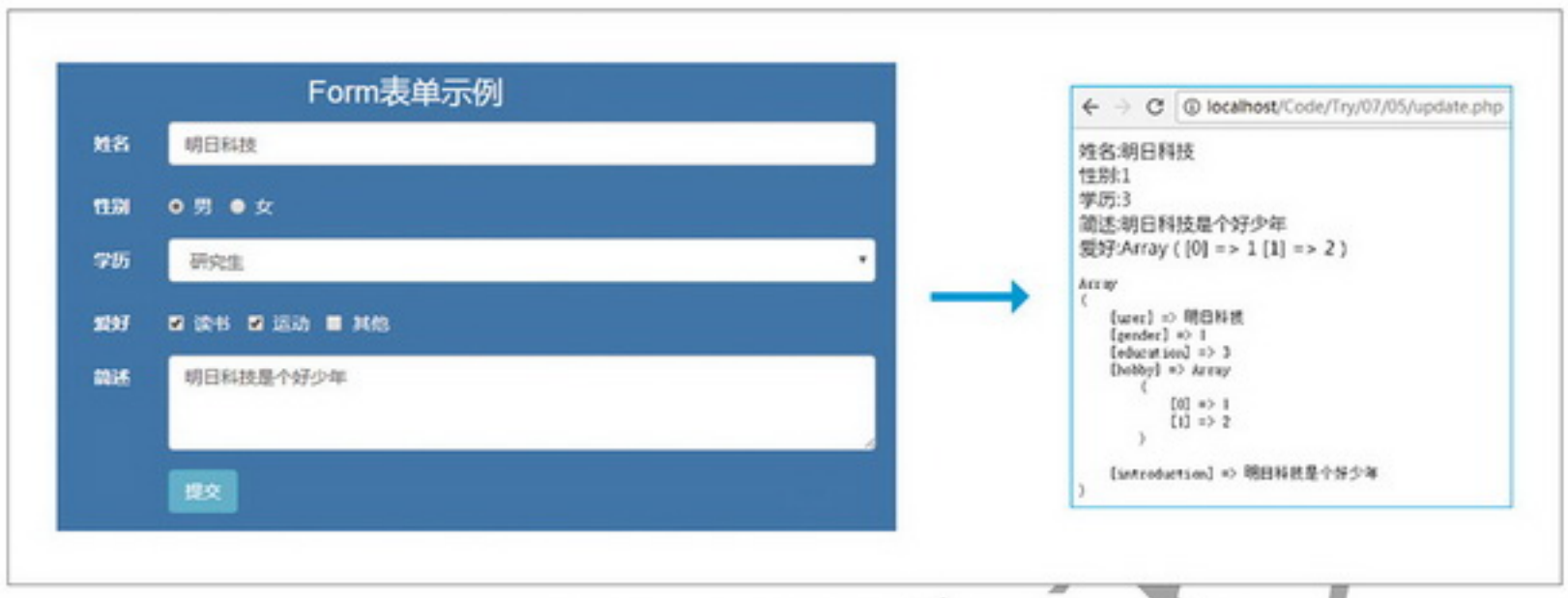


图 7.20　POST 提交用户信息

（2）创建一个用户登录页面“login.html”，使用 POST 方式提交到“checkLogin.php”，在 checkLogin.php 文件中，输出填写的登录信息。如图 7.21 所示。（光盘\Code\Try\07\06）

图 7.21　POST 方式提交用户登录信息

7.5.2 获取 GET 方式提交的表单数据

GET 方式是 <form> 表单中 method 属性的默认方法。使用 GET 方式提交的表单数据被附加到 URL 后，并作为 URL 的一部分发送到服务器端。在程序的开发过程中，由于 GET 方式提交的数据是附加到 URL 上发送的，因此，在 URL 的地址栏中将会显示“URL+ 用户传递的参数”。

GET 方式传递的参数格式如图 7.22 所示。

图 7.22　GET 方式传递的参数示意图

其中，url 为表单响应地址（如 localhost/index.php），name1 为表单元素的名称，value1 为表单元素的值。url 和表单元素之间用“?”隔开，而多个表单元素之间用“&”隔开，每个表单元素的格式都是“name=value”，固定不变。

注意：若要使用 GET 方式发送表单，URL 的长度应限制在 1MB 字符以内。如果发送的数据量太大，数据将被截断，从而导致意外或失败的处理结果。

实例 04　使用 GET 方式提交表单数据

实例位置：光盘\Code\SL\07\04

创建一个表单来实现应用 GET 方式提交用户名和密码，并显示在 URL 地址栏中。添加一个文本框，命名为 user，添加一个密码域，命名为 pwd，将表单的 method 属性设置为 GET 方式，实例代码如下：

实例04-1

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>PHP零基础</title>
    <!-- 引入Bootstrap前端UI框架 -->
    <link   href="bootstrap/css/bootstrap.css" rel="stylesheet">
</head>
<body class="bg-primary">
<h3 class="col-sm-offset-2">Form表单GET示例</h3>
<form class="form-horizontal" role="form" method="get" action="#">
    <div class="form-group">
        <label class="col-sm-2 control-label">姓名</label>
        <div class="col-sm-3">
            <input type="text" name="username" class="form-control"
                   placeholder="请输入用户名">
        </div>
    </div>
    <div class="form-group">
        <label  class="col-sm-2 control-label">密码</label>
        <div class="col-sm-3">
            <input type="password" name="password"  class="form-control"
                   placeholder="请输入密码">
        </div>
    </div>
    <div class="form-group">
        <div class="col-sm-offset-2 col-sm-10">
            <button type="submit" class="btn btn-info">提交</button>
        </div>
    </div>
```

```
31  </form>
32  </body>
33  </html>
```

多学两招：上述代码中，使用了 Bootstrap 前端 UI 框架。Bootstrap，来自 Twitter 公司，是目前最受欢迎的前端框架。Bootstrap 是基于 HTML、CSS、JavaScript 开发的，它简洁灵活，使得 Web 开发更加快捷。中文网址：http://www.bootcss.com。

运行本实例，在文本框中输入用户名“明日科技”和密码“mrsoft”，单击“提交”按钮，文本框内的信息就会显示在 URL 地址栏中，如图 7.23 所示。

图 7.23　使用 GET 方式提交表单

显而易见，这种方法会将参数暴露。如果用户传递的参数是非保密性的参数（如 id=8），那么采用 GET 方式传递数据是可行的，如果用户传递的是保密性的参数（如密码），这种方法就会不安全。而使用 POST 方式则更为安全。

练一练：

（1）在明日学院用户列表中，用户单击“编辑”按钮，浏览器将使用 GET 方式发送请求，并且传递用户 id 到 editUser.php 文件。在 editUser.php 文件中，输出用户 id，如图 7.24 所示。试着实现该功能。（光盘\Code\Try\07\07）

图 7.24　GET 方式接收用户 id

（2）在明日学院用户列表中，用户单击“删除”按钮，浏览器将使用 GET 方式发送请求，并且传递用户 id 到 deleteUser.php 文件，在 deleteUser.php 文件中，输出用户 id。试着实现该功能。（光盘\Code\Try\07\08）

7.6 难点解答

7.6.1 Web 工作原理

Web 工作原理实际上比 7.1.2 小节的描述要复杂得多，每一个步骤都可以用一本书来讲解，但作为初学者，只要求掌握大概流程即可。比如，了解 Web 工作原理，就会明白为什么不能在 HTML 文件中写 PHP 代码，因为 HTML 文件不会请求 PHP 引擎，自然不会解析 <?php ?> 标签，PHP 代码会以字符串形式原样输出。但是在 PHP 文件中，可以编写 HTML 代码。

7.6.2 JavaScript 和 Java 关系

JavaScript 和 Java 的关系就像雷锋和雷峰塔的关系。它们是两门不同的编程语言。当时网景公司之所以将 LiveScript 命名为 JavaScript，是因为 Java 是当时最流行的编程语言，带有“Java”的名字有助于这门新生语言的传播。

7.6.3 JavaScript 和 jQuery 的关系

jQuery 是对 JavaScript 的一个扩展、封装，让 JavaScript 更好用、更简单。其核心理念是“write less, do more（写的更少，做的更多）”。例如，获取一个表单中“id="start"”的元素 value 值，使用 JavaScript 代码如下：

```
document.getElementById("start").value;
```

使用 jQuery 的代码如下：

```
$('#start').val();
```

在使用 jQuery 时，需要先引入 jQuery。虽然使用 jQuery 比 JavaScript 更简单，但是 JavaScript 才是根本，所以一般要先学习 JavaScript，再学习 jQuery。

7.7 小　结

本章内容涉及知识比较广泛，既有前端 HTML、CSS 和 JavaScript 技术，又有后端 PHP 使用两种方式接收表单数据的知识。相信读者在学习完本章后，可以对表单应用自如，从而轻松实现“人机交互”。掌握了本章的技术要点，就意味着已经有了开发动态网页的能力，为下一步的深入学习奠定良好的基础。

7.8 动手纠错

1．运行“光盘\Code\Debug\07\01”文件夹下的程序，PHP 代码没有解析，导致昵称为空，如图 7.25 所示。请根据注释改正程序。

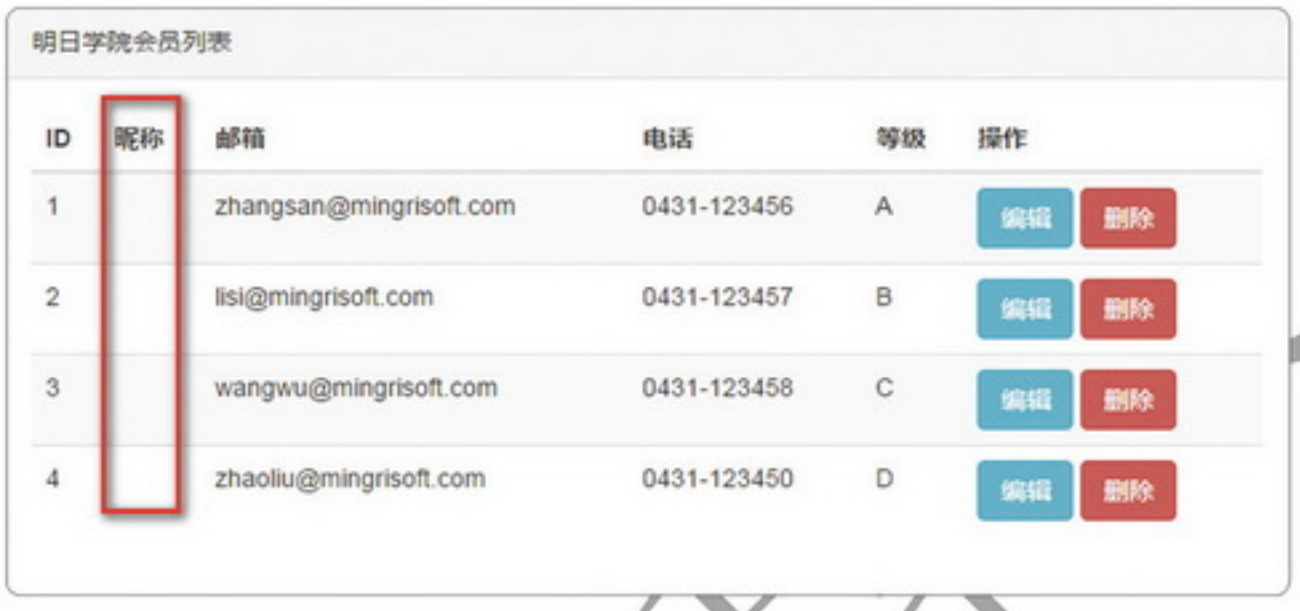


图 7.25 昵称为空

2．运行“光盘\Code\Debug\07\02”文件夹下的程序，Form 表单无法提交。请根据注释改正程序。

3．运行“光盘\Code\Debug\07\03”文件夹下的程序，学历 select 下拉列表默认选中“本科”无效，请根据注释改正程序。

4．运行“光盘\Code\Debug\07\04”文件夹下的程序，“爱好”复选框选择 3 个选项，提交后只接收到一个。请根据注释改正程序。

5．运行“光盘\Code\Debug\07\05”文件夹下的程序，添加头像后，单击“提交”按钮后，无法接收上传的文件信息，如图 7.26 所示，请根据注释改正程序。

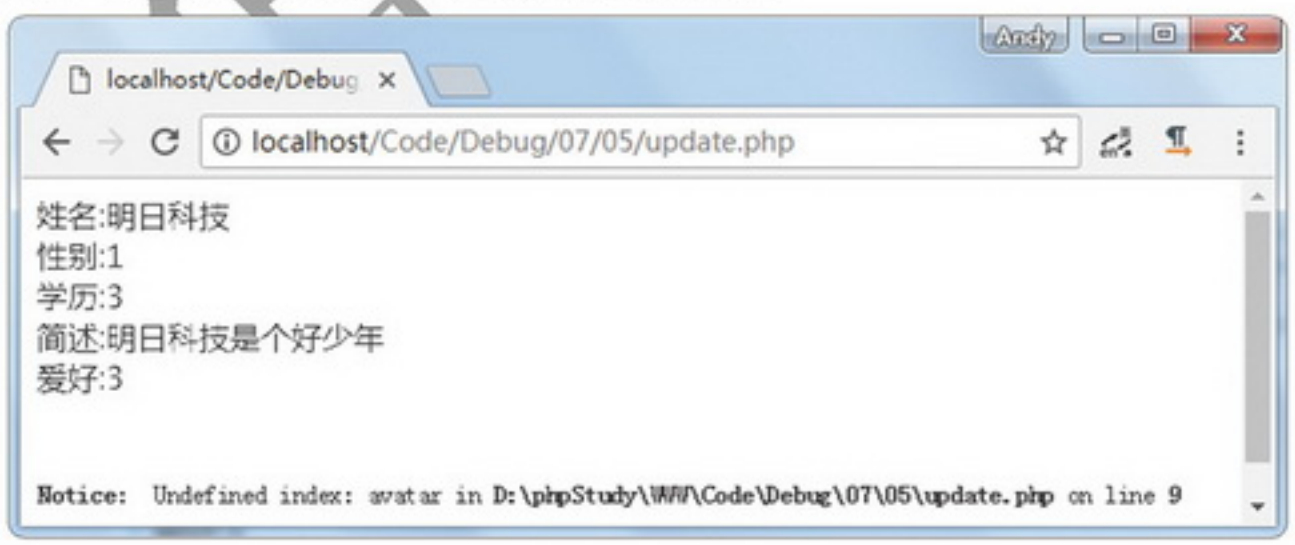


图 7.26 无法接收文件上传信息

第 8 章

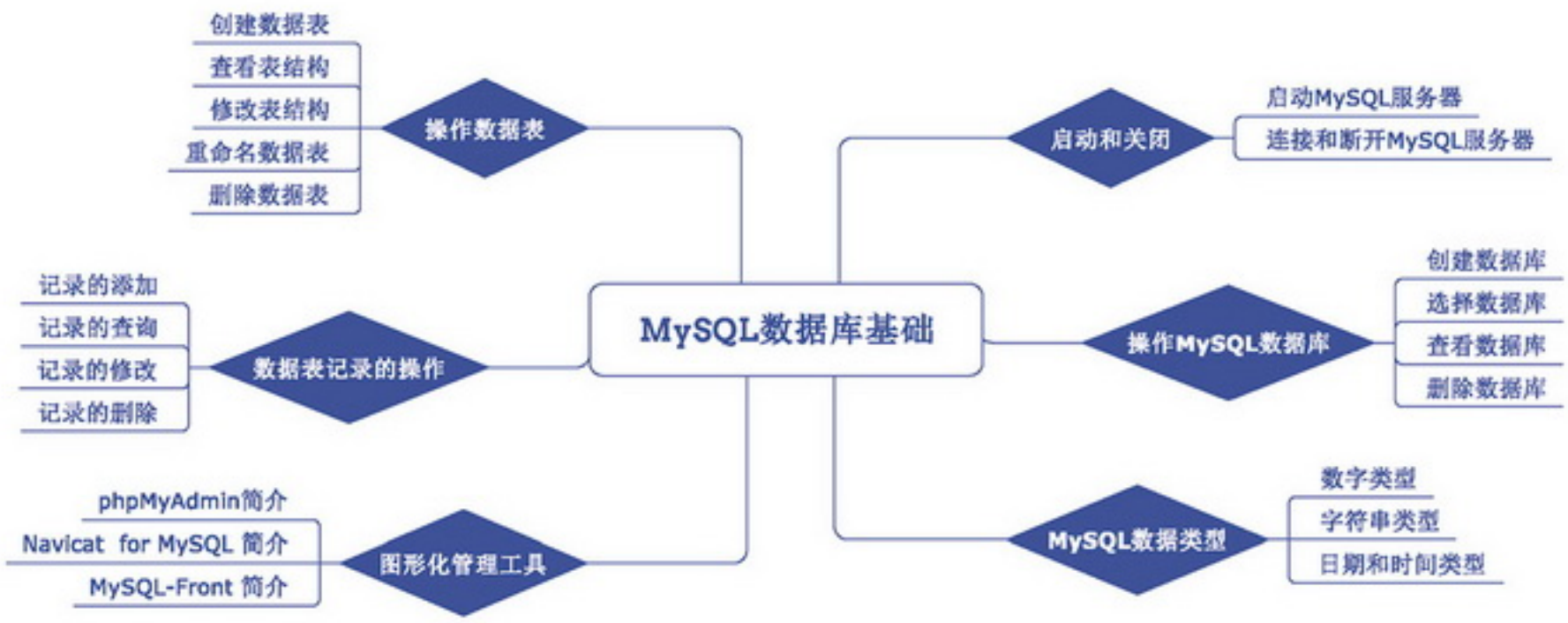
MySQL 数据库基础

（视频讲解：58 分）

本章概览

只有与数据库相结合，才能充分发挥动态网页编程语言的魅力，因为网络上的众多应用都是基于数据库的。PHP 支持多种数据库，尤其与 MySQL 被称为“黄金搭档”。MySQL 命令行通过 sql 语句对数据库进行操作，本章将详细介绍 MySQL 数据库的基础知识。通过本章的学习，读者不但可以轻松掌握操作 MySQL 数据库、数据表的方法，还可以对 MySQL 数据库进行查询等操作。

知识框架



8.1 MySQL 概述

MySQL 是目前最为流行的开源数据库，是完全网络化的跨平台关系型数据库系统，它是由瑞典的 MySQL AB 公司开发的。该公司由 MySQL 的初始开发人员 David Axmark 和 Michael Monty Widenius 于 1995 年建立。它的象征符号是一只名为 Sakila 的海豚，如图 8.1 所示，代表着 MySQL 数据库和团队的速度、能力、精确和优秀本质。

图 8.1　MySQL 图标

除了具有许多其他数据库所不具备的功能和选择之外，MySQL 数据库还是一种完全免费的产品，用户可以直接从网上下载使用，而不必支付任何费用。MySQL 的特点如下：

◆ 功能强大：MySQL 中提供了多种数据库存储引擎，这些引擎各有所长，适用于不同的应用场合，用户可以选择最合适的引擎以得到最高的性能，甚至可以处理每天访问量数亿的高强度 Web 搜索站点。MySQL 支持事务、视图、存储过程和触发器等。

◆ 支持跨平台：MySQL 支持至少 20 种以上的开发平台，包括 Linux、Windows、FreeBSD、IBMAIX、AIX 和 FreeBSD 等。这使得在任何平台下编写的程序都可以进行移植，而不需要对程序做任何修改。

◆ 运行速度快：高速是 MySQL 的显著特性。MySQL 使用了极快的 B 树磁盘表（MyISAM）和索引压缩；通过使用优化的单扫描多连接，能够极快地实现连接；SQL 函数使用高度优化的类库实现，运行速度极快。

◆ 成本低：MySQL 数据库是一种完全免费的产品，用户可以直接从网上下载。

◆ 支持各种开发语言：MySQL 为各种流行的程序设计语言提供支持，为它们提供了很多的 API 函数，包括 PHP、ASP.NET、Java、Eiffel、Python、Ruby、Tcl、C、C++ 和 Perl 等。

◆ 数据库存储容量大：MySQL 数据库的最大有效表尺寸通常是由操作系统对文件大小的限制决定的，而不是由 MySQL 内部限制决定的。InnoDB 存储引擎将 InnoDB 表保存在一个表空间内，该表空间可由数个文件创建，表空间的最大容量为 64TB，可以轻松处理拥有上千万条记录的大型数据库。

8.2 启动和关闭 MySQL 服务器

8.2.1 启动 MySQL 服务器

由于我们使用的 phpStudy 集成开发环境中已经内置了 MySQL，所以，读者无需再重复安装 MySQL。当启动 phpStudy 时，MySQL 也随着默认启动，如图 8.2 所示。

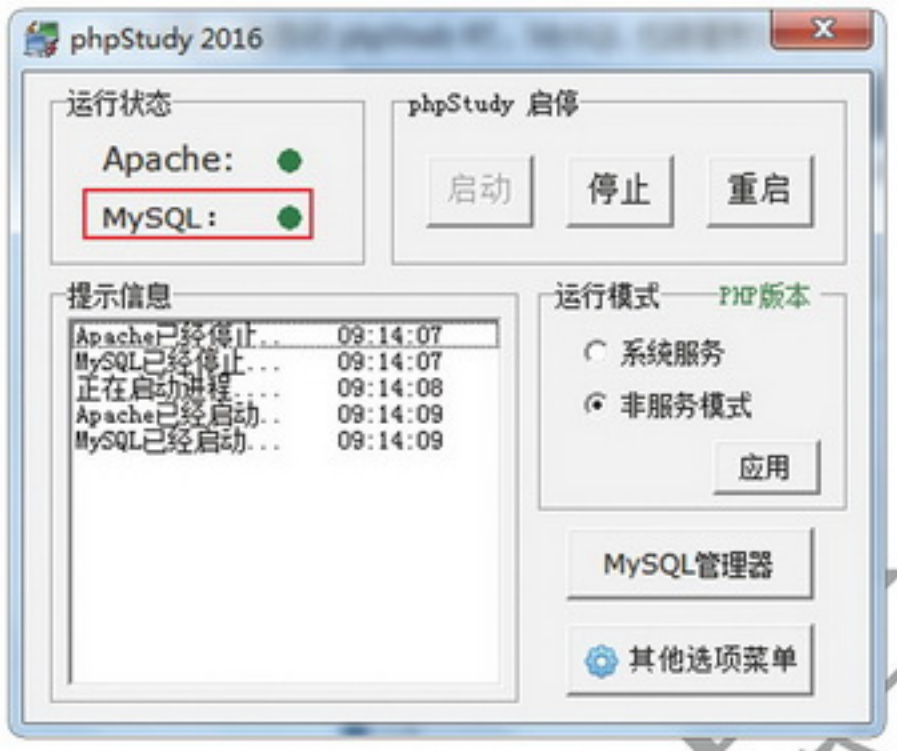


图 8.2　默认启动 MySQL

8.2.2 连接和断开 MySQL 服务器

1. 连接 MySQL 服务器

MySQL 服务器启动后，下面连接服务器。MySQL 提供了 MySQL console 命令窗口客户端实现了与 MySQL 服务器之间的交互。操作步骤如下：

选择“开始”→“运行”，在弹出的“运行”窗口中输入“cmd”（命令提示符）命令，如图 8.3 所示。按下 <Enter> 键后进入 DOS 窗口，如图 8.4 所示。

图 8.3　Windows 7 系统下的“运行”窗口

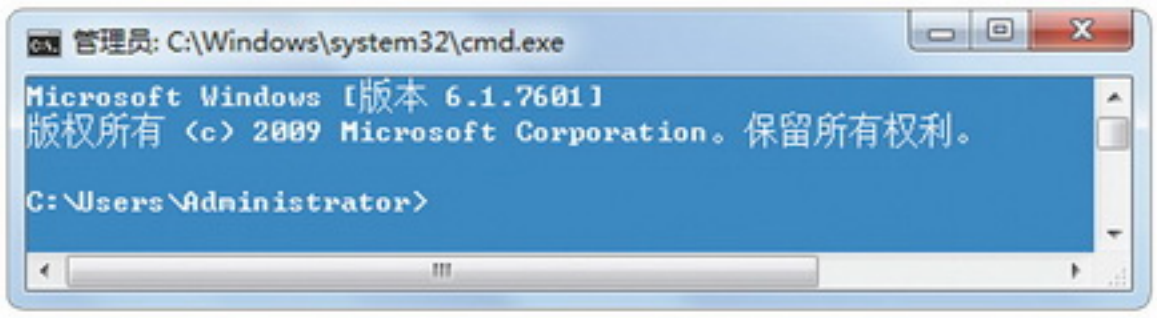


图 8.4　DOS 窗口

要使用 MySQL 命令，首先需要切换到 MySQL 命令行目录，即“D:\phpStudy\MySQL\bin”，操作方法如图 8.5 所示。

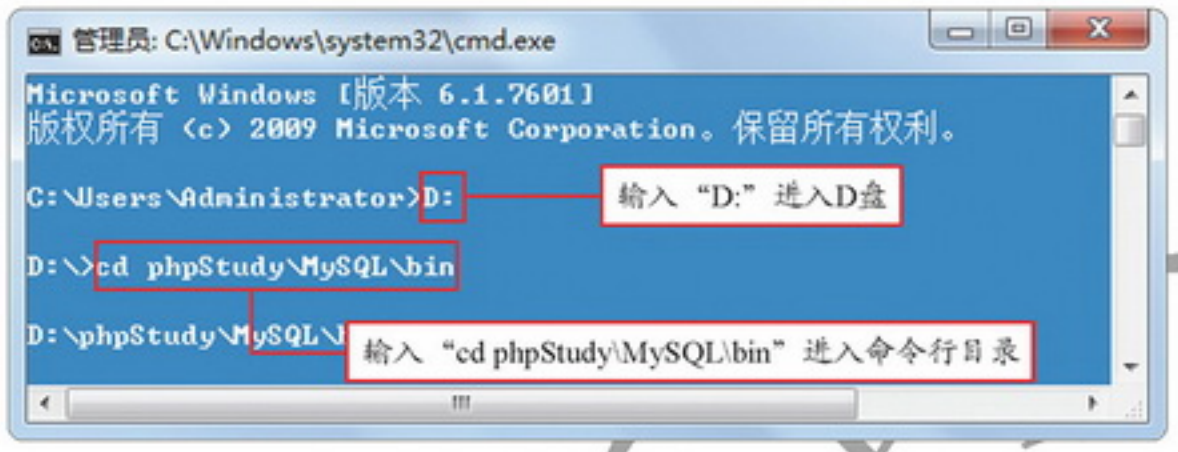


图 8.5　进入 MySQL 命令行目录

在命令提示符下，输入如下命令连接 MySQL：

```
mysql  -uroot  -proot
```

上述命令中，“-uroot”表示用户名为“root”，“-proot”表示密码为“root”。phpStudy 中 MySQL 的默认账号和密码都是 root。

注意：“-uroot”中字母之间没有空格，“-proot”也没有空格。

输入完命令语句后，按下 <Enter> 键即可连接 MySQL 服务器，如图 8.6 所示。

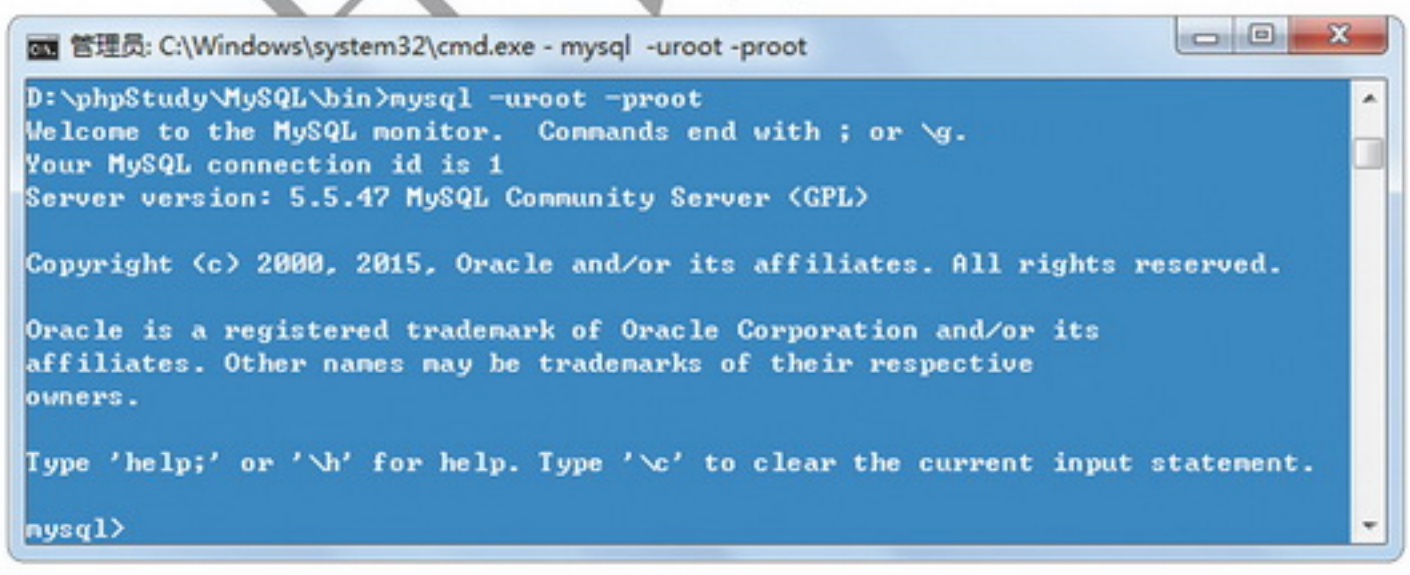


图 8.6　成功连接 MySQL 服务器

2. 断开 MySQL 连接

连接到 MySQL 服务器后，可以通过在 MySQL 提示符下输入“exit”或者“quit”命令，然后按下 <Enter> 键来断开 MySQL 连接，如图 8.7 所示。

图 8.7　断开 MySQL 连接

3. 设置系统的环境变量

每次使用 MySQL 命令，都要切换到 MySQL 命令行目录，即“D:\phpStudy\MySQL\bin”，如果在其他目录下执行 MySQL 命令则会提示“‘mysql’不是内部或外部命令”错误信息，如图 8.8 所示。

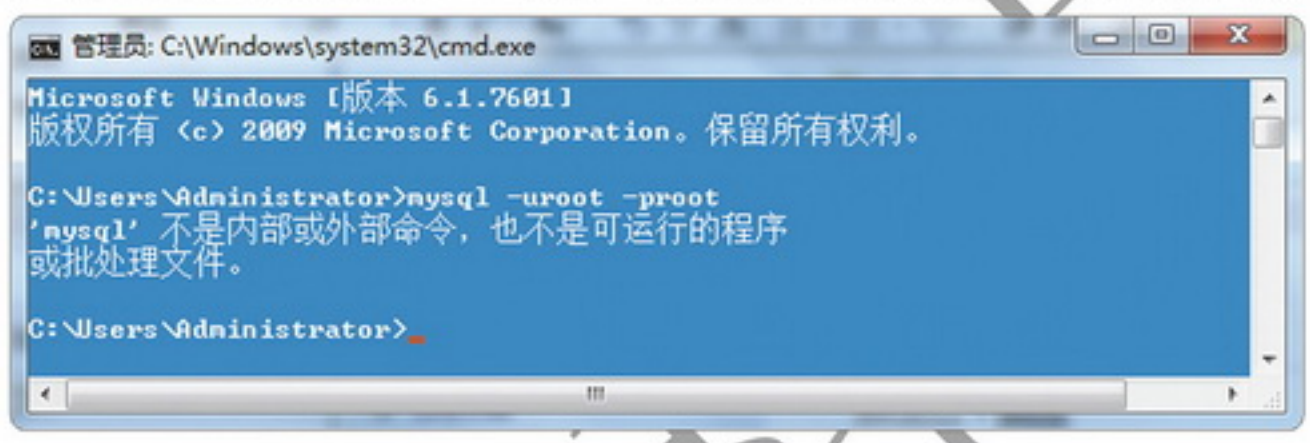


图 8.8 “mysql”不是内部或外部命令的错误提示

通过设置环境变量，可以实现在任何目录下都能使用 MySQL 命令的功能。下面介绍设置环境变量的方法。其步骤如下：

（1）鼠标右键单击“计算机”图标，选择“属性”，在弹出的对话框中单击“高级系统设置”，如图 8.9 所示。在弹出的“系统属性”对话框中，单击“环境变量”，如图 8.10 所示。

图 8.9　高级系统设置对话框

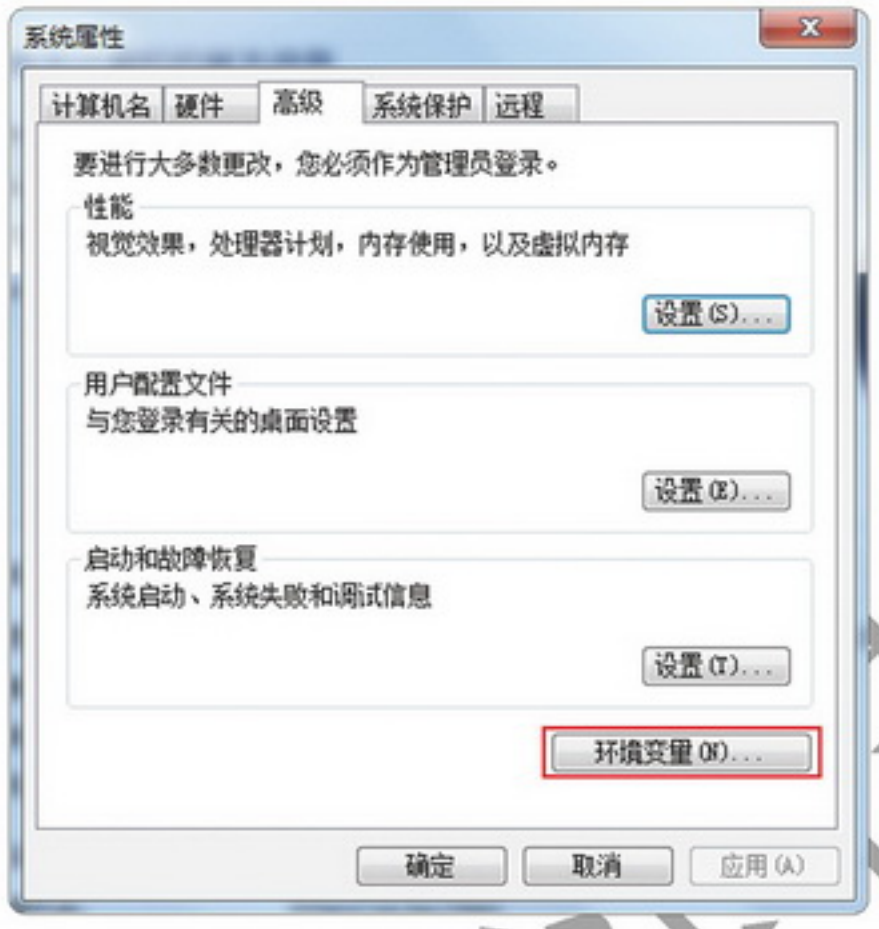


图 8.10 “系统属性”对话框

（2）在弹出的“环境变量”对话框中，选择“PATH”，如图 8.11 所示。单击“编辑”按钮，将弹出“编辑用户变量”对话框，如图 8.12 所示。

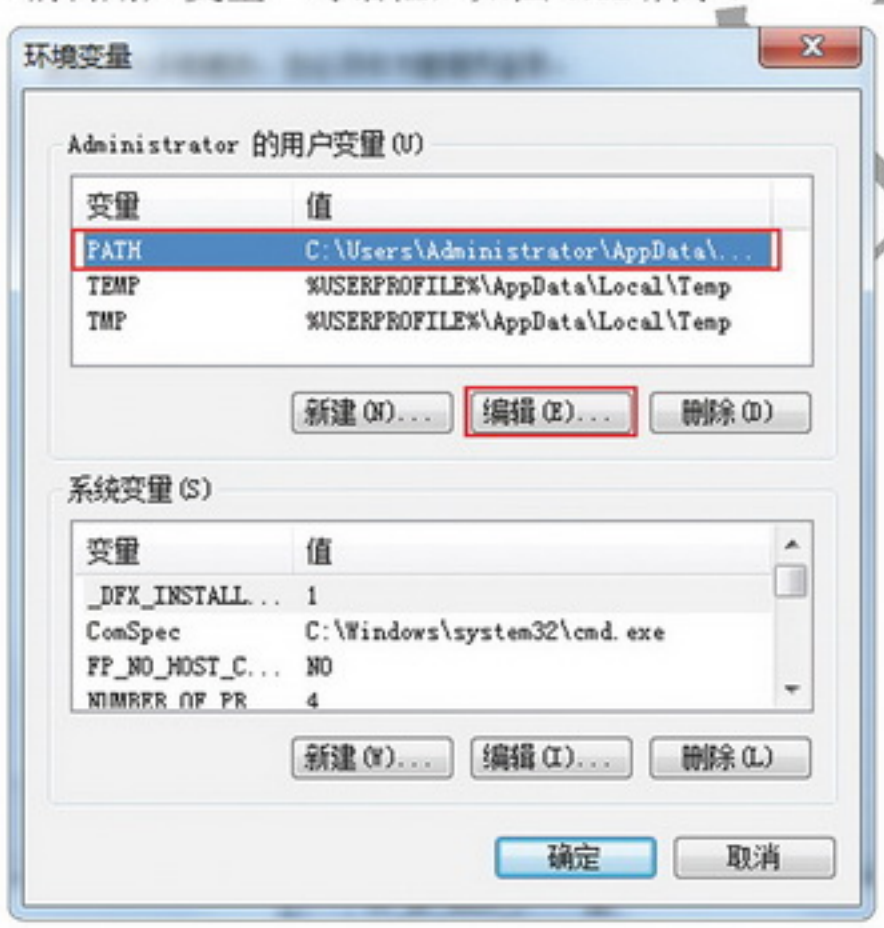


图 8.11 “环境变量”对话框

图 8.12 “编辑用户变量”对话框

在“编辑用户变量”对话框中，将 MySQL 服务器的 bin 文件夹路径（D:\phpStudy\MySQL\bin）添加到变量值文本框中，注意要使用“;”与其他变量值进行分隔，最后，单击“确定”按钮。环境变量设置完成后，即可在任何目录使用 MySQL 命令了。例如，在 cmd 进入的初始目录中使用 MySQL 命令，如图 8.13 所示。

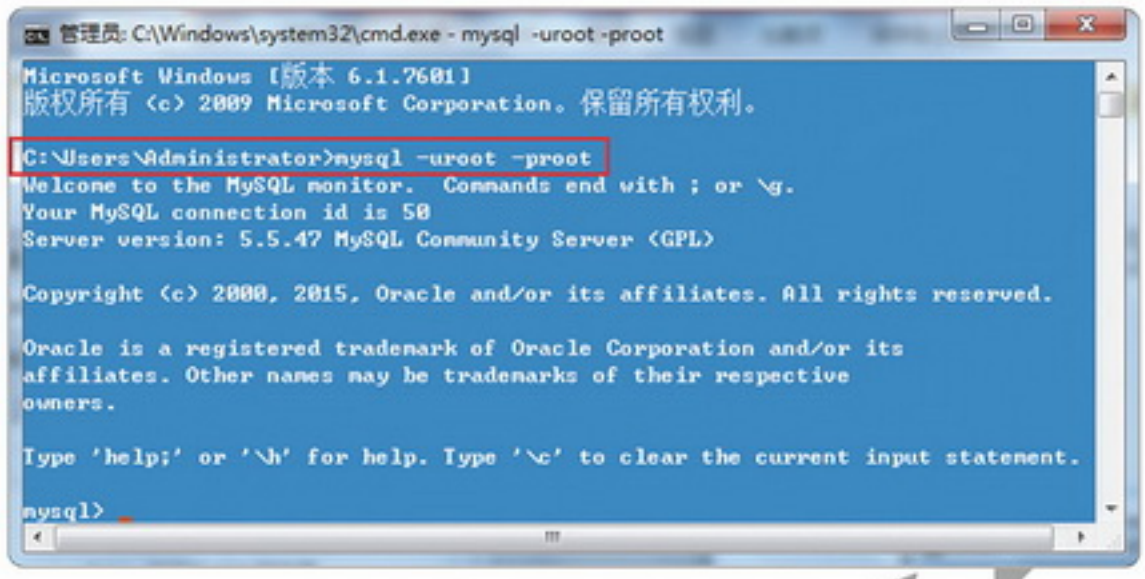


图 8.13　在初始目录使用 MySQL 命令

8.3　操作 MySQL 数据库

针对 MySQL 数据库的操作可以分为创建、选择、查看和删除 4 种，下面分别进行介绍。

8.3.1　创建数据库

在 MySQL 中，应用 create database 语句创建数据库。其语法格式如下：

```
create database 数据库名;
```

在创建数据库时，数据库的命名要遵循如下规则：

◆ 不能与其他数据库重名。

◆ 名称可以由任意字母、阿拉伯数字、下划线（_）或者“$”组成，可以使用上述的任意字符开头，但不能使用单独的数字，否则会造成它与数值相混淆。

◆ 名称最长可为 64 个字符组成（还包括表、列和索引的命名），而别名最多可达 256 个字符。

◆ 不能使用 MySQL 关键字作为数据库、表名。

◆ 默认情况下，Windows 下数据库名、表名的字母大小写是不敏感的，而在 Linux 下数据库名、表名的字母大小写是敏感的。为了便于数据库在平台间进行移植，建议读者采用小写字母来定义数据库名和表名。

下面通过 create database 语句创建一个名称为 db_users 的数据库。在创建数据库时，首先连接 MySQL 服务器，然后编写“create database db_users;”SQL 语句，数据库创建成功。运行结果如图 8.14 所示。

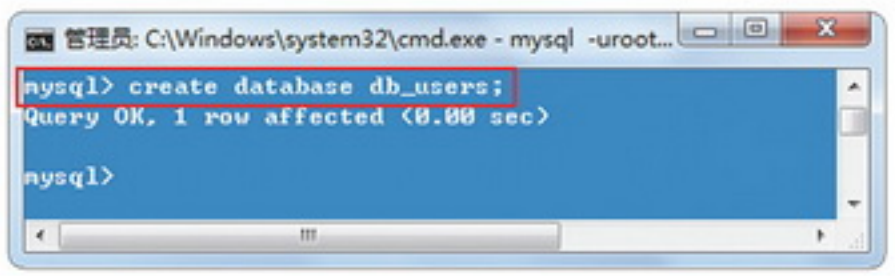


图 8.14　创建数据库

创建 db_users 数据库后，MySQL 管理系统会自动在“D:\phpStudy\MySQL\data”目录下创建 db_users 数据库文件夹及相关文件，实现对该数据库的文件管理。

📖 **说明：** “D:\phpStudy\MySQL\data”目录是 MySQL 配置文件 my.ini 中设置的数据库文件的存储目录。用户可以通过修改配置选项 datadir 的值来对数据库文件的存储目录进行重新设置。

8.3.2 选择数据库

use 语句用于选择一个数据库，使其成为当前默认数据库。其语法如下：

```
use 数据库名;
```

例如，选择名称为 db_users 的数据库，操作命令如图 8.15 所示。

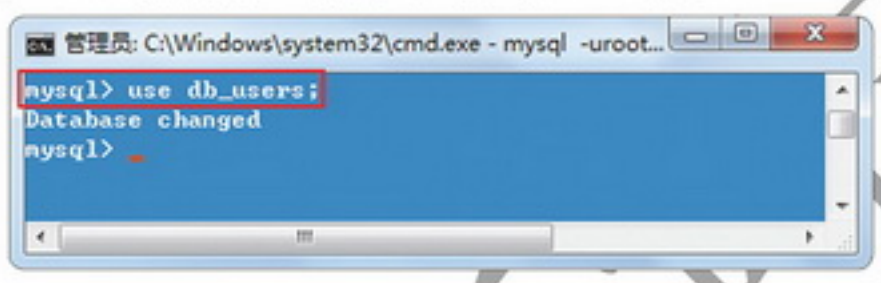


图 8.15　选择数据库

选择了 db_users 数据库之后，才可以操作该数据库中的所有对象。

8.3.3 查看数据库

数据库创建完成后，可以使用 show databases 命令查看 MySQL 数据库中所有已经存在的数据库。语法如下：

```
show databases
```

例如，使用 show databases 命令显示本地 MySQL 数据库中所有存在的数据库名，如图 8.16 所示。

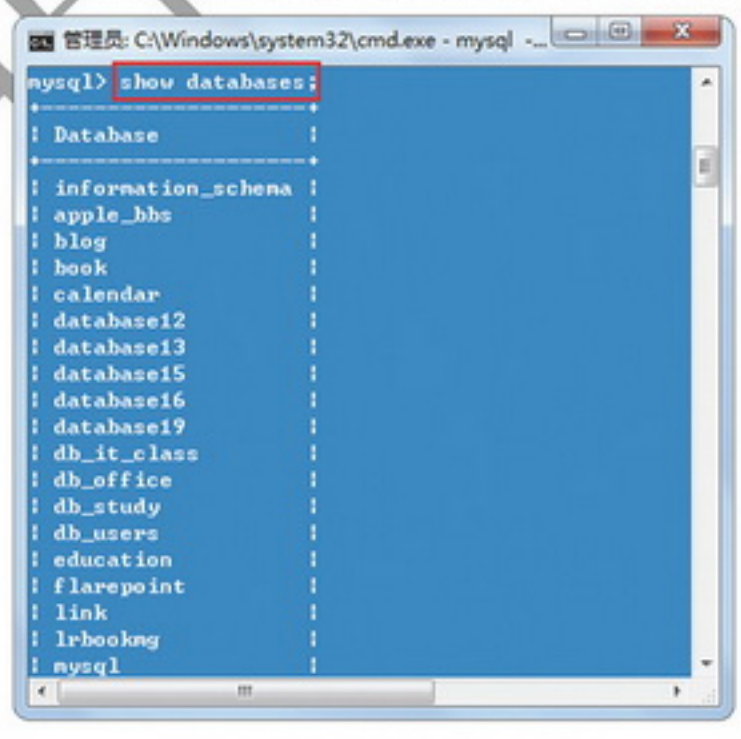


图 8.16　显示所有数据库名

注意：“show databases”是复数形式，并且所有命令都以英文分号“;”结尾。

8.3.4 删除数据库

删除数据库使用的是 drop database 语句，语法如下：

```
drop database 数据库名;
```

例如，在 MySQL 命令窗口中使用“drop database db_users;”SQL 语句即可删除 db_users 数据库，如图 8.17 所示。删除数据库后，MySQL 管理系统会自动删除“D:\phpStudy\MySQL\data”目录下的 db_users 目录及相关文件。

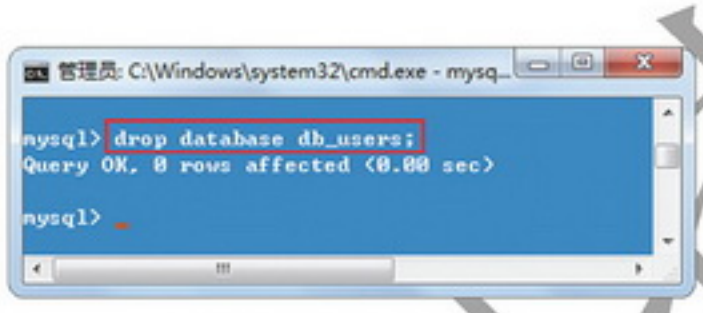


图 8.17　删除数据库

注意：对于删除数据库的操作，应该谨慎使用，一旦执行这项操作，数据库的所有结构和数据都会被删除，没有恢复的可能，除非数据库有备份。

8.4 MySQL 数据类型

在 MySQL 数据库中，每一条数据都有其数据类型。MySQL 支持的数据类型主要分成三类：数字类型、字符串（字符）类型、日期和时间类型。

8.4.1 数字类型

MySQL 支持的数字类型包括准确数字的数据类型（NUMERIC、DECIMAL、INTEGER 和 SMALLINT），还包括近似数字的数据类型（FLOAT、REAL 和 DOUBLE PRECISION）。其中，关键字 INT 是 INTEGER 的简写，关键字 DEC 是 DECIMAL 的简写。

一般来说，数字类型可以分成整型和浮点型两类，详细内容如表 8.1 和表 8.2 所示。

表 8.1　整型数据类型

数 据 类 型	取 值 范 围	说　明	存 储 容 量
TINYINT	有符号值：–128 ～ 127 无符号值：0 ～ 255	最小的整数	1 字节
BIT	有符号值：–128 ～ 127 无符号值：0 ～ 255	最小的整数	1 字节

续表

数据类型	取值范围	说明	存储容量
BOOL	有符号值：-128 ～ 127 无符号值：0 ～ 255	最小的整数	1 字节
SMALLINT	有符号值：-32768 ～ 32767 无符号值：0 ～ 65535	小型整数	2 字节
MEDIUMINT	有符号值：-8388608 ～ 8388607 无符号值：0 ～ 16777215	中型整数	3 字节
INT	有符号值：-2147683648 ～ 2147683647 无符号值：0 ～ 4294967295	标准整数	4 字节
BIGINT	有符号值： -9223372036854775808 ～ 9223372036854775807 无符号值：0 ～ 18446744073709551615	大型整数	8 字节

表 8.2　浮点型数据类型

数据类型	取值范围	说明	存储容量
FLOAT	-3.402823466E+38 ～ +3.402823466E+38	单精度浮点数	8 字节或 4 字节
DOUBLE	-1.7976931348623157E+308 ～ +1.7976931348623157E+308	双精度浮点数	8 字节
DECIMAL	可变	一般整数	自定义长度

📖 **说明**：在创建表时，使用哪种数字类型，应遵循以下原则：

（1）选择最小的可用类型，如果值永远不超过 127，则使用 TINYINT 要比使用 INT 好。

（2）对于完全都是数字的，可以选择整数类型。

（3）浮点类型用于可能具有小数部分的数。例如，货物单价、网上购物支付金额等。

8.4.2 字符串类型

字符串类型可以分为三类：普通的文本字符串类型（CHAR 和 VARCHAR）、可变类型（TEXT 和 BLOB）和特殊类型（SET 和 ENUM）。它们之间都有一定的区别，取值的范围不同，应用的地方也不同。

（1）普通的文本字符串类型，即 CHAR 和 VARCHAR 类型，CHAR 列的长度在创建表时指定，取值在 0 ～ 255 之间；VARCHAR 列的值是变长的字符串，取值和 CHAR 一样。普通的文本字符串类型如表 8.3 所示。

表 8.3　普通的文本字符串类型

类　　型	取 值 范 围	说　　明
[national] CHAR(M) [binary|ASCII|unicode]	0 ～ 255	固定长度为 M 的字符串，其中 M 的取值范围为 0 ～ 255。national 关键字指定了应该使用的默认字符集。binary 关键字指定了数据是否区分大小写（默认是区分大小写的）。ASCII 关键字指定了在该列中使用 latin1 字符集。unicode 关键字指定了使用 UCS 字符集
CHAR	0 ～ 255	CHAR(M) 类似
[national] VARCHAR(M) [binary]	0 ～ 65535	长度可变，其他和 CHAR(M) 类似

（2）TEXT 和 BLOB 类型。它们的大小可以改变，TEXT 类型适合存储长文本，而 BLOB 类型适合存储二进制数据，支持任何数据，如文本、声音和图像等。TEXT 和 BLOB 类型如表 8.4 所示。

表 8.4　TEXT 和 BLOB 类型

类　　型	取 值 范 围	说　　明
TINYBLOB(n)	达到 n(<=255)	小 BLOB 字段
TINYTEXT(n)	达到 n(<=255)	小 TEXT 字段
BLOB(n)	达到 n(<=65535)	常规 BLOB 字段
TEXT(n)	达到 n(<=65535)	常规 TEXT 字段
MEDIUMBLOB(n)	达到 n(<=1.67E+7)	中型 BLOB 字段
MEDIUMTEXT(n)	达到 n(<=1.67E+7)	中型 TEXT 字段
LONGBLOB(n)	达到 n(<=4.29E+9)	长 BLOB 字段
LONGTEXT(n)	达到 n(<=4.29E+9)	长 TEXT 字段

（3）特殊类型 SET 和 ENUM

特殊类型 SET 和 ENUM 的介绍如表 8.5 所示。

表 8.5　SET 和 ENUM 类型

类　　型	说　　明	存 储 容 量
Enum ('value1', 'value2', …)	该类型的列只可以容纳所列值之一或为 NULL	1 或 2 字节
Set ('value1', 'value2', …)	该类型的列可以容纳一组值或为 NULL	1,2,3,4 或 8 字节

📖 **说明**：在创建表时，使用字符串类型时应遵循以下原则：

（1）从速度方面考虑，要选择固定的列，可以使用 CHAR 类型。

（2）要节省空间，使用动态的列，可以使用 VARCHAR 类型。

（3）要将列中的内容限制在一种选择，可以使用 ENUM 类型。
（4）允许在一列中有一个数组，可以使用 SET 类型。
（5）如果要搜索的内容不区分大小写，可以使用 TEXT 类型。
（6）如果要搜索的内容区分大小写，可以使用 BLOB 类型。

8.4.3 日期和时间类型

日期和时间类型包括：DATETIME、DATE、TIMESTAMP、TIME 和 YEAR，每种类型都有相应的取值范围，如赋予它一个不合法的值，将会被“0”代替。日期和时间数据类型如表 8.6 所示。

表 8.6 日期和时间数据类型

类　型	取值范围	说　明
DATE	1000-01-01 9999-12-31	日期，格式 YYYY-MM-DD
TIME	-838:58:59 835:59:59	时间，格式 HH:MM:SS
DATETIME	1000-01-01 00:00:00 9999-12-31 23:59:59	日期和时间，格式 YYYY-MM-DD HH:MM:SS
TIMESTAMP	1970-01-01 00:00:01 2038-01-19 03:14:07	时间标签，在处理报告时使用的显示格式取决于 M 的值
YEAR	1901-2155	年份可指定两位数字和四位数字的格式

在 MySQL 中，日期的顺序是按照标准的 ANSISQL 格式进行输入的。

8.5 操作数据表

数据库创建完成后，即可在命令提示符下对数据库中的数据表进行操作，如创建数据表、更改数据表结构以及删除数据表等。

8.5.1 创建数据表

MySQL 数据库中，可以使用 create table 命令创建数据表。语法如下：

```
create[TEMPORARY] table [IF NOT EXISTS] 数据表名
[(create_definition,…)][table_options] [select_statement]
```

create table 语句的参数说明如表 8.7 所示。

表 8.7　create table 语句的参数说明

参　数	说　明
TEMPORARY	如果使用该关键字，表示创建一个临时表
IF NOT EXISTS	该关键字用于避免表存在时 MySQL 报告错误
create_definition	这是表的列属性部分。MySQL 要求在创建表时，表要至少包含一列
table_options	表的一些特性参数
select_statement	SELECT 语句描述部分，用它可以快速地创建表

下面介绍列属性 create_definition 的使用方法，每一列具体的定义格式如下：

```
col_name  type [NOT NULL | NULL] [DEFAULT default_value] [AUTO_INCREMENT]
               [PRIMARY KEY ] [reference_definition]
```

属性 create_definition 的参数说明如表 8.8 所示。

表 8.8　属性 create_definition 的参数说明

参　数	说　明
col_name	字段名
type	字段类型
NOT NULL | NULL	指出该列是否允许是空值，但是数据“0”和空格都不是空值，系统一般默认允许为空值，所以当不允许为空值时，必须使用 NOT NULL
DEFAULT default_value	表示默认值
AUTO_INCREMENT	表示是否是自动编号，每个表只能有一个 AUTO_INCREMENT 列，并且必须被索引
PRIMARY KEY	表示是否为主键。一个表只能有一个 PRIMARY KEY。如表中没有 PRIMARY KEY，而某些应用程序要求 PRIMARY KEY，MySQL 将返回第一个没有任何 NULL 列的 UNIQUE 键，作为 PRIMARY KEY
reference_definition	为字段添加注释

在实际应用中，使用 create table 命令创建数据表时，只需指定最基本的属性即可，格式如下：

```
create table table_name (列名1 属性, 列名2 属性 …);
```

例如，在命令提示符下应用 create database db_users 创建 db_users 数据库，然后使用 create table 命令在数据库 db_users 中创建一个名为 tb_users 的数据表，表中包括 id、user、pwd 和 createtime 等字段，实现过程如图 8.18 所示。

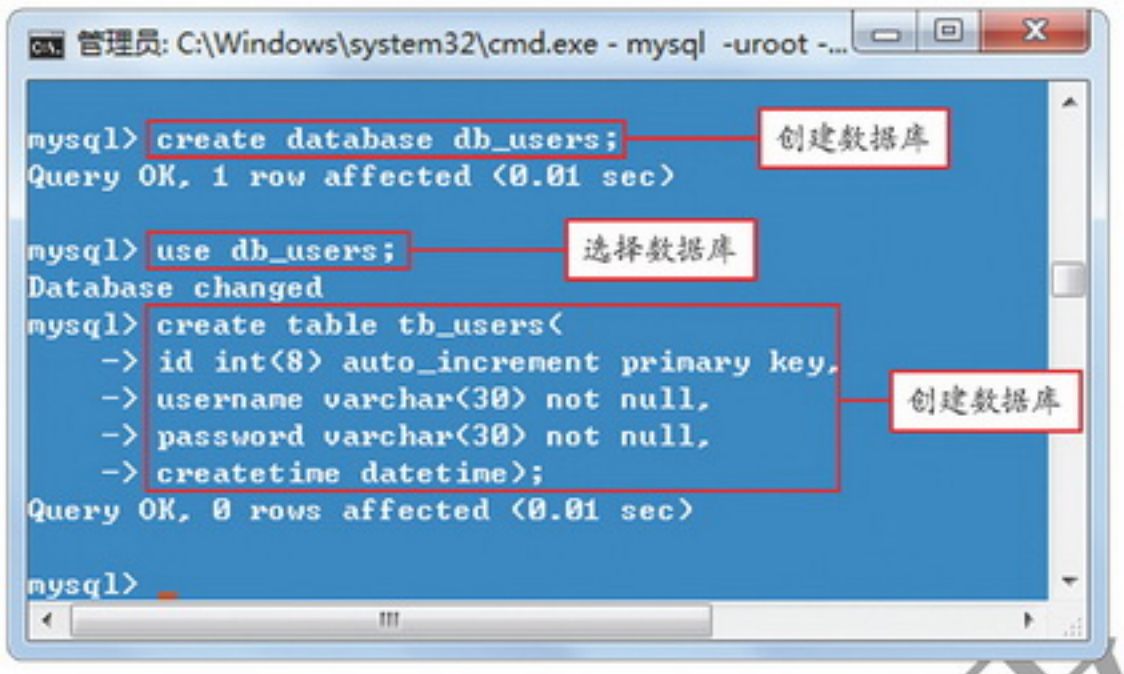


图 8.18　创建 MySQL 数据表

📖 **说明**：按下 <Enter> 键即可换行，结尾分号（;）表示该行语句结束。

8.5.2 查看表结构

成功创建数据表后，可以使用 show columns 命令或 describe 命令查看指定数据表的表结构。下面分别对这两个语句进行介绍。

1. show columns 命令

show columns 命令的语法格式如下：

```
show [full] columns  from 数据表名 [from 数据库名];
```

或写成：

```
show  [full] columns  FROM 数据库名.数据表名;
```

例如，应用 show columns 命令查看数据表 tb_users 表结构，如图 8.19 所示。

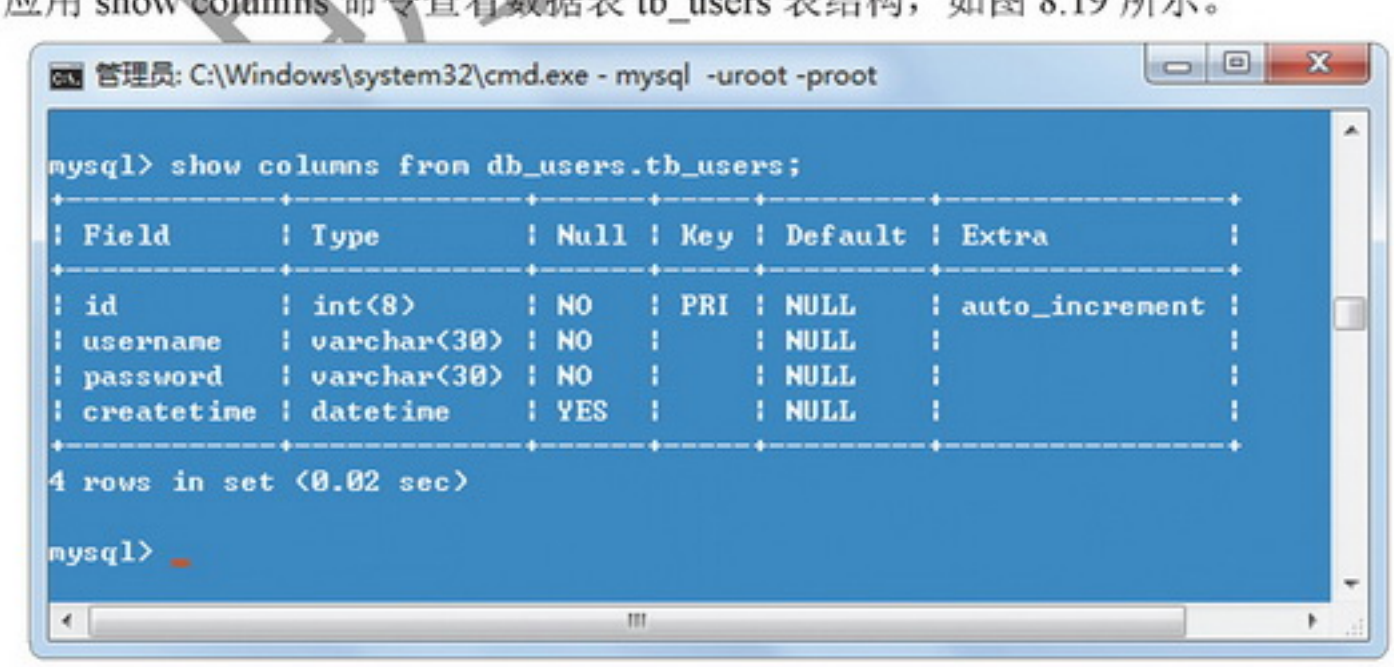


图 8.19　查看 tb_users 表结构

2. describe 命令

describe 命令的语法格式如下：

```
describe 数据表名;
```

其中，describe 可以简写为 desc。在查看表结构时，也可以只列出某一列的信息，语法格式如下：

```
describe 数据表名 列名;
```

例如，应用 describe 命令的简写形式查看数据表 tb_users 的某一列信息，如图 8.20 所示。

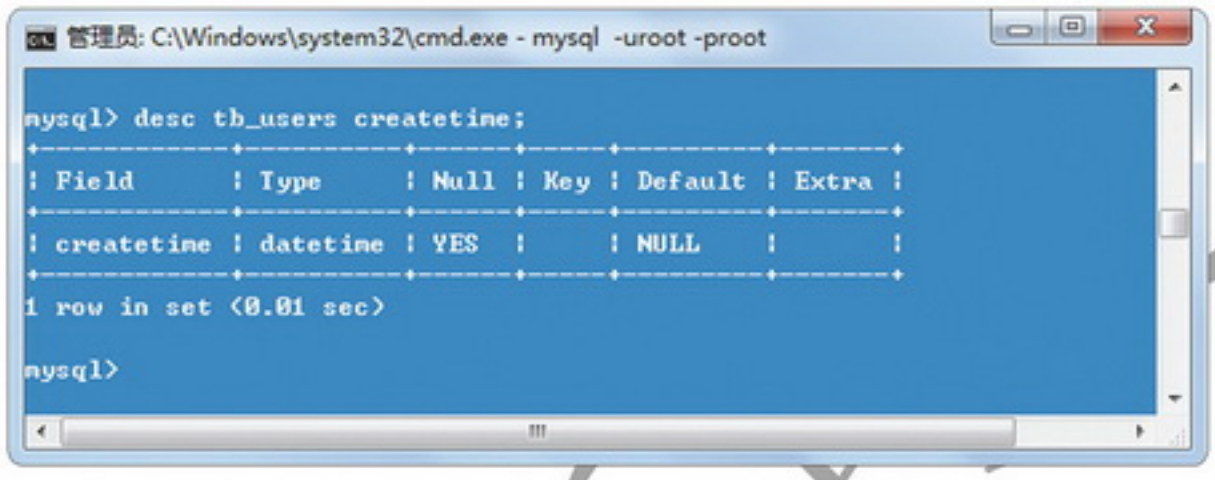


图 8.20　查看 tb_users 表 createtime 列的信息

8.5.3 修改表结构

修改表结构采用 alter table 命令。修改表结构指增加或者删除字段、修改字段名称或者字段类型、设置取消主键或外键、设置取消索引以及修改表的注释等。

语法如下：

```
alter [IGNORE] table 数据表名 alter_spec[,alter_spec]…
```

注意，当指定IGNORE时，如果出现重复关键的行，则只执行一行，其他重复的行被删除。其中，alter_spec 子句用于定义要修改的内容，语法如下：

```
alter_specification:
    ADD [COLUMN] create_definition [FIRST | AFTER column_name ]   --添加新字段
    |ADD INDEX [index_name] (index_col_name,…)                    --添加索引名称
    |ADD PRIMARY KEY (index_col_name,…)                           --添加主键名称
    |ADD UNIQUE [index_name] (index_col_name,…)                   --添加唯一索引
    |ALTER [COLUMN] col_name {SET DEFAULT literal | DROP DEFAULT}--修改字段名称
    |CHANGE [COLUMN] old_col_name create_definition               --修改字段类型
    |MODIFY [COLUMN] create_definition                           --修改子句定义字段
    |DROP [COLUMN] col_name                                      --删除字段名称
    |DROP PRIMARY KEY                                            --删除主键名称
    |DROP INDEX index_name                                       --删除索引名称
    |RENAME [AS] new_tbl_name                                    --更改表名
    |table_options
```

alter table 语句允许指定多个动作，动作间使用逗号分隔，每个动作表示对表的一个修改。

例如，向 tb_users 表中添加一个新的字段 address，类型为 varchar(60)，并且不为空值“not null”，将字段 username 的类型由 varchar(30) 改为 varchar(50)，然后再用 show colume 命令查看修改后的表结构，如图 8.21 所示。

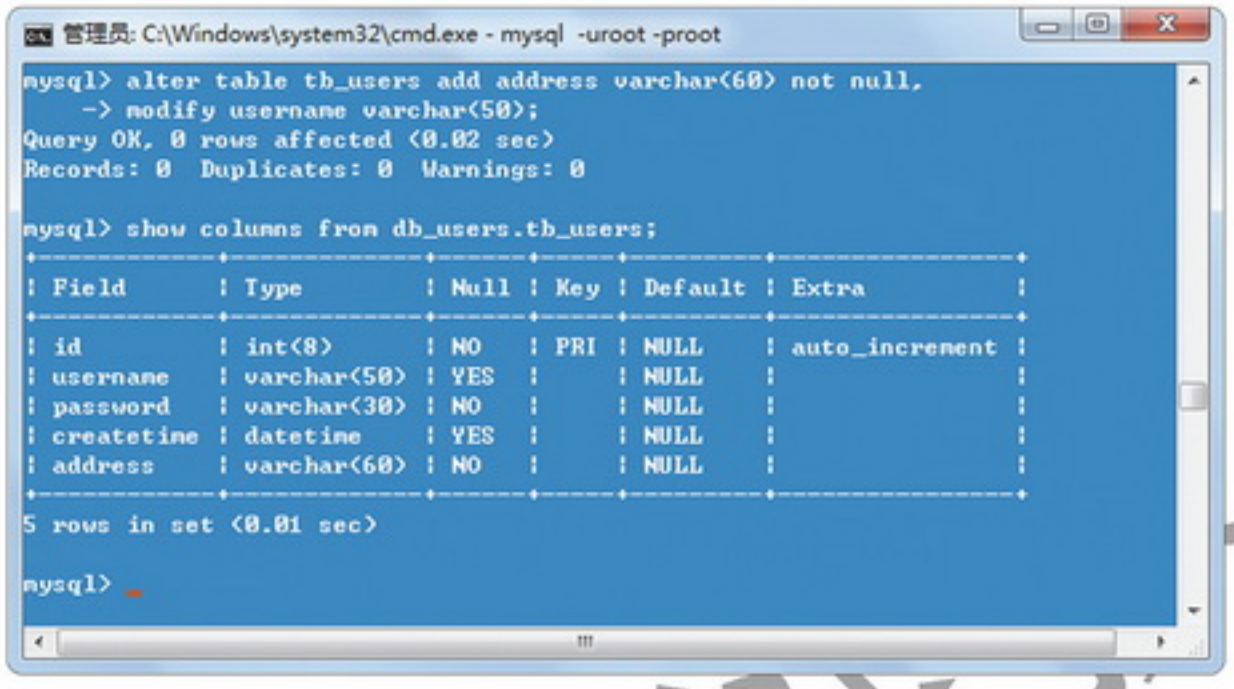


图 8.21　修改 tb_users 表结构

8.5.4 重命名数据表

重命名数据表采用 rename table 命令，语法格式如下：

```
rename table 数据表名1 to 数据表名2;
```

例如，对数据表 tb_users 进行重命名，更名后的数据表为 tb_member，只需要在 MySQL 命令窗口中使用“rename table tb_users to tb_member;”语句即可。此时使用 show columns 查看 tb_users 表将输出错误信息，提示“tb_users 表不存在”。因为 tb_users 表已经变成 tb_member 表，可以使用 show columns 查看 tb_member 表，运行结果如图 8.22 所示。

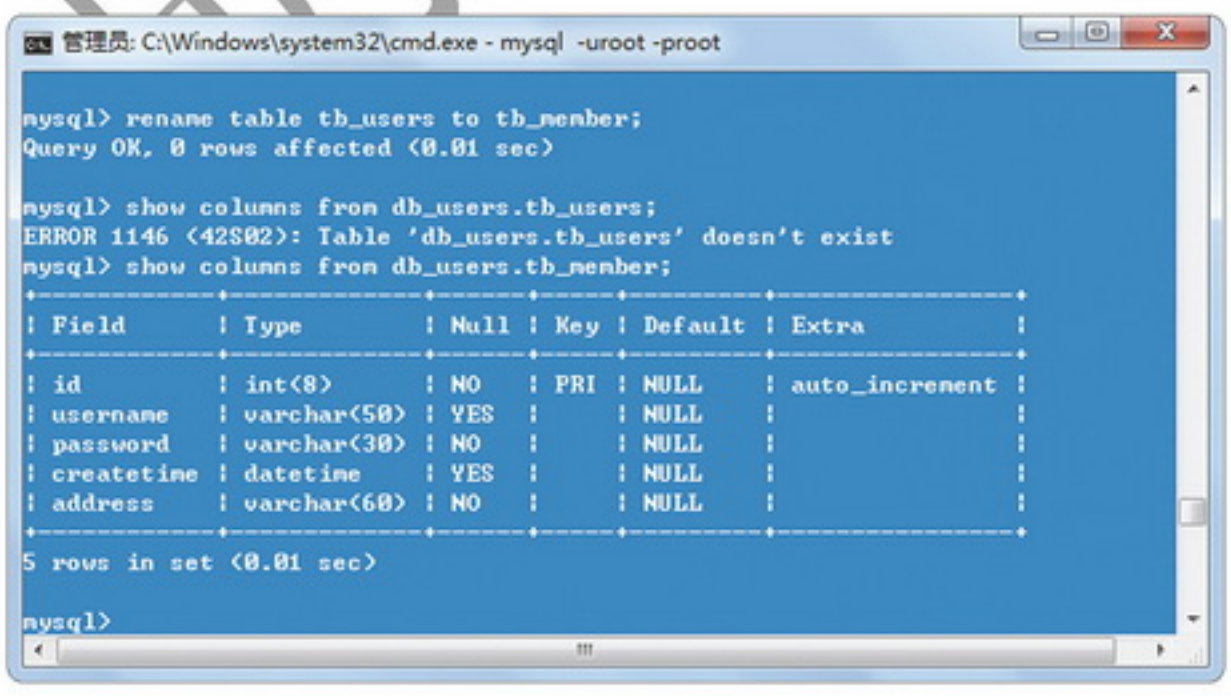


图 8.22　将 tb_users 表名更改为 tb_member 数据表

📖 **说明**：该语句可以同时对多个数据表进行重命名，多个表之间以逗号“,”分隔。

8.5.5 删除数据表

删除数据表的操作很简单，与删除数据库的操作类似，使用 drop table 命令即可实现。格式如下：

```
drop table 数据表名;
```

例如，在 MySQL 命令窗口中使用“drop table tb_member;”语句即可删除 tb_member 数据表。删除数据表后，MySQL 管理系统会自动删除“D:\phpStudy\MySQL\data\db_member”目录下的表文件。

注意：删除数据表的操作应该谨慎使用。一旦删除了数据表，那么表中的数据将会全部清除，如果没有备份则无法恢复。

在删除数据表的过程中，如果删除一个不存在的表将会产生错误，这时在删除语句中加入 if exists 关键字就可以避免出错。格式如下：

```
drop table if exists 数据表名;
```

注意：在对数据表进行操作之前，首先必须选择数据库，否则是无法对数据表进行操作的。

例如，先使用 drop table 语句删除一个 tb_users 表，查看提示信息，然后使用 drop table if exists 语句删除 tb_users 表。运行结果如图 8.23 所示。

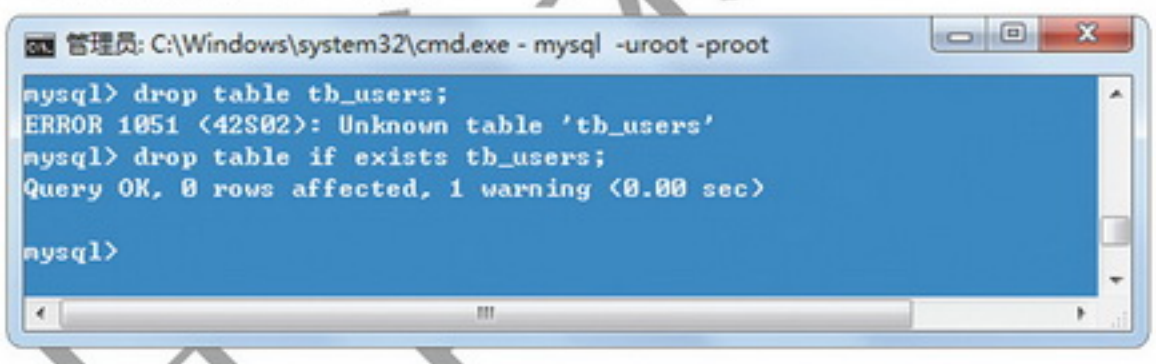


图 8.23　删除 tb_users 数据表

8.6 数据表记录的操作

数据库中包含数据表，而数据表中包含数据。在 MySQL 与 PHP 的结合应用中，真正被操作的是数据表中的数据，因此如何更好地操作和使用这些数据才是使用 MySQL 数据库的根本。

向数据表中插入、修改和删除记录可以在 MySQL 命令行中使用 SQL 语句完成。下面介绍如何在 MySQL 命令行中执行基本的 SQL 语句。

8.6.1 数据表记录的添加

建立一个空的数据库和数据表时，首先要想到的就是如何向数据表中添加数据。这项操作可以通过 insert 命令来实现。

语法格式如下：

```
insert into 数据表名(column_name,column_name2, … ) values (value1, value2, … );
```

在 MySQL 中，一次可以同时插入多行记录，各行记录的值清单在 values 关键字后以逗号“,”分隔，而标准的 SQL 语句一次只能插入一行。

📖 **说明**：值列表中的值应与字段列表中字段的个数和顺序相对应，值列表中值的数据类型必须与相应字段的数据类型保持一致。

例如，向用户信息表 tb_member 中插入一条数据信息，如图 8.24 所示。

图 8.24　向 tb_member 表插入新记录

当向数据表中的所有列添加数据时，insert 语句中的字段列表可以省略，例如，

```
insert into tb_member values('2', '小明','xiaoming','2017-6-20 12:12:12','长春市');
```

8.6.2 数据表记录的查询

向数据表中插入数据后，可以使用 select 命令来查询数据表中的数据。该语句的格式如下：

```
select selection_list  from 数据表名 where condition;
```

其中，selection_list 是要查找的列名，如果要查询多个列，可以用“,”隔开；如果查询所有列，可以用“*”代替。where 子句是可选的，如果给出该子句，将查询出指定记录。

例如，查询 tb_member 表中所有数据。运行结果如图 8.25 所示。

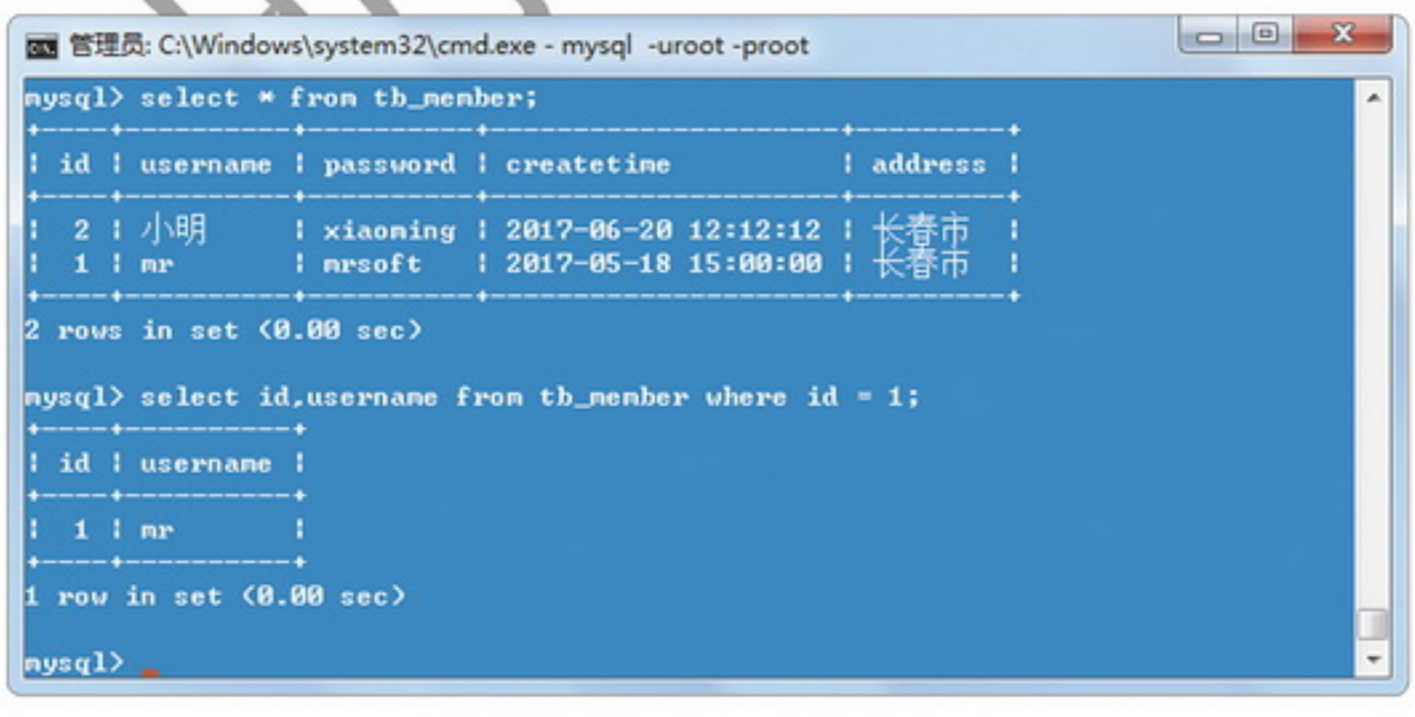


图 8.25　使用 select 命令查找数据

8.6.3 数据表记录的修改

要执行数据修改的操作可以使用 update 命令，该语句的格式如下：

```
update 数据表名set column_name1 = new_value1,column_name2 = new_value2, …where condition;
```

其中，set 子句指出要修改的列及其给定的值；where 子句是可选的，如果给出该子句将指定记录中哪行应该被更新，否则，所有的记录行都将被更新。

例如，将用户信息表 tb_member 中用户名为 mr 的管理员密码“mrsoft”修改为“mingrisoft”，SQL 语句如下：

```
update tb_member set password='mingrisoft' where username='mr';
```

运行结果如图 8.26 所示。

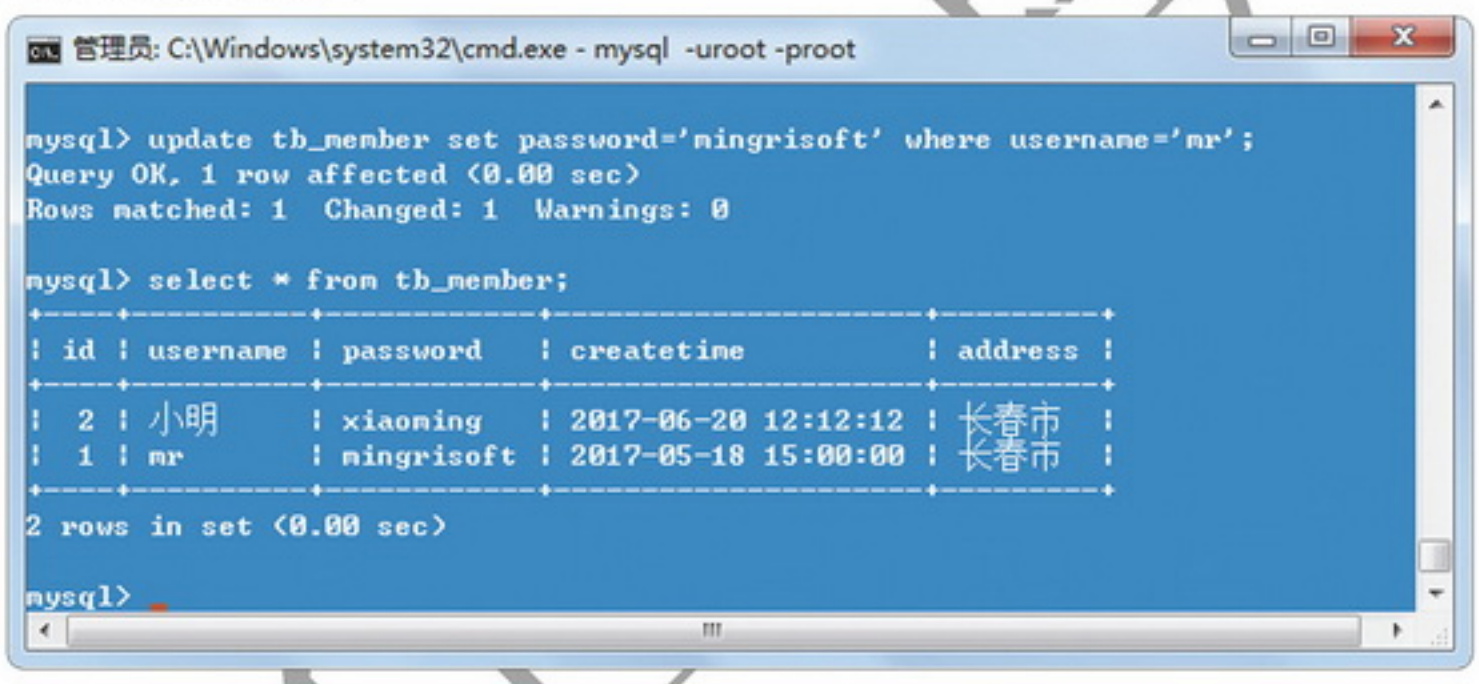


图 8.26　更改数据表记录

8.6.4 数据表记录的删除

在数据库中有些数据已经失去意义或者是错误的，这时就需要将它们删除，此时可以使用 delete 命令。该命令的格式如下：

```
delete from 数据表名 where condition;
```

注意：该语句在执行过程中，如果没有指定 where 条件，将删除所有的记录；如果指定了 where 条件，将按照指定的条件进行删除。

使用 delete 命令删除整个表的效率并不高，还可以使用 truncate 命令，利用它可以快速删除表中所有的内容。

例如，删除用户信息表 tb_member 中用户名为“mr”的记录信息，SQL 语句如下：

```
delete from tb_member where username = 'mr';
```

删除后，使用 select 命令查看结果。运行结果如图 8.27 所示。

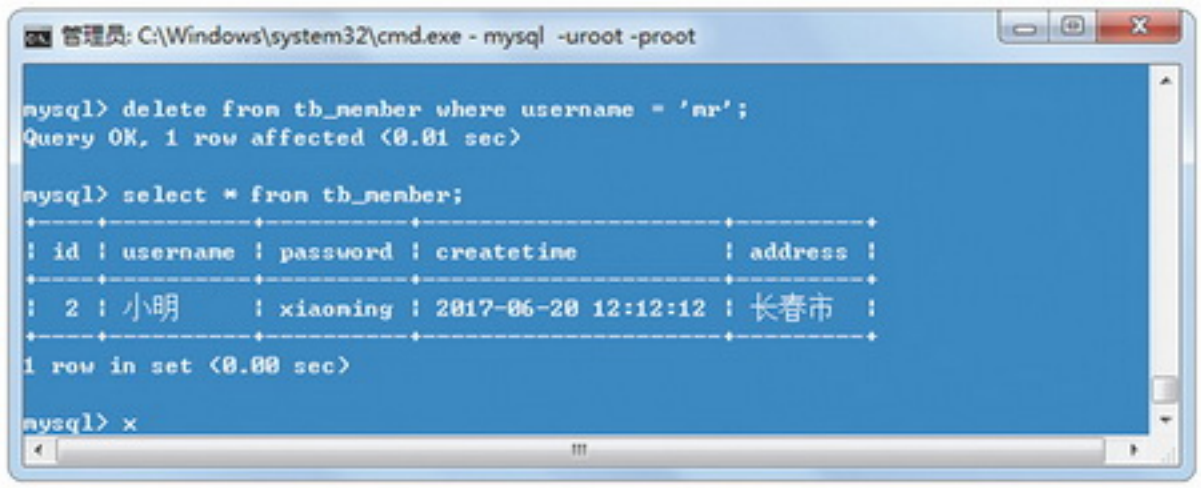


图 8.27　使用 delete 命令删除记录

8.7 数据表记录的查询操作

对于数据表的“增删改查”，最常用的就是查询操作。在 8.6.2 小节中，我们只是介绍了最基础的查询操作，实际应用中查询的条件要复杂的多。再来看一下比较复杂的 select 语法：

```
select selection_list                    --要查询的内容，选择哪些列
from 数据表名                            --指定数据表
where primary_constraint                 --查询时需要满足的条件
group by grouping_columns                --如何对结果进行分组
order by sorting_cloumns                 --如何对结果进行排序
having secondary_constraint              --查询时满足的第二条件
limit count                              --限定输出的查询结果
```

下面对它的参数进行详细的讲解。

1. selection_list

设置查询内容。如果要查询表中所有列，可以将其设置为“*”；如果要查询表中某一列或多列，则直接输入列名，并以“,”为分隔符。例如，查询 tb_mrbook 数据表中所有列和查询 id 和 bookname 列的代码如下：

```
01  select * from tb_mrbook;                 //查询数据表中所有数据
02  select id,bookname from tb_mrbook;       //查询数据表中id和bookname列的数据
```

2. table_list

指定查询的数据表。既可以从一个数据表中查询，也可以从多个数据表中进行查询，多个数据表之间用“,”进行分隔，并且通过 where 子句使用连接运算来确定表之间的联系。

例如，从 tb_mrbook 和 tb_bookinfo 数据表中查询“bookname='PHP 自学视频教程'”的 id 编号、书名、作者和价格，其代码如下：

```
select tb_mrbook.id,tb_mrbook.bookname,
```

```
    -> author,price from tb_mrbook,tb_bookinfo
    -> where tb_mrbook.bookname = tb_bookinfo.bookname and
    -> tb_bookinfo.bookname = 'php自学视频教程';
```

在上面的 SQL 语句中，因为两个表都有 id 字段和 bookname 字段，为了告诉服务器要显示的是哪个表中的字段信息，要加上前缀。语法如下：

```
表名.字段名
```

tb_mrbook.bookname = tb_bookinfo.bookname 将表 tb_mrbook 和 tb_bookinfo 连接起来，叫作等同连接；如果不使用 tb_mrbook.bookname = tb_bookinfo.bookname，那么产生的结果将是两个表的笛卡尔积，叫作全连接。

多学两招：笛卡尔乘积是指在数学中，两个集合 X 和 Y 的笛卡尔积（Cartesian product），又称直积，表示为 X × Y，第一个对象是 X 的成员而第二个对象是 Y 的所有可能有序对的其中一个成员。

3. where 条件语句

在使用查询语句时，如要从很多的记录中查询出想要的记录，就需要一个查询的条件。只有设定了查询的条件，查询才有实际的意义。设定查询条件应用的是 where 子句。

where 子句的功能非常强大，通过它可以实现很多复杂的条件查询。在使用 where 子句时，需要使用一些比较运算符，常用的比较运算符如表 8.9 所示。

表 8.9　where 子句常用的比较运算符

运　算　符	名　　称	示　　例	运　算　符	名　　称	示　　例
=	等于	id=10	is not null	n/a	id is not null
>	大于	id>10	between	n/a	id between1 and 10
<	小于	id<10	in	n/a	id in (4,5,6)
>=	大于或等于	id>=10	not in	n/a	name not in (a,b)
<=	小于或等于	id<=10	like	模式匹配	name like ('abc%')
!= 或 <>	不等于	id!=10	not like	模式匹配	name not like ('abc%')
is null	n/a	id is null	regexp	常规表达式	name 正则表达式

表 8.9 中列举的是 where 子句常用的比较运算符，示例中的 id 是记录的编号，name 是表中的用户名。

例如，应用 where 子句，查询 tb_mrbook 表，条件是 type（类别）为 PHP 的所有图书，代码如下：

```
select * from tb_mrbook where type = 'PHP';
```

4. distinct 在结果中去除重复行

使用 distinct 关键字，可以去除结果中的重复行。

例如，查询 tb_mrbook 表，并在结果中去掉类型字段 type 中的重复数据，代码如下：

```
select distinct type from tb_mrbook;
```

5. order by 对结果排序

使用 order by 可以对查询的结果进行升序和降序（desc）排列，在默认情况下，order by 按升序输出结果。如果要按降序排列可以使用 desc 来实现。

对含有 NULL 值的列进行排序时，如果是按升序排列，NULL 值将出现在最前面，如果是按降序排列，NULL 值将出现在最后。例如，查询 tb_mrbook 表中的所有信息，按照“id”进行降序排列，并且只显示五条记录。其代码如下：

```
select * from tb_mrbook order by id desc limit 5;
```

6. like 模糊查询

like 属于较常用的比较运算符，通过它可以实现模糊查询。它有两种通配符：“%”和下划线“_”。“%”可以匹配一个或多个字符，而“_”只匹配一个字符。例如，查找所有书名（bookname 字段）包含 PHP 的图书，代码如下：

```
select * from tb_mrbook where bookname like('%PHP%');
```

📖 说明：无论是一个英文字符还是中文字符都算作一个字符，在这一点上英文字母和中文没有什么区别。

7. concat 联合多列

使用 concat 函数可以联合多个字段，构成一个总的字符串。例如，把 tb_mrbook 表中的书名（bookname）和价格（price）合并到一起，构成一个新的字符串。代码如下：

```
select id,concat(bookname,":",price) as info,type from tb_mrbook;
```

其中，合并后的字段名为 concat 函数形成的表达式“bookname:price”，看上去十分复杂，通过 as 关键字给合并字段取一个别名，这样看上去就清晰了。如《PHP 项目开发实战入门》这本书定价为 69.80 元，concat 查询的结果中的 info 字段值则是“PHP 项目开发实战入门 :69.80”。

8. limit 限定结果行数

limit 子句可以对查询结果的记录条数进行限定，控制它输出的行数。例如，查询 tb_mrbook 表，并按照图书价格升序排列显示十条记录，代码如下：

```
select * from tb_mrbook order by price asc limit 10;
```

使用 limit 还可以从查询结果的中间部分取值。首先要定义两个参数，参数 1 是开始读取的第一条记录的编号（在查询结果中，第一个结果的记录编号是 0，而不是 1）；参数 2 是要查询记录的个数。

例如，查询 tb_mrbook 表，从第 3 条记录开始，查询 6 条记录，代码如下：

```
select * from tb_mrbook limit 2,6;
```

9. 使用函数和表达式

在 MySQL 中，还可以使用表达式来计算各列的值，作为输出结果。表达式还可以包含一些函数。

例如，计算 tb_mrbook 表中各类图书的总价格，代码如下：

```
select sum(price) as totalprice,type from tb_mrbook group by type;
```

在对 MySQL 数据库进行操作时，有时需要对数据库中的记录进行统计，例如求平均值、最小值、最大值等，这时可以使用 MySQL 中的统计函数，其常用的统计函数如表 8.10 所示。

表 8.10　MySQL 中常用的统计函数

名　称	说　明
avg（字段名）	获取指定列的平均值
count（字段名）	如指定了一个字段，则会统计出该字段中的非空记录。如在前面增加 DISTINCT，则会统计不同值的记录，相同的值当作一条记录。如使用 COUNT（*）则统计包含空值的所有记录数
min（字段名）	获取指定字段的最小值
max（字段名）	获取指定字段的最大值
std（字段名）	指定字段的标准背离值
stdtev（字段名）	与 STD 相同
sum（字段名）	获取指定字段所有记录的总和

除了使用函数之外，还可以使用算术运算符、字符串运算符，以及逻辑运算符来构成表达式。例如，计算图书打九折之后的价格，代码如下：

```
select *, (price * 0.9) as '90%' from tb_mrbook;
```

10. group by 对结果分组

通过 group by 子句可以将数据划分到不同的组中，实现对记录进行分组查询。在查询时，所查询的列必须包含在分组的列中，目的是使查询到的数据没有矛盾。在与 avg() 函数或 sum() 函数一起使用时，group by 子句能发挥最大作用。例如，查询 tb_mrbook 表，按照 type 进行分组，求每类图书的平均价格，代码如下：

```
select avg(price),type from tb_mrbook group by type;
```

11. 使用 having 子句设定第二个查询条件

having 子句通常和 group by 子句一起使用。在对数据结果进行分组查询和统计之后，还可以使用 having 子句来对查询的结果进行进一步的筛选。having 子句和 where 子句都用于指定查询条件，不同的是 where 子句在分组查询之前应用，而 having 子句在分组查询之后应用，而且 having 子句中还可以包含统计函数。例如，计算 tb_mrbook 表中各类图书的平均价格，并筛选出图书的平均价格大于 60 的记录，代码如下：

```
select avg(price),type from tb_mrbook group by type having avg(price)>60;
```

8.8 MySQL 中的特殊字符

当 SQL 语句中存在特殊字符时，需要使用“\”对特殊字符进行转义，否则将会出现错误。这些特殊字符及转义后对应的字符如表 8.11 所示。

表 8.11　MySQL 中的特殊字符

特 殊 字 符	转义后的字符	特 殊 字 符	转义后的字符
\'	单引号	\t	制表符
\"	双引号	\0	0 字符
\\	反斜杠	\%	% 字符
\n	换行符	_	_ 字符
\r	回车符	\b	退格符

例如，向用户信息表 tb_member 中添加一条用户名为 O'Neal 的记录，然后查询表中的所有记录，SQL 语句如下：

```
insert into tb_member values(null,'O\'Neal','123456','2015-6-20 12:12:12','长春市');
select * from tb_member;
```

运行结果如图 8.28 所示。

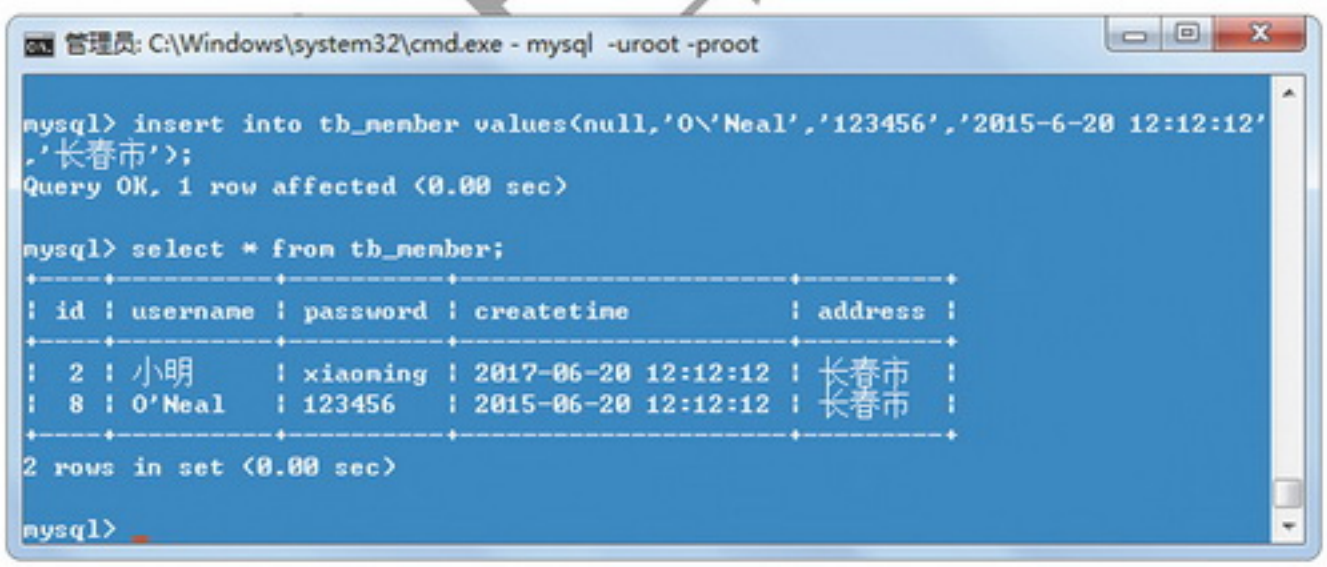


图 8.28　插入记录并查询数据表

8.9 MySQL 图形化管理工具

在命令提示符下操作 MySQL 数据库的方式对 PHP 初学者并不友好，而且需要有专业的 SQL 语

言知识，所以各种 MySQL 图形化管理工具应运而生。下面只简要介绍 phpMyAdmin、Navicat for MySQL 和 MySQL-Front 这三种工具的使用。

8.9.1 phpMyAdmin 简介

phpMyAdmin 是众多 MySQL 图形化管理工具中使用最广泛的一种，是一款使用 PHP 开发的 B/S 模式的 MySQL 客户端软件，该工具是基于 Web 跨平台的管理程序，并且支持简体中文。通过该管理工具可以对 MySQL 进行各种操作，如创建数据库、数据表和生成 MySQL 数据库脚本文件等。

phpStudy 集成开发环境中已经安装了 phpMyAdmin 图形化管理工具，所以用户无需再下载，可以按图 8.29 方式打开。

图 8.29　打开 phpMyAdmin

打开后，输入数据库的用户名“root”，密码“root”，如图 8.30 所示。

图 8.30　phpMyAdmin 连接 MySQL

单击“执行”按钮，进入 phpMyAdmin 管理平台，如图 8.31 所示。

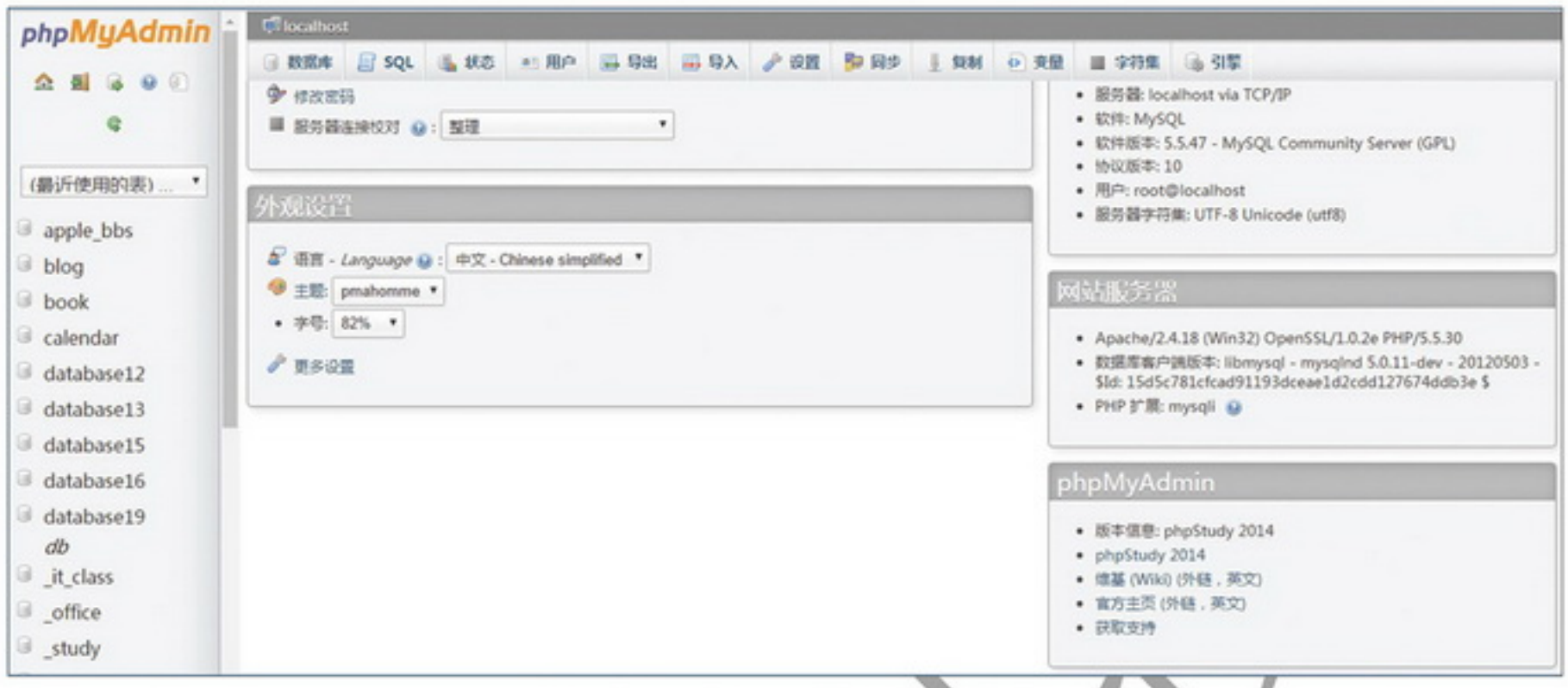


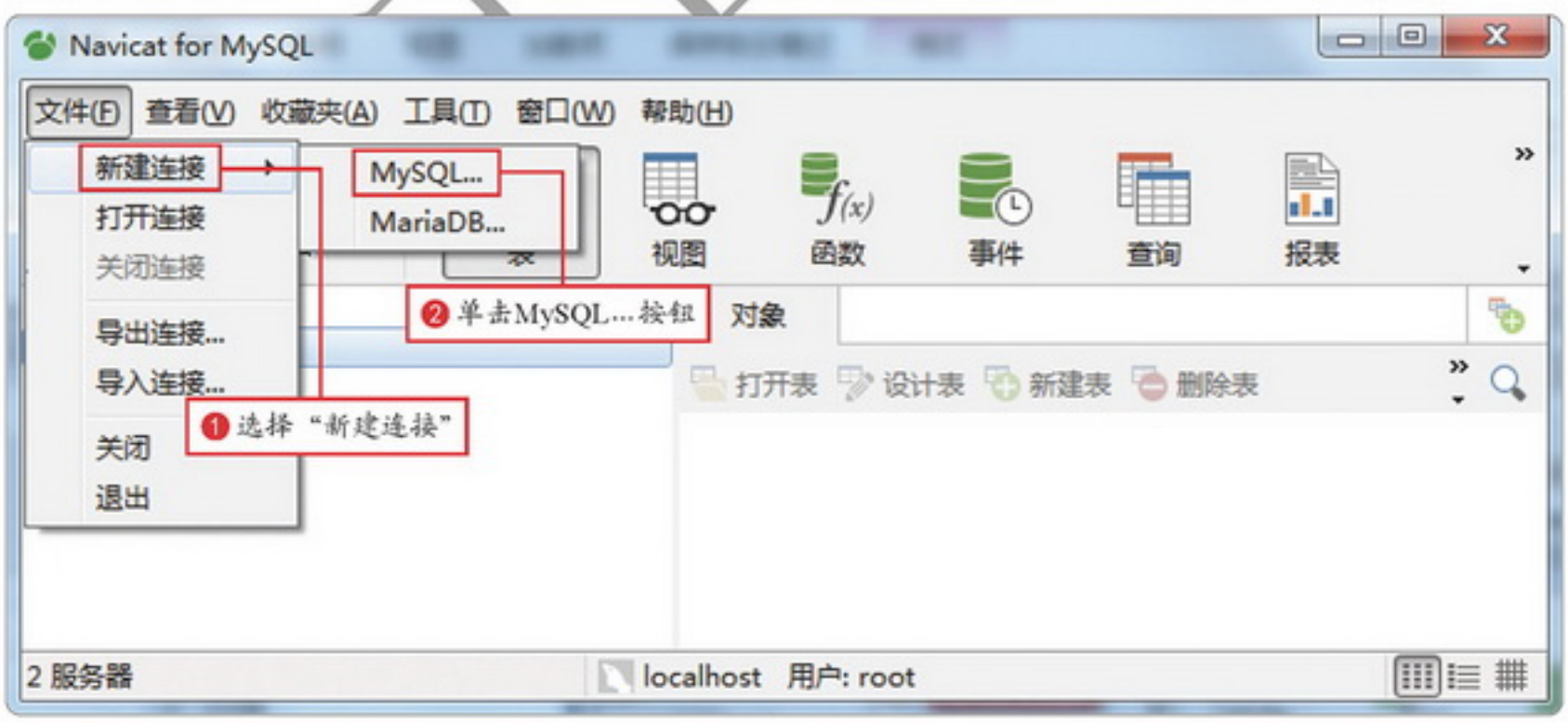
图 8.31　phpMyAdmin 管理平台首页

📖 说明：phpMyAdmin 的更多操作，请查阅相关资料。

8.9.2 Navicat for MySQL 简介

Navicat 是一个桌面版的 MySQL 数据库管理和开发工具。和微软 SQLServer 的管理器很像，易学易用。Navicat 使用图形化的用户界面，可以让用户更为轻松地使用和管理。官方网址为：https://www.navicat.com.cn。

首先下载、安装 Navicat for MySQL，然后按照步骤新建 MySQL 连接，如图 8.32 所示。



8.32　新建 MySQL 连接

在弹出的“连接属性”对话框中，输入连接信息，如图 8.33 所示，其中密码为“root”。

图 8.33　输入连接信息

单击“确定”按钮，创建完成。此时，双击“localhost”，即进入“localhost”数据库。

说明：Navicat for MySQL 的更多操作，请查阅相关资料。

8.9.3 MySQL-Front 简介

MySQL-Front 也是 phpStudy 集成的 MySQL 图形化管理工具，可以按如图 8.34 所示步骤直接打开。

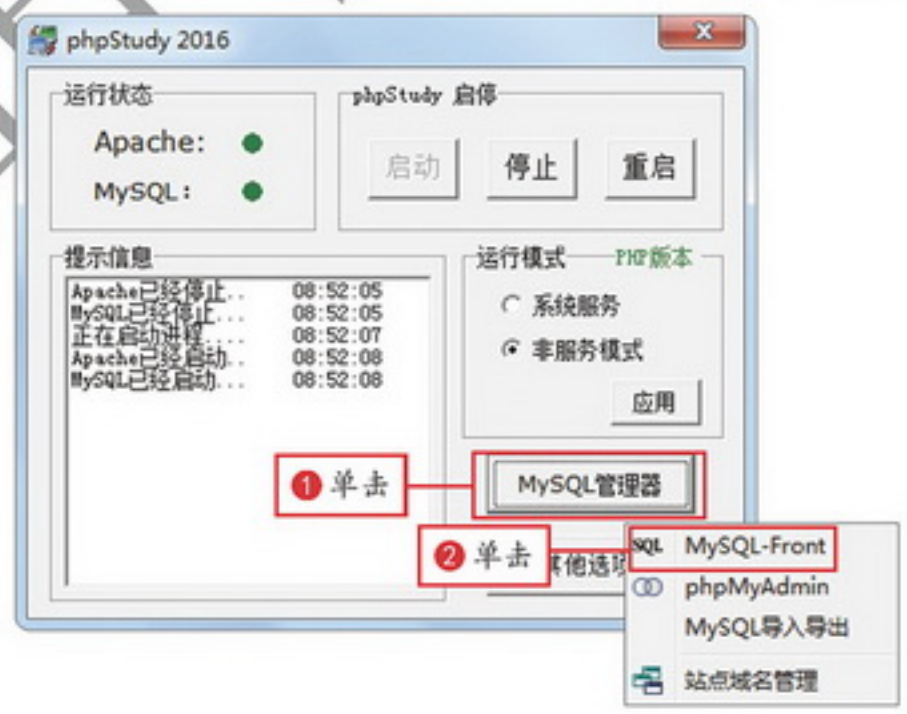


图 8.34　打开 MySQL-Front

打开以后，输入数据库连接信息，与 Navicat for MySQL 的连接方式相同，如图 8.35 所示。

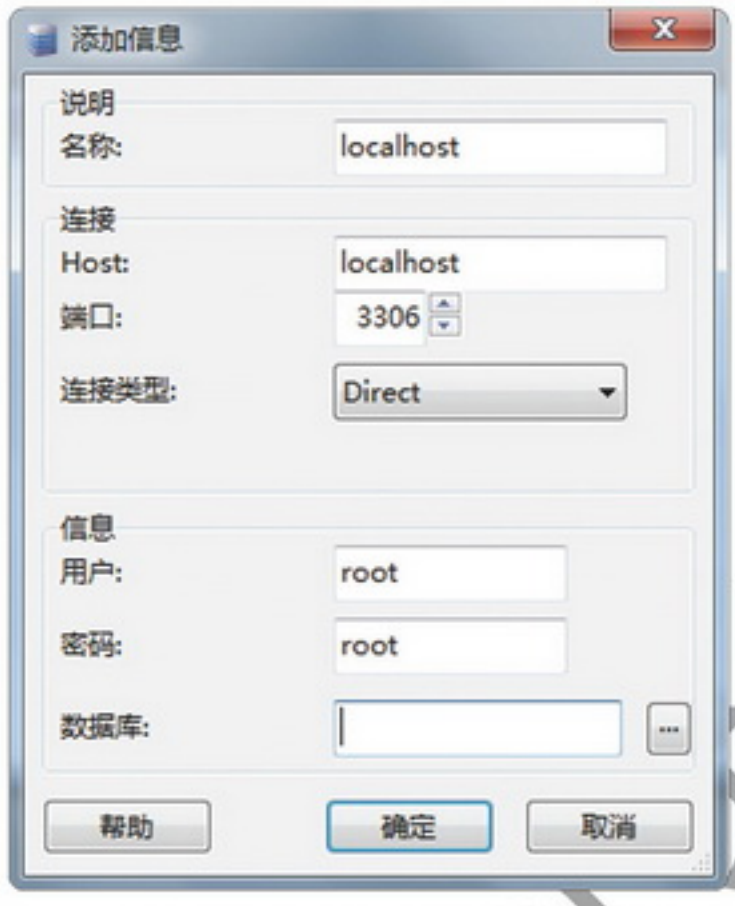


图 8.35　填写数据库连接信息

添加完信息后，单击“确定”按钮。然后，选择“localhost”，单击“打开”按钮，如图 8.36 所示。

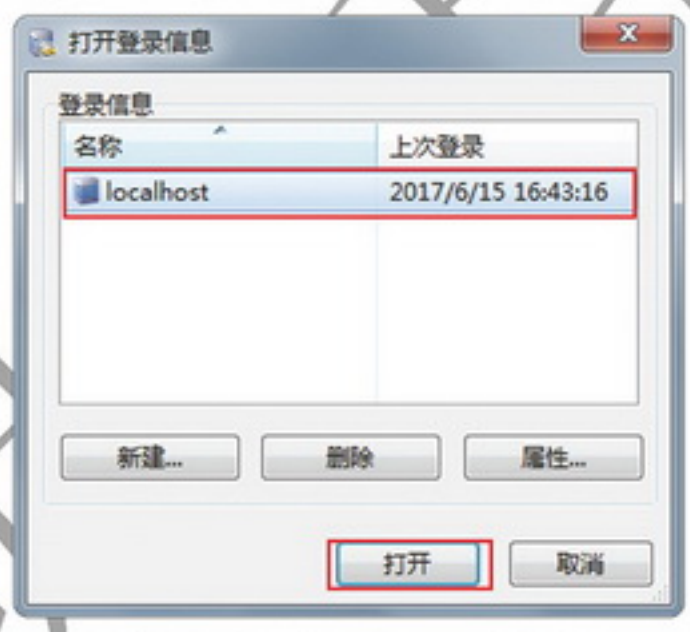


图 8.36　打开 localhost 数据库

📖 说明：MySQL-Front 的更多操作，请查阅相关资料。

8.10 难点解答

8.10.1 drop、delete 和 truncate 的区别

◆ delete 和 truncate 操作只删除表中数据，而不删除表结构；使用 delete 删除时，对于 auto_increment 类型的字段，值不会从 1 开始，而 truncate 可以实现删除数据后，auto_increment 类型的字段值从 1 开

始。但是 drop 语句将删除表的结构、被依赖的约束（constraint）、触发器（trigger）、索引（index）等。依赖于该表的存储过程或函数将保留，但是变为 invalid 状态。

◆ 属于不同类型的操作，delete 属于 DML，这个操作会放到 rollback segement 中，事务提交之后才生效；如果有相应的 trigger，执行的时候将被触发。而 truncate 和 drop 属于 DDL，操作立即生效，原数据不放到 rollback segment 中，不能回滚，操作不触发 trigger。

◆ 执行速度：drop > truncate > delete。

◆ 安全性：小心使用 drop 和 truncate，尤其没有备份的时候。具体使用时，想删除部分数据行用 delete，注意带上 where 子句。此外，回滚段要足够大。

◆ 使用建议：完全删除表使用 drop ；想保留表而将所有数据删除，如果和事务无关，使用 truncate，如果和事务有关，或者想触发 trigger，使用 delete。

8.10.2 主键、外键和索引的区别

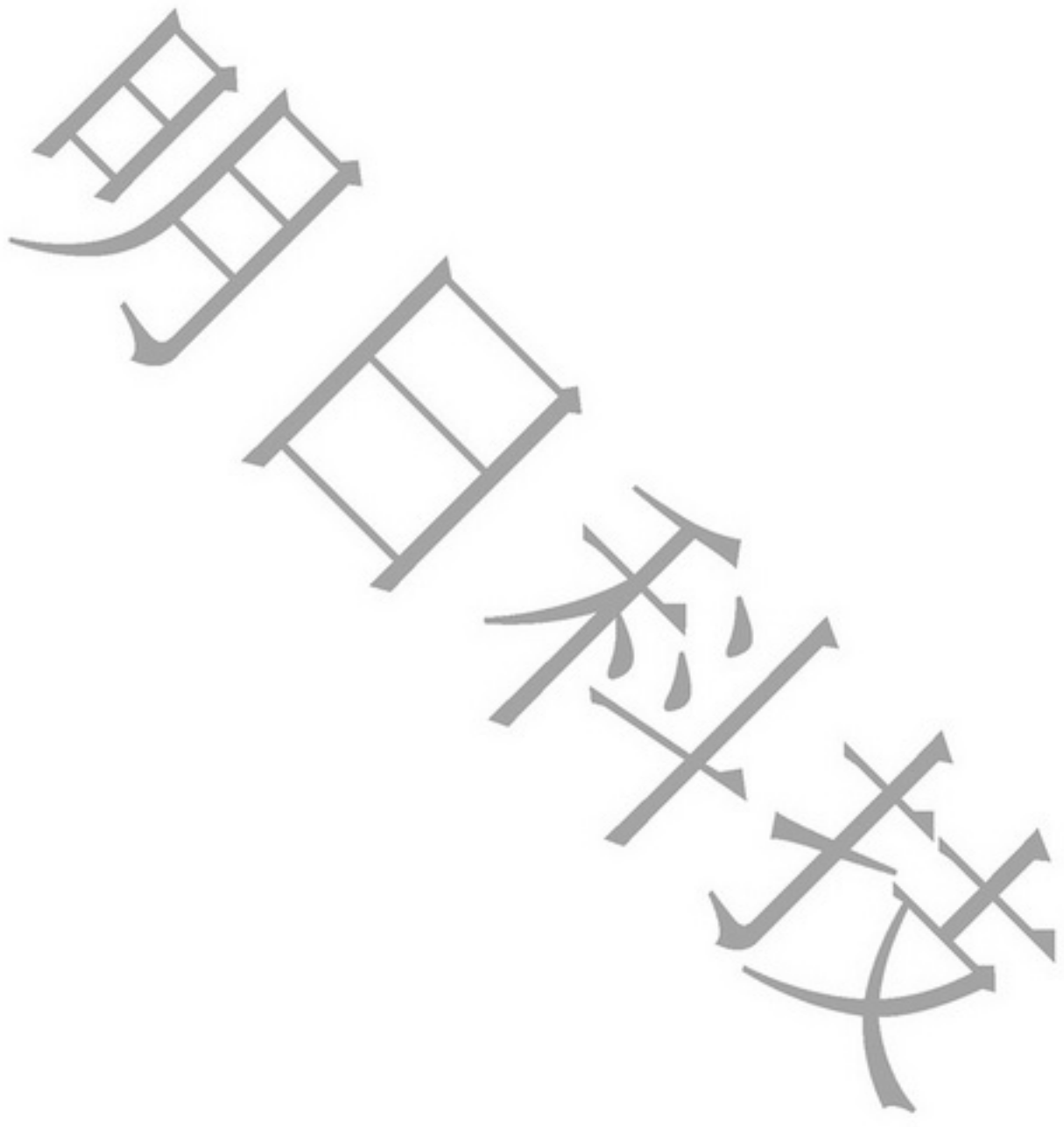
主键、外键和索引的区别主要有三个方面，如表 8.12 所示。

表 8.12　MySQL 中主键、外键和索引的关系

	主　键	外　键	索　引
定义：	唯一标识一条记录，不能有重复的，不允许为空	表的外键是另一表的主键，外键可以有重复的，可以是空值	该字段没有重复值，但可以有一个空值
作用：	用来保证数据完整性	用来和其他表建立联系用的	是提高查询排序的速度
个数：	主键只能有一个	一个表可以有多个外键	一个表可以有多个唯一索引

8.11 小　结

本章主要介绍 MySQL 数据库的基本操作，包括创建、查看、选择、删除数据库；创建、修改、更名、删除数据表；插入、浏览、修改、删除记录，这些是程序开发人员必须掌握的内容。如果用户不习惯在命令提示符下管理数据库，可以在可视化的图形工具中轻松操作和管理数据库。另外，本章还介绍了启动、连接和断开 MySQL 服务器的方法，要求读者熟练掌握。



第 9 章

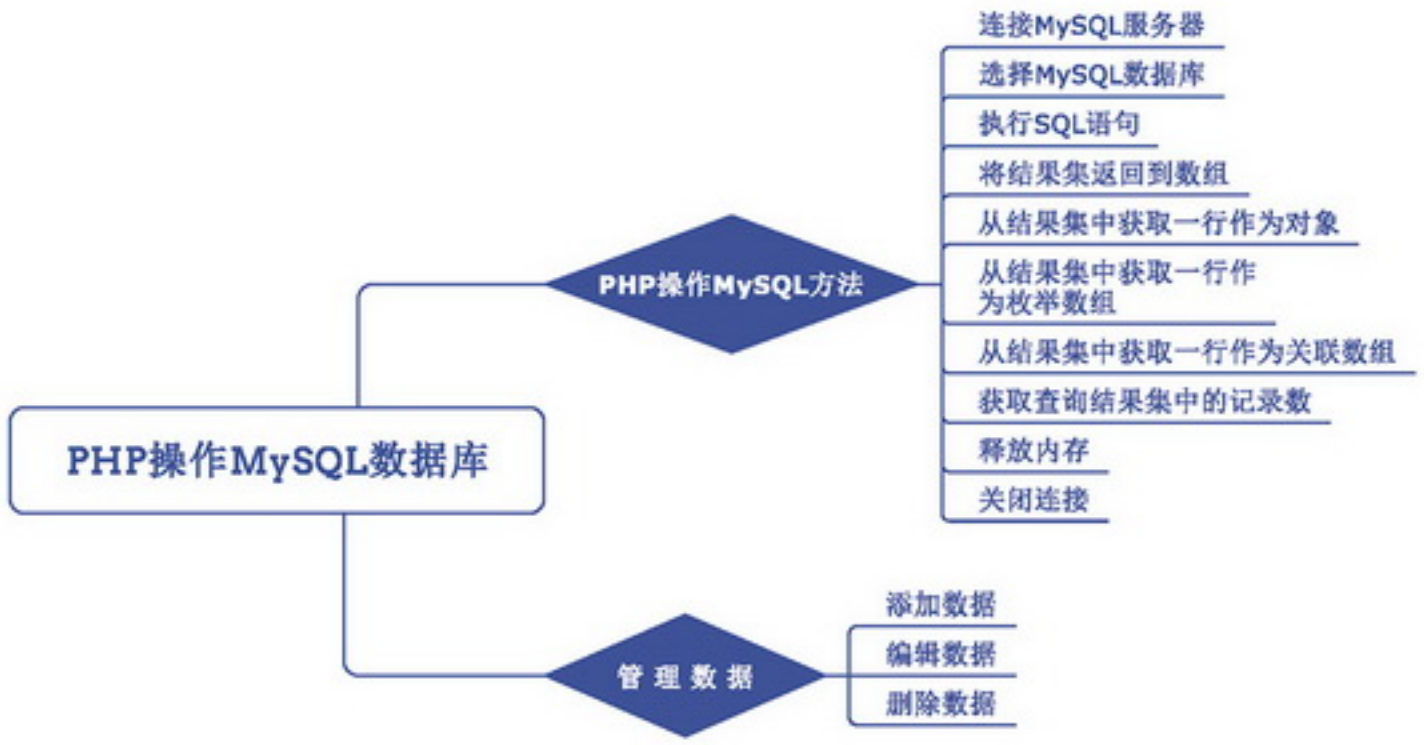
PHP 操作 MySQL 数据库

（视频讲解：52 分）

本章概览

第 8 章中，我们学习了通过 MySQL 命令行或视图管理工具来操作 MySQL 数据库，本章将介绍如何使用 PHP 来操作 MySQL 数据库。很长时间以来，PHP 操作 MySQL 数据库时使用的是 mysql 扩展库提供的相关函数，但是，随着 MySQL 的发展，mysql 扩展库开始出现一些问题，因此逐渐被 mysqli 扩展库取代。本章将介绍如何通过 mysqli 扩展库来操作 MySQL 数据库。

知识框架



9.1 PHP 操作 MySQL 数据库的方法

mysqli 扩展库和 mysql 扩展库的应用基本类似，而且大部分函数的使用方法都一样，唯一的区别就是 mysqli 扩展库中的函数名称都是以 mysqli 开头的。

9.1.1 连接 MySQL 服务器

PHP 操作 MySQL 数据库，首先要建立与 MySQL 数据库的连接。在第 8 章中，使用如下命令连接数据库：

```
mysql  -uroot -proot
```

现在，使用 mysqli 扩展库提供的 mysqli_connect() 函数来实现与 MySQL 数据库的连接，语法如下：

```
mysqli mysqli_connect ( [string $host [, string $username [, string $password [, string
                        $dbname [, int $port [, string $socket]]]]]] )
```

mysqli_connect() 函数用于打开一个到 MySQL 服务器的连接，如果成功则返回一个 MySQL 连接标识，失败则返回 false。该函数的参数如表 9.1 所示。

表 9.1 mysqli_connect() 函数的参数说明

参　数	说　明
host	MySQL 服务器地址
username	用户名，默认值是服务器进程所有者的用户名
password	密码，默认值是空密码
dbname	连接的数据库名称
port	MySQL 服务器使用的端口号
socket	UNIX 域 socket

例如，应用 mysqli_connect() 函数创建与 MySQL 服务器的连接，MySQL 数据库服务器地址为 localhost，用户名为“root”，密码为“root”，代码如下：

```
<?php
    $host = "localhost";                    //MySQL服务器地址，本地测试也可以填写127.0.0.1
    $userName = "root";                     //用户名
    $password = "root";                     //密码
    if ($link = mysqli_connect($host, $userName, $password)){
        //建立与MySQL数据库的连接，并弹出提示对话框
        echo "<script type='text/javascript'>alert('数据库连接成功！');</script>";
```

```
08        }else{
09            echo "<script type='text/javascript'>alert('数据库连接失败！');</script>";
10        }
11    ?>
```

运行上述代码，如果在本地计算机中安装了 MySQL 数据库，并且连接数据库的用户名为“root”，密码为“root”，则会弹出如图 9.1 所示的对话框。

图 9.1　数据库连接成功

说明：代码中使用了 JavaScript 的 alert() 方法弹出提示框。

9.1.2 选择 MySQL 数据库

数据库连接完成以后，需要选择数据库。第 8 章中选择数据库的命令如下：

```
use database db_users
```

现在，使用 mysqli 扩展库提供的 mysqli_connect() 函数可以创建与 MySQL 服务器的连接，同时也可以指定要选择的数据库名称，例如，在连接 MySQL 服务器的同时选择名称为 db_users 的数据库，代码如下：

```
$link= mysqli_connect("localhost", "root", "root", "db_users");
```

除此之外，mysqli 扩展库还提供了 mysqli_select_db() 函数用来选择 MySQL 数据库。其语法如下：

```
bool mysqli_select_db ( mysqli $link, string $dbname )
```

◆ 参数 link 为必选参数，应用 mysqli_connect() 函数成功连接 MySQL 数据库服务器后返回的连接标识。

◆ 参数 dbname 为必选参数，用户指定要选择的数据库名称。

例如，首先创建 database9 数据库，然后使用 mysqli_connect() 函数建立与 MySQL 数据库的连接，最后使用 mysqli_select_db() 函数选择 database9 数据库，实现代码如下：

```
01  <?php
02  $host = "localhost";                                    //MySQL服务器地址
03  $userName = "root";                                     //用户名
04  $password = "root";                                     //密码
05  $dbName = "database9";                                  //数据库名称
06  $link = mysqli_connect($host, $userName, $password);    //建立与MySQL数据库服务器的连接
```

```
07  if(mysqli_select_db($link, $dbName)){                    //选择数据库
08      echo "数据库选择成功！";
09  }else{
10      echo "数据库选择失败！";
11  }
12  ?>
```

运行上述代码，如果本地 MySQL 数据库服务器中存在名为 database9 的数据库，将在页面中输出如下内容：

```
数据库选择成功!
```

否则输出：

```
数据库选择失败!
```

📖 **说明**：在实际的程序开发过程中，通常将 MySQL 服务器的连接和数据库的选择存储于一个单独文件中，在需要使用的脚本中通过 require 语句包含这个文件即可。这样做既有利于程序的维护，同时也避免了代码的冗余。

9.1.3 执行 SQL 语句

在第 8 章中，使用 SQL 语句对数据库中的表进行操作。在 mysqli 扩展库中，同样使用 SQL 语句对数据表进行操作，但是需要使用 mysqli_query() 函数来执行 SQL 语句。其语法如下：

```
mixed mysqli_query( mysqli $link, string $query [, int $resultmode] )
```

◆ 参数 link 为必选参数，mysqli_connect() 函数成功连接 MySQL 数据库服务器后所返回的连接标识。

◆ 参数 query 为必选参数，所要执行的 SQL 语句。

◆ 参数 resultmode 为可选参数，该参数取值有 MYSQLI_USE_RESULT 和 MYSQLI_STORE_RESULT。其中 MYSQLI_STORE_RESULT 为该函数的默认值。如果返回大量数据可以应用 MYSQLI_USE_RESULT，但应用该值时，以后的查询调用可能返回一个 commands out of sync 错误，解决办法是应用 mysqli_free_result() 函数释放内存。

如果 SQL 语句是查询指令 select，成功则返回查询结果集，否则返回 false；如果 SQL 语句是 insert、delete、update 等操作指令，成功则返回 true，否则返回 false。

下面看一下如何通过 mysqli_query() 函数执行简单的 SQL 语句。

执行一个添加会员记录的 SQL 语句的代码如下：

```
$result = mysqli_query($link,"insert into tb_member values('mrsoft','123',
                        'mrsoft@mrsoft.com')");
```

执行一个修改会员记录的 SQL 语句的代码如下：

```
$result = mysqli_query($link,"update tb_member set user='mrbook',pwd='mrsoft'
                        where user='mrsoft'");
```

执行一个删除会员记录的 SQL 语句的代码如下：

```
$result = mysqli_query($link,"delete from tb_member where user='mrbook'");
```

执行一个查询会员记录的 SQL 语句的代码如下：

```
$result = mysqli_query($link,"select * from tb_member");
```

mysqli_query() 函数不仅可以执行诸如 select、update 和 insert 等 SQL 指令，而且可以选择数据库和设置数据库编码格式。选择数据库的功能与 mysqli_select_db() 函数是相同的，代码如下：

```
mysqli_query($link,"use database9");                    //选择数据库database9
```

设置数据库的编码格式的代码如下：

```
mysqli_query($link,"set names utf8");                   //设置数据库的编码为UTF-8
```

9.1.4 将结果集返回到数组

使用 mysqli_query() 函数执行 select 语句，如果成功将返回查询结果集。下面介绍一个对查询结果集进行操作的函数 mysqli_fetch_array()，它将结果集返回到数组中。其语法如下：

```
array mysqli_fetch_array ( resource $result [, int $result_type] )
```

- 参数 result：资源类型的参数，要传入的是由 mysqli_query() 函数返回的数据指针。
- 参数 result_type：可选项，设置结果集数组的表述方式。有以下 3 种取值：

（1）MYSQLI_ASSOC：返回一个关联数组。数组下标由表的字段名组成，如“id”“name”。

（2）MYSQLI_NUM：返回一个索引数组。数组下标由数字组成，如“0”“1”“2”。

（3）MYSQLI_BOTH：返回一个同时包含关联和数字索引的数组。默认值是 MYSQLI_BOTH。

注意：本函数返回的字段名要区分大小写，这是初学者最容易忽略的问题。

到此，已经介绍了 PHP 操作 MySQL 数据库的部分方法，可以实现 MySQL 服务器的连接、选择数据库、执行查询语句，并且可以将查询结果集中的数据返回到数组中。下面编写一个实例，通过 PHP 操作 MySQL 数据库，来读取数据库中存储的数据。

实例 01　使用 mysqli_fetch_array() 函数读取数据　　实例位置：光盘\Code\SL\09\01

本实例将利用 mysqli_fetch_array() 函数，读取 database9 数据库中 books 图书表中的数据。具体步骤如下：

（1）创建 database9 数据库并选择该数据库。SQL 语句如下：

实例01-1

```
create database database9;
use database9;
```

（2）创建 books 数据表，并设置数据库的编码格式为 UTF-8。创建数据表的 SQL 语句如下：

实例01-2

```
DROP TABLE IF EXISTS `books`;
CREATE TABLE `books` (
  `id` int(8) NOT NULL AUTO_INCREMENT,
  `name` varchar(50) NOT NULL,
  `category` varchar(50) NOT NULL,
  `price` decimal(10,2) DEFAULT NULL,
  `publish_time` date DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=MyISAM AUTO_INCREMENT=1 DEFAULT CHARSET=utf8;
```

上述 SQL 语句创建了 books 表，该表共 5 个字段。其中，“id”字段是表的主键，并且是自增的；“name”字段用于保存图书名称，“category”字段用于保存图书分类，“price”字段用于保存图书价格；“publish_time”字段用于保存出版时间。

（3）插入测试数据。为了显示图书信息，需要先在 books 表中插入几条测试数据。插入数据的 SQL 语句如下：

实例01-3

```
INSERT INTO `books` VALUES ('1', 'PHP从入门到精通', 'PHP', '50.00', '2017-02-17');
INSERT INTO `books` VALUES ('2', 'PHP自学宝典', 'PHP', '69.00', '2017-02-17');
INSERT INTO `books` VALUES ('3', 'PHP项目实战入门', 'PHP', '70.00', '2017-02-17');
INSERT INTO `books` VALUES ('4', '零基础学PHP', 'PHP', '68.80', '2017-02-17');
```

（4）连接数据库，获取数据。创建 index.php 文件，具体代码如下：

实例01-4

```
<?php
    //连接MySQL服务器，选择数据库
    $link = mysqli_connect("localhost", "root", "root", "database9") or die("连接数据库
                            服务器失败！".mysqli_error());
    mysqli_query($link,"set names utf8");                    //设置数据库编码格式UTF-8
    $result = mysqli_query($link,"select * from books");     //执行查询语句
    include_once('lists.html');                              //引入模板
```

在上述代码中，首先使用 mysqli_connect() 函数连接数据库，如果连接失败，则终止程序，并使用 mysqli_error() 函数来显示错误信息。然后设置数据库编码格式为 UTF-8。代码第 6 行使用 mysqli_query 执行 select 语句，从数据库中查询获取结果集。最后使用 include_once() 函数，引入模板文件，即 HTML 页面。

注意：为保证数据能正确显示，建议读者保持以下几个编码格式统一为 UTF-8：PHP 文件的编码格式和 HTML 文件的编码格式（可以使用 PhpStorm 编辑器设置）；数据表编码格式（可以使用图形化工具设置）。与 PHP 中不同的是，MySQL 中指定的编码格式是“utf8”，而不是“utf-8”。

（5）显示图书信息。创建 lists.html 文件，该文件就是 HTML 模板文件。使用 mysqli_fetch_array() 函数将结果集返回到数组中，通过 while 语句循环遍历图书数组，将每本图书数据插入到 <table> 表格中，具体代码如下：

实例01-5

```
<!DOCTYPE html>
<html lang="en" class="is-centered is-bold">
```

```
<head>
    <meta charset="UTF-8">
    <title>零基础</title>
    <link href="css/bootstrap.css" rel="stylesheet">
    <script src="js/jquery.min.js"></script>
</head>
<body>
<div class="container">
    <div class="col-sm-offset-2 col-sm-8">
        <div class="panel panel-default">
            <div class="panel-heading">
                图书列表
            </div>
            <div class="panel-body">
                <table class="table table-striped task-table">
                    <thead>
                        <tr>
                            <th>ID</th>
                            <th>图书名称</th>
                            <th>分类</th>
                            <th>价格</th>
                            <th>出版日期</th>
                            <th>操作</th>
                        </tr>
                    </thead>
                    <tbody>
                        <?php while($rows = mysqli_fetch_array($result,MYSQLI_ASSOC)) { ?>
                        <tr>
                            <td class="table-text">
                                <?php echo $rows['id'] ?>
                            </td>
                            <td class="table-text">
                                <?php echo $rows['name'] ?>
                            </td>
                            <td class="table-text">
                                <?php echo $rows['category'] ?>
                            </td>
                            <td class="table-text">
                                <?php echo $rows['price'] ?>
                            </td>
                            <td><?php echo $rows['publish_time'] ?></td>
                            <td>
                                <a href='editBook.php?id=<?php echo $rows['id'] ?>'>
                                    <button class="btn btn-info edit" >编辑</button>
                                </a>
                                <a href='deleteBook.php?id=<?php echo $rows['id'] ?>'>
                                    <button class="btn btn-danger delete">删除</button>
```

```
                                    </a>
                                </td>
                            </tr>
                            <?php } ?>
                        </tbody>
                    </table>
                </div>
            </div>
        </div>
    </div>
</body>
</html>
```

使用浏览器访问“localhost/Code/SL/09/01/index.php”，运行结果如图 9.2 所示。

图 9.2　显示图书列表

练一练：

（1）修改实例 01 步骤（3）中的查询语句，根据价格升序筛选图书，运行结果如图 9.3 所示。（光盘\Code\Try\09\01）

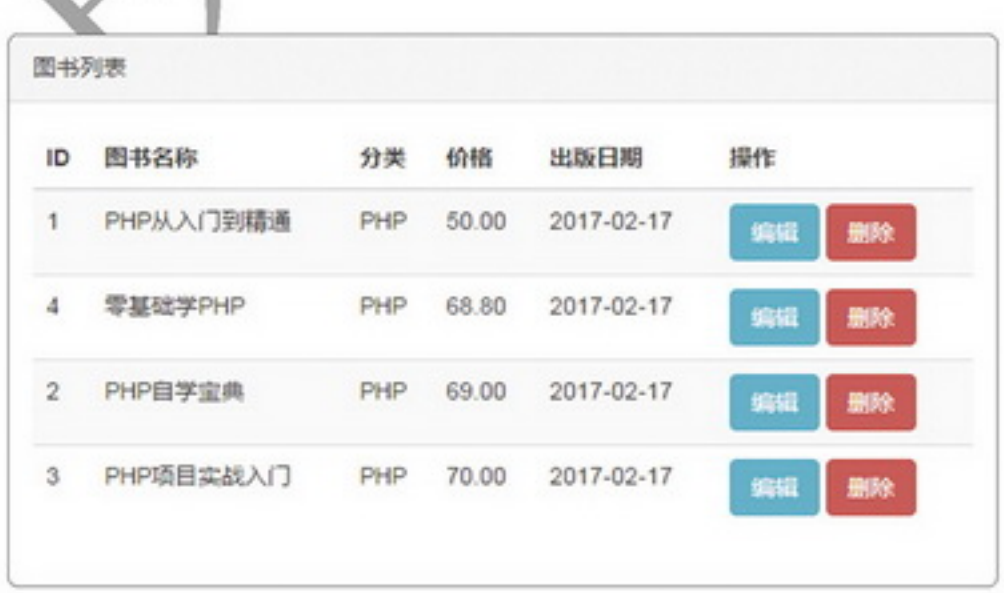


图 9.3　根据图书价格升序展示

（2）修改实例 01 步骤（3）中的查询语句，筛选出所有包含“入门”关键字的图书，运行结果如图 9.4 所示。（光盘\Code\Try\09\02）

图 9.4　筛选包含关键字“入门”的图书

9.1.5 从结果集中获取一行作为对象

9.1.4 小节中讲解了应用 mysqli_fetch_array() 函数来获取结果集中的数据。除了这个方法以外，应用 mysqli_fetch_object() 函数也可以轻松实现这一功能，下面通过同一个实例的不同方法来体验一下这两个函数在使用上的区别。

首先介绍 mysqli_fetch_object() 函数，其语法如下：

```
mixed mysqli_fetch_object ( resource result )
```

mysqli_fetch_object() 函数和 mysqli_fetch_array() 函数类似，只有一点区别：它返回的是一个对象而不是数组，即该函数只能通过字段名来访问数组。访问结果集行中元素的语法结构如下：

```
$row->col_name                    //col_name为字段名，$row代表结果集
```

例如，从某数据表中检索 id 和 name 值，可以用 $row->id 和 $row-> name 访问行中的元素值。

注意：mysqli_fetch_object() 函数返回的字段名同样是区分大小写的。

实例 02　使用 mysqli_fetch_object() 函数读取所有图书数据　　实例位置：光盘\Code\SL\09\02

本实例同样是读取 database9 数据库中 books 数据表中的数据，不同的是应用 mysqli_fetch_object() 函数逐行获取结果集中的记录。由于在实例 01 中，已经创建了数据库和数据表，并且连接了数据库，所以只需要修改 lists.html 文件即可。

在 lists.html 文件中，使用 mysqli_fetch_object() 函数逐行获取结果集，该结果集是一个对象，使用 while 语句循环遍历该对象，将每本图书数据插入到 <table> 表格中，关键代码如下：

实例02-1

```
<!DOCTYPE html>
<html lang="en" class="is-centered is-bold">
<head>
    <!-- 省略重复代码 -->
```

```
</head>
<body>
<div class="container">
    <!-- 省略重复代码 -->
    <tbody>
    <?php  while($obj = mysqli_fetch_object($result)) {
        if(is_object($obj)){   //判断对象是否存在
    ?>
    <tr>
        <td class="table-text">
            <?php echo $obj->id ?>
        </td>
        <td class="table-text">
            <?php echo $obj->name ?>
        </td>
        <td class="table-text">
            <?php echo $obj->category ?>
        </td>
        <td class="table-text">
            <?php echo $obj->price ?>
        </td>
        <td><?php echo $obj->publish_time ?></td>
        <td>
            <a href='editBook.php?id=<?php echo $obj->id ?>'>
                <button class="btn btn-info edit" >编辑</button>
            </a>
            <a href='deleteBook.php?id=<?php echo $obj->id ?>'>
                <button class="btn btn-danger delete">删除</button>
            </a>
        </td>
    </tr>
    <?php }
        }
    ?>
    </tbody>
    </table>
    <!-- 省略重复代码 -->
</div>
</body>
</html>
```

本实例的运行结果与实例 01 相同，如图 9.2 所示。

练一练：

（1）修改实例 02 步骤（3）中的查询语句，根据价格升序筛选图书，运行结果如图 9.5 所示。(光盘\Code\Try\09\03)

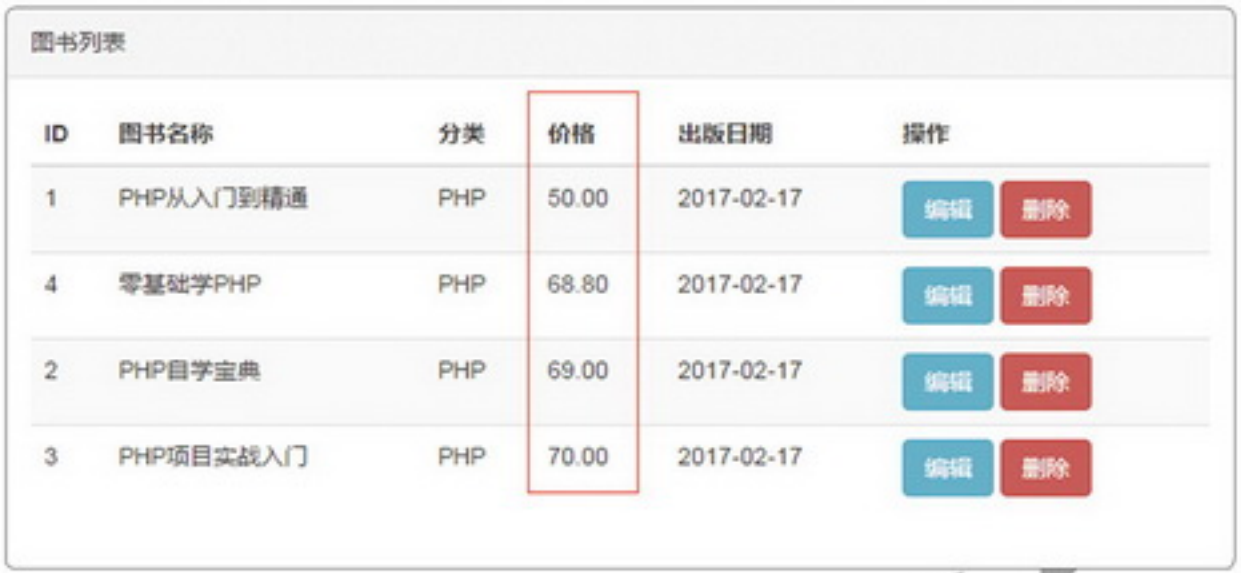


图 9.5　价格升序筛选图书

（2）修改实例 02，从 books 表中只筛选图书的名称和价格，运行结果如图 9.6 所示。（光盘\Code\Try\09\04）

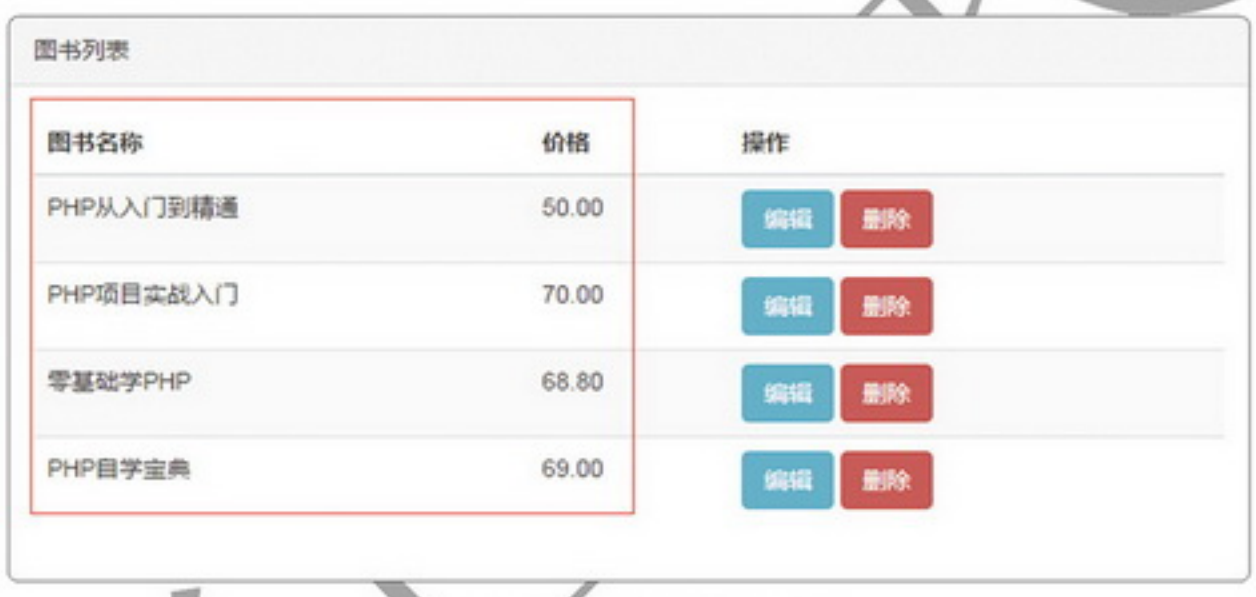


图 9.6　只筛选名称和价格

9.1.6 从结果集中获取一行作为枚举数组

mysqli_fetch_row() 函数可以从结果集中取得一行作为枚举数组，即数组的键用数字索引来表示。其语法如下：

```
mixed mysqli_fetch_row ( resource $result )
```

mysqli_fetch_row() 函数返回根据所取得的行生成的数组，如果没有更多行则返回 null。返回数组的偏移量从 0 开始，即以 $row[0] 的形式访问第一个元素（只有一个元素时也是如此）。

例如，使用 mysqli_fetch_row() 函数实现图书列表的功能，只需修改 lists.html 文件，修改代码如下：

```
01 //省略重复代码
02 <tbody>
03 <?php while($rows = mysqli_fetch_row($result)) { ?>
04     <tr>
05         <td class="table-text">
```

```
06              <?php echo $rows[0] ?>
07          </td>
08          <td class="table-text">
09              <?php echo $rows[1] ?>
10          </td>
11          <td class="table-text">
12              <?php echo $rows[2] ?>
13          </td>
14          <td class="table-text">
15              <?php echo $rows[3] ?>
16          </td>
17          <td><?php echo $rows[4] ?></td>
18          <td>
19             <a href='editBook.php?id=<?php echo $rows[0] ?>'>
20              <button class="btn btn-info edit">编辑</button>
21             </a>
22                    <a href='deleteBook.php?id=<?php echo $rows[0] ?>'>
23              <button class="btn btn-danger delete" >删除
24                    </a>
25          </td>
26      </tr>
27  <?php } ?>
28  </tbody>
29  //省略重复代码
```

上述代码在使用 mysqli_fetch_row() 函数逐行获取结果集中的记录时，只能使用数字索引来读取数组中的数据，而不能像 mysqli_fetch_array() 函数那样可以使用关联索引获取数组中的数据。

本实例的运行结果与实例 01 相同，如图 9.2 所示。

9.1.7 从结果集中获取一行作为关联数组

mysqli_fetch_assoc() 函数可以从结果集中取得一行作为关联数组，即数组的键用字段名来表示。其语法如下：

```
mixed mysqli_fetch_assoc ( resource $result )
```

mysqli_fetch_assoc() 函数返回根据所取得的行生成的数组，如果没有更多行则返回 null。该数组的下标为数据表中字段的名称。

```
mysqli_fetch_assoc($result)
```

等价于：

```
mysqli_fetch_array($result,MYSQLI_ASSOC)
```

9.1.8 获取查询结果集中的记录数

使用 mysqli_num_rows() 函数，可以获取由 select 语句查询到的结果集中行的数目。mysqli_num_rows() 函数的语法如下：

```
int mysqli_num_rows ( resource $result )
```

mysqli_num_rows() 函数返回结果集中行的数目。此函数仅对 select 语句有效。要取得被 insert、update 或者 delete 语句所影响到的行的数目，则要使用 mysqli_affected_rows() 函数。

实例 03　使用 mysqli_num_rows() 函数获取图书总数　　实例位置：光盘\Code\SL\09\03

本实例中应用 mysqli_fetch_array() 函数逐行获取结果集中的记录，同时应用 mysqli_ num_rows() 函数获取结果集中行的数目，并输出返回值。具体步骤如下：

由于本实例是在实例 01 的基础上进行操作，所以这里只给出关键代码，不再赘述它的创建步骤。

（1）在 index.php 文件中，增加 mysqli_num_rows() 函数，获取结果集中记录数。代码如下：

实例03-1

```
<?php
    //连接MySQL服务器，选择数据库
    $link = mysqli_connect("localhost", "root", "root", "database9") or
            die("连接数据库服务器失败！".mysqli_error());
    mysqli_query($link,"set names utf8");                    //设置数据库编码格式UTF-8
    $result = mysqli_query($link,"select * from books");     //执行查询语句
    $number = mysqli_num_rows($result);                      //获取查询条数
    include_once('lists.html');                              //引入模板
```

（2）在 lists.html 文件中，新增显示记录条数代码，关键代码如下：

实例03-2

```
<p class="text-primary text-center ">共计<?php echo $number ?>条</p>
```

运行结果如图 9.7 所示。

图 9.7　获取查询结果的记录数

练一练：

（1）修改实例 03，从图书列表中获取包含关键字“入门”的图书数量。（光盘\Code\Try\09\05）

（2）修改实例 03，从图书列表中获取价格大于 60 的图书数量。（光盘\Code\Try\09\06）

9.1.9 释放内存

mysqli_free_result() 函数用于释放内存，数据库操作完成后，需要关闭结果集，以释放系统资源，该函数的语法格式如下：

```
void mysqli_free_result(resource $result);
```

mysqli_free_result() 函数将释放所有与结果标识符 $result 相关联的内存。该函数仅需要在考虑到返回很大的结果集会占用较多内存时调用。在执行结束后所有关联的内存都会被自动释放。

9.1.10 关闭连接

完成对数据库的操作后，需要及时断开与数据库的连接并释放内存，否则会浪费大量的内存空间，在访问量较大的 Web 项目中，很可能导致服务器崩溃。在 MySQL 函数库中，使用 mysqli_close() 函数断开与 MySQL 服务器的连接，该函数的语法格式如下：

```
bool mysqli_close ( mysqli $link )
```

参数 link 为 mysqli_connect() 函数成功连接 MySQL 数据库服务器后所返回的连接标识。如果成功则返回 true，失败则返回 false。

例如，读取 database9 数据库中 books 数据表中的数据，然后使用 mysqli_free_result() 函数释放内存并使用 mysqli_close() 函数断开与 MySQL 数据库的连接。代码如下：

```
<?php
    //连接MySQL服务器，选择数据库
    $link = mysqli_connect("localhost", "root", "root", "database9") or
            die("连接数据库服务器失败！".mysqli_error());
    mysqli_query($link,"set names utf8");                    //设置数据库编码格式utf8
    $result=mysqli_query($link,"select * from books");       //执行查询语句
    $number = mysqli_num_rows($result);                      //获取查询条数
    include_once('lists.html');                              //引入模板
    mysqli_free_result($result);                             //释放内存
    mysqli_close($link);                                     //断开与数据库连接
```

说明：PHP 中与数据库的连接是非持久连接的，系统会自动回收，一般不用设置关闭。但如果一次性返回的结果集比较大，或网站访问量比较多，则最好使用 mysqli_close() 函数手动进行释放。

9.2 管理 MySQL 数据库中的数据

在开发网站的后台管理系统中，对数据库的操作不仅局限于查询，对数据的添加、修改和删除等操作也是必不可少的。本节重点介绍如何在 PHP 页面中对数据库进行增、删、改的操作。

9.2.1 添加数据

实例 04　向图书信息表中添加图书信息　　实例位置：光盘\Code\SL\09\04

本实例将通过 insert 语句和 mysqli_query() 函数向图书信息表中添加一条记录。具体步骤如下：

（1）创建 Form 表单页面 add.html，表单中包含“name”（书名）、“category”（分类）、“price”（价格）、“publish_time”（出版时间）4 个字段。当单击“提交”按钮时，将表单提交到“addBook.php”文件。具体代码如下：

实例04-1

```
<!DOCTYPE html>
 <html lang="en" class="is-centered is-bold">
 <head>
     <meta charset="utf-8">
     <title>零基础</title>
     <!-- 省略部分代码 -->
 </head>
 <body>
 <div class="container">
 <!-- 省略部分代码 -->
     <div class="panel-body">
         <form class="form-horizontal" role="form" method="POST"
               action="addBook.php">
             <div class="row">
                 <div class="col-md-8">
                     <div class="form-group">
                         <label for="name" class="col-md-2 control-label">
                             名称
                         </label>
                         <div class="col-md-10">
                             <input type="text" class="form-control" name="name"
                                    id="name" value="">
                         </div>
                     </div>
                     <div class="form-group">
                         <label for="name" class="col-md-2 control-label">
```

```
27                          分类
28                      </label>
29                      <div class="col-md-10">
30                          <select class="form-control" name="category">
31                              <option value="PHP">PHP</option>
32                              <option value="Java">Java</option>
33                              <option value="C++">C++</option>
34                          </select>
35                      </div>
36                  </div>
37                  <div class="form-group">
38                      <label for="price" class="col-sm-2 control-label">
39                          价格
40                      </label>
41                      <div class="col-md-10">
42                          <input type="text" class="form-control" name="price"
43                                 id="price" value="">
44                      </div>
45                  </div>
46                  <div class="form-group">
47                      <label class="col-sm-2 control-label">出版时间</label>
48                      <div class='col-md-10' >
49                          <div class='input-group date' id='publish_time'>
50                              <input type='text' class="form-control"
51                                     name="publish_time"/>
52                              <span class="input-group-addon">
53                                  <span class="glyphicon glyphicon-calendar"></span>
54                              </span>
55                          </div>
56                      </div>
57                  </div>
58              </div>
59          </div>
60          <div class="col-md-8">
61              <div class="form-group">
62                  <div class="col-md-10 col-md-offset-2">
63                      <button type="submit" class="btn btn-primary btn-lg">
64                          <i class="fa fa-disk-o"></i>
65                          提交
66                      </button>
67                  </div>
68              </div>
69          </div>
70      </form>
71      <!-- 省略部分代码 -->
```

```
72          </div>
73      <script>
74          //省略部分代码
75      </script>
76      </body>
77      </html>
```

运行结果如图 9.8 所示。

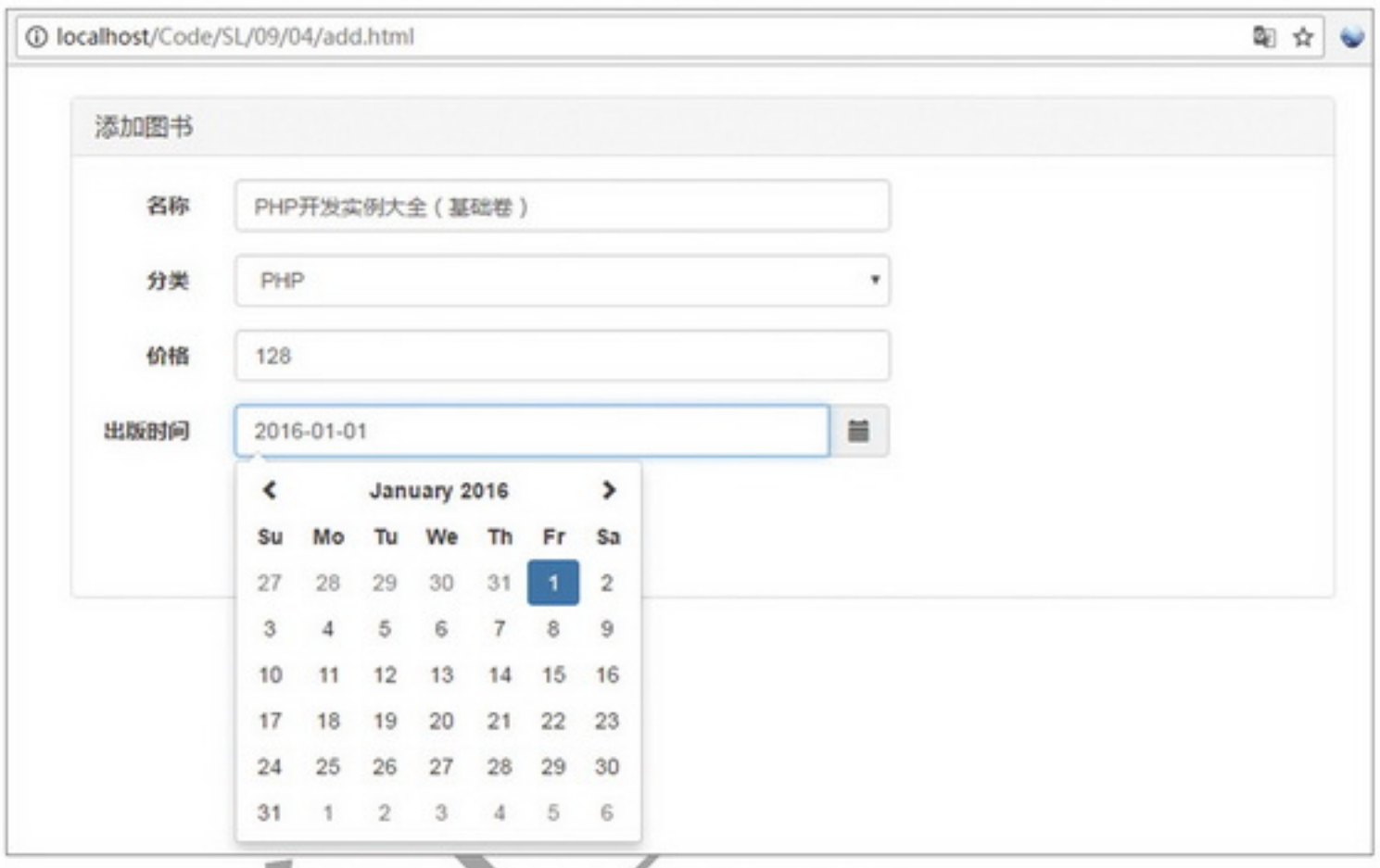


图 9.8　添加图书页面效果

（2）创建 addBook.php 文件，用于连接数据库、发送查询以及检查结果。此时，发送的查询是 insert 而不是 select。在将数据插入数据库时，为避免 SQL 注入攻击，需要使用 prepare 语句进行预处理，然后使用 bind_param() 函数进行参数绑定，最后使用 execute() 函数执行 SQL 语句。具体代码如下：

实例04-2

```
01  <?php
02  $link = mysqli_connect('localhost', 'root', 'root', 'database9');
03  mysqli_query($link,"set names utf8");          //设置数据库编码格式UTF-8
04  /* 检测连接是否成功 */
05  if (!$link) {
06      printf("Connect failed: %s\n", mysqli_connect_error());
07      exit();
08  }
09  /* 检测是否生成MySQLi_STMT类 */
10  $stmt = mysqli_prepare($link, "insert into books (name,category,price,publish_time)
11                                 VALUES (?, ?, ?, ?)");
12  if ( !$stmt ) {
13      die('mysqli error: '.mysqli_error($link));
```

```
14  }
15  /* 获取POST提交数据 */
16  $name     = $_POST['name'];
17  $category = $_POST['category'];
18  $price    = $_POST['price'];
19  $publish_time = $_POST['publish_time'];
20  /* 参数绑定 */
21  mysqli_stmt_bind_param($stmt, 'ssds', $name, $category, $price, $publish_time);
22  /* 执行prepare语句 */
23  mysqli_stmt_execute($stmt);
24  /* 根据执行结果，跳转页面 */
25  if(mysqli_stmt_affected_rows($stmt)){
26      echo "<script>alert('添加成功');window.location.href='index.php';</script>";
27  }else{
28      echo "<script>alert('添加失败');</script>";
29  }
30
31  ?>
```

上述代码中，在使用 mysqli_prepare() 函数时，SQL 语句中包含了 4 个“?”，它们是占位符，没有实际意义，后面会被 mysqli_stmt_bind_param() 函数中的相应参数替换。mysqli_stmt_bind_param() 函数的语法如下：

```
bool mysqli_stmt_bind_param ( mysqli_stmt $stmt , string $types , mixed &$var1 [, mixed &$... ] )
```

- stmt：statement 标识。
- types：绑定的变量的数据类型，它接受的字符种类包括 4 个，如表 9.2 所示。

表 9.2　绑定变量的数据类型

字 符 种 类	代表的数据类型
I	integer
D	double
S	string
B	blob

- var：绑定的变量，其数量必须要与 SQL 语句中的参数数量保持一致。

本实例中应用到的该函数的代码如下：

```
mysqli_stmt_bind_param($stmt, 'ssds', $name, $category, $price, $publish_time);
```

在代码中，“ssds”分别表示 $name 为字符串类型，$category 为字符串类型，$price 为双精度浮点型，$publish_time 为字符串类型。

mysqli_stmt_affected_rows() 函数获取受影响的行数，如果返回 0，则表示没有记录被更新，或者查询语句条件不匹配，或者没有执行查询语句。如果返回 -1，则表示查询返回错误。

填写表单，单击“提交”按钮，运行结果如图 9.9 所示。

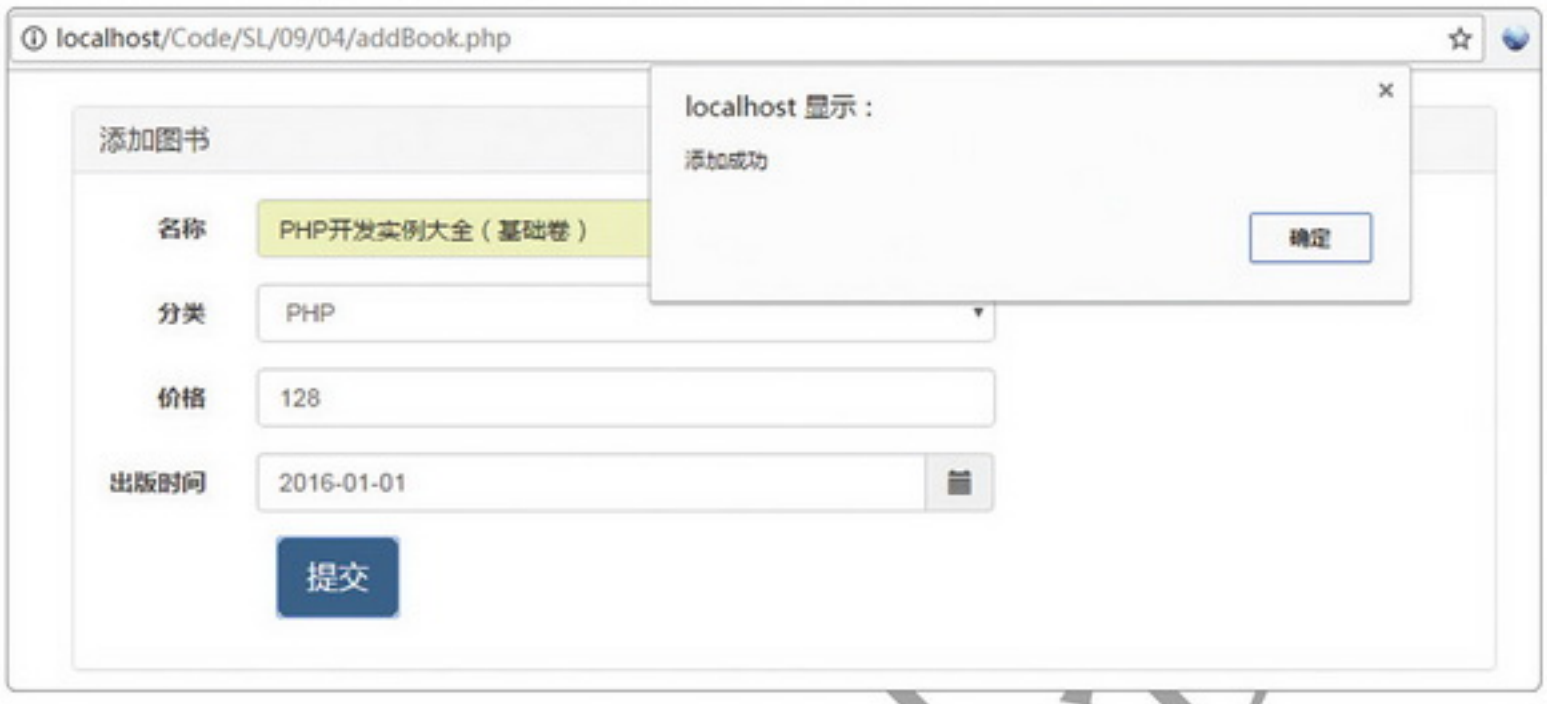


图 9.9　添加成功页面

添加成功后，页面跳转到列表页。在列表页中，会显示提交的图书，如图 9.10 所示。

图 9.10　列表页显示提交的图书

练一练：

（1）结合第 7 章实例 03，在 database9 数据库中新建 member 表，实现会员注册功能。注册页面如图 9.11 所示，注册成功后，member 表数据如图 9.12 所示。（光盘\Code\Try\09\07）

（2）在 database9 数据库中新建 blog 表，包含“title”“author”“post”“publish_time”四个字段。实现博客的添加文章功能，如图 9.13 所示。（光盘\Code\Try\09\08）

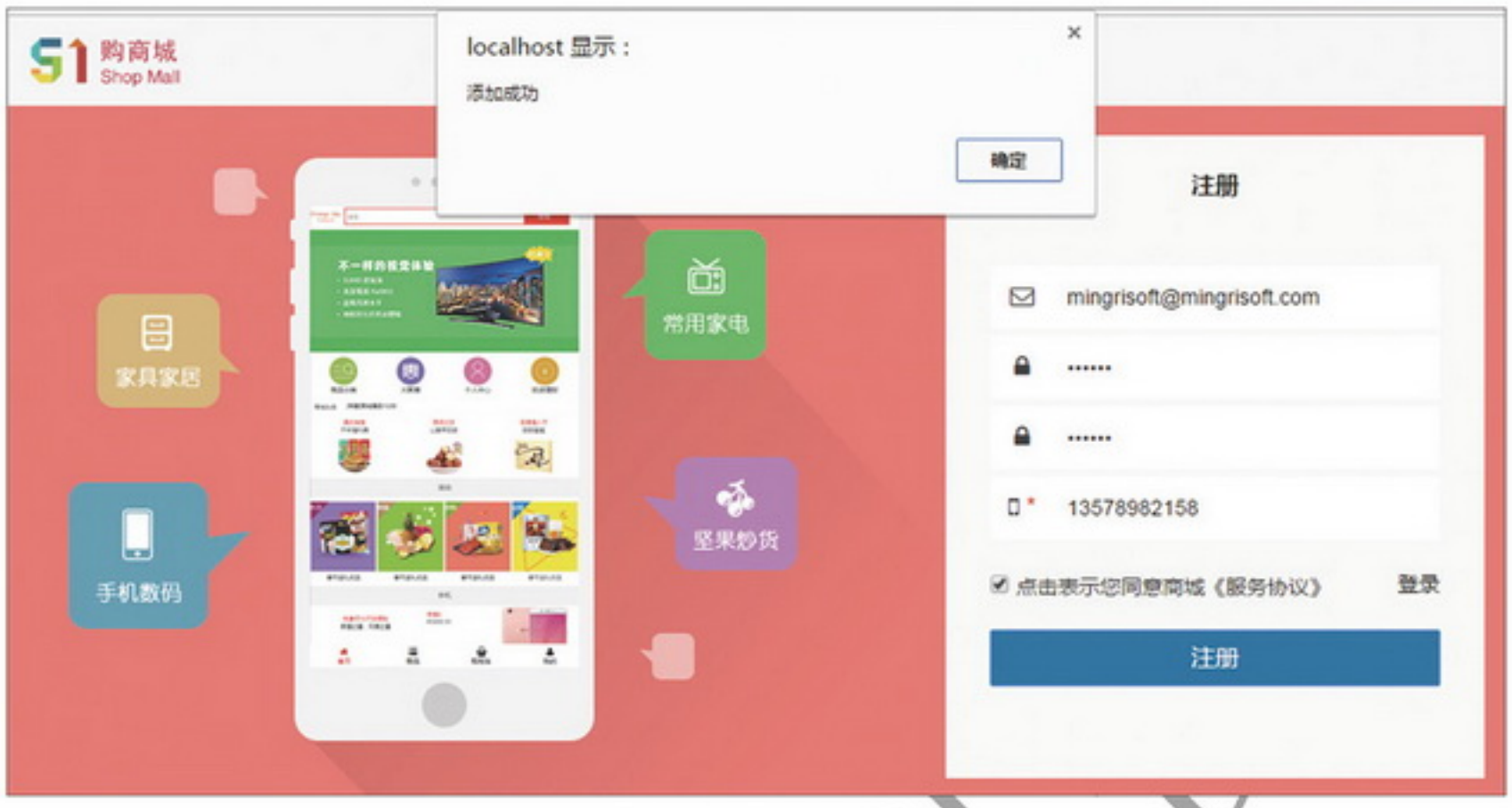


图 9.11　注册页面

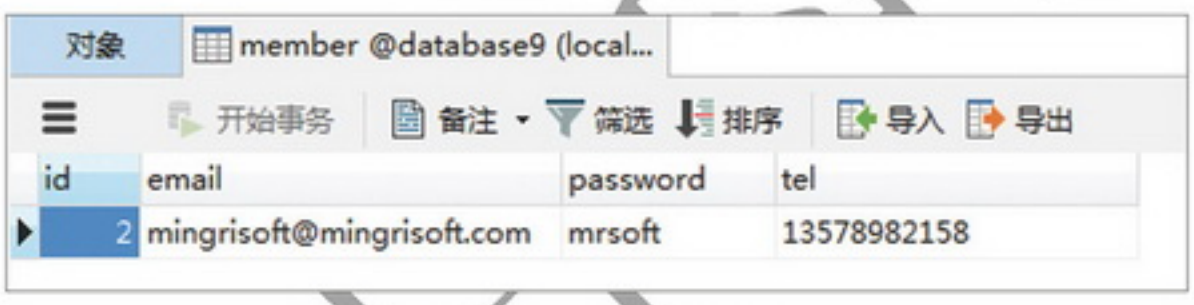


图 9.12　member 表数据

图 9.13　添加博客页面

9.2.2 编辑数据

有时插入数据后，才发现录入的是错误信息或一段时间后需要更新数据，这时就要对数据进行编辑。数据更新使用 update 语句，依然通过 mysqli_query() 函数来执行该语句。

实例 05　编辑图书信息　　实例位置：光盘\Code\SL\09\05

本实例将通过 update 语句和 mysqli_query() 函数实现对数据的更新操作。具体步骤如下：

（1）创建 editBook.php 文件，获取需要编辑的图书信息。在 lists.html 图书列表页中，有如下代码：

实例05-1

```
<a href='editBook.php?id=<?php echo $rows['id'] ?>'>
    <button class="btn btn-info edit">编辑</button>
</a>
```

当单击“编辑”按钮时，页面跳转至 editBook.php 文件，并传递图书 id 参数。在 editBook.php 文件中，接收传递的 id，根据 id 查找图书信息。editBook.php 文件具体代码如下：

实例05-2

```
<?php
    //连接MySQL服务器，选择数据库
    $link = mysqli_connect("localhost", "root", "root", "database9") or
            die("连接数据库服务器失败！".mysqli_error());
    mysqli_query($link,"set names utf8");                    //设置数据库编码格式UTF-8
    $id    = $_GET['id'];                                    //获取id
    $query = 'select * from books where id ='.$id;
    $result = mysqli_query($link,$query);                    //执行查询语句
    $data   = mysqli_fetch_assoc($result);                   //获取关联数组形式的结果集
    include_once('edit.html');                               //引入模板
```

（2）创建 edit.html 文件。图书编辑页面和新增图书页面相似，不同之处在于编辑页面需要显示输入框内的值，即 value 的值。edit.html 文件关键代码如下：

实例05-3

```
//省略重复代码
<form class="form-horizontal" role="form" method="POST" action="updateBook.php">
    <input type="hidden" name="id" value="<?php echo $data['id'] ?>">
    <div class="row">
        <div class="col-md-8">
            <div class="form-group">
                <label for="name" class="col-md-2 control-label">
                    名称
                </label>
                <div class="col-md-10">
                    <input type="text" class="form-control" name="name" id="name"
                           value="<?php echo $data['name'] ?>">
                </div>
            </div>
```

传递id

```
15              <div class="form-group">
16                  <label for="name" class="col-md-2 control-label">
17                      分类
18                  </label>
19                  <div class="col-md-10">
20                      <select class="form-control" name="category">
21                          <option value="PHP"  <?php if($data["category"] == "PHP")
22                                                { echo 'selected'; } ?> >PHP</option>
23                          <option value="Java" <?php if($data["category"] == "Java")
24                                                { echo 'selected'; } ?> >Java</option>
25                          <option value="C++"  <?php if($data["category"] == "c++")
26                                                { echo 'selected'; } ?> >C++</option>
27                      </select>
28                  </div>
29              </div>
30              <div class="form-group">
31                  <label for="price" class="col-sm-2 control-label">
32                      价格
33                  </label>
34                  <div class="col-md-10">
35                      <input type="text" class="form-control" name="price" id="price"
36                             value="<?php echo $data['price'] ?>">
37                  </div>
38              </div>
39
40              <div class="form-group">
41                  <label class="col-sm-2 control-label">出版时间</label>
42                  <div class='col-md-10' >
43                      <div class='input-group date' id='publish_time'>
44                          <input type='text' class="form-control" name="publish_time"
45                                 value="<?php echo $data['publish_time'] ?>"/>
46                          <span class="input-group-addon">
47                              <span class="glyphicon glyphicon-calendar"></span>
48                          </span>
49                      </div>
50                  </div>
51              </div>
52          </div>
53      </div>
54      <div class="col-md-8">
55          <div class="form-group">
56              <div class="col-md-10 col-md-offset-2">
57                  <button type="submit" class="btn btn-primary btn-lg">
58                      <i class="fa fa-disk-o"></i>
59                      提交
```

获取下拉菜单的选中值

为value属性赋值

```
60                    </button>
61                </div>
62            </div>
63        </div>
64    </form>
65    //省略重复代码
```

上述代码中，为 <input> 标签的 value 属性赋值后，编辑页面中将会显示相应的图书内容。此外还需要注意两点：

◆ 使用 <input> 标签的隐藏域 type=hidden 传递图书 id，为下一步保存图书信息做准备。

◆ 在图书类别 select 下拉列表中，默认选中的是 select 的第一项（本代码中为“PHP”），使用 selected 属性可以设置选中为当前项。所以在每个 <option> 标签中，使用 if 语句来判断当前选项是否被选中。

在浏览器中运行 index.php 图书列表页，选择 id 为 2 的记录（“PHP 自学宝典”），单击右侧“编辑”按钮，进入编辑页面。运行效果如图 9.14 所示。

图 9.14　图书编辑页面

（3）创建 updateBook.php 文件，获取表单中提交的数据，根据隐藏域传递的 id 值，定义更新语句完成数据的更新操作，代码如下：

实例05-4

```
01  <?php
02  /* 获取POST提交数据 */
03  $id         = $_POST['id'];
04  $name       = $_POST['name'];
05  $category = $_POST['category'];
06  $price      = $_POST['price'];
07  $publish_time = $_POST['publish_time'];
08
09  $link = mysqli_connect('localhost', 'root', 'root', 'database9');
```

```php
mysqli_query($link,"set names utf8");    //设置数据库编码格式utf8
/* 检测连接是否成功 */
if (!$link) {
    printf("Connect failed: %s\n", mysqli_connect_error());
    exit();
}
/* 检测是否生成MySQLi_STMT类 */
$query = "update books set name = ?,category = ? ,price = ? ,publish_time = ?
          where id = ".$id;
$stmt  = mysqli_prepare($link, $query);
if ( !$stmt ) {
    die('mysqli error: '.mysqli_error($link));
}
/* 参数绑定 */
mysqli_stmt_bind_param($stmt, 'ssds', $name, $category, $price, $publish_time);
/* 执行prepare语句 */
mysqli_stmt_execute($stmt);
/* 根据执行结果，跳转页面 */
if(mysqli_stmt_affected_rows($stmt)){
    echo "<script>alert('修改成功');window.location.href='index.php';</script>";
}else{
    echo "<script>alert('修改失败');</script>";
}

?>
```

上述代码中，预处理和参数绑定的内容与新增图书相同，不再赘述。SQL 语句需要使用 update 语句和 where 条件来实现数据更新。

运行本实例，修改 id 为 2 的记录（“PHP 自学宝典”），修改效果如图 9.15 所示。

图 9.15　编辑图书页面

修改成功后，单击“确定”按钮，页面跳转到图书列表页，显示修改后的信息，效果如图 9.16 所示。

图 9.16　编辑后图书列表页

练一练：

（1）实现在“个人中心”页面修改用户手机号码的功能，如图 9.17 所示。（光盘\Code\Try\09\09）

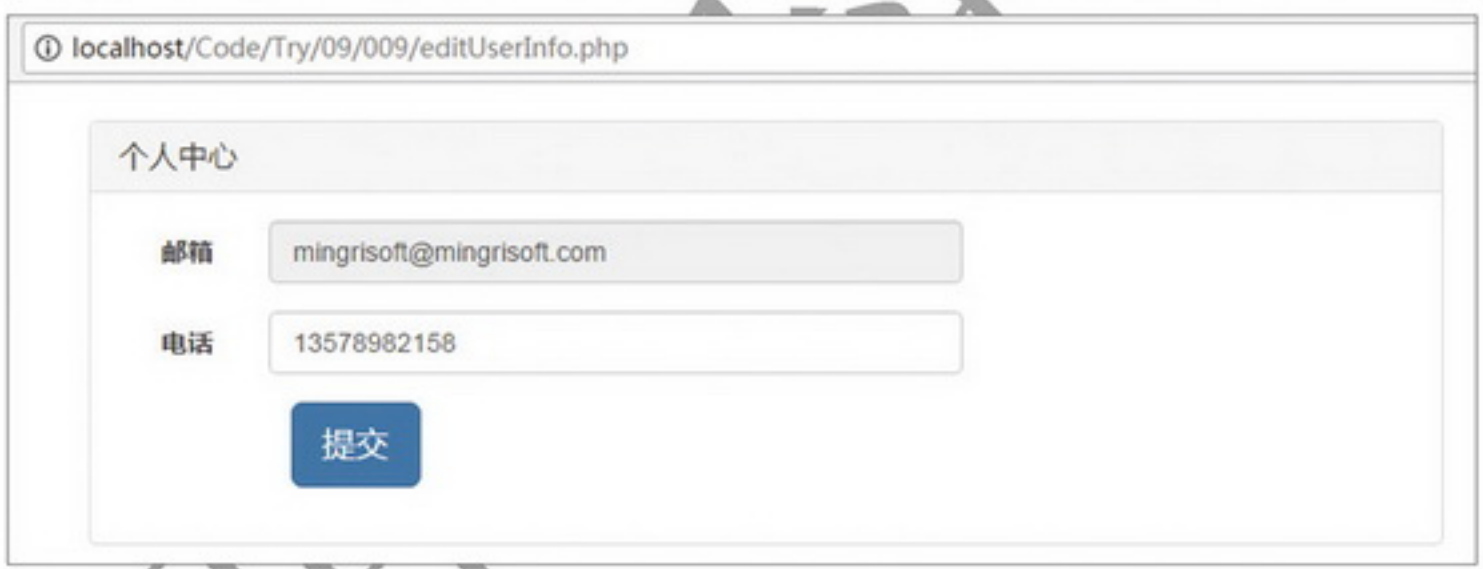


图 9.17　修改手机号码

（2）实现在“个人中心”页面修改密码的功能。（光盘\Code\Try\09\10）

9.2.3 删除数据

删除数据库中的数据，应用的是 delete 语句，如果在不指定删除条件的情况下，那么将删除指定数据表中所有的数据，如果定义了删除条件，那么将删除数据表中指定的记录。执行删除操作时一定要非常慎重，因为一旦执行该操作，数据就没有恢复的可能。

实例 06	删除图书信息	实例位置：光盘\Code\SL\09\06

在添加图书过程中，如果输入了无效的图书信息，就会用到删除数据的功能。删除数据只需利用 mysqli_query() 函数执行 delete 语句即可。在 lists.html 图书列表页中，有如下代码：

实例06-1

```
<a href='deleteBook.php?id=<?php echo $rows['id'] ?>'>
        <button class="btn btn-danger delete">删除</button>

</a>
```

当单击“删除”按钮时，页面跳转至 deleteBook.php 文件，并传递图书 id 参数。在 deleteBook.php 文件中，接收传递的 id，删除该 id 的图书记录。deleteBook.php 文件具体代码如下：

实例06-2

```
<?php
    /* 连接数据库 */
    $link = mysqli_connect('localhost','root','root','database9');
    if(!$link){
        die('mysqli connect error:'.mysqli_connect_error());
    }
    $id    = $_GET['id'];                          //获取id
    $query = "delete from books where id = ".$id;  //SQL删除语句
    /* 判断删除成功或失败 */
    if(mysqli_query($link,$query) === true ){
        echo "<script>alert('删除成功');window.location.href='index.php'</script>";
    }else{
        echo "<script>alert('删除失败');</script>";
    }
```

运行本实例，单击“id”为 2 的记录右侧的“删除”按钮，运行结果如图 9.18 所示。

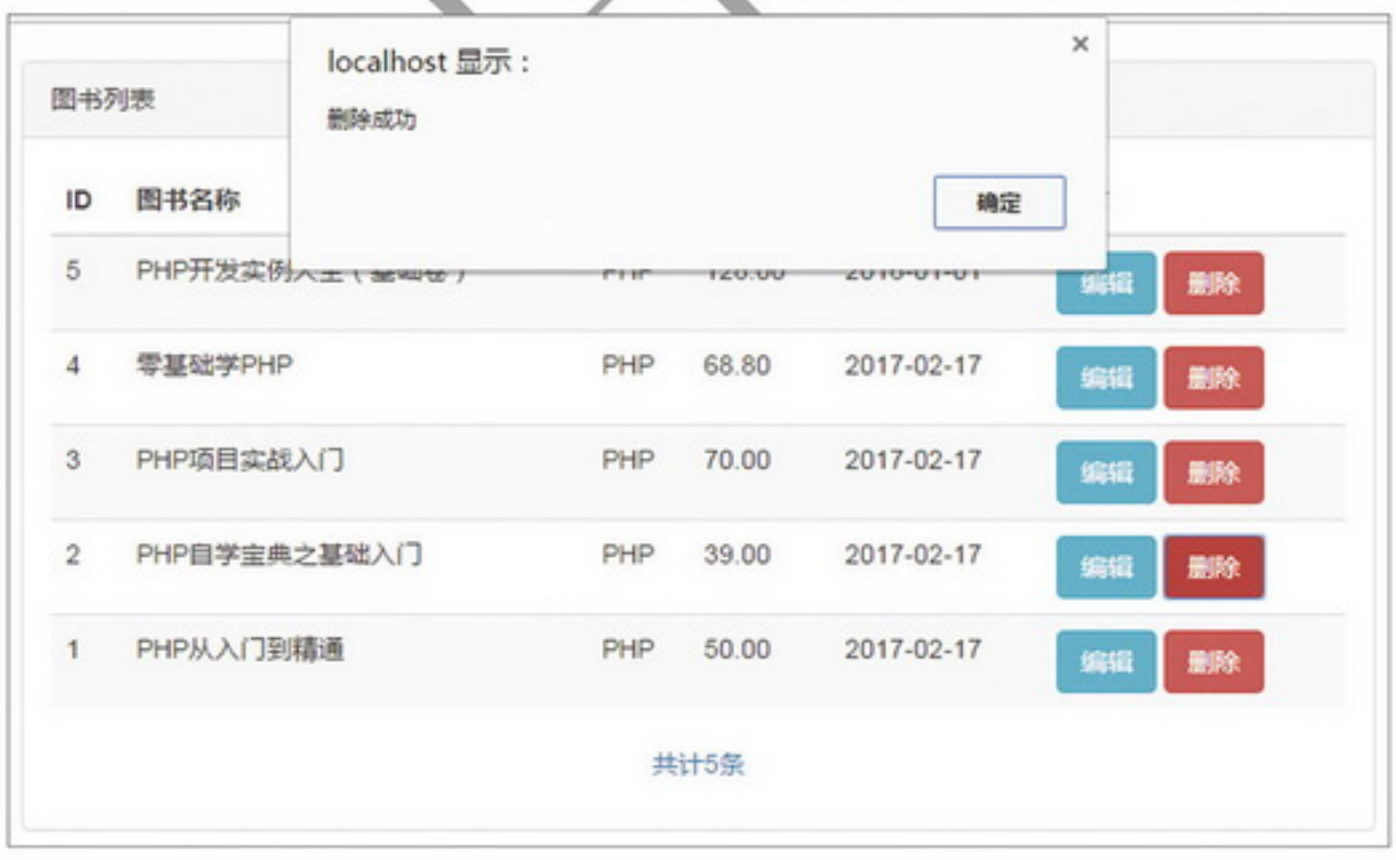


图 9.18　删除数据成功

单击“确定”按钮，页面跳转至图书列表页，此时 id 为 2 的记录被删除，不会在列表页中显示。删除后的列表页如图 9.19 所示。

图书列表

ID	图书名称	分类	价格	出版日期	操作
5	PHP开发实例大全（基础卷）	PHP	128.00	2016-01-01	编辑 删除
4	零基础学PHP	PHP	68.80	2017-02-17	编辑 删除
3	PHP项目实战入门	PHP	70.00	2017-02-17	编辑 删除
1	PHP从入门到精通	PHP	50.00	2017-02-17	编辑 删除

共计4条

图 9.19　删除数据后的列表页

注意：由于删除后，数据不可恢复，通常会在删除前弹出提示框，确定是否删除。当单击“确认”按钮后，再执行删除操作。

练一练：

（1）修改实例 06，单击“删除”按钮，弹出提示框，提示“确定删除？”，单击“确定”按钮后，再执行删除，否则不删除，如图 9.20 所示。（光盘\Code\Try\09\11）

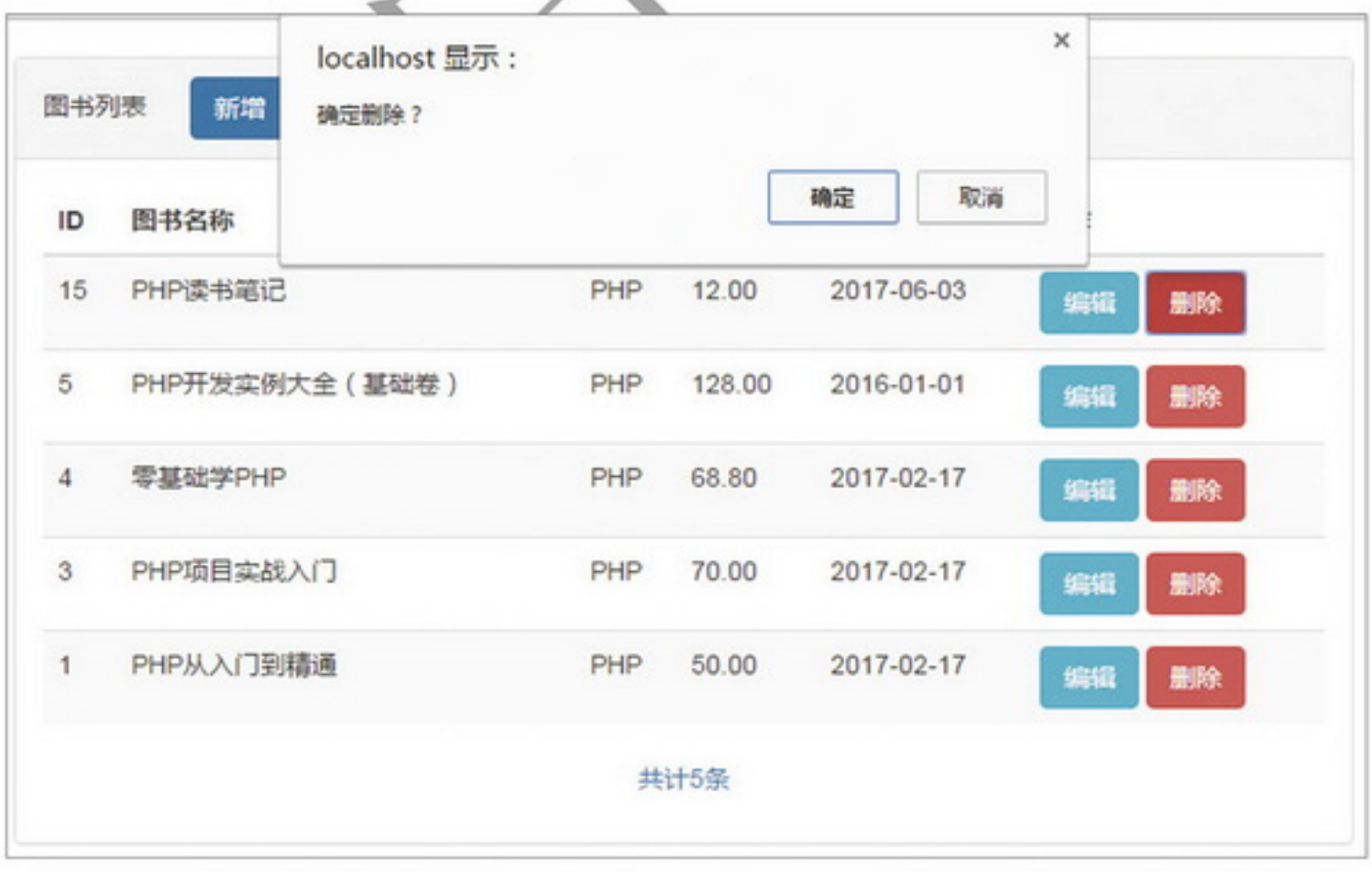


图 9.20　新增删除确认框

（2）修改实例 06，实现多选删除功能，如图 9.21 所示。（光盘\Code\Try\09\12）

图 9.21　删除多条记录

9.3 难点解答

9.3.1 四种查询函数的区别

◆ mysql_fetch_array() 函数，从结果集中取得一行作为关联数组，或数字数组，或二者兼有，除了将数据以数字索引方式储存在数组外，还可以将数据作为关联索引储存，用字段名作为键名。

◆ mysqli_fetch_object() 函数，顾名思义，从结果集中取得一行作为对象，并将字段名字作为属性。

◆ mysqli_fetch_assoc($result) 等价于 mysql_fetch_array($result,MYSQL_ASSOC)。

◆ mysqli_fetch_row($result) 等价于 mysql_fetch_array($result,MYSQL_NUM)。

9.3.2 两种预处理函数的区别

mysqli_prepare() 函数和 mysqli_stmt_prepare() 函数都能够实现 mysqli 的预处理功能。mysqli_prepare() 函数是 mysqli_stmt_prepare() 函数的简写形式，mysqli_prepare() 函数等价于如下代码：

```
$stmt = mysqli_stmt_init();                    //初始化MySQL_STMT
mysqli_stmt_prepare($sql);                     //预处理
```

9.4 小　结

本章主要介绍了使用 PHP 操作 MySQL 数据库的方法。通过本章的学习，读者能够掌握 PHP 操作 MySQL 数据库的一般流程，掌握 mysqli 扩展库中常用函数的使用方法，并能够具备独立完成基本数据库操作的能力。希望本章能够起到抛砖引玉的作用，帮助读者在此基础上更深层次地学习 PHP 操作 MySQL 数据库的相关技术，并进一步学习使用面向对象的方式操作 MySQL 数据库的方法。

9.5 动手纠错

1．运行“光盘\Code\Debug\09\01”文件夹下的 add.html 文件，添加图书成功后，页面跳转到图书列表页，出现中文乱码，如图 9.22 所示，请根据注释改正程序。

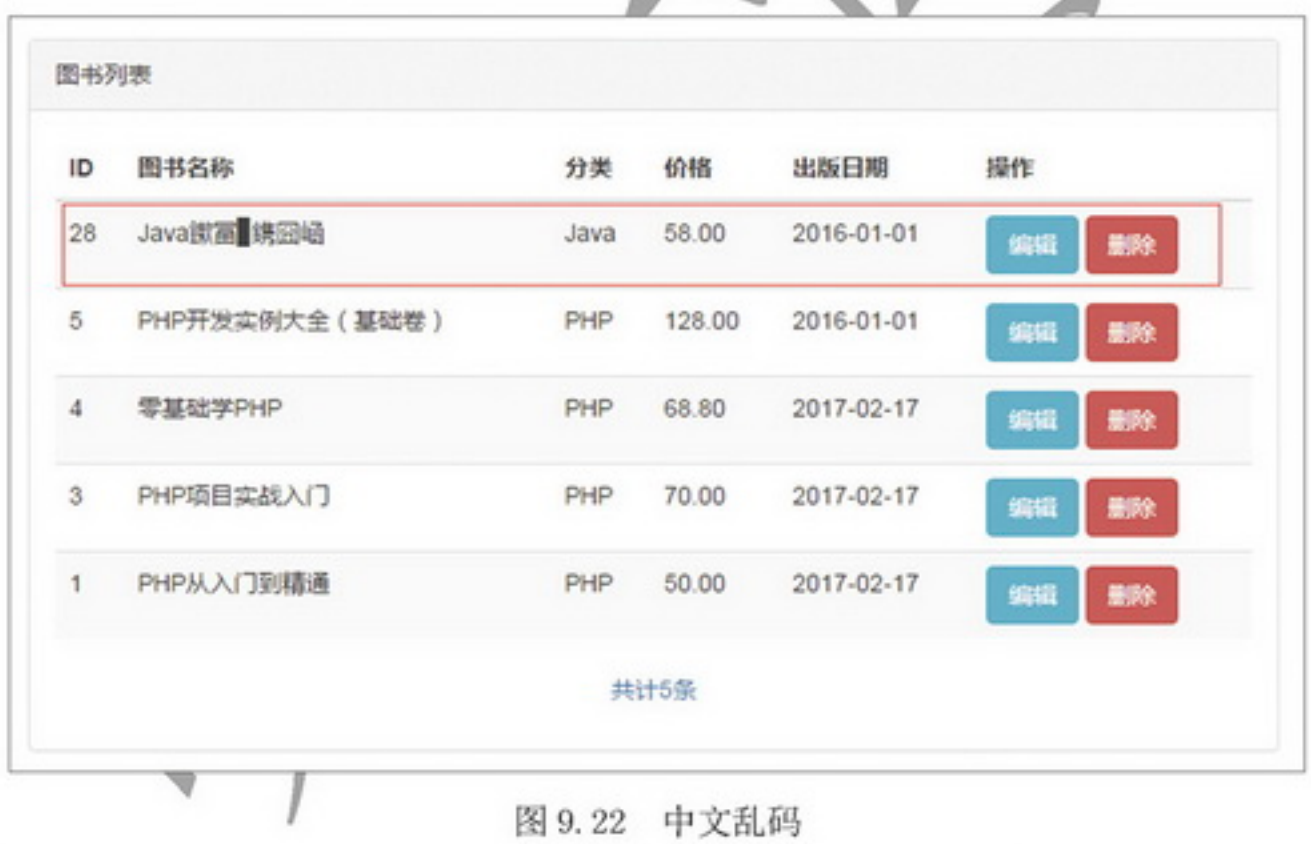


图 9.22　中文乱码

2．运行“光盘\Code\Debug\09\02”文件夹下的 index.php 文件，出现的错误提示如图 9.23 所示，请根据注释改正程序。

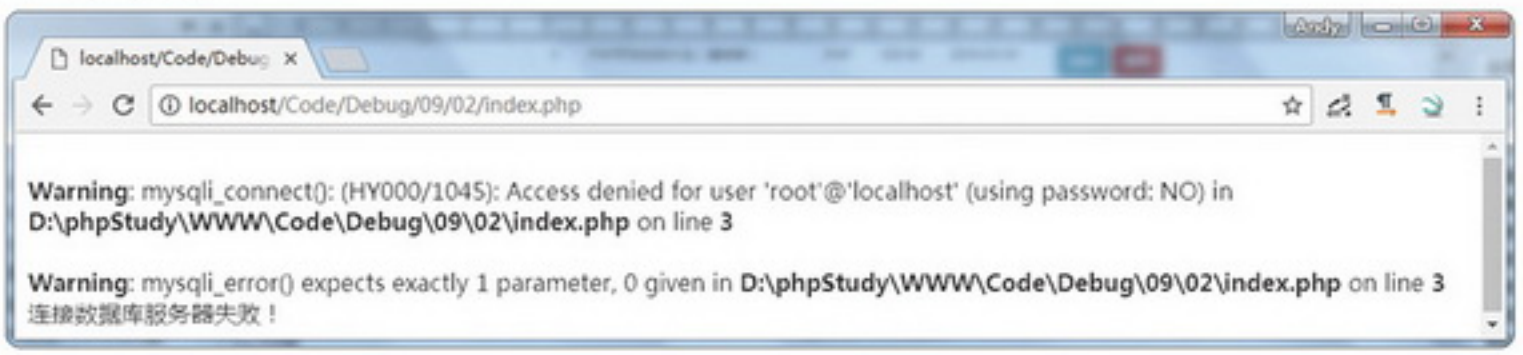


图 9.23　连接数据库服务器失败提示

3．运行“光盘\Code\Debug\09\03”文件夹下的 index.php 文件，出现“下标位置不存在”的错误提示，如图 9.24 所示，请根据注释改正程序。

图 9.24　下标位置不存在错误提示

4．运行“光盘\Code\Debug\09\04”文件夹下的 add.html 文件，添加博客后，页面跳转至博客列表页，出现“发布时间”错误。如图 9.25 所示，请根据注释改正程序。

图 9.25　发布时间错误

5．运行“光盘\Code\Debug\09\05”文件夹下的 index.php 文件，选择多条记录，单击“删除”按钮，提示“删除失败”，如图 9.26 所示，请根据注释改正程序。

图 9.26　删除多条记录失败

第 10 章

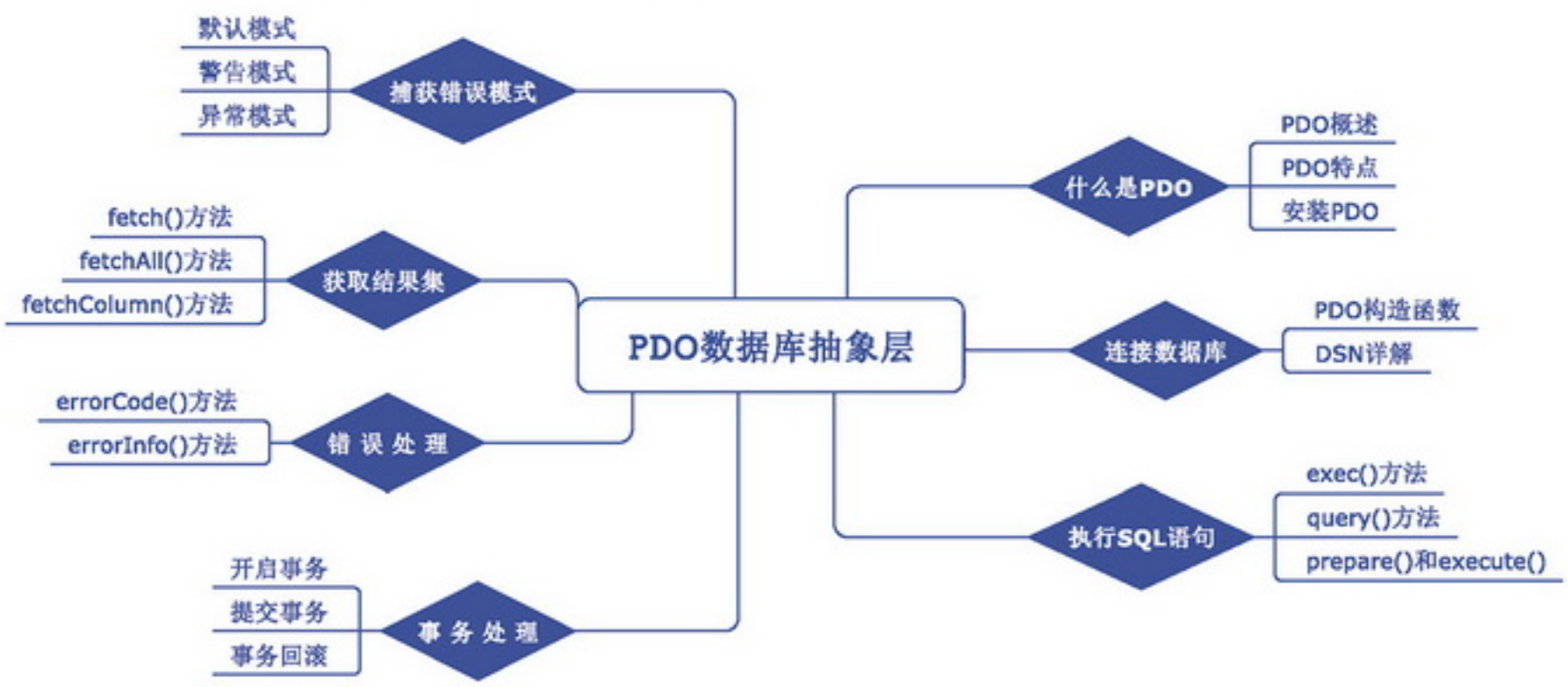
PDO 数据库抽象层

（视频讲解：48 分）

本章概览

为了解决 PHP 早期版本维护困难、可移植性差等问题，PHP 的开发人员编写了一种轻型、便利的 API 来统一各种数据库的共性，从而达到 PHP 脚本最大程度的抽象性和兼容性，它就是数据库抽象层。目前，PHP 抽象层中最为流行的是 PDO 数据库抽象层。本章不仅介绍了 PDO 数据库抽象层的特点和安装方法，还详细讲解了如何连接不同的数据库、如何执行 SQL 语句、如何获取结果集等。学习过本章内容后，读者只需要使用 PDO 接口中的方法，就可以对数据库进行操作。

知识框架



10.1 什么是 PDO

10.1.1 PDO 概述

PDO 是 PHP Date Object（PHP 数据对象）的简称，它是与 PHP 5.1 版本一起发行的，目前支持的数据库包括 Firebird、FreeTDS、Interbase、MySQL、MS SQL Server、ODBC、Oracle、Postgre SQL、SQLite 和 Sybase。有了 PDO，就不必再使用 mysql_* 函数、oci_* 函数或者 mssql_* 函数，也不必再为它们封装数据库的操作类，只需要使用 PDO 接口中的方法就可以对数据库进行操作。在选择不同的数据库时，只需修改 PDO 的 DSN（数据源名称）即可。

注意：从 PHP 5.1 开始附带了 PDO，PDO 需要 PHP 5 核心的新 OO（面向对象）特性，因此不能在较早版本的 PHP 上运行。

10.1.2 PDO 特点

PDO 是一个“数据库访问抽象层”，作用是统一各种数据库的访问接口，与 MySQL 函数库相比，PDO 让跨数据库的使用更具有亲和力；与 ADODB 和 MDB2 相比，PDO 更高效。此外，PDO 还具有以下特点：

◆ PDO 将通过一种轻型、清晰、方便的函数，来统一各种不同 RDBMS 库的共有特性，从而实现 PHP 脚本最大程度的抽象性和兼容性。

◆ PDO 吸取现有数据库扩展成功和失败的经验教训，利用 PHP 5 的最新特性，可以轻松地与各种数据库进行交互。

◆ PDO 扩展是模块化的，能够在运行时为数据库后端加载驱动程序，而不必重新编译或重新安装整个 PHP 程序。例如，PDO_MySQL 扩展会替代 PDO 扩展，实现 MySQL 数据库 API。还有一些用于 Oracle、PostgreSQL、ODBC 和 Firebird 的驱动程序。

10.1.3 安装 PDO

默认情况下，PDO 在 PHP 5.2 中为开启状态，但是要启用对某个数据库驱动程序的支持，仍需要进行相应的配置操作。

在 Windows 环境下，PDO 在 php.ini 文件中进行配置，如图 10.1 所示。

图 10.1　在 Windows 环境下配置 PDO

要启用 PDO，首先必须加载“extension=php_pdo.dll”，如果要想其支持某个具体的数据库，那么还要加载对应的数据库选项。例如，要支持 MySQL 数据库，则需要加载“extension=php_pdo_mysql.dll”选项。

注意：在完成数据库的加载后，要保存 php.ini 文件，并且重新启动 Apache 服务器，这样修改才能够生效。

10.2 PDO 连接数据库

10.2.1 PDO 构造函数

在 PDO 中，要建立与数据库的连接需要实例化 PDO 的构造函数，PDO 构造函数的语法如下：

```
__construct(string $dsn[,string $username[,string $password[,array $driver_options]]])
```

构造函数的参数说明如下：

- dsn：数据源名，包括主机名端口号和数据库名称。
- username：连接数据库的用户名。
- password：连接数据库的密码。
- driver_options：连接数据库的其他选项。

通过 PDO 连接 MySQL 数据库的代码如下：

```
01  <?php
02  header("Content-Type:text/html;charset=utf-8");          //设置页面的编码格式
03  $dbms='mysql';                                          //数据库类型
04  $dbName='database10';                                   //使用的数据库名称
05  $user='root';                                           //使用的数据库用户名
06  $pwd='root';                                            //使用的数据库密码
07  $host='localhost';                                      //使用的主机名称
08  $dsn="$dbms:host=$host;dbname=$dbName";
09  try {                                                   //捕获异常
10      $pdo=new PDO($dsn,$user,$pwd);                      //实例化对象
11      echo "PDO连接MySQL成功";
12  } catch (Exception $e) {
13      echo $e->getMessage()."<br>";
14  }
15  ?>
```

10.2.2 DSN 详解

DSN 是 Data Source Name（数据源名称）的首字母缩写。DSN 提供连接数据库需要的信息。PDO 的 DSN 包括三部分：PDO 驱动名称（例如：mysql、sqlite 或者 pgsql）、冒号和驱动特定的语法。每种

数据库都有其特定的驱动语法。

数据库服务器是完全独立于 PHP 的实体。数据库服务器可能与 Web 服务器不是在同一台计算机上，此时要通过 PDO 连接数据库时，就需要修改 DSN 中的主机名称。

每种数据库服务器都有一个默认的端口号（MySQL 默认是 3306），数据库管理员可以对端口号进行修改，因此 PHP 有可能找不到数据库的端口，此时就需要在 DSN 中包含端口号。

另外由于一个数据库服务器中可能拥有多个数据库，所以在通过 DSN 连接数据库时，通常都包括数据库名称，这样可以确保连接的是需要的数据库，而不是其他的数据库。

10.3 PDO 中执行 SQL 语句

在 PDO 中，可以使用下面的 3 种方法来执行 SQL 语句。

1. exec() 方法

exec() 方法返回执行后受影响的行数，其语法如下：

```
int PDO::exec ( string $statement )
```

参数 $statement 是要执行的 SQL 语句。该方法返回执行查询时受影响的行数，通常用于 insert、delete 和 update 语句中。

例如，使用 exec() 方法执行删除操作，删除 member 表（会员表）中 id 为 1 的记录。代码如下：

```
01 <?php
02 /** 连接数据库 **/
03 $dbh = new PDO("mysql:host=localhost;databasename=database10", "root", "root");
04 /** 执行SQL语句 **/
05 $count = $dbh->exec("DELETE FROM member WHERE id = 1");
06 /** 返回被删除的行数 **/
07 print("Deleted $count rows.\n");
08 ?>
```

2. query() 方法

query() 方法用于返回执行查询后的结果集。其语法如下：

```
PDOStatement PDO::query ( string $statement )
```

参数 $statement 是要执行的 SQL 语句，它返回的是一个 PDOStatement 对象。

例如，使用 query() 方法执行查询操作，查询 member 表中所有记录的 id。代码如下：

```
01 <?php
02     $pdo=new PDO("mysql:host=localhost;dbname=database10","root","root"); //实例化对象
03     $sql = 'SELECT * FROM member';                                          //SQL语句
```

```
04      foreach ($pdo->query($sql) as $row) {      //执行SQL语句，遍历数据
05          print $row['id'] . "\n";
06      }
07  ?>
```

3. 预处理语句——prepare() 方法和 execute() 方法

预处理语句包括 prepare() 和 execute() 两个方法。首先，通过 prepare() 方法做查询的准备工作，语法如下：

```
PDOStatement PDO::prepare ( string $statement [, array $driver_options] )
```

参数 $statement 是要执行的 SQL 语句。它返回的是一个 PDOStatement 对象。

然后，通过 execute() 方法执行查询。并且还可以通过 bindParam() 方法来绑定参数提供给 execute() 方法。其语法如下：

```
bool PDOStatement::execute ( [array $input_parameters] )
```

例如，查询 member 表中 id 大于 2 且会员等级为 C 的所有记录。代码如下：

```
01  <?php
02      //实例化对象
03      $pdo=new PDO("mysql:host=localhost;dbname=database10","root","root");
04      //prepare预处理
05      $sth = $pdo->prepare('SELECT * FROM member WHERE id > ? AND level = ?');
06      $sth->execute(array(2, 'C'));                    //execute()方法执行SQL语句，并替换参数
07      $res = $sth->fetchAll();                         //获取执行结果
08      var_dump($res);
09  ?>
```

10.4 PDO 中获取结果集

在 PDO 中获取结果集常用 3 种方法：fetch() 方法、fetchAll() 方法和 fetchColumn() 方法。

10.4.1 fetch() 方法

fetch() 方法可以获取结果集中的下一行记录，其语法格式如下：

```
mixed PDOStatement::fetch ( [int $fetch_style [, int $cursor_orientation [,
                            int $cursor_offset]]] )
```

◆ 参数 fetch_style：控制结果集的返回方式，其可选方式如表 10.1 所示。

表 10.1　fetch_style 控制结果集的可选值

值	说　明
PDO::FETCH_ASSOC	关联数组形式
PDO::FETCH_NUM	数字索引数组形式
PDO::FETCH_BOTH	两种数组形式都有，这是缺省的
PDO::FETCH_OBJ	按照对象的形式，类似于以前的 mysqli_fetch_object()
PDO::FETCH_BOUND	以布尔值的形式返回结果，同时将获取的列值赋给 bindParam() 方法中指定的变量
PDO::FETCH_LAZY	以关联数组、数字索引数组和对象 3 种形式返回结果

◆ 参数 cursor_orientation：PDOStatement 对象的一个滚动游标，可用于获取指定的一行。

◆ 参数 cursor_offset：游标的偏移量。

实例 01　使用 fetch() 方法获取明日学院会员列表

实例位置：光盘\Code\SL\10\01

本实例通过 fetch() 方法获取结果集中下一行的数据，然后应用 while 语句完成数据库中数据的循环输出，具体步骤如下：

（1）创建 member 数据表。首先创建 database10 数据库，并设置数据库的编码格式为 UTF-8。创建 member 表的 SQL 语句如下：

实例01-1

```
DROP TABLE IF EXISTS `member`;
CREATE TABLE `member` (
  `id` int(8) NOT NULL AUTO_INCREMENT,
  `nickname` varchar(255) NOT NULL,
  `email` varchar(255) DEFAULT NULL,
  `phone` varchar(11) DEFAULT NULL,
  `level` char(10) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=MyISAM AUTO_INCREMENT=1 DEFAULT CHARSET=utf8;
```

（2）插入测试数据。为了显示图书信息，需要先在 member 表中插入几条测试数据。插入数据的 SQL 语句如下：

实例01-2

```
INSERT INTO `member` VALUES ('1', '张三', 'zhangsan@mingrisoft.com', '0431-123456', 'A');
INSERT INTO `member` VALUES ('2', '李四', 'lisi@mingrisoft.com', '0431-123457', 'B');
INSERT INTO `member` VALUES ('3', '王五', 'wangwu@mingrisoft.com', '0431-123458', 'C');
INSERT INTO `member` VALUES ('4', '赵六', 'zhaoliu@mingrisoft.com', '0431-123450', 'D');
```

（3）创建数据库配置文件 config.php，代码如下：

实例01-3

```
<?php
define('DB_HOST','localhost');
define('DB_USER','root');
```

```
04  define('DB_PWD','root');
05  define('DB_NAME','database10');
06  define('DB_PORT','3306');
07  define('DB_TYPE','mysql');
08  define('DB_CHARSET','utf8');
09  define('DB_DSN', "mysql:host=".DB_HOST.";dbname=".DB_NAME.";charset=".DB_CHARSET);
10  ?>
```

（4）创建 index.php 文件，用于连接数据库，执行查询语句，并引入模板文件。index.php 文件的具体代码如下：

实例01-4

```
01  <?php
02  require "config.php";                          //引入配置文件
03  try{
04      //连接数据库、选择数据库
05      $pdo = new PDO(DB_DSN,DB_USER,DB_PWD);
06  }catch(PDOException $e){
07      //输出异常信息
08      echo $e->getMessage();
09  }
10
11  $query  = "select * from member";              //定义SQL语句
12  $result = $pdo->prepare($query);               //准备查询语句
13  $result->execute();                            //执行查询语句，并返回结果集
14  include_once('lists.html');                    //引入模板
15  ?>
```

（5）创建 lists.html 文件，显示会员信息。使用 $result->fetch(MYSQLI_ASSOC)，将结果集返回到数组中，通过 while 语句循环遍历数组，将各会员的数据插入到 <table> 表格中，具体代码如下：

实例01-5

```
01  <!DOCTYPE html>
02  <html lang="en" class="is-centered is-bold">
03  <head>
04      <meta charset="UTF-8">
05      <title>零基础</title>
06      <link href="css/bootstrap.css" rel="stylesheet">
07  </head>
08  <body>
09  <div class="container">
10      <div class="col-sm-offset-2 col-sm-8">
11          <div class="panel panel-default">
12              <div class="panel-heading">
13                  明日学院会员列表
14              </div>
```

```
15              <div class="panel-body">
16                  <table class="table table-striped task-table">
17                      <thead>
18                      <tr>
19                          <th>ID</th>
20                          <th>昵称</th>
21                          <th>邮箱</th>
22                          <th>电话</th>
23                          <th>等级</th>
24                          <th>操作</th>
25                      </tr>
26                      </thead>
27                      <tbody>
28                      <?php while($row = $result->fetch(PDO::FETCH_ASSOC)) { ?>
29                      <tr>
30                          <td class="table-text">
31                              <?php echo $row['id'] ?>
32                          </td>
33                          <td class="table-text">
34                              <?php echo $row['nickname'] ?>
35                          </td>
36                          <td class="table-text">
37                              <?php echo $row['email'] ?>
38                          </td>
39                          <td class="table-text">
40                              <?php echo $row['phone'] ?>
41                          </td>
42                          <td><?php echo $row['level'] ?></td>
43                          <td>
44                              <button class="btn btn-info edit">编辑</button>
45                              <button class="btn btn-danger delete">删除</button>
46                          </td>
47                      </tr>
48                      <?php } ?>
49                      </tbody>
50                  </table>
51              </div>
52          </div>
53      </div>
54  </div>
55  </body>
56  </html>
```

使用浏览器访问 index.php 文件，运行结果如图 10.2 所示。

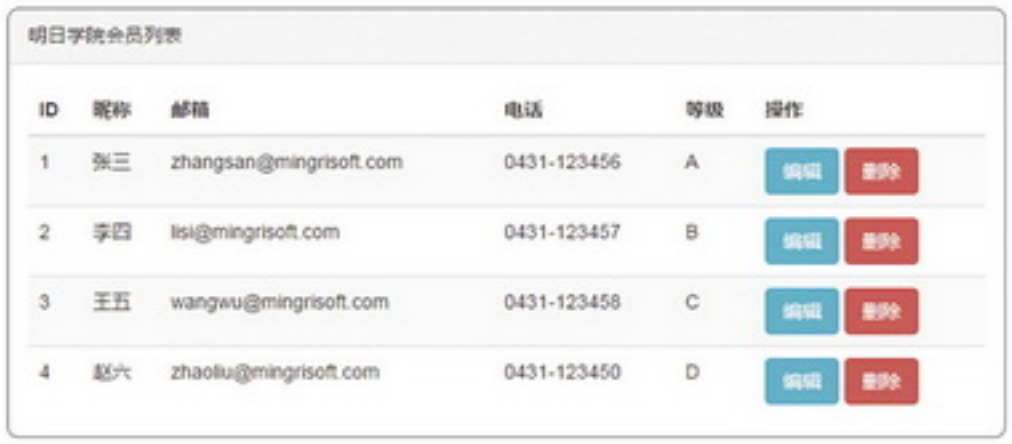


图 10.2　显示会员列表

练一练：

（1）试着修改实例 01，使用 PDO::FETCH_NUM 参数，以数字索引数组形式获取会员信息。（光盘\Code\Try\10\01）

（2）试着修改实例 01，使用 PDO::FETCH_OBJ 参数，以对象的形式获取会员信息。（光盘\Code\Try\10\02）

10.4.2 fetchAll() 方法

fetchAll() 方法可以获取结果集中的所有行。其语法如下：

```
array PDOStatement::fetchAll ( [int $fetch_style [, int $column_index]] )
```

- 参数 fetch_style：控制结果集中数据的显示方式。
- 参数 column_index：字段的索引。
- 返回值：是一个包含结果集中所有数据的二维数组。

实例 02　使用 fetchAll() 方法获取明日学院会员列表　　实例位置：光盘\Code\SL\10\02

本实例通过 fetchAll() 方法获取结果集中所有行，并且通过 foreach 语句读取二维数组中的数据，完成数据库中数据的循环输出。由于开发流程与实例 01 相似，这里只介绍修改的关键代码。

（1）创建 index.php 文件，用于连接数据库，执行查询语句，并引入模板文件。使用 fetchAll(PDO::FETCH_ASSOC) 方法获取结果集中所有行，并将结果赋值给 $data 数组。index.php 文件修改后代码如下：

实例02-1

```
<?php
require "config.php";                                    //引入配置文件
try{
    //连接数据库、选择数据库
    $pdo = new PDO(DB_DSN,DB_USER,DB_PWD);
}catch(PDOException $e){
    //输出异常信息
    echo $e->getMessage();
}
```

```php
$query  = "select * from member";                    //定义SQL语句
$result = $pdo->prepare($query);                     //准备查询语句
$result->execute();                                  //执行查询语句，并返回结果集
$data = $result->fetchAll(PDO::FETCH_ASSOC);         //获取全部数据
include_once('lists.html');                          //引入模板
?>
```

（2）创建 lists.html 文件。使用 foreach 语句遍历 $data 数组，将相应数据写入 <table> 表格，关键代码如下：

实例02-2

```php
<tbody>
<?php foreach($data as $row){ ?>
<tr>
    <td class="table-text">
        <?php echo $row['id'] ?>
    </td>
    <td class="table-text">
        <?php echo $row['nickname'] ?>
    </td>
    <td class="table-text">
        <?php echo $row['email'] ?>
    </td>
    <td class="table-text">
        <?php echo $row['phone'] ?>
    </td>
    <td><?php echo $row['level'] ?></td>
    <td>
        <button class="btn btn-info edit">编辑</button>
        <button class="btn btn-danger delete">删除</button>
    </td>
</tr>
<?php } ?>
</tbody>
```

运行结果与实例 01 相同。

练一练：

（1）试着修改实例 02，使用 PDO::FETCH_NUM 参数，以数字索引数组形式获取会员信息数组。(光盘\Code\Try\10\03)

（2）试着修改实例 02，使用 PDO::FETCH_OBJ 参数，以对象的形式获取会员信息数组。(光盘\Code\Try\10\04)

10.4.3 fetchColumn() 方法

fetchColumn() 方法可以获取结果集中下一行指定列的值。其语法如下：

```
string PDOStatement::fetchColumn ( [int $column_number] )
```

可选参数 column_number 设置行中列的索引值，该值从 0 开始。如果省略该参数则从第 1 列开始取值。

实例 03　使用 fetchColumn() 方法读取会员名

实例位置：光盘\Code\SL\10\03

本实例通过 fetchColumn() 方法获取结果集中下一行中指定列的值，注意这里是“指定列的值”，然后输出 member 表（会员表）中 nickname（用户昵称）的值。具体步骤如下：

（1）创建 index.php 文件，用于连接数据库，执行查询语句，并引入模板文件。index.php 文件的具体代码如下：

实例03-1

```
<?php
require "config.php";                                    //引入配置文件
try{
    //连接数据库、选择数据库
    $pdo = new PDO(DB_DSN,DB_USER,DB_PWD);
}catch(PDOException $e){
    //输出异常信息
    echo $e->getMessage();
}

$query  = "select nickname from member";                 //定义SQL语句
$result = $pdo->prepare($query);                         //预处理语句
$result->execute();                                      //执行查询语句，并返回结果集
include_once('lists.html');                              //引入模板
?>
```

（2）创建 lists.html 文件，使用 fetchColumn() 方法获取昵称。关键代码如下：

实例03-2

```
<tbody>
    <tr>
        <td class="table-text">
            <?php echo  $result->fetchColumn() ?>
        </td>
    </tr>
    <tr>
        <td class="table-text">
            <?php echo  $result->fetchColumn() ?>
        </td>
    </tr>
    <tr>
        <td class="table-text">
            <?php echo $result->fetchColumn() ?>
        </td>
    </tr>
    <tr>
        <td class="table-text">
```

```
19                <?php echo $result->fetchColumn() ?>
20            </td>
21        </tr>
22    </tbody>
```

运行结果如图 10.3 所示。

图 10.3　使用 fetchColumn() 方法获取用户昵称

练一练：

（1）试着修改实例 03，使用 fetchColumn() 方法获取所有会员的“邮箱”信息。（光盘\Code\Try\10\05）

（2）试着修改实例 03，使用 fetchColumn() 方法获取会员总数。（光盘\Code\Try\10\06）

10.5 PDO 中捕获 SQL 语句中的错误

在 PDO 中有 3 种方法可以捕获 SQL 语句中的错误，分别为：默认模式 (PDO::ERRMODE_SILENT)、警告模式 (PDO::ERRMODE_WARNING) 和异常模式 (PDO::ERRMODE_EXCEPTION)。下面就对这 3 种方法分别进行讲解。

10.5.1 默认模式

在默认模式中设置 PDOStatement 对象的 errorCode 属性，但不进行其他任何操作。

通过 prepare() 方法和 execute() 方法向数据库中添加数据，设置 PDOStatement 对象的 errorCode 属性，手动检测代码中的错误。

实例 04　使用默认模式捕获 SQL 语句中的错误　　实例位置：光盘\Code\SL\10\04

创建 index.php 文件。通过 PDO 连接 MySQL 数据库，通过预处理语句 prepare() 方法和 execute() 方法执行 insert 添加语句，向数据表中添加数据，并且设置 PDOStatement 对象的 errorCode 属性，检测代码中的错误。代码如下：

实例04-1

```
01  <?php
02  require "config.php";                    //引入配置文件
```

```php
try{
    //连接数据库、选择数据库
    $pdo = new PDO(DB_DSN,DB_USER,DB_PWD);
}catch(PDOException $e){
    //输出异常信息
    echo $e->getMessage();
}
//定义SQL语句
$query  = "insert into members(nickname , email) values ('mr','mr@mrsoft.com')";
$result = $pdo->prepare($query);
$result->execute();
if(!$result->errorCode()){
    echo "数据添加成功！";
}else{
    echo '错误信息：<br/>';
    echo 'SQL Query:'.$query;
    echo  '<pre>';
    print_r($result->errorInfo());
}

?>
```

在本实例中，在定义 insert 添加语句时，使用了错误的数据表名称 members（正确名称是 member），导致输出结果如图 10.4 所示。

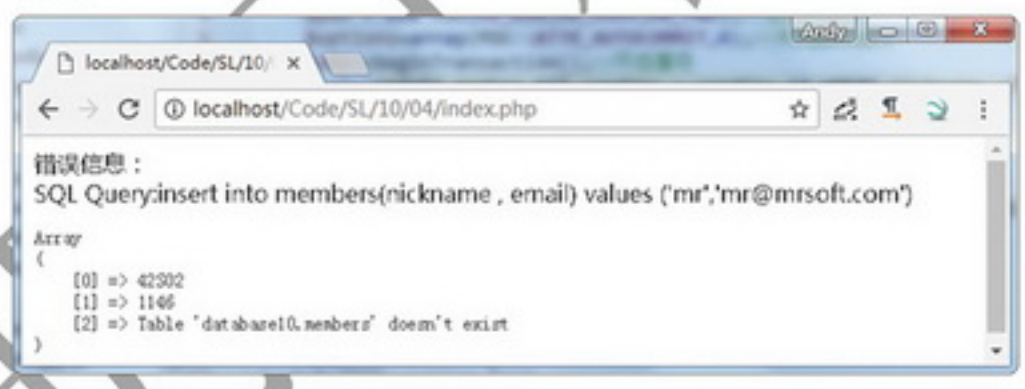


图 10.4　在默认模式中捕获 SQL 中的错误

练一练：

（1）试着修改实例 04，使用 PDO 的 setAttribute() 方法设置 PDO::ERRMODE_SILENT，输出异常信息。（光盘\Code\Try\10\07）

（2）试着删除实例 04 中 config.php 文件中的数据库密码，使用默认模式捕获 SQL 语句中的错误，如图 10.5 所示。（光盘\Code\Try\10\08）

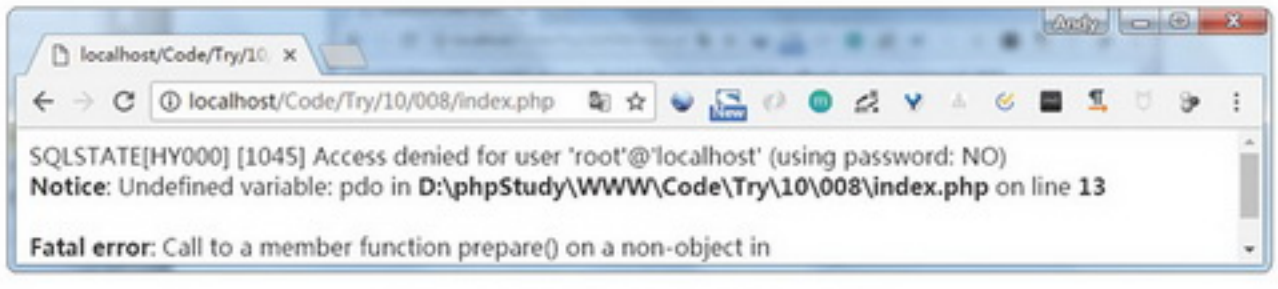


图 10.5　捕获 SQL 语句中的错误

10.5.2 警告模式

警告模式会产生一个 PHP 警告，并设置 errorCode 属性。如果设置的是警告模式，那么除非明确地检查错误代码，否则程序将继续按照其方式运行。

例如，设置警告模式，通过 prepare() 方法和 execute() 方法读取数据库中数据，体会在设置成警告模式后执行错误的 SQL 语句。具体代码如下：

```
<?php
require "config.php";                                          //引入配置文件
try{
    //连接数据库、选择数据库
    $pdo = new PDO(DB_DSN,DB_USER,DB_PWD);
}catch(PDOException $e){
    //输出异常信息
    echo $e->getMessage();
}

$pdo->setAttribute(PDO::ATTR_ERRMODE,PDO::ERRMODE_WARNING); //设置为警告模式
$query="select * from members";                                //定义SQL语句
$result=$pdo->prepare($query);                                 //预处理语句
$result->execute();                                            //执行查询语句，并返回结果集
?>
```

在设置为警告模式后，如果 SQL 语句出现错误将给出一个提示信息，但是程序仍能够继续执行下去，其运行结果如图 10.6 所示。

图 10.6　设置警告模式后捕获的 SQL 语句错误

10.5.3 异常模式

异常模式会创建一个 PDOException，并设置 errorCode 属性。它可以将执行的代码封装到一个 try{…}catch{…} 语句块中。未捕获的异常将会导致脚本中断，并显示堆栈跟踪，让用户了解是哪里出现的问题。

例如，在执行数据库中数据的删除操作时，设置为异常模式，并且编写一个错误的 SQL 语句（操作错误的数据表 members），体会异常模式与警告模式和默认模式的区别。具体代码如下：

```
<?php
require "config.php";                                          //引入配置文件
```

```
try{
    //连接数据库、选择数据库
    $pdo = new PDO(DB_DSN,DB_USER,DB_PWD);
    $pdo->setAttribute(PDO::ATTR_ERRMODE,PDO::ERRMODE_EXCEPTION);
    $query="delete from members where id = 1";          //定义SQL语句
    $result=$pdo->prepare($query);                      //预处理语句
    $result->execute();                                 //执行SQL语句
}catch(PDOException $e){
    //输出异常信息
    echo 'PDO 异常捕获: ';
    echo 'SQL Query: '.$query;
    echo '<pre>';
    echo "Error: " . $e->getMessage(). "<br/>";
    echo "Code: " . $e->getCode(). "<br/>";
    echo "File: " . $e->getFile(). "<br/>";
    echo "Line: " . $e->getLine(). "<br/>";
    echo "Trace: " . $e->getTraceAsString(). "<br/>";
    echo '</pre>';
}
?>
```

在设置为异常模式后，执行错误的 SQL 语句返回的结果如图 10.7 所示。

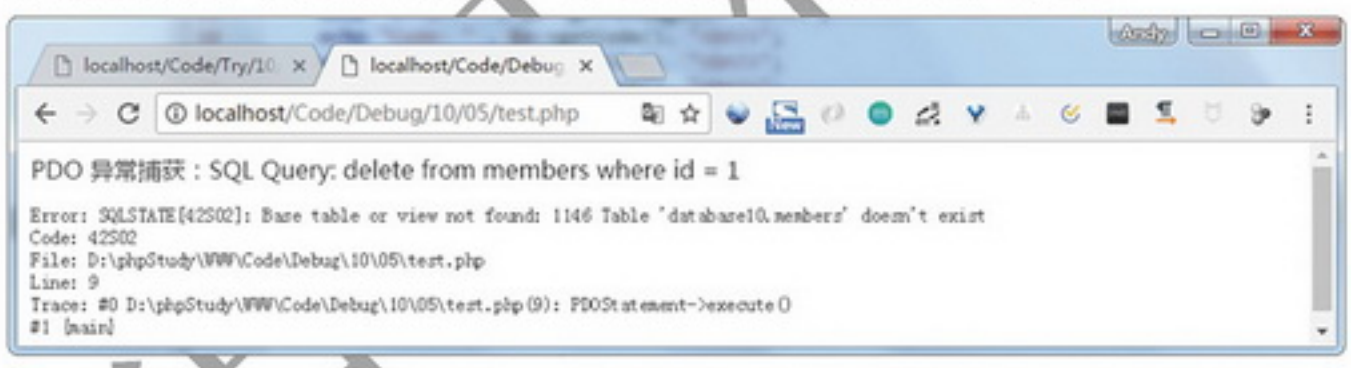


图 10.7　异常模式捕获的 SQL 语句错误信息

10.6 PDO 中的错误处理

在 PDO 中有两个获取程序中错误信息的方法：errorCode() 方法和 errorInfo() 方法。

10.6.1 errorCode() 方法

errorCode() 方法用于获取在操作数据库句柄时所发生的错误代码，这些错误代码被称为 SQLSTATE 代码。其语法格式如下：

```
int PDOStatement::errorCode ( void )
```

errorCode() 方法返回一个 SQLSTATE，SQLSTATE 是由 5 个数字和字母组成的代码。例如在 10.5.1 小节的实例中，就是通过 errorCode() 方法返回错误代码，判断 insert 插入是否成功。

10.6.2 errorInfo() 方法

errorInfo() 方法用于获取操作数据库句柄时所发生的错误信息。其语法格式如下：

```
array PDOStatement::errorInfo ( void )
```

errorInfo() 方法的返回值为一个数组，该数组包含了最后一次操作数据库的错误信息描述，如表 10.2 所示。

表 10.2　错误信息描述

数 组 元 素	信　　息
0	SQLSTATE 错误码（5 个字母或数字组成的在 ANSI SQL 标准中定义的标识符）
1	错误代码
2	错误信息

在 10.5.1 小节的实例中，使用“$result->errorInfo()”输出如下错误信息：

```
Array
(
    [0] => 42S02
    [1] => 1146
    [2] => Table 'database10.members' doesn't exist
)
```

10.7 PDO 中的事务处理

事务 (transaction) 是由查询和更新语句的序列组成。用 begin、start transaction 开始一个事务，rollback 回滚事务，commit 提交事务。在开始一个事务后，可以有若干个 SQL 查询或更新语句，每个 SQL 递交执行后，还应该有判断是否正确执行的语句，以确定下一步是否回滚，若都被正确执行则最后提交事务。事务一旦回滚，数据库则保持开始事务前的状态。所以，事务可被视为原子操作，事务中的 SQL，要么全部执行，要么一句都不执行。

PDO 中实现事务处理的方法如下：

◆ 开启事务——beginTransaction() 方法：beginTransaction() 方法将关闭自动提交（autocommit）模式，直到事务提交或者回滚以后才恢复。

◆ 提交事务——commit() 方法：commit() 方法完成事务的提交操作，成功则返回 true，否则返回 false。

◆ 事务回滚——rollback() 方法：rollback() 方法执行事务的回滚操作。

实例 05　使用 PDO 事务实现积分的获取

实例位置：光盘\Code\SL\10\05

现有一个问答网站，张三提问一个问题，需要花费 10 个积分。李四回答张三的问题，且被张三采纳，则李四获得 10 个积分，且两个事件应该同时完成。任何一个事件出错，则回滚到最初状态。使用 PDO 事务处理，具体实现如下：

（1）在 member 表中创建 credits 积分字段，初始值为 100。SQL 语句如下：

实例05-1

```
DROP TABLE IF EXISTS `member`;
CREATE TABLE `member` (
  `id` int(8) NOT NULL AUTO_INCREMENT,
  `nickname` varchar(255) NOT NULL COMMENT '昵称',
  `email` varchar(255) DEFAULT NULL,
  `phone` varchar(11) DEFAULT NULL,
  `level` char(10) DEFAULT NULL,
  `credits` int(10) NOT NULL DEFAULT '100',
  PRIMARY KEY (`id`)
) ENGINE=MyISAM AUTO_INCREMENT=1 DEFAULT CHARSET=utf8;
INSERT INTO `member` VALUES ('1', '张三', 'zhangsan@mingrisoft.com', '0431-123456', 'A', '100');
INSERT INTO `member` VALUES ('2', '李四', 'lisi@mingrisoft.com', '0431-123457', 'B', '100');
INSERT INTO `member` VALUES ('3', '王五', 'wangwu@mingrisoft.com', '0431-123458', 'C', '100');
INSERT INTO `member` VALUES ('4', '赵六', 'zhaoliu@mingrisoft.com', '0431-123450', 'D', '100');
```

创建完成后，member 表数据如图 10.8 所示。

对象　member @database10 (loc...

开始事务　备注　筛选　排序　导入　导出

id	nickname	email	phone	level	credits
1	张三	zhangsan@mingrisoft.cor	0431-123456	A	100
2	李四	lisi@mingrisoft.com	0431-123457	B	100
3	王五	wangwu@mingrisoft.com	0431-123458	C	100
4	赵六	zhaoliu@mingrisoft.com	0431-123450	D	100

图 10.8　执行事务前 member 表数据

（2）创建 index.php 文件，开启事务，执行事务，如果执行成功提示“操作成功”，否则回滚到初始状态。index.php 文件具体代码如下：

实例05-2

```
<?php
require "config.php";                                    //引入配置文件
try{
    $pdo = new PDO(DB_DSN,DB_USER,DB_PWD);               //连接数据库
    $options=array(PDO::ATTR_AUTOCOMMIT,0);              //关闭自动提交
    $pdo->beginTransaction();                            //开启事务
    $sql='update member SET credits = credits-10 WHERE nickname ="张三"';
    $res1=$pdo->exec($sql);
```

```
09      if($res1==0){
10          throw new PDOException('张三扣除积分失败');
11      }
12      $res2=$pdo->exec('update member SET credits = credits+10 WHERE nickname ="李四"');
13      if($res2==0){
14          throw new PDOException('李四获取积分失败');
15      }
16      //提交事务
17      $pdo->commit();
18      echo "采纳成功! ";
19  }catch(PDOException $e){
20      $pdo->rollBack();                    //回滚事务
21      echo $e->getMessage();               //输出异常
22  }
23  ?>
```

运行 index.php 文件，执行成功后，member 表数据如图 10.9 所示。

对象 member @database10 (loc...

开始事务 备注 筛选 排序 积分变化

id	nickname	email	phone	level	credits
1	张三	zhangsan@mingrisoft.cor	0431-123456	A	90
2	李四	lisi@mingrisoft.com	0431-123457	B	110
3	王五	wangwu@mingrisoft.com	0431-123458	C	100
4	赵六	zhaoliu@mingrisoft.com	0431-123450	D	100

图 10.9　执行事务后 member 表数据

练一练：

（1）问答网站为鼓励用户参与回答问题，被采纳的回答赠送 10 积分，未被采纳的赠送 2 积分。修改实例 05，张三提问，李四、王五、赵六均回答了张三的问题，且李四的回答被采纳，实现这个功能，运行结果如图 10.10 所示。（光盘\Code\Try\10\09）

对象 member @database10 (loc...

开始事务 备注 筛选 排序 导入 导出

id	nickname	email	phone	level	credits
1	张三	zhangsan@mingrisoft.cor	0431-123456	A	76
2	李四	lisi@mingrisoft.com	0431-123457	B	120
3	王五	wangwu@mingrisoft.com	0431-123458	C	102
4	赵六	zhaoliu@mingrisoft.com	0431-123450	D	102

图 10.10　数据库变更结果

（2）已知公司来了一名新员工，需要为新员工创建一组记录，分配一个为 23 的 id。除了将此人的基本数据记录在 staff 表（员工表）之外，还需要将他的工资记录在 salary 表（工资表）中。使用 PDO::beginTransaction() 开启事务和 PDO::commit() 自动提交事务实现该功能。（光盘\Code\Try\10\10）

10.8 难点解答

10.8.1 为什么 PDO 能够防止 SQL 注入

使用 PDO 的预处理功能可以防止 SQL 注入，那么什么是 SQL 注入呢？SQL 注入就是通过把 SQL 命令插入到 Web 表单提交或输入域名或页面请求的查询字符串，最终达到欺骗服务器执行恶意的 SQL 命令。

使用 PDO 预处理，可以将 SQL 模板和变量分两次发送给 MySQL，然后由 MySQL 完成变量的转义处理。既然变量和 SQL 模板是分两次发送的，那么就不存在 SQL 注入的问题了，但需要在 DSN 中指定 charset 属性，如：

```
$pdo = new PDO('mysql:host=localhost;dbname=test;charset=utf8', 'root');
```

10.8.2 PDO 类和 PDOStatement 的关系

PDO 类和 PDOStatement 的关系，与 mysqli_connect() 函数和 mysqli_query() 函数的关系类似。PDO 类用来执行 SQL 和管理链接，而 PDOStatement 类只用来处理结果集。

10.9 小　结

本章重点介绍了数据库抽象层——PDO，从它的概述、特点和安装开始讲解，到它的实际应用，包括：如何连接不同的数据库、如何执行 SQL 语句、如何获取结果集，以及错误处理，再到它的高级应用事务都进行了详细的讲解，并且配有相应的实例。通过本章的学习，相信读者能够掌握 PDO 技术的应用。

10.10 动手纠错

1．运行“光盘\Code\Debug\10\01”文件夹下的 index.php 文件，出现“could not find driver”的错误提示，如图 10.11 所示，请根据注释改正程序。

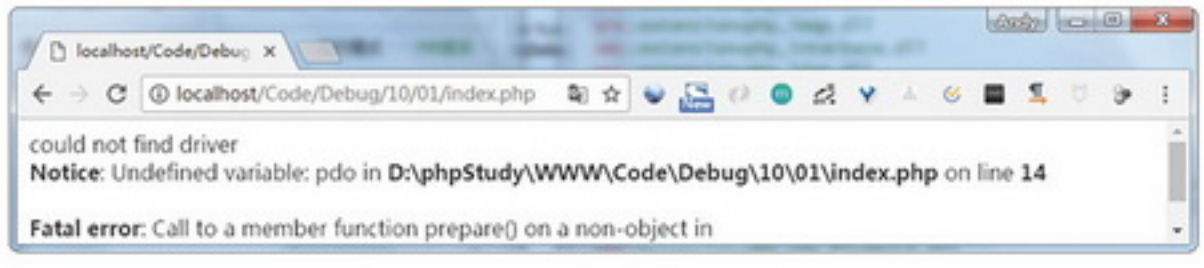


图 10.11　没有 PDO 驱动错误提示

2．运行“光盘\Code\Debug\10\02”文件夹下的 index.php 文件，出现“中文乱码”，如图 10.12 所示，请根据注释改正程序。

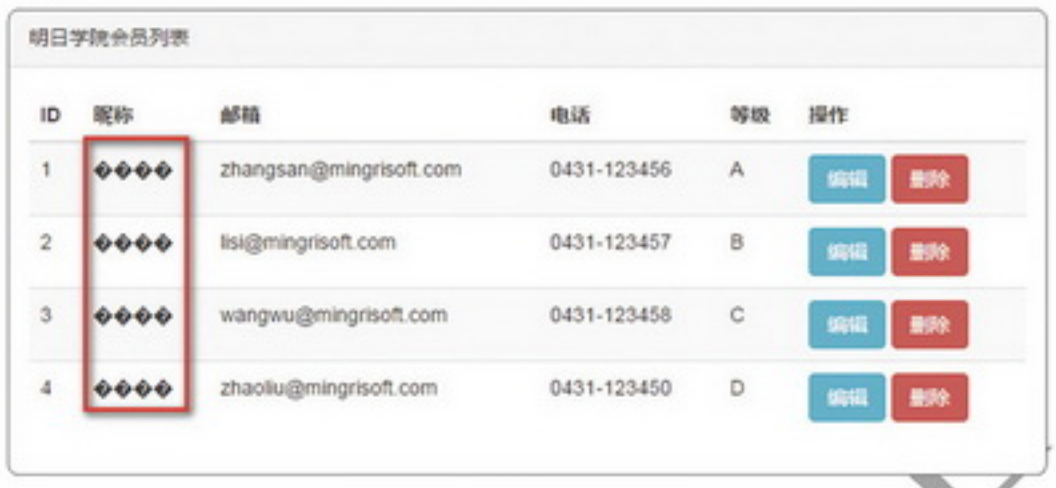


图 10.12　中文乱码

3．运行“光盘\Code\Debug\10\03”文件夹下的 index.php 文件，使用 order by 和 limit 查询条件后，筛选数据返回空值，如图 10.13 所示。请根据注释改正程序。

图 10.13　筛选数据为空

4．运行“光盘\Code\Debug\10\04”文件夹下的 index.php 文件，出现“Undefine offset”的错误提示。如图 10.14 所示，请根据注释改正程序。

图 10.14　没有定义索引错误提示

5．运行“光盘\Code\Debug\10\05”文件夹下的 index.php 文件，出现“Call to undefined method PDOStatement::exec()”的错误提示，如图 10.15 所示。请根据注释改正程序。

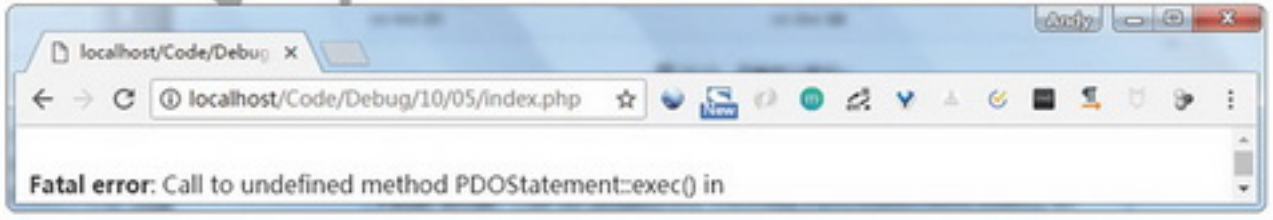


图 10.15　PDOStatement 类的 exec() 方法不存在错误提示

第 3 篇

高级应用

- 第 11 章　Cookie 与 Session
- 第 12 章　图形图像处理技术
- 第 13 章　文件系统
- 第 14 章　PHP 与 Ajax 技术
- 第 15 章　ThinkPHP 框架

第 11 章

Cookie 与 Session

（视频讲解：48 分）

本章概览

Cookie 和 Session 是两种不同的存储机制，前者是从一个 Web 页到下一个页面的数据传递方法，存储在客户端；后者是让数据在页面中持续有效的方法，存储在服务器端。

本章主要讲解了创建、读取、删除 Cookie 的具体操作方法，以及 Session 管理和高级应用。掌握 Cookie 和 Session 技术，对于提高 Web 网站页面间信息传递的安全性是必不可少的，希望读者可以熟练掌握并灵活运用。

知识框架

当我们运行一个应用程序（如 QQ）时，会打开它，进行某些操作，然后关闭它，这个过程很像一次会话。计算机清楚你是谁，它知道何时启动应用程序，并在何时终止。但是在 Internet 上，存在一个问题：服务器不知道你是谁以及你做什么，这是因为 HTTP 地址不能维持状态，所以需要通过在服务器上存储用户信息以便后面使用。Cookie 和 Session 解决了这个问题（比如保存用户名称、购买商品等）。不过，会话信息是临时的，在用户离开网站后或会话过期后将被系统自动删除。如果需要永久储存信息，可以把数据存储在数据库中。下面就学习 Cookie 和 Session 的相关知识。

11.1 Cookie 管理

Cookie 是在 HTTP 协议下，服务器或脚本维护客户工作站上信息的一种方式。Cookie 的使用很普遍，许多提供个人化服务的网站都是利用 Cookie 来区别不同用户，以显示与用户相应的内容，如 Web 接口的免费 e-mail 网站，就需要用到 Cookie。有效地使用 Cookie 可以轻松完成很多复杂任务。下面对 Cookie 的相关知识进行详细介绍。

11.1.1 了解 Cookie

1. 什么是 Cookie

Cookie 是一种在远程浏览器端存储数据并以此来跟踪和识别用户的机制。简单地说，Cookie 是 Web 服务器暂时存储在用户硬盘上的一个文本文件，并被 Web 浏览器读取。当用户再次访问 Web 网站时，网站通过读取 Cookie 文件记录这位访客的特定信息（如上次访问的位置、花费的时间、用户名和密码等），从而迅速做出响应，如在页面中不需要输入用户的 id 和密码即可直接登录网站等。

举个简单的例子，如果用户的系统盘为 C 盘，操作系统为 Windows 7，当使用 IE 浏览器访问 Web 网站时，Web 服务器会生成相应的 Cookie 文本文件，并存储在用户硬盘的指定位置，如图 11.1 所示。

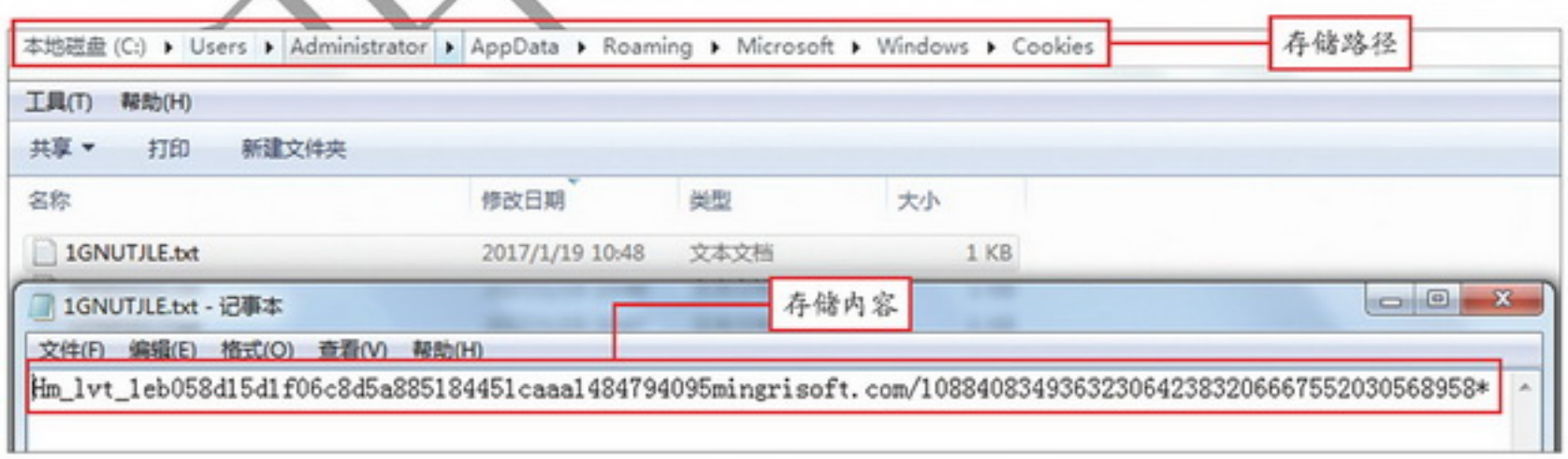


图 11.1　Cookie 文件的存储路径

说明：谷歌浏览器的 Cookie 数据位于：C:\Users\Administrator\AppData\Local\Google\Chrome\User Data\Default\Cookies。其中，Administrator 是电脑用户名。在 IE 浏览器中，IE 将各个站点的 Cookie 分别保存为一个 XXX.txt 的纯文本文件（文件个数可能很多，但文件都较小）；而 Firefox 和 Chrome 是将所有的 Cookie 都保存在一个文件中（文件较大），该文件为 SQLite3 数据库格式的文件。

2. Cookie 的功能

Web 服务器可以应用 Cookie 包含信息的任意性来筛选并经常性维护这些信息，并以此判断在 HTTP 传输中的状态。Cookie 常用于以下 3 个方面：

◆ 记录访客的某些信息。如可以利用 Cookie 记录用户访问网页的次数，或者记录访客曾经输入过的信息，另外，某些网站可以使用 Cookie 自动记录访客上次登录的用户名。

◆ 在页面之间传递变量。浏览器并不会保存当前页面上的任何变量信息，当页面被关闭时，页面上的所有变量信息将随之消失。如果用户声明一个变量 id=8，要把这个变量传递到另一个页面，可以把变量 id 以 Cookie 形式保存下来，然后在下一页通过读取该 Cookie 来获取变量的值。

◆ 将所查看的网页存储在 Cookie 临时文件夹中，可以提高以后浏览的速度。

注意：一般不要用 Cookie 保存数据集或其他量大的数据。并非所有的浏览器都支持 Cookie，并且数据信息是以明文文本的形式保存在客户端计算机中，因此最好不要保存敏感的或未加密的数据，否则会影响网络的安全性。

11.1.2 创建 Cookie

在 PHP 中通过 setcookie() 函数创建 Cookie。在创建 Cookie 之前必须了解的是，Cookie 是 HTTP 头标的组成部分，而头标必须在页面其他内容之前发送，它必须最先输出。若在 setcookie() 函数前输出一个 HTML 标签或 echo 语句，甚至一个空行都会导致程序出错。

语法格式如下：

```
bool setcookie(string name[,string value[,int expire[, string path[,string domain[,int
                secure]]]]])
```

setcookie() 函数的参数说明如表 11.1 所示。

表 11.1　setcookie() 函数的参数说明

参　数	说　明	举　例
name	Cookie 的变量名	可以通过 $_COOKIE["cookiename"] 调用变量名为 cookiename 的 Cookie
value	Cookie 变量的值，该值保存在客户端，不能用来保存敏感数据	可以通过 $_COOKIE["values"] 获取名为 values 的值
expire	Cookie 的失效时间，expire 是标准的 UNIX 时间标记，可以用 time() 函数获取，单位为秒	如果不设置 Cookie 的失效时间，那么 Cookie 将永远有效，除非手动将其删除
path	Cookie 在服务器端的有效路径	如果该参数设置为“/”，则它在整个 domain 内有效，如果设置为“/11”，则它在 domain 下的 /11 目录及子目录内有效。默认是当前目录
domain	Cookie 有效的域名	如果要使 Cookie 在 mingrisoft.com 域名下的所有子域都有效，应该设置为 mingrisoft.com

续表

参　数	说　明	举　例
secure	指明 Cookie 是否仅通过安全的 HTTPS，值为 0 或 1	如果值为 1，则 Cookie 只能在 HTTPS 连接上有效；如果值为默认值 0，则 Cookie 在 HTTP 和 HTTPS 连接上均有效

例如，使用 setcookie() 函数创建 Cookie，代码如下：

```
<?php
setcookie("MRSOFT",'www.mingrisoft.com');
setcookie("MRBOOK", 'www.mrbccd.com', time()+60);          //设置Cookie的有效时间为60秒
?>
```

在谷歌浏览器下运行本实例，按如下步骤查看 Cookie。

首先鼠标右键单击浏览器页面，弹出如图 11.2 所示对话框，然后单击“检查”选项。在弹出的对话框中单击 Application 后，在对话框左侧选择 Storage → Cookies → http://localhost，即可看到 Cookie 内容，操作步骤如图 11.3 所示。

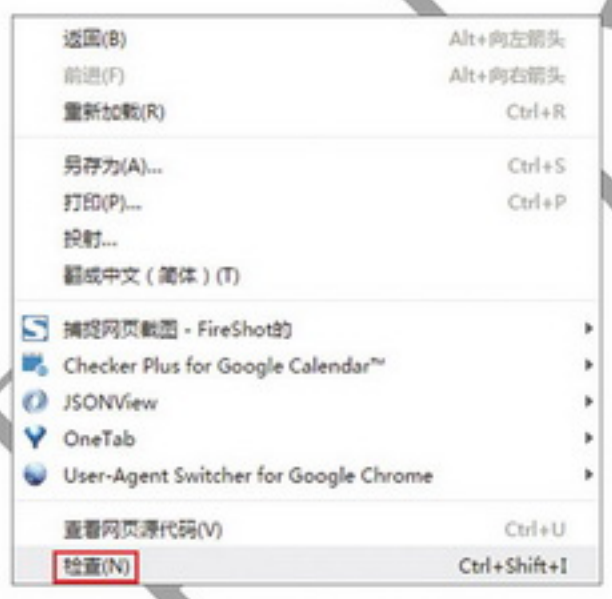


图 11.2　检查元素

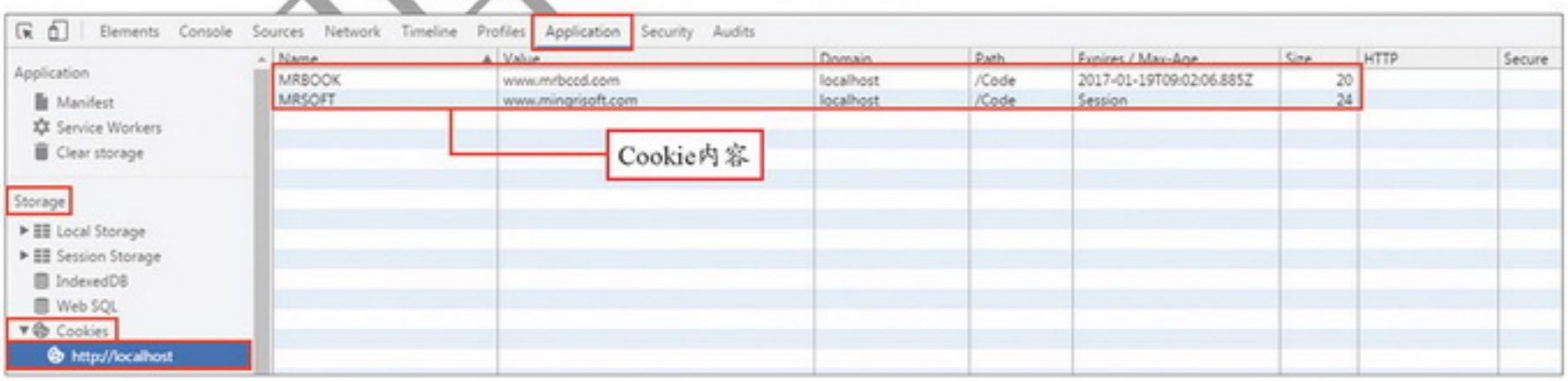


图 11.3　查看 Cookie 内容

11.1.3　读取 Cookie

在 PHP 中可以直接通过超级全局数组 $_COOKIE[] 来读取客户端的 Cookie 值。

例如，使用 $_COOKIE[] 读取 Cookie 变量，代码如下：

```php
<?php
    date_default_timezone_set('PRC');                          //设置时区
    if(!isset($_COOKIE["visittime"])){                         //检测Cookie文件是否存在，如果不存在
        setcookie("visittime",date("Y-m-d H:i:s"));            //设置一个Cookie变量
        echo "欢迎您第一次访问网站！";                          //输出字符串
    }else{                                                     //如果Cookie存在
        setcookie("visittime",date("Y-m-d H:i:s"),time()+60);  //设置保存Cookie失效时间
        echo "您上次访问网站的时间为：".$_COOKIE["visittime"];   //输出上次访问网站的时间
        echo "<br>";                                           //输出回车符
    }
    echo "您本次访问网站的时间为：  ".date("Y-m-d H:i:s"); //输出当前的访问时间
?>
```

在上面的代码中，首先使用 isset() 函数检测 Cookie 文件是否存在，如果不存在，则使用 setcookie() 函数创建一个 Cookie，并输出相应的字符串；如果 Cookie 文件存在，则使用 setcookie() 函数设置 Cookie 文件失效的时间，然后输出用户上次访问网站的时间。最后在页面输出本次访问网站的当前时间。

首次运行本实例，由于没有检测到 Cookie 文件，运行结果如图 11.4 所示。如果用户在 Cookie 设置到期时间（本例为 60 秒）前刷新或再次访问该实例，运行结果如图 11.5 所示。

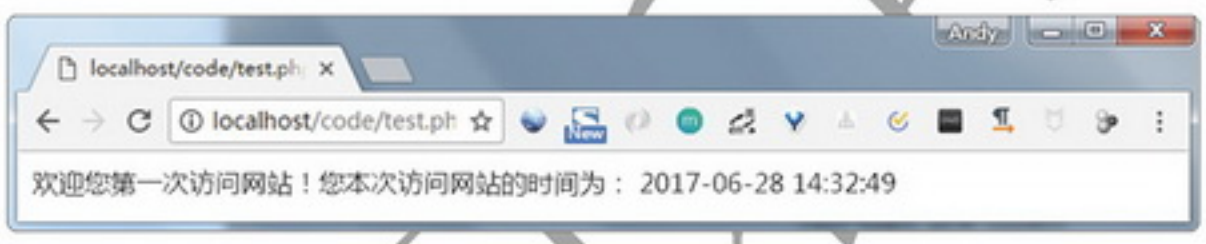


图 11.4　第一次访问网页的运行结果

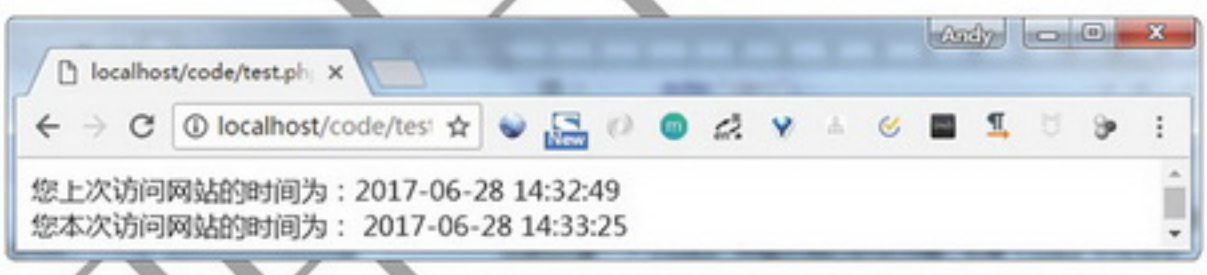


图 11.5　刷新或再次访问本网页后的运行结果

注意：如果未设置 Cookie 的到期时间，则在关闭浏览器时自动删除 Cookie 数据。如果为 Cookie 设置了到期时间，浏览器将会记住 Cookie 数据，即使用户重启计算机，只要没到期，再访问网站时也会获得图 11.5 所示的数据信息。

11.1.4 删除 Cookie

当 Cookie 被创建后，如果没有设置它的失效时间，Cookie 文件会在关闭浏览器时被自动删除。如果要在关闭浏览器之前删除 Cookie 文件，方法有两种：一种是使用 setcookie() 函数删除，另一种是在浏览器中手动删除。下面分别进行介绍。

1. 使用 setcookie() 函数删除 Cookie

删除和创建 Cookie 的方式基本类似，删除 Cookie 也使用 setcookie() 函数。删除 Cookie 只需要将

setcookie() 函数中的第二个参数设置为空值，将第 3 个参数 Cookie 的过期时间设置为小于系统的当前时间即可。

例如，将 Cookie 的过期时间设置为当前时间减 1 秒，代码如下：

```
setcookie("name", "", time()-1);
```

在上面的代码中，time() 函数返回以秒表示的当前时间戳，把过期时间减 1 秒就会得到过去的时间，从而删除 Cookie。

注意：把过期时间设置为 0，可以直接删除 Cookie。

2. 在浏览器中手动删除 Cookie

在使用 Cookie 时，Cookie 自动生成一个文本文件存储在客户端电脑上。不同浏览器或者同一浏览器的不同版本（如 IE 6 和 IE 11）手动删除 Cookie 的方式都不相同。

11.1.5 Cookie 的生命周期

如果 Cookie 不设定时间，就表示它的生命周期为浏览器会话的时间，只要关闭浏览器，Cookie 就会自动消失。这种 Cookie 被称为会话 Cookie，一般不保存在硬盘上，而是保存在内存中。

如果设置了过期时间，那么浏览器会把 Cookie 保存到硬盘中，再次打开 IE 浏览器时会依然有效，直到它的有效期超时。

虽然 Cookie 可以长期保存在客户端浏览器中，但也不是一成不变的。因为浏览器最多允许存储 300 个 Cookie 文件，而且每个 Cookie 文件支持的最大容量为 4KB；每个域名最多支持 20 个 Cookie，如果达到限制时，浏览器会自动地随机删除 Cookie。

11.1.6 7 天免登录功能的实现

登录“明日学院”网站时，有一个“7 天免登录”功能选项。当选择这个选项并登录成功后，7 天之内浏览“明日学院”网站，就不需要再次登录，网站会自动保留登录信息。下面就来实现这个功能。

实例 01　实现 7 天免登录功能　　实例位置：光盘\Code\SL\11\01

实现 7 天免登录功能的具体步骤如下：

（1）创建数据表。创建 database11 数据库，在该数据库中创建 users 数据表及数据。SQL 语句如下：

实例01-1

```
DROP TABLE IF EXISTS `users`;
CREATE TABLE `users` (
    `id` int(8) NOT NULL AUTO_INCREMENT,
    `username` char(50) NOT NULL,
    `password` varchar(255) DEFAULT NULL,
    PRIMARY KEY (`id`)
```

```
) ENGINE=MyISAM AUTO_INCREMENT=1 DEFAULT CHARSET=utf8;

INSERT INTO `users` VALUES ('1', 'mr', 'fdb390e945559e74475ed8c8bbb48ca5');
```

创建完成后，如图 11.6 所示。

图 11.6　新增 users 表及数据

（2）创建登录页。创建一个 login.php 文件，该文件中包含一个 Form 表单，表单内有“用户名”“密码”“是否开启 7 天免登录”三个字段。login.php 文件具体代码如下：

实例01-2

```
<!DOCTYPE html>
<html lang="en" class="is-centered is-bold">
<head>
    <meta charset="UTF-8">
    <title>零基础</title>
    <link href="css/main.css" rel="stylesheet">
</head>
<body>
<section style="background: transparent">
    <form class="box py-3 px-4 px-2-mobile" role="form" method="post"
          action="checkLogin.php" onsubmit="return check()">
        <div class="is-flex is-column is-justified-to-center">
            <h1 class="title is-3 mb-a has-text-centered">
                登录
            </h1>
            <div class="inputs-wrap py-3">
                <div class="control">
                    <input type="text" id="username" name="username" class="input"
                           placeholder="用户名" value="" required>
                </div>
                <div class="control">
                    <input type="password" id="password" name="password" class="input"
                           placeholder="密码" required>
                </div>
                <div class="control">
                    <button type="submit" class="button is-submit is-primary is-outlined">
                        提交
                    </button>
                </div>
```

```
30              </div>
31              <footer class="is-flex is-justified-space-between">
32                  <div>
33                    <input type="checkbox" name="keep" id="keep" checked value="">7天免登录
34                  </div>
35                  <a href="register.html">
36                      暂无账号，点击去注册
37                  </a>
38              </footer>
39          </div>
40      </form>
41  </section>
42  <script>
43      function check() {
44          //判断是否勾选免登录
45          if(document.getElementById("keep").checked){
46              document.getElementById("keep").value = 1;
47          }
48      }
49  </script>
50  </body>
51  </html>
```

上述代码中，使用 checkbox 复选框，设置属性为 checked，即表示默认情况下是选中状态。如果用户选中“7 天免登录”，则该复选框的 value 值为 1，否则为空。运行效果如图 11.7 所示。

图 11.7　登录页面

（3）检测是否登录成功。当用户在填写完“用户名”和“密码”后，单击“提交”按钮，将表单提交到 checkLogin.php 文件。在该文件中处理业务逻辑。首先，以 PDO 方式连接数据库，然后在

users 表中查找用户名和密码。如果存在这条记录，则判断用户是否勾选“7 天免登录”。如果勾选，则将用户名存入 Cookie，并保存 7 天，否则使用 Cookie 的默认保存时间。如果不存在这条记录，则直接提示“用户名和密码不匹配”。checkLogin.php 文件具体代码如下：

实例01-3

```php
<?php
if(isset($_POST['username']) && isset($_POST['password'])){
    $username = trim($_POST['username']);                    //trim()函数去除前后空格
    $password = md5(trim($_POST['password']));               //trim()函数去除前后空格，使用md5加密
    require "config.php";                                    //引入配置文件
    try{
        //连接数据库、选择数据库
        $pdo = new PDO("mysql:host=".DB_HOST.";dbname=".DB_NAME,DB_USER,DB_PWD);
    }catch(PDOException $e){
        //输出异常信息
        echo $e->getMessage();
    }
    //users表中查找输入的用户名和密码是否匹配
    $sql = 'select * from users where username = :username and password = :password';
    $res = $pdo->prepare($sql);
    $res->bindParam(':username',$username);                   //绑定参数
    $res->bindParam(':password',$password);                   //绑定参数
    if($res->execute()){
        $rows = $res->fetch(PDO::FETCH_ASSOC);               //返回一个索引为结果集列名的数组
        if($rows){
            if(isset($_POST['keep'])){                       //勾选“7天免登录”
                setcookie('username',$rows['username'],time()+604800); //设置cookie
            }else{                                           //正常登录
                setcookie('username',$rows['username']); //设置cookie
            }
            echo "<script>alert('恭喜您，登录成功！');
                        window.location.href='index.php'; </script>";
        }else{
            echo "<script>alert('用户名或密码错误，登录失败！');history.back();</script>";
            exit ();
        }
    }
}else{                                                       //如果不是POST提交，跳转到登录页
    echo "<script>window.location.href='login.html'</script>";
}

?>
```

输入用户名“mr”，密码“mrsoft”，登录成功后，运行效果如图 11.8 所示。输入错误的用户名或密码登录时，运行效果如图 11.9 所示。

图 11.8　登录成功

图 11.9　登录失败

（4）自动登录。登录成功后，页面跳转到“index.php”文件。在该文件中，判断 $_COOKIE['username'] 是否存在，如果存在，表示登录成功，显示该页面，否则跳转到登录页。如果勾选“7 天免登录”，当关闭 index.php 页面后，再次访问该页面，会直接显示页面内容，不需要重新登录。index.php 文件具体代码如下：

实例01-4

```
<?php
    if(!isset($_COOKIE['username'])){
        echo "<script>alert('请先登录');window.location.href='login.php'</script>";
    }
?>
<!DOCTYPE html>
<html lang="en" class="is-centered is-bold">
<head>
    <meta charset="UTF-8">
    <title>零基础</title>
    <link rel="stylesheet" href="css/bootstrap.css">
</head>
<body class="container">
<div class="jumbotron" style="background-color: #17ecf1;">
    <h1>欢迎
        <span style="color: white">
            <?php echo $_COOKIE['username']?>
        </span>
         登录网站
    </h1>
    <p><a class="btn btn-primary btn-lg" href="logout.php" role="button">退出登录</a></p>
</div>
```

```
23  </body>
24  </html>
```

运行效果如图 11.10 所示。

图 11.10　index.php 页面效果

（5）退出登录。在 index.php 页面，单击“退出登录”按钮，页面跳转到 logout.php 文件。在该文件中，使用 setcookie() 函数删除 Cookie，并跳转到登录页面。此时，再次访问 index.php 页面，由于 Cookie 已经被删除，所以还会跳转到 login.php 登录页面。logout.php 文件代码如下：

实例01-5

```
01  <?php
02      setcookie("username", "", time()-1);
03      echo "<script>window.location.href='login.php'</script>";
04  ?>
```

练一练：

（1）试着修改实例 01，在用户登录成功时，将登录时间存入 session['time']，在 index.php 文件中，根据当前时间和 session['time']，统计用户在线时间，运行效果如图 11.11 所示。（光盘\Code\Try\11\01）

图 11.11　统计在线时间

（2）试着修改实例 01，在 index.php 文件中，新增“清除 Cookie”按钮，如图 11.12 所示。单击该按钮，清除所有 Cookie。（光盘\Code\Try\11\02）

图 11.12　清除 Cookie

11.2 Session 管理

对比 Cookie 和 Session 会话文件中保存的数据是以变量的形式创建的，创建的会话变量在生命周期（24 分钟）中可以被跨页的请求所引用。另外，Session 是存储在服务器端的会话，相对安全，并且不像 Cookie 那样有存储长度的限制。

11.2.1 了解 Session

1. 什么是 Session

Session 译为“会话”，其本义是指有始有终的一系列动作或消息，如打电话时从拿起电话拨号到挂断电话的一系列过程可以称为一个 Session。

在计算机专业术语中，Session 是指一个终端用户与交互系统进行通信的时间间隔，通常指从注册进入系统到注销退出系统所经过的时间。因此，Session 实际上是一个特定的时间概念。

2. Session 工作原理

当启动一个 Session 会话时，会生成一个随机且唯一的 session_id，也就是 Session 的文件名，此时 session_id 存储在服务器的内存中，当关闭页面时此 id 会自动注销，重新登录此页面，会再次生成一个随机且唯一的 id。

3. Session 的功能

Session 在 Web 技术中非常重要。由于网页是一种无状态的连接程序，因此无法得知用户的浏览状态。通过 Session 则可记录用户的有关信息，以供用户再次以此身份对 Web 服务器提交要求时做确认。例如，在电子商务网站中，通过 Session 记录用户登录的信息，以及用户所购买的商品，如果没有 Session，那么用户每进入一个页面都需要登录一次用户名和密码。

另外，Session 会话适用于存储信息量比较少的情况。如果用户需要存储的信息量相对较少，并且不需要长期存储，那么使用 Session 把信息存储到服务器端比较合适。

11.2.2 创建会话

创建一个会话需要通过以下步骤：

启动会话 → 存储会话 → 读取会话 → 删除会话

1. 启动会话

使用 session_start() 函数启动 PHP 会话。

语法格式如下：

```
bool session_start(void) ;
```

注意：通常，session_start() 函数在页面开始位置调用，会话变量被存储到 $_SESSION 中。

2. 存储会话

$_SESSION 变量是个数组，开启会话之后，就可以使用 $_SESSION 变量来存取信息了。当要把信息存入 Session 的时候，可编写如下代码：

```
$_SESSION['username'] = '张三';
```

例如，判断存储用户名的 Session 会话变量是否为空，如果不为空，则将该会话变量赋给 $myvalue，代码如下：

```
<?php
$_SESSION['username'] = '张三';
if ( !empty ( $_SESSION['username']))     //判断用于存储用户名的Session会话变量是否为空
    $myvalue = $_SESSION['username'] ;    //将会话变量赋给一个变量$myvalue
?>
```

3. 读取会话

读取会话很简单，就像使用数组一样，代码如下：

```
$userName = $_SESSION['username'];
```

4. 删除会话

删除会话的方法主要有删除单个会话、删除多个会话和结束当前会话 3 种，下面分别进行介绍。

（1）删除单个会话

删除会话变量，同数组的操作一样，直接注销 $_SESSION 数组的某个元素即可。

例如，注销 $_SESSION['user'] 变量，可以使用 unset() 函数，代码如下：

```
unset( $_SESSION['username'] );
```

注意：使用 unset() 函数时，如 unset($_SESSION) 函数会将全局变量 $_SESSION 销毁，而且没有办法将其恢复，用户也不能再注册 $_SESSION 变量。如果要删除多个或全部会话，可采用下面的两种方法。

（2）删除多个会话

如果想要一次注销所有的会话变量，则可以将一个空的数组赋值给 $_SESSION，代码如下：

```
$_SESSION = array() ;
```

（3）结束当前会话

如果整个会话已经结束，首先应该注销所有的会话变量，然后使用 session_destroy() 函数清除结束当前的会话，并清空会话中的所有资源，彻底销毁 Session，代码如下：

```
session_destroy() ;
```

11.2.3 使用 Session 实现判断用户登录功能

Cookie 数据存储在客户的浏览器上，Session 数据存储在服务器上；Cookie 数据不安全，别人可以分享存放在本地的 Cookie 数据并进行 Cookie 欺骗，因此考虑到安全性应当使用 Session 数据。

实例 02　使用 Session 实现判断用户是否登录　　实例位置：光盘\Code\SL\11\02

本实例通过 Session 技术实现判断用户是否登录。具体步骤如下：

（1）创建 login.php 文件作为登录页面，该页面与实例 01 登录页面相似。关键代码如下：

实例02-1

```
//省略其余代码
<form class="box py-3 px-4 px-2-mobile" role="form" method="post"
          action="checkLogin.php">
    <div class="is-flex is-column is-justified-to-center">
        <h1 class="title is-3 mb-a has-text-centered">
            登录
        </h1>
        <div class="inputs-wrap py-3">
            <div class="control">
                <input type="text" id="username" name="username" class="input"
                       placeholder="用户名" value="" required>
            </div>
            <div class="control">
                <input type="password" id="password" name="password" class="input"
                       placeholder="密码" required>
            </div>
            <div class="control">
                <button type="submit" class="button is-submit is-primary is-outlined">
                    提交
                </button>
            </div>
        </div>
    </div>
</form>
//省略其余代码
```

运行效果如图 11.13 所示。

图 11.13　登录页面

（2）单击“提交”按钮，表单提交到 checkLogin.php 文件，并在该文件中处理登录逻辑。登录成功后，使用 session_start() 函数初始化 Session 变量，将 username 存储到 Session 中。关键代码如下：

实例02-2

```
//省略其余代码
//users表中查找输入的用户名和密码是否匹配
$sql = 'select * from users where username = :username and password = :password';
$res = $pdo->prepare($sql);
$res->bindParam(':username',$username);                    //绑定参数
$res->bindParam(':password',$password);                    //绑定参数
if($res->execute()){
    $rows = $res->fetch(PDO::FETCH_ASSOC);                 //返回一个索引为结果集列名的数组
    if($rows){
        session_start();                                   //启动Session
        $_SESSION['username'] = $rows['username'];         //Session赋值
        echo "<script>alert('恭喜您，登录成功！');window.location.href='index.php';
                </script>";
    }else{
        echo "<script>alert('用户名或密码错误，登录失败！');history.back();</script>";
        exit ();
    }
}
//省略其余代码
```

运行效果如实例 01 中图 11.8 和 11.9 所示。

（3）登录成功后，username 存储在 Session 中，可以使用 $_SESSION['username'] 获取到该值。在 index.php 文件中，关键代码如下：

实例02-3

```
<?php
    session_start();                                       //启动Session
    if(!isset($_SESSION['username'])){                     //Session取值
        echo "<script>alert('请先登录');window.location.href='login.php'</script>";
    }
?>
```

运行效果与实例 01 相同。

（4）退出登录。创建 logout.php 文件，使用 unset() 函数清除 Session。代码如下：

实例02-4

```
<?php
    session_start();                                       //启动Session
    unset( $_SESSION['username'] );                        //清除Session
    echo "<script>window.location.href='login.php'</script>";
?>
```

清除 Session 后，再次访问 index.php 文件，页面将跳转到登录页。

常见错误：使用 Session 前，一定要先使用 session_start() 函数启动 Session，否则将提示“_SESSION 不存在”。

练一练：

（1）应用 Session 可以在页面传递数据的特性，实现聊天室换肤的功能。单击不同的颜色值，更换聊天室的背景颜色。运行效果如图 11.14 所示。（光盘\Code\Try\11\03）

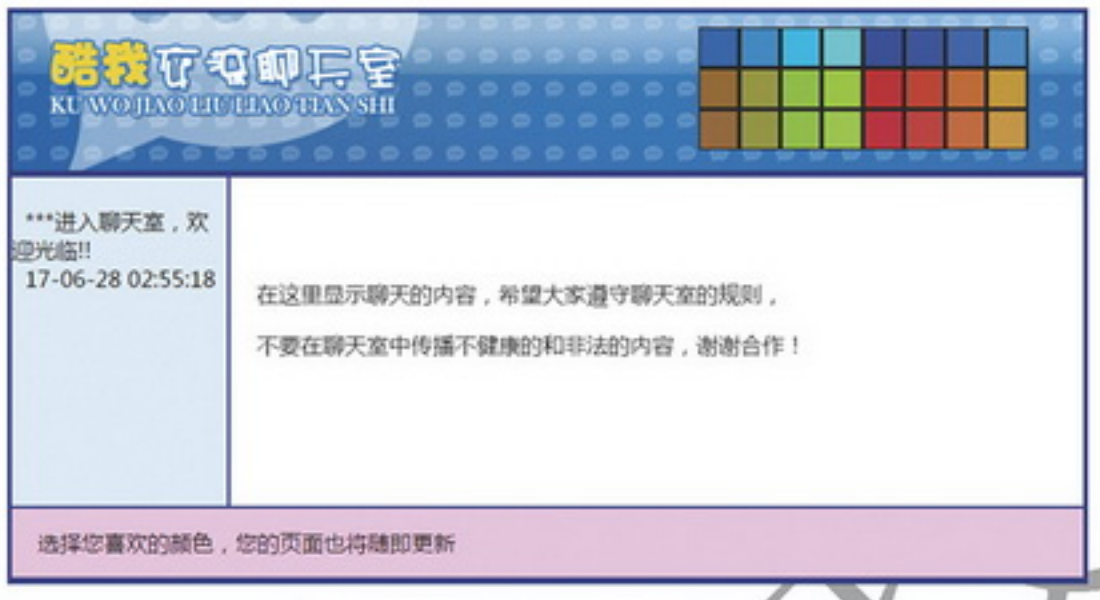


图 11.14　更换聊天室背景颜色

（2）模拟京东商城购物车，创建一个 goods 商品表和一个 order 订单表，使用 Session 实现用户在未登录的情况下，加入购物车的功能，如图 11.15 所示。（光盘\Code\Try\11\04）

图 11.15　加入购物车

11.3 Session 高级应用

11.3.1 Session 临时文件

在服务器中，如果将所有用户的 Session 都保存到临时目录中，则会降低服务器的安全性和效率，

打开服务器存储站点会非常慢。

使用 PHP 函数 session_save_path() 存储 Session 临时文件，可以缓解因临时文件的存储导致服务器效率降低和站点打开缓慢的问题。代码如下：

```
<?php
$path = './tmp/';                          //设置Session存储路径
session_save_path($path);
session_start();                           //初始化Session
$_SESSION['username'] = 'mr';
?>
```

注意：session_save_path() 函数应在 session_start() 函数之前调用。

运行结果如图 11.16 所示。

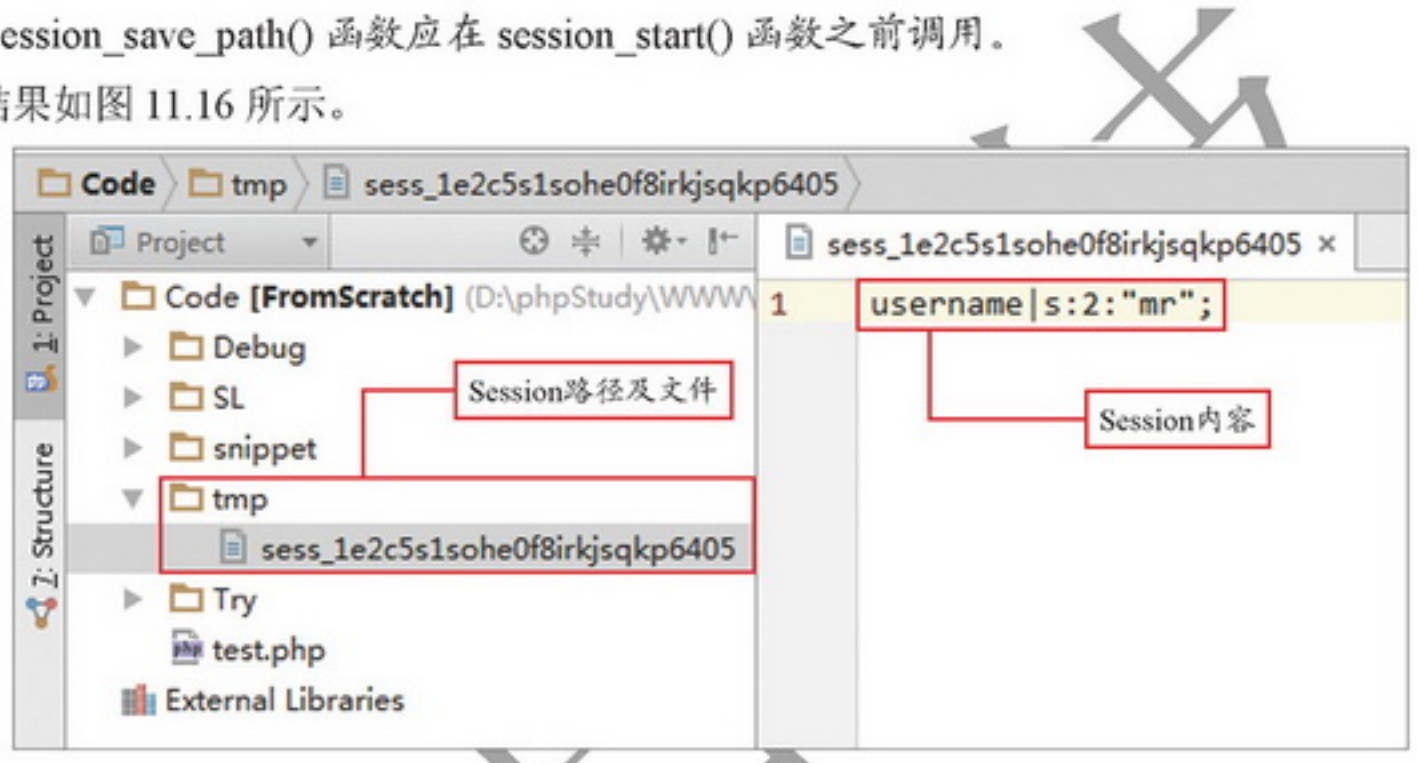


图 11.16　设置缓存路径

11.3.2 Session 缓存

Session 缓存是将网页中的内容临时存储到客户端的文件夹下，并且可以设置缓存的时间。当第一次浏览网页后，页面的部分内容在规定的时间内就被临时存储在客户端的临时文件夹中，这样在下次访问这个页面时，就可以直接读取缓存中的内容，从而提高网站的浏览效率。

使用 session_cache_limiter() 函数实现 Session 缓存，语法如下：

```
string session_cache_limiter ( [string cache_limiter])
```

参数 cache_limiter 为 public 或 private。同时 Session 缓存并不是指在服务器端而是在客户端缓存，但在服务器中没有显示。

使用 session_cache_expire() 函数设置缓存时间，语法如下：

```
int session_cache_expire ( [int new_cache_expire])
```

参数 new_cache_expire 是 Session 缓存的时间数字，单位是分。

注意：这两个 Session 缓存函数必须在 session_start() 函数之前调用，否则会出错。

下面通过例子来了解 Session 缓存过程，代码如下：

```
<?php
/* 设置缓存限制为 "private" */
session_cache_limiter('private');
$cache_limiter = session_cache_limiter();
/* 设置缓存过期时间为10分 */
session_cache_expire(10);
$cache_expire = session_cache_expire();
/* 开始会话 */
session_start();
echo "缓存限制为：". $cache_limiter."<br/>";
echo "客户端缓存时间为：". $cache_expire ."分";
?>
```

运行结果如下：

```
缓存限制为：private
客户端缓存时间为：10分
```

11.3.3 Session 数据库存储

PHP 默认采用文件的方式来保存 Session，虽然通过改变 Session 存储文件夹，使 Session 不至于因为临时文件夹被填满而造成站点瘫痪，但是可以计算一下：如果一个大型网站一天登录 1000 人，一个月登录了 30000 人，这时站点中就存在 30000 个 Session 文件，要在这 30000 个文件中查询一个 session_id 应该不是件轻松的事情，这时就可以应用 Session 数据库存储，也就是 PHP 中的 session_set_save_handler() 设置自定义会话存储函数。

session_set_save_handler() 函数语法格式如下：

```
bool session_set_save_handler ( SessionHandlerInterface $sessionhandler [, bool
                                $register_shutdown = true ] )
```

session_set_save_handler() 函数的参数说明如下：

◆ sessionhandler 实现了 SessionHandlerInterface 接口的对象，例如 SessionHandler。自 PHP 5.4 之后可以使用。

◆ register_shutdown 将函数 session_write_close() 注册为 register_shutdown_function() 函数。

◆ 返回值：成功时返回 true，或者在失败时返回 false。

实例 03　使用 session_set_save_handler() 函数将 Session 存入数据库　　实例位置：光盘\Code\SL\11\03

创建一个 Session 存储类，使用 session_set_save_handler() 函数实现 Session 数据库存储。具体步骤如下：

（1）在 database11 数据库中，创建一个 sessions 表，包含 3 个字段：id（主键）、data（Session 存储内容）、last_accessed（最后访问时间，根据当前时间戳更新），如图 11.17 所示。

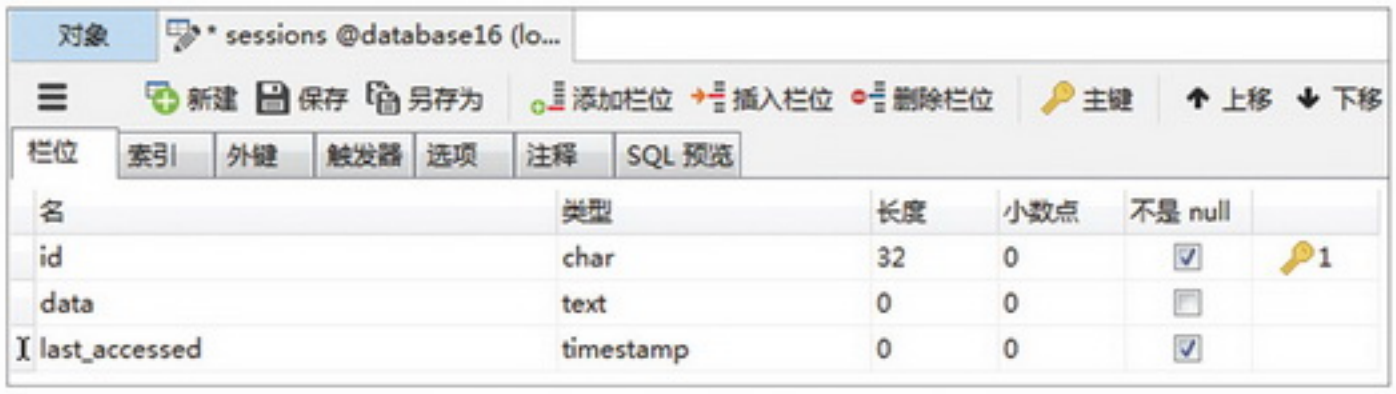


图 11.17　sessions 表数据结构

（2）创建 Session 类，该类中包含 session_set_save_handler() 函数参数。具体代码如下：

实例03-1

```php
<?php

class mysqlSession implements SessionHandlerInterface{
    private $pdo     = null;            //数据库链接句柄
    private $dbtable = 'sessions';
    /**
     * session_start() 开始会话后第一个调用的函数，类似于构造函数的作用
     * @param string $save_path 默认的保存路径
     * @param string $session_name 默认的参数名（PHPSESSID）
     * @return bool
     */
    public function open($save_path, $session_name)
    {
        $dsn = DB_TYPE.":host=" .DB_HOST.";dbname=".DB_NAME;
        try {
            $this->pdo = new PDO($dsn, DB_USER, DB_PWD);
            return true;
        } catch (PDOException $e) {
            return false;
        }
    }

    /**
     * 类似于析构函数，在write()之后调用或者session_write_close()函数之后调用
     * @return bool
     */
    public function close()
    {
        $this->pdo = null;
        return true;
    }

    /**
     * 读取session信息
     * @param string $sessionId 通过该ID（客户端的PHPSESSID）唯一确定对应的session数据
```

```
36          * @return session信息或者空串（没有存储session信息）
37          */
38         public function read($sessionId)
39         {
40             try {
41                 $sql = 'SELECT * FROM '. $this->dbtable.' WHERE id = ? LIMIT 1';
42                 $res = $this->pdo->prepare($sql);
43                 $res->execute(array($sessionId));
44                 if ($ret = $res->fetch(PDO::FETCH_ASSOC)) {
45                     return $ret['data'];
46                 } else {
47                     return '';
48                 }
49             } catch (PDOException $e) {
50                 return '';
51             }
52         }
53
54         /**
55          * 写入或修改session数据
56          * @param string $sessionId 要写入数据的session对应的id（PHPSESSID）
57          * @param string $sessionData 要写入的是数据，已经序列化过的
58          * @return bool
59          */
60         public function write($sessionId, $sessionData)
61         {
62             try {
63                 $sql = 'REPLACE INTO '. $this->dbtable. '(id, data) VALUES (?, ?)';
64                 $res = $this->pdo->prepare($sql);
65                 $res->execute(array($sessionId, $sessionData));
66                 return true;
67             } catch (PDOException $e) {
68                 return false;
69             }
70         }
71
72         /**
73          * 主动销毁session会话
74          * @param string $sessionId 要销毁的会话的唯一id
75          * @return bool
76          */
77         public function destroy($sessionId)
78         {
79             try {
80                 $sql = 'DELETE FROM '. $this->dbtable. 'WHERE id =  ?';
```

```
            $res = $this->pdo->prepare($sql);
            $res->execute(array($sessionId));
            return true;
        } catch (PDOException $e) {
            return false;
        }
    }

    /**
     * 清理会话中的过期数据
     * @param int $maxlifetime 有效期（自动读取php.ini 中的 session.gc_maxlifetime 配置项）
     * @return bool
     */
    public function gc($maxlifetime)
    {
        try {
            $sql = 'DELETE FROM '. $this->dbtable.
                    'WHERE DATE_ADD (last_accessed,INTERVAL ? SECOND) < NOW()';
            $res = $this->pdo->prepare($sql);
            $res->execute(array($maxlifetime));
            return true;
        } catch (PDOException $e) {
            return false;
        }
    }
}
```

上述代码中，write() 函数使用了 REPLACE INTO 语句，REPLACE INTO 语句和 INSERT INTO 语句功能类似，区别是：REPLACE INTO 语句首先尝试将数据插入表中，如果发现表中已经有此行数据（根据主键或者唯一索引判断）则先删除此行数据，然后插入新的数据；否则，直接插入新数据。要注意的是：插入数据的表必须有主键或者是唯一索引。否则，INSERT INTO 语句会直接插入数据，导致表中出现重复的数据。

（3）创建 index.php 文件，使用 Session 类实现将 Session 存入数据库。index.php 文件具体代码如下：

实例03-2

```
<?php
    require_once('./config.php');                    //引入配置文件
    require_once('./session.php');                   //引入mysqlSession类文件
    $sess = new mysqlSession();                      //实例化mysqlSession类
    session_set_save_handler($sess);                 //调用session_set_save_handler()方法
    session_start();                                 //开启session
    //设置Session
    $_SESSION['username'] = 'mr';
```

```
09      $_SESSION['password'] = md5('123456');
10      echo "<pre>";
11      var_dump($_SESSION);
12  ?>
```

上述代码中，首先引入 config.php 数据库配置文件、session.php 类文件。然后实例化 mysqlSession 类，最后调用 session_set_save_handler() 方法，并传入 $sess 对象。当使用 session_start() 函数开启 Session 后，mysqlSession 类中方法的执行顺序如下：

```
open()→read()→write()→close()
```

使用谷歌浏览器运行 index.php 文件，结果如图 11.18 所示。

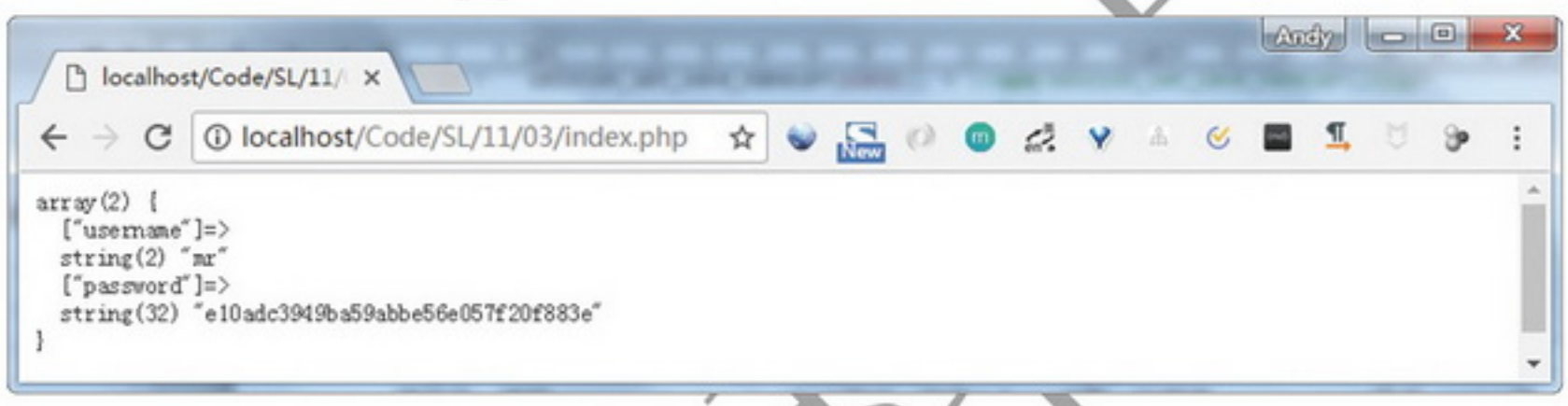


图 11.18　index.php 文件运行结果

在谷歌浏览器下查看 Session ID，具体步骤为：首先鼠标右键单击浏览器页面，在弹出的如图 11.2 所示对话框中单击“检查”选项；然后在弹出的对话框中单击 Network 选项，按下 <F5> 键刷新页面后，选择 index.php → Headers；最后在下方找到 Request Hearders 选项，即可查看到 Session ID 的内容，如图 11.19 所示。

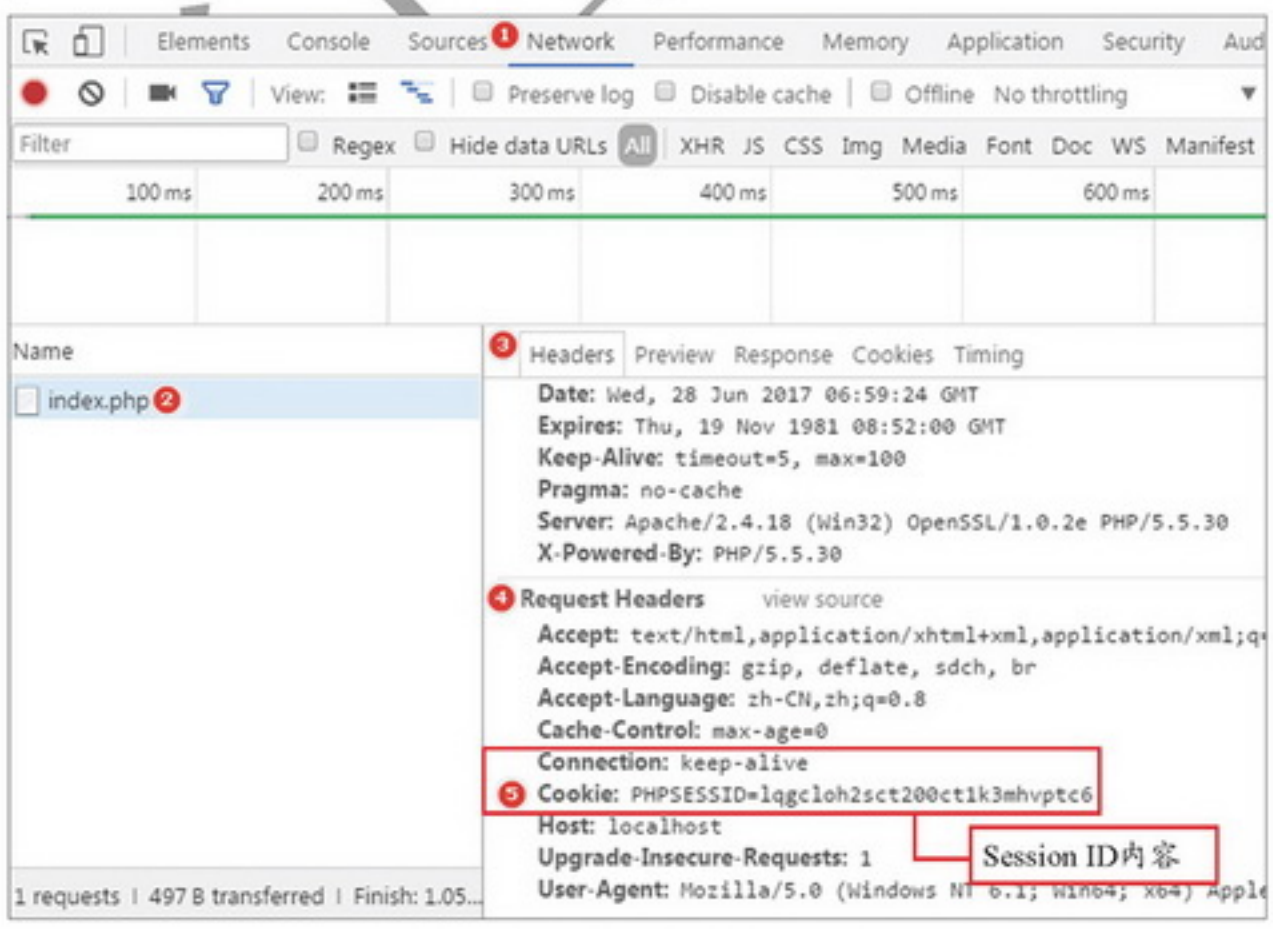


图 11.19　查看 Session ID

查看 database11 数据库的 sessions 表的数据内容如图 11.20 所示。

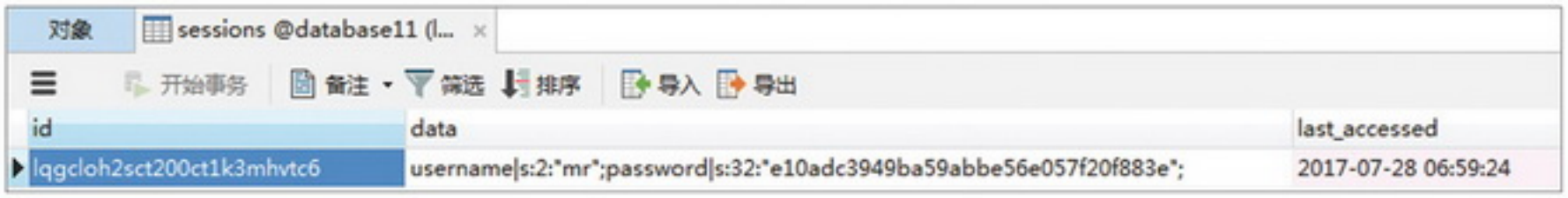


图 11.20　sessions 表数据

图 11.19 中的 PHPSESSID 和图 11.20 中 sessions 表的 id 相同，说明已经成功地将 Session 数据存储到 MySQL 数据表中。当使用 session_destroy() 函数时，会调用 mysqlSession 类中的 destroy() 方法，清除 sessions 表中的数据。

11.4 难点解答

11.4.1 Cookie 和 Session 的区别

◆ 存储位置

Session 存储在服务器上，可以通过 php.ini 文件配置 Session 相关设置。

Cookie 存储在客户端上，有两种情况：

（1）持久性 Cookie，设置了 Cookie 的时间，以文件方式存储在硬盘上。

（2）会话 Cookie，没有设置 Cookie 时间，Cookie 的生命周期在关闭浏览器前就消失，一般不会保存在硬盘，而是保存在内存上。

◆ 容量限制

Cookie 有大小和数量的限制，每个站点最多 20 个，最大 4KB；Session 没有限制，但设置太多当访问用户很多时会很占服务器内存。

◆ 安全性

因为保存在客户端，Cookie 可能会泄露用户隐私，带来其他安全问题。用户隐私等数据存放在 Session 中比较稳妥；其他数据可以存放到 Cookie 中。

11.4.2 Cookie 和 Session 的关系

当程序需要为某个客户端的请求创建一个 Session 时，服务器首先检查这个客户端的请求里是否已包含了一个 Session 标识（称为 Session ID），如果已包含，则说明之前已经为此客户端创建过 Session，服务器就按照 Session ID 把这个 Session 检索出来。如果客户端请求不包含 Session ID，则为此客户端创建一个 Session 并且生成一个与此 Session 相关联的 Session ID，Session ID 的值应该是一个既不会重复，又不容易被找到规律来仿造的字符串，这个 Session ID 将被在本次响应中返回给客户端保存。保存这个 Session ID 的方式可以采用 Cookie，这样在交互过程中浏览器可以自动地按照规则把这个标识发送给服务器。

11.5 小　结

本章主要介绍了 Cookie 和 Session 的基础知识，包括它们的创建、读取以及删除等，并且重点介绍了 Session 的一些高级应用。通过完整的实例，可以使读者加深对 Cookie 和 Session 的理解及运用。希望通过本章的学习，读者能够了解 Cookie 和 Session 的关系和区别，以及它们各自的应用场景。

11.6 动手纠错

1．在未登录情况下，运行“光盘\Code\Debug\11\01”文件夹下的 index.php 文件，出现“Undefined index: username”的错误提示，如图 11.21 所示，请根据注释改正程序。

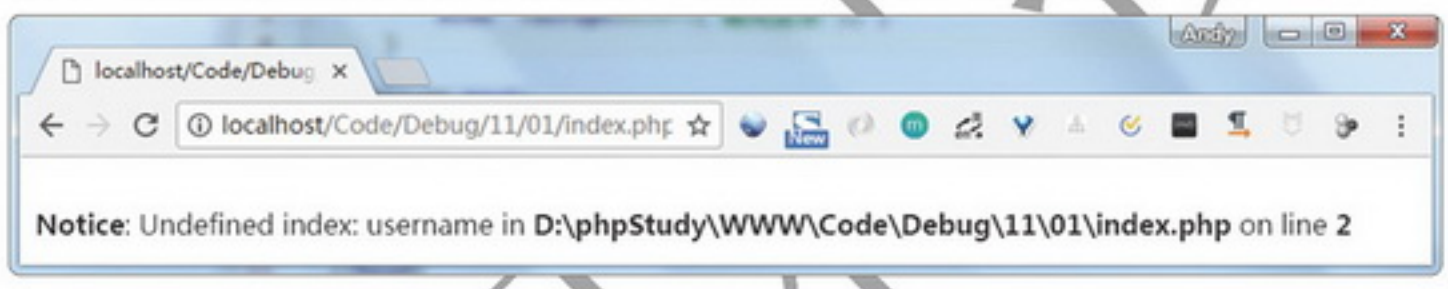


图 11.21　没有定义 username 索引错误提示

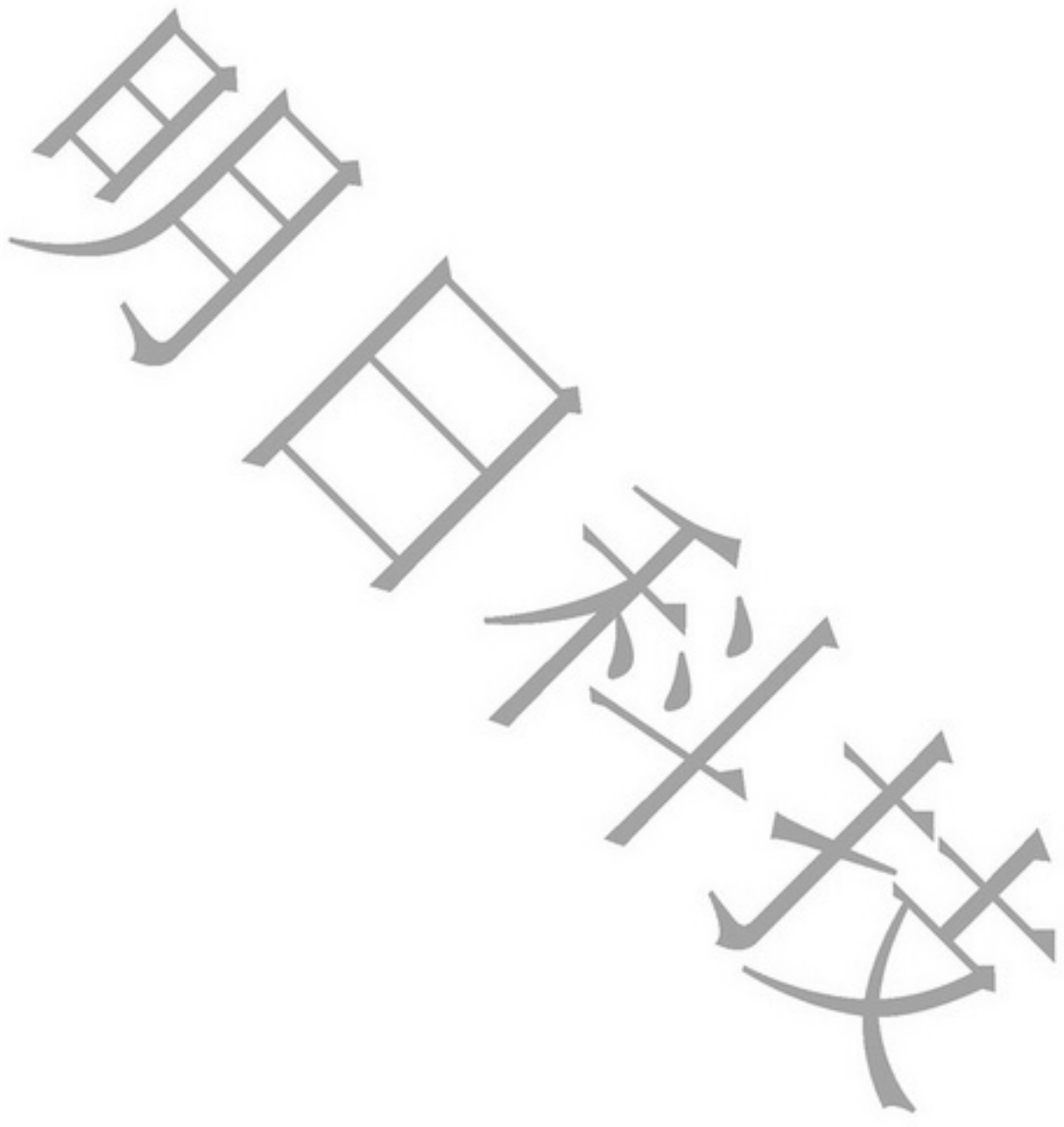
2．运行“光盘\Code\Debug\11\02”文件夹下的 shopInfo.html 文件，单击“加入购物车”按钮，出现“Undefined variable: _SESSION”的错误提示，如图 11.22 所示。请根据注释改正程序。

图 11.22　没有开启 Session 错误提示

3．运行“光盘\Code\Debug\11\03”文件夹下的 index.php 文件，输出的 Session 数组为空值，请根据注释改正程序。

4．运行“光盘\Code\Debug\11\04”文件夹下的 index.php 文件，没有按照预期将 Session 文件存储在 tmp 临时文件夹中，请根据注释改正程序。

5．运行“光盘\Code\Debug\11\05”文件夹下的 index.php 文件，使用数据库存储 Sesssion，调用 mysql Session 类中的 destroy() 方法删除数据无效，请根据注释改正程序。



第 12 章

图形图像处理技术

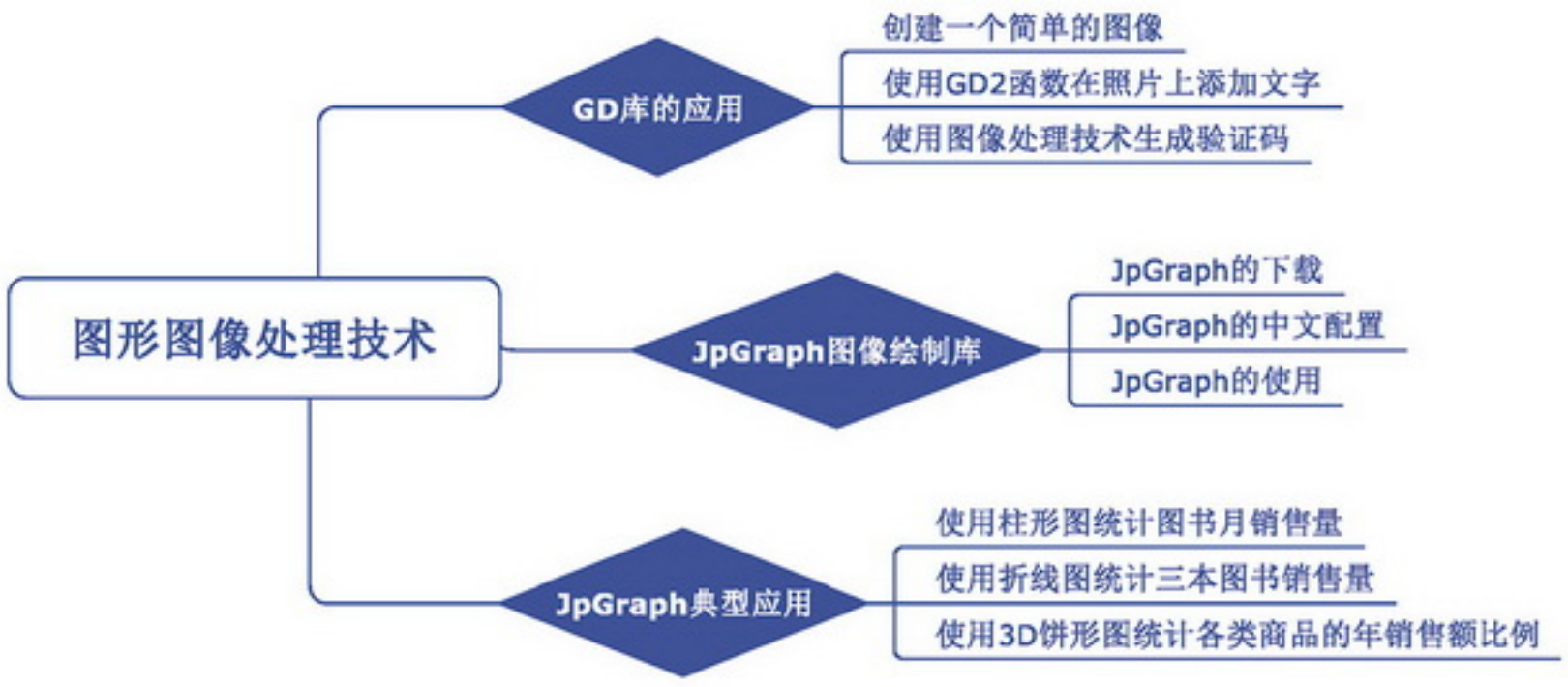
（视频讲解：46 分）

本章概览

由于 GD 库的支持，不仅使得 PHP 的图像处理功能非常强大，而且便捷易用。另外，PHP 图形化类库——JpGraph 也是一款非常强大、好用的图形处理工具，可以绘制各种统计图和曲线图，也可以自定义设置颜色和字体等元素。

图像处理技术中的经典应用是绘制饼形图、柱形图和折线图，这是对数据进行图形化分析的最佳方法。本章将分别对 GD2 函数及 JpGraph 类库进行详细讲解。

知识框架



12.1 在 PHP 中加载 GD 库

GD 库在 PHP 5 中是默认安装的，但要激活 GD 库，必须先修改 php.ini 文件。将该文件中的“;extension=php_gd2.dll”选项前的分号“;”删除（phpStudy 已经默认开启），保存修改后的文件并重新启动 Apache 服务器即可生效。

在成功加载 GD2 函数库后，可以通过 phpinfo() 函数来获取 GD2 函数库的安装信息，验证 GD 库是否安装成功。在浏览器的地址栏中输入“localhost/phpinfo.php”并按下 <Enter> 键，在打开的页面中检索到如图 12.1 所示的 GD 库的安装信息，即说明 GD 库安装成功。

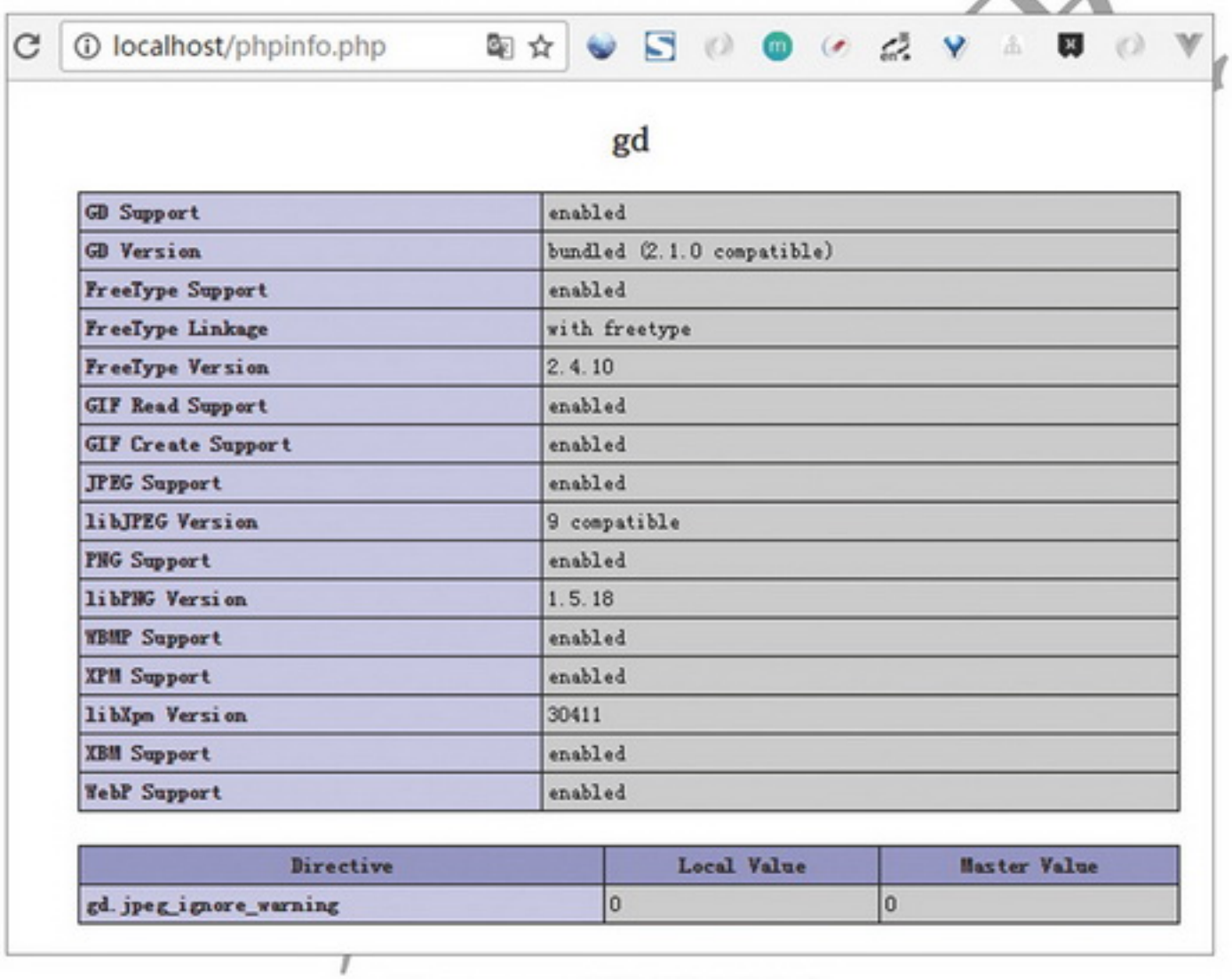


图 12.1　GD2 函数库的安装信息

12.2 GD 库的应用

12.2.1 创建一个简单的图像

使用 GD2 函数库可以实现各种图形图像的处理。创建画布是使用 GD2 函数库来创建图像的第一步，无论创建什么样的图像，首先都需要创建一个画布，其他操作都将在这个画布上完成。在 GD2 函

数库中创建画布，可以通过 imagecreate() 函数实现。

例如，使用 imagecreate() 函数创建一个宽 200 像素、高 60 像素的画布，并且设置画布背景颜色 RGB 值为（225，66，159），最后输出一个 png 格式的图像。代码如下：

```
01  <?php
02  $im = imagecreate(200,60);                          //建立一幅200×60的图像
03  $bg = imagecolorallocate($im,225,66,159);           //设置背景颜色
04  header("Content-type:image/png");                   //输出图像
05  imagepng($im);                                      //生成png格式的图像
06  ?>
```

运行效果如图 12.2 所示。

图 12.2　生成画布

12.2.2 使用 GD2 函数在照片上添加文字

GD2 函数库中的 imageTTFText() 函数可以实现用 TrueType 字体向图像写入文本的功能，语法格式如下：

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

imageTTFText() 函数的参数说明如表 12.1 所示。

表 12.1　imageTTFText() 函数的参数说明

参　数	说　明
image	由图像创建函数（例如 imagecreatetruecolor()）返回的图像资源
size	字体的尺寸。根据 GD 库的版本，为像素尺寸（GD1）或点（磅）尺寸（GD2）
angle	角度制表示的角度，0 度为从左向右读的文本。更高数值表示逆时针旋转。例如 90 度表示从下向上读的文本
x	由 x，y 所表示的坐标定义了第一个字符的基本点（大概是字符的左下角）。这和 imagestring() 不同，其 x，y 定义了第一个字符的左上角。例如 "top left" 为 0，0
y	y 坐标。它设定了字体基线的位置，不是字符的最底端
color	颜色索引。使用负的颜色索引值具有关闭防锯齿的效果
fontfile	是想要使用的 TrueType 字体的路径
text	UTF-8 编码格式的文本字符串

实例 01　在明日学院幻灯片背景图上添加文字

实例位置：光盘\Code\SL\12\01

使用 GD2 函数在照片上添加文字的具体步骤如下：

（1）使用 imagecreatefromjpeg() 函数载入图片。

（2）使用 imagecolorallocate() 函数设置字体颜色。

（3）使用 imageTTFText() 函数向图片中写入文本。

（4）使用 imagejpeg() 函数创建 jpg 图片。

（5）使用 imagedestroy() 函数销毁图像，释放内存空间。

使用 imageTTFText() 函数将文字“明日学院”输出到图像中。代码如下：

实例01-1

```
<?php
header("content-type:image/jpeg");                          //定义输出为图像类型
$path = "mingri.jpg";                                       //图片路径
$im    = imagecreatefromjpeg($path);                        //载入图片
$textcolor = imagecolorallocate($im,255,255,255);           //设置字体颜色，值为RGB颜色值
$fnt = "c:/windows/fonts/simfang.ttf";                      //定义字体
$str = '明日学院';
imageTTFText($im,100,0,50,200,$textcolor,$fnt,$str);        //写TTF文字到图中
imagejpeg($im);                                             //创建jpg图片
imagedestroy($im);                                          //结束图形，释放内存空间
?>
```

运行前、后的效果如图 12.3 和图 12.4 所示。

图 12.3　运行前效果图

图 12.4　运行后效果图

常见错误：如图片不显示或中文汉字乱码，请先检查 index.php 文件的编码格式是否为 UTF-8；然后检查定义的字体是否支持中文，在 C:\Windows\Fonts 文件夹下，查找支持中文的字体，如“黑体”为“simhei.ttf”，“仿宋”为“simfang.ttf”。

练一练：

（1）试着生成一张图片的缩略图，缩放比例为 0.2，缩放前后对比如图 12.5 所示。（光盘\Code\Try\12\01）

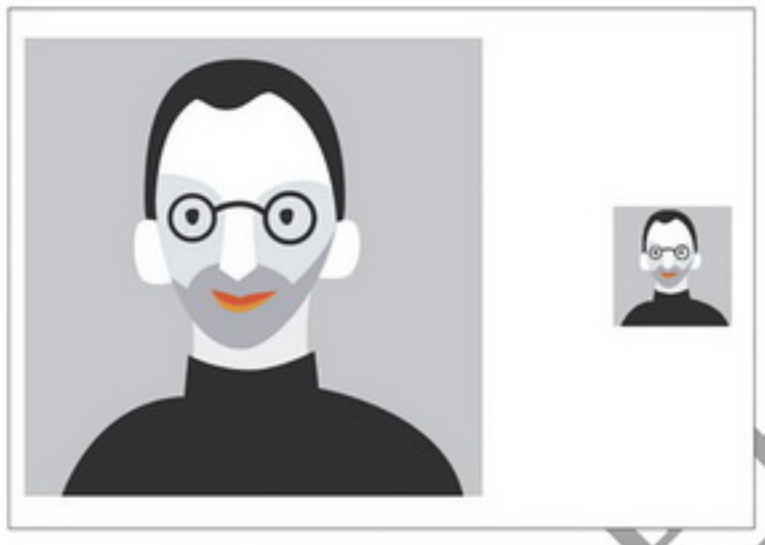


图 12.5　图片缩放前后对比

（2）试着制作一个“告别图片生成器”，输入英文姓名，单击“确定”按钮后，即可生成带有输入内容的图片。（光盘\Code\Try\12\02）

12.2.3 使用图像处理技术生成验证码

验证码功能的实现方法很多，有数字验证码、图形验证码和文字验证码等。在本节中将介绍一种使用图像处理技术生成的验证码。

实例 02	使用 GD2 函数生成验证码	实例位置：光盘\Code\SL\12\02

图像处理技术生成验证码常用在用户登录过程中，下面在登录页面实现该过程。程序的开发步骤如下：

（1）生成验证码。创建 verify.php 文件，用于生成验证码。在该文件中使用 GD2 函数创建一个 4 位的验证码，并且将生成的验证码保存在 Session 变量中，代码如下：

实例02-1

```
<?php
    session_start();                                    //初始化Session变量
    header("content-type:image/png");                   //设置创建图像的格式
    $image_width  = 76;                                 //设置图像宽度
    $image_height = 40;                                 //设置图像高度
    $lenth        = 4;                                  //字符串长度
    //除去0、1、o、l容易混淆字符
    $str = "23456789abcdefghijkmnpqrstuvwxyzABCDEFGHIJKLMNPQRSTUVW";
    $code = '';
    for ($i=0; $i<$lenth; $i++){
        $code.= $str[mt_rand(0, strlen($str)-1)];//从字符串中随机选择
    }
    $_SESSION['verify'] = $code;                        //将获取的随机数验证码写入到Session变量中
```

```
14      $image = imagecreate($image_width,$image_height); //创建一个画布
15      imagecolorallocate($image,255,255,255);           //设置画布的颜色
16      for($i=0;$i<strlen($_SESSION['verify']);$i++){    //循环读取Session变量中的验证码
17          $font  = mt_rand(3,5);                        //设置随机的字体
18          $x     = mt_rand(1,8)+$image_width*$i/4;      //设置随机字符所在位置的X坐标
19          $y     = mt_rand(8,$image_height/4);          //设置随机字符所在位置的Y坐标
20          //设置字符的颜色
21          $color = imagecolorallocate($image,mt_rand(0,100),mt_rand(0,150),mt_rand(0,200));
22          imagestring($image,$font,$x,$y,$_SESSION['verify'][$i],$color); //水平输出字符
23      }
24
25      //绘制干扰点元素
26      $pixel=30;
27      $black = imagecolorallocate($image, 0, 0, 0);
28      for($i=0;$i<$pixel;$i++){
29      imagesetpixel($image, mt_rand(0, $image_width-1),mt_rand(0, $image_height-1),$black);
30      }
31      imagepng($image);                                 //生成png格式的图像
32      imagedestroy($image);                             //释放图像资源
33  ?>
```

在上面的代码中，对验证码进行输出时，每个字符的位置、颜色和字体都是通过随机数来获取的，并且使用 imagesetpixel() 函数设置干扰点，不仅可以在浏览器中生成各式各样的验证码，还可以防止恶意用户攻击网站系统。此外，为了后续检测验证码，使用 session_start() 函数开启 Session，并将生成的验证码存入 Session。运行结果如图 12.6 所示。

图 12.6　生成的验证码

（2）显示验证码。创建 login.php 文件，该文件包含用户登录的表单，并调用 checks.php 文件，在表单页中输出验证码图像的内容。代码如下：

实例02-2

```
01  <!DOCTYPE html>
02  <html lang="en" class="is-centered is-bold">
03  <head>
04      <meta charset="UTF-8">
05      <title>零基础</title>
06      <link href="css/main.css" rel="stylesheet">
07  </head>
```

```
<body>
<section style="background: transparent">
    <form class="box py-3 px-4 px-2-mobile" role="form" method="post"
          action="checkLogin.php">
        <div class="is-flex is-column is-justified-to-center">
            <h1 class="title is-3 mb-a has-text-centered">
                登录
            </h1>
            <div class="inputs-wrap py-3">
                <div class="control">
                    <input type="text" id="username" name="username" class="input"
                           placeholder="用户名" value="" required>
                </div>
                <div class="control">
                    <input type="password" id="password" name="password" class="input"
                           placeholder="密码" required>
                </div>
                <div class="control">
                    <input type="text" id="verify" name="verify" class="input"
                           style="width: 70%" placeholder="验证码" required>
                    <a href="javascript:;">
                        <!-- 显示验证码、点击重新生成验证码 -->
                        <img src="verify.php"
                             onClick="this.src=this.src+'?'+Math.random()">
                    </a>
                </div>
                <div class="control">
                    <button type="submit" class="button is-submit is-primary is-outlined">
                        提交
                    </button>
                </div>
            </div>
        </div>
    </form>
</section>
</body>
</html>
```

上述代码中，<img> 标签中有如下代码：

```
<img src="verify.php" onclick="this.src=this.src">
```

src 属性值为 verify.php，由于 verify.php 使用：

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

即生成内容为图片格式。所以，登录页面会显示 verify 生成的验证码图片。运行结果如图 12.7 所示。

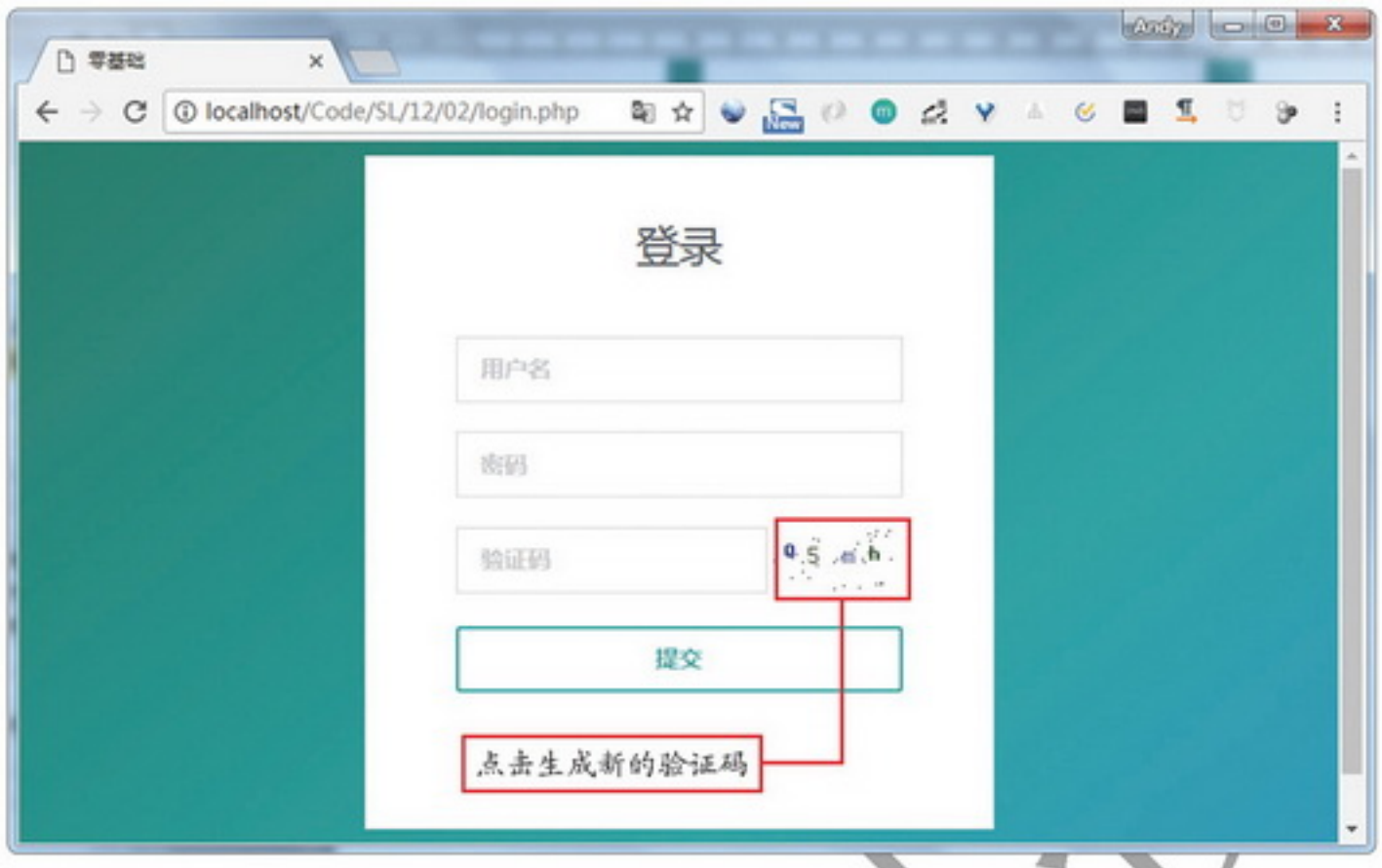


图 12.7　登录页面显示验证码

（3）检测验证码。在登录页面单击“提交”按钮后，会将表单提交到 checkLogin.php 页面。创建 checkLogin.php 文件，用于检测用户提交的验证码是否正确。由于在 verify.php 文件中，已经将生成的验证码存入 Session 中，所以只需要判断用户输入的验证码和 Session 值是否相等即可。checkLogin.php 文件代码如下：

实例02-3

```
<?php
session_start();                                          //初始化Session
if(isset($_POST["username"]) && isset($_POST["password"])){
    $checks = $_POST["verify"];                           //获取验证码文本框的值
    if($checks == ""){                                    //如果验证码的值为空，则弹出提示信息
        echo "<script> alert('验证码不能为空');
              window.location.href='login.php';</script>";
    }
    //如果用户输入验证码的值与随机生成的验证码的值相等，则弹出登录成功提示
    if($checks == $_SESSION['verify']){
        /**省略用户名密码验证过程**/
        echo "<script> alert('用户登录成功!');</script>";
    }else{                                                //否则弹出验证码不正确的提示信息
        echo "<script> alert('您输入的验证码不正确!');
              window.location.href='login.php';</script>";
    }
}
?>
```

在登录页面中，输入用户名和密码，在“验证码”文本框中输入验证码信息，单击“提交”按钮，对验证码的值进行判断，注意区分字母大小写。验证码正确运行效果如图 12.8 所示，验证码错误运行效果如图 12.9 所示。

图 12.8　验证码正确运行效果图

12.9　验证码错误运行效果图

练一练：

（1）修改实例 02，生成数字图像验证码。（光盘\Code\Try\12\03）

（2）修改实例 02，生成中文图像验证码，如图 12.10 所示。（光盘\Code\Try\12\04）

图 12.10　生成中文验证码

12.3　JpGraph 图像绘制库

JpGraph 是一种面向对象的图像绘制库，是基于 GD2 函数库，并对其中的函数进行封装，可以直接使用生成统计图的函数。JpGraph 可以生成 X-Y 坐标图、X-Y-Y 坐标图、柱形图、饼图、3D 饼图等统计图，并会自动生成坐标轴、坐标轴刻度、图例等信息，帮助我们快速生成所需样式的统计图。

JpGraph 这个强大的绘图组件能根据用户的需要绘制任意图形。用户只需要提供数据，就能自动调用绘图函数，把要处理的数据输入，并自动绘制。JpGraph 提供了多种方法创建各种统计图，包括折线图、柱形图和饼形图等。JpGraph 是一个完全使用 PHP 语言编写的类库，并可以应用在任何 PHP 环境中。

12.3.1 JpGraph 的下载

JpGraph 可以从其官方网站（网址为：http://JpGraph.net/download）下载。注意，JpGraph 支持 PHP 5 和 PHP 7，目前最新的版本是 JpGraph 4.0.2。

JpGraph 的安装方法非常简单，文件下载后，安装步骤如下：

（1）将下载的压缩包解压。解压后，将 jpgraph-4.0.2 文件夹下的 src 文件夹复制到项目文件夹下。这里复制到 D:\phpStudy\WWW\Code\SL\12\ 文件夹下。

（2）将 src 文件夹重命名为 jpgraph。目录结构如图 12.11 所示。

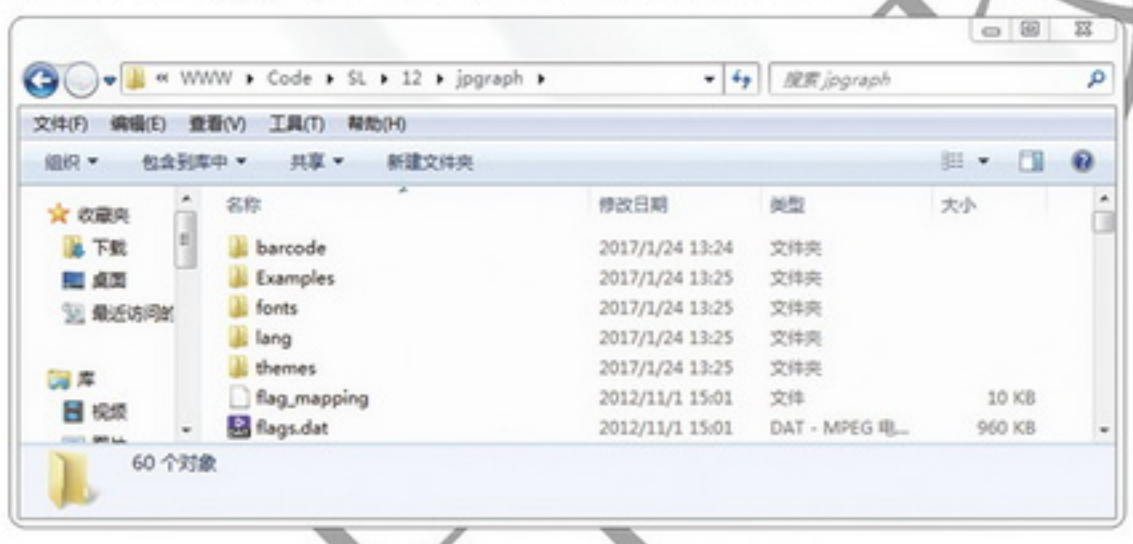


图 12.11　jpgraph 文件目录结构

12.3.2 JpGraph 的中文配置

JpGraph 生成的图片包含中文时，会出现中文乱码现象。要解决此问题，需对下面 3 个文件进行修改。

◆ 修改 jpgraph_ttf.inc.php。路径为：D:\phpStudy\WWW\Code\SL\12\jpgraph\jpgraph_ttf.inc.php。在 jpgraph_ttf.inc.php 文件中，将代码：

```
define('CHINESE_TTF_FONT','bkai00mp.ttf');
```

修改为：

```
define('CHINESE_TTF_FONT','simhei.ttf');
```

其中 simhei.ttf 是中文黑体，更多中文字体可以在 C:\Windows\Fonts\ 文件夹下选择。

◆ 修改 jpgraph_legend.inc.php，路径：D:\phpStudy\WWW\Code\SL\12\jpgraph\jpgraph_legend.inc.php。在 jpgraph_legend.inc.php 文件中，将代码：

```
public $font_family=FF_DEFAULT,$font_style=FS_NORMAL,$font_size=8;
```

修改为：

```
public $font_family=FF_CHINESE,$font_style=FS_NORMAL,$font_size=8;
```

◆ 修改 jpgraph.php，路径：D:\phpStudy\WWW\Code\SL\12\jpgraph\jpgraph.php。将 jpgraph.php 文件中：

```
public $font_family=FF_DEFAULT,$font_style=FS_NORMAL,$font_size=8,$label_angle=0;
```

修改为：

```
public $font_family=FF_CHINESE,$font_style=FS_NORMAL,$font_size=8,$label_angle=0;
```

12.3.3 JpGraph 的使用

完成 12.3.2 小节的中文配置后，本节以基本的折线图为例，讲解如何使用 JpGraph，以及如何显示中文字体。生成折线图步骤如下：

（1）引入类文件。首先引入 jpgraph.php 文件，由于要画折线图，接下来引入 jpgraph_line.php 折线图类文件。

（2）创建 Graph 类，设置相关属性，包括 X 轴、Y 轴坐标刻度，折线图标题字体、标题、X 轴数据等。

（3）创建 LinePlot 坐标类，并导入 Y 轴数据。

（4）坐标类注入图表类。

（5）显示图片。

以明日学院小班课报名人数为例，生成折线图。在折线图中，X 轴显示月份、Y 轴显示人数，并设置折线为蓝色。具体代码如下：

```
<?php
    require_once ('jpgraph/jpgraph.php');                          //必须要引用的文件
    require_once ('jpgraph/jpgraph_line.php');                     //包含曲线图文件
    //创建Graph类，650为宽度，350长度
    $graph = new Graph(650,350);
    //设置刻度类型，X轴刻度可作为文本标注的直线刻度，Y轴为直线刻度
    $graph->SetScale('textlin');
    $graph->title->SetFont(FF_CHINESE);                            //设置字体
    $graph->title->Set('明日学院小班课报名人数');                    //设置标题
    //设置X轴数据
    $graph->xaxis->SetTickLabels(array('1月','2月','3月','4月','5月','6月','7月',
                                       '8月','9月'));
    $ydata = array(220,430,580,420,330,220,440,340,230);            //Y轴数据，以数组形式赋值
    $lineplot=new LinePlot($ydata);                                //创建坐标类，将Y轴数据注入
    $lineplot->SetColor('blue');                                   //Y轴连线设定为蓝色
    $graph->Add($lineplot);                                        //坐标类注入图表类
    $graph->Stroke();                                              //显示图表
?>
```

运行效果如图 12.12 所示。

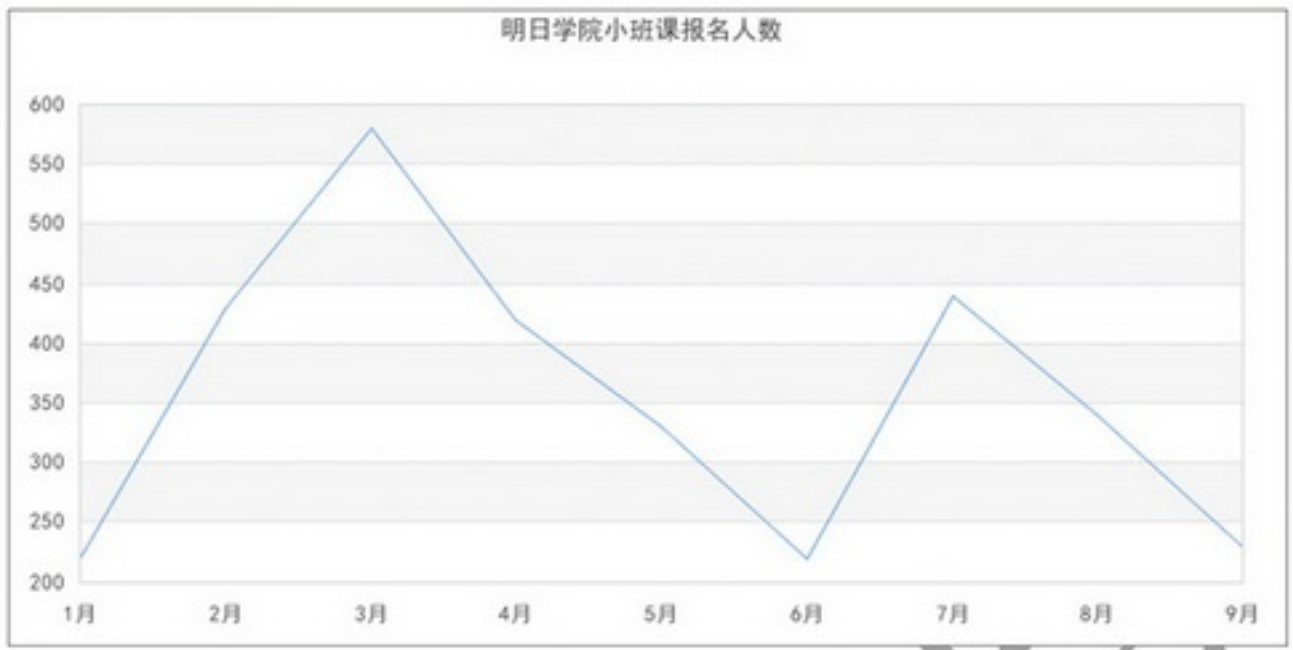


图 12.12　报名人数折线图

注意：在使用“$graph->title->Set()”设置标题前，如果标题为中文，需要先使用“$graph->title->SetFont（FF_CHINESE）”设置字体。

12.4 JpGraph 典型应用

网页中如果没有丰富多彩的图形图像总是缺少生气，漂亮的图形图像能让整个网页看起来更富有吸引力，使许多文字难以表达的思想一目了然，并且可以清晰地表达出数据之间的关系。下面对图形图像处理的各种技术进行讲解。

12.4.1 使用柱形图统计图书月销售量

柱形图的使用在 Web 网站中非常广泛，它可以直观地显示数据信息，使数据对比和变化趋势一目了然，从而可以更加准确、直观地表达信息和观点。

实例 03	使用柱形图统计图书月销售情况	实例位置：光盘\Code\SL\12\03

本实例将使用 JpGraph 类库实现柱形图统计图书月销售情况。创建柱形分析图的详细步骤如下：

（1）使用 require_once 语句引用 jpgraph.php 文件。

（2）采用柱形图进行统计分析，需要创建 BarPlot 对象，BarPlot 类在 jpgraph_bar.php 中定义，需要使用 require_once 语句引用该文件。

（3）创建 Graph 对象，生成一个 850×600 像素大小的画布，设置 X 轴、Y 轴刻度类型，设置 X 轴、Y 轴数据。

（4）创建一个矩形的对象 BarPlot，设置其柱形图的颜色、柱体间距。

（5）将绘制的柱形图添加到画布中。

（6）添加标题名称。

（7）输出图像。

本实例的完整代码如下：

实例03-1

```
<?php
require_once ('../jpgraph/jpgraph.php');                    //必须要引用的文件
require_once ('../jpgraph/jpgraph_bar.php');                //包含柱状图文件
$graph = new Graph(850,600,'auto');                         //设置画布大小
//设置刻度类型，X轴刻度可作为文本标注的直线刻度，Y轴为直线刻度
$graph->SetScale("textlin");
//设置X轴数据
$graph->xaxis->SetTickLabels(array('1月','2月','3月','4月','5月','6月','7月','8月',
                                   '9月','10月','11月','12月'));
//Y轴数据以数组形式赋值
$datay  = array(220,430,580,420,330,220,440,340,230,432,562,523);
$b1plot = new BarPlot($datay);                              //创建柱状坐标类，将y轴数据注入
$graph->Add($b1plot);                                       //柱状坐标类注入图表
$b1plot->SetColor("white");                                 //设置柱状边框颜色
$b1plot->SetFillGradient("#4B0082","white",GRAD_LEFT_REFLECTION); //设置柱体颜色
$b1plot->SetWidth(35);                                      //设置柱状间距
$graph->title->SetFont(FF_CHINESE);
$graph->title->Set("2016年PHP从入门到精通销售情况");
$graph->Stroke();                                           //显示图表
?>
```

本实例的运行结果如图 12.13 所示。

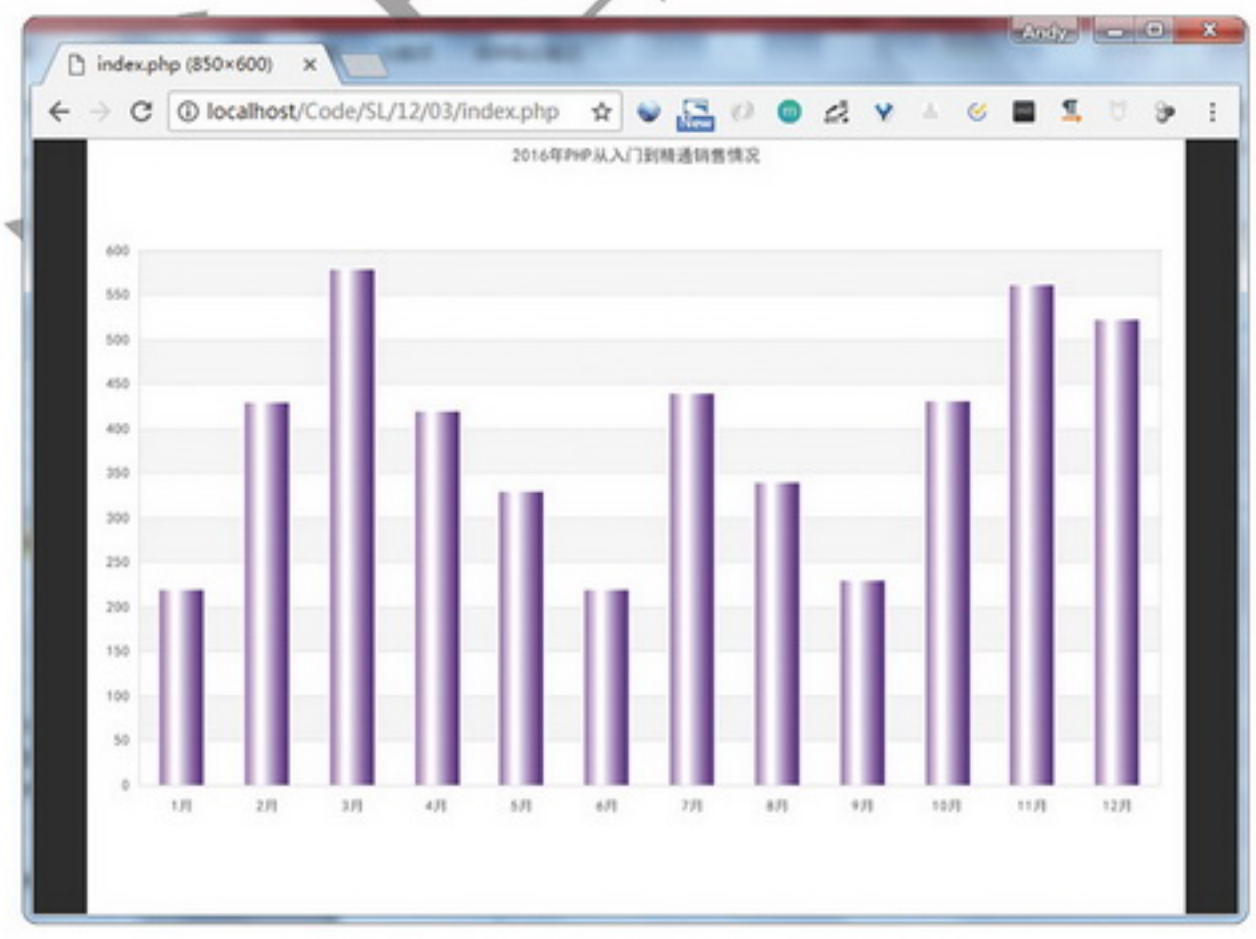


图 12.13　应用柱形图统计图书的月销量

12.4.2 使用折线图统计三本图书销售量

折线图的使用同样十分广泛，如商品的价格走势、股票在某一时间段的涨跌等，都可以使用折线图来分析。

实例 04	使用折线图统计三本图书销售量	实例位置：光盘\Code\SL\12\04

本实例将使用 JpGraph 类库实现折线图统计三本图书上半年销售量。创建折线图的步骤如下：

（1）使用 require_once 语句引用 JpGraph_line.php 文件。

（2）采用折线图进行统计分析，需要创建 LinePlot 对象，而 LinePlot 类在 JpGraph_line.php 中定义，需要应用 require_once 语句引用该文件。

（3）创建 Graph 对象，生成一个 850×600 像素大小的画布，设置 X 轴、Y 轴刻度类型，设置 X 轴、Y 轴数据。

（4）创建三个对象 LinePlot，设置折线的颜色和图例名称。

（5）将绘制的折线图添加到画布中。

（6）输出图像。

本实例的完整代码如下：

实例04-1

```
<?php
require_once ('../jpgraph/jpgraph.php');                    //必须要引用的文件
require_once ('../jpgraph/jpgraph_line.php');               //包含折线图文件

$graph = new Graph(850,600,'auto');                         //设置画布大小
//设置刻度类型，X轴刻度可作为文本标注的直线刻度，Y轴为直线刻度
$graph->SetScale("textlin");
$graph->title->SetFont(FF_CHINESE);                         //设置中文字体
$graph->title->Set('2016上半年PHP图书销售情况');              //设置标题
$graph->yaxis->HideZeroLabel();                             //设置Y轴数据不显示0
$graph->xaxis->SetTickLabels(array('1月','2月','3月','4月','5月','6月')); //设置X轴数据
$graph->xgrid->SetColor('#E3E3E3');
$graph->xgrid->Show();                                      //显示X轴交叉线

//设置Y轴数据
$datay1 = array(200,150,230,150,234,252);
$datay2 = array(120,90,420,80,322,342);
$datay3 = array(50,170,320,240,254,332);
//创建第一条线
$p1 = new LinePlot($datay1);
$graph->Add($p1);
$p1->SetColor("#6495ED");
$p1->SetLegend('PHP从入门到精通');
```

```
24  //创建第二条线
25  $p2 = new LinePlot($datay2);
26  $graph->Add($p2);
27  $p2->SetColor("#B22222");
28  $p2->SetLegend('PHP项目开发案例整合');
29  //创建第三条线
30  $p3 = new LinePlot($datay3);
31  $graph->Add($p3);
32  $p3->SetColor("#FF1493");
33  $p3->SetLegend('PHP开发实例大全');
34
35  $graph->legend->SetFrameWeight(1);                    //设置图例边框
36  $graph->Stroke();                                     //显示图片
37  ?>
```

本实例的运行结果如图 12.14 所示。

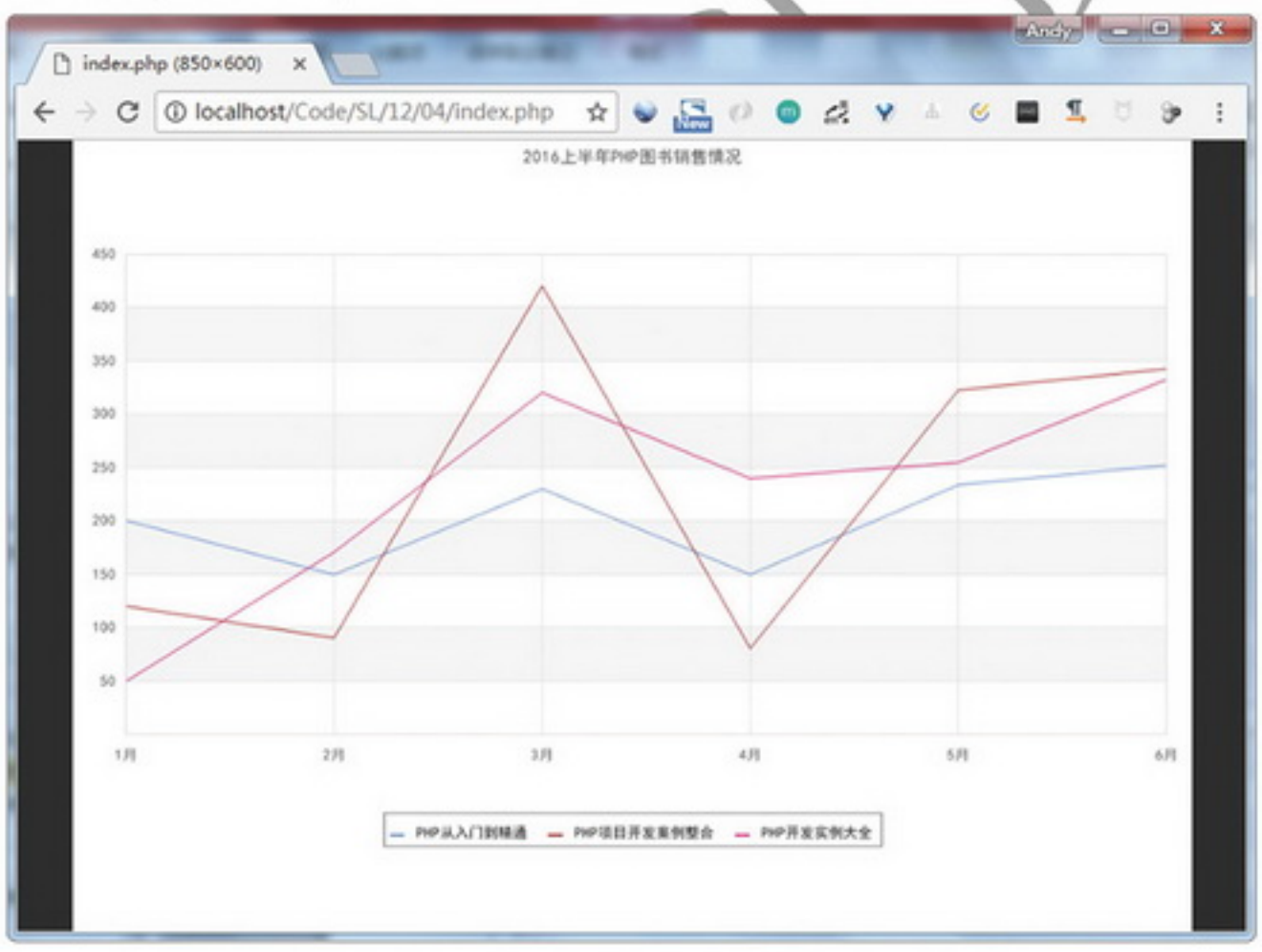


图 12.14　应用折线图统计图书上半年销售量

12.4.3 使用 3D 饼形图统计各类商品的年销售额比例

饼形图是一种非常实用的数据分析技术，可以清晰地表达出数据之间的关系。在调查商场某类商品的年销售额比例时，最好的显示方式就是使用饼形图，通过饼形图可以直观地看到某类产品的销售额在所有商品中所占的比例。

实例 05　统计各类商品的年销售额比例

实例位置：光盘\Code\SL\12\05

使用 3D 饼形图统计各类商品的年销售额比例的步骤与创建其他图形的步骤大致相同，不再赘述，程序完整代码如下：

实例05-1

```
<?php
    require_once '../jpgraph/jpgraph.php';                    //导入Jpgraph类库
    require_once '../jpgraph/jpgraph_pie.php';                //导入Jpgraph类库的饼形图功能
    require_once '../jpgraph/jpgraph_pie3d.php';              //导入Jpgraph类库的3D饼形图功能

    $data = array(69, 78, 99, 63, 98);                        //设置统计数据
    $graph = new PieGraph(600, 300);                          //设置画布大小
    $graph->title->SetFont(FF_CHINESE);                       //设置中文字体
    $graph->title->Set('明日科技部门业绩比较表');                 //设置标题
    $pieplot = new PiePlot3D($data);                          //创建3D饼图对象
    $pieplot->SetCenter(0.5, 0.5);                            //设置饼图居中
    $department = array('IT数码', '家电', '日用', '服装', '食品'); //设置文字框对应的内容
    $pieplot->SetLegends($department);                        //添加图例
    $graph->legend->SetFont(FF_SIMSUN, FS_BOLD);              //设置字体
    $graph->legend->SetLayout(LEGEND_HOR);
    $graph->legend->Pos(0.5, 0.98, 'center', 'bottom');       //图例文字框的位置
    $graph->Add($pieplot);                                    //将3D饼图添加到统计图对象中
    $graph->Stroke();                                         //输出图像
?>
```

运行结果如图 12.15 所示。

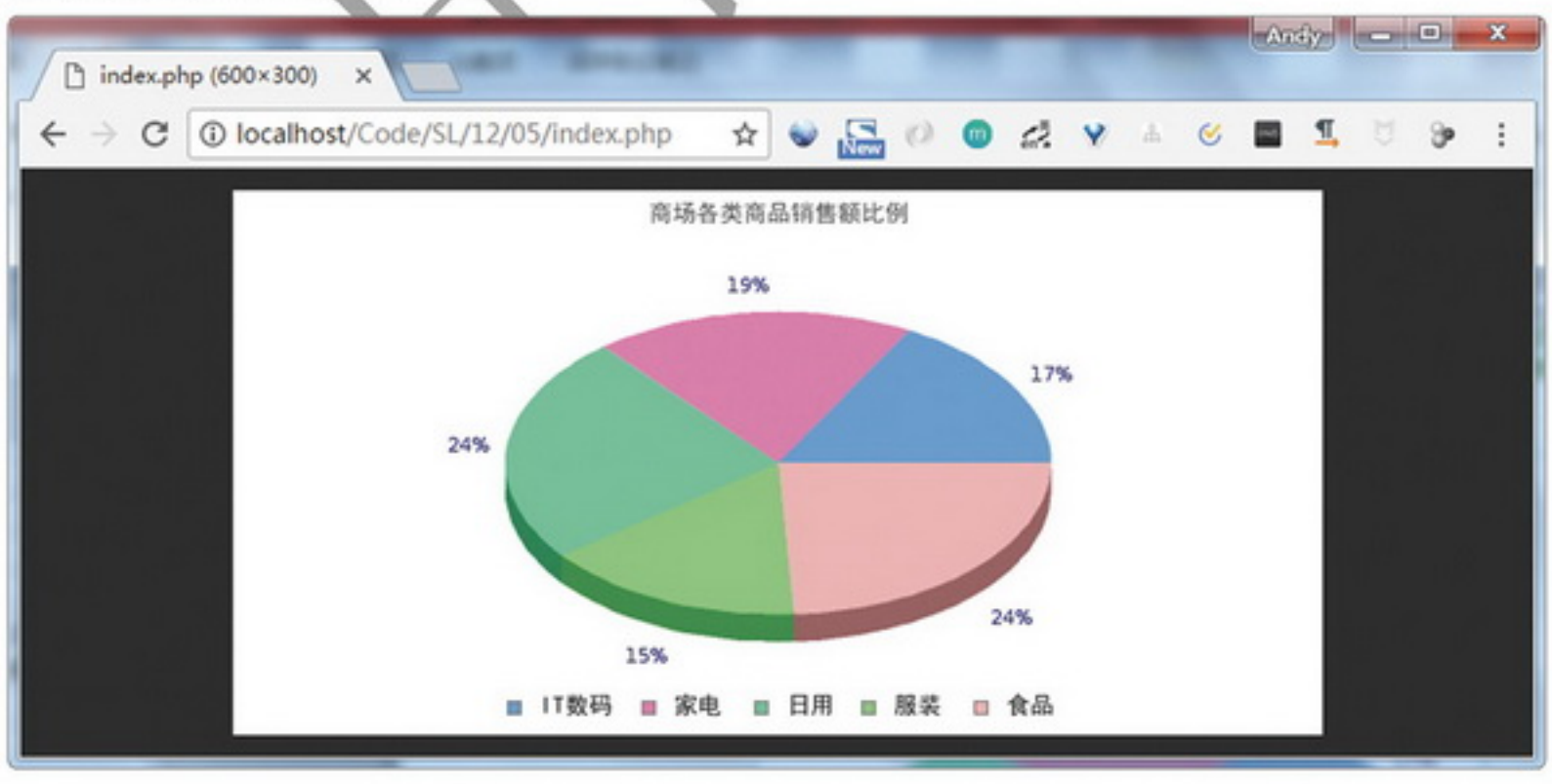


图 12.15　应用 3D 饼形图统计各类商品的年销售额比例

12.5 难点解答

12.5.1 JpGraph 中文乱码

JpGraph 生成图片时出现中文乱码是一个常见的问题，使用 JpGraph 前请按照 12.3.2 小节介绍的方法进行相应修改。修改完成后，在输出中文文字前，需要设置文字字体，如“$graph->title->SetFont(FF_CHINESE);”。此外，需要注意设置的字体在“C:\Windows\Fonts”路径下必须存在。

12.5.2 如何使用 JpGraph 的其他图形

使用 JpGraph 除了可以生成折线图、柱状图和饼图外，还可以生成散点图、脉冲图、样条图等等。在使用这些图形前，需要先查看官方手册，手册网址：http://jpgraph.net/download/manuals/chunkhtml/index.html。例如，需要画散点图，在手册中搜索“Scatter graphs”，单击进入“Scatter graphs”手册，在手册中查找相应的示例代码，然后根据个人需求，修改相应代码即可。

12.6 小　结

本章首先介绍了 GD2 函数库的安装方法，以及应用 GD2 函数创建图像，使读者对 GD2 函数有一个初步的认识。接着介绍了一个专门用于绘制统计图的类库——JpGraph。通过讲解 JpGraph 类库的安装、配置到实际的应用过程，指导读者熟练使用该类库，完成更复杂的图形图像的开发。

12.7 动手纠错

1．运行“光盘\Code\Debug\12\01”文件夹下的程序，图片不显示。使用谷歌浏览器审查元素，提示“未定义 imagecreatefromjpeg() 函数”，如图 12.16 所示，请根据注释改正程序。

图 12.16　未定义 imagecreatefromjpeg() 函数提示

2．运行“光盘\Code\Debug\12\02”文件夹下的程序，图片不显示。使用谷歌浏览器审查 verify.php 页面元素，提示“无效的字体文件名”，如图 12.17 所示，请根据注释改正程序。

图 12.17　无效的字体文件名提示

3．运行“光盘\Code\Debug\12\03”文件夹下的 login.php 文件，出现验证码文字显示不全的现象，如图 12.18 所示，请根据注释改正程序。

图 12.18　验证码显示不全

4．运行“光盘\Code\Debug\12\04”文件夹下的 index.php 文件，出现如图 12.19 所示错误信息，请根据注释改正程序。

图 12.19　文件不存在错误信息

5．运行“光盘\Code\Debug\12\05”文件夹下的程序，出现如图 12.20 所示错误提示，请根据注释改正程序。

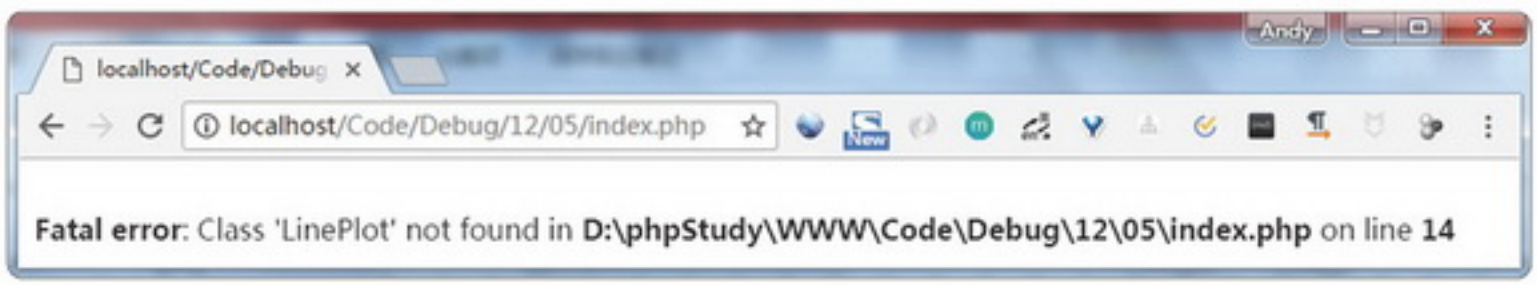


图 12.20　LinePlot 类不存在错误信息

第 13 章

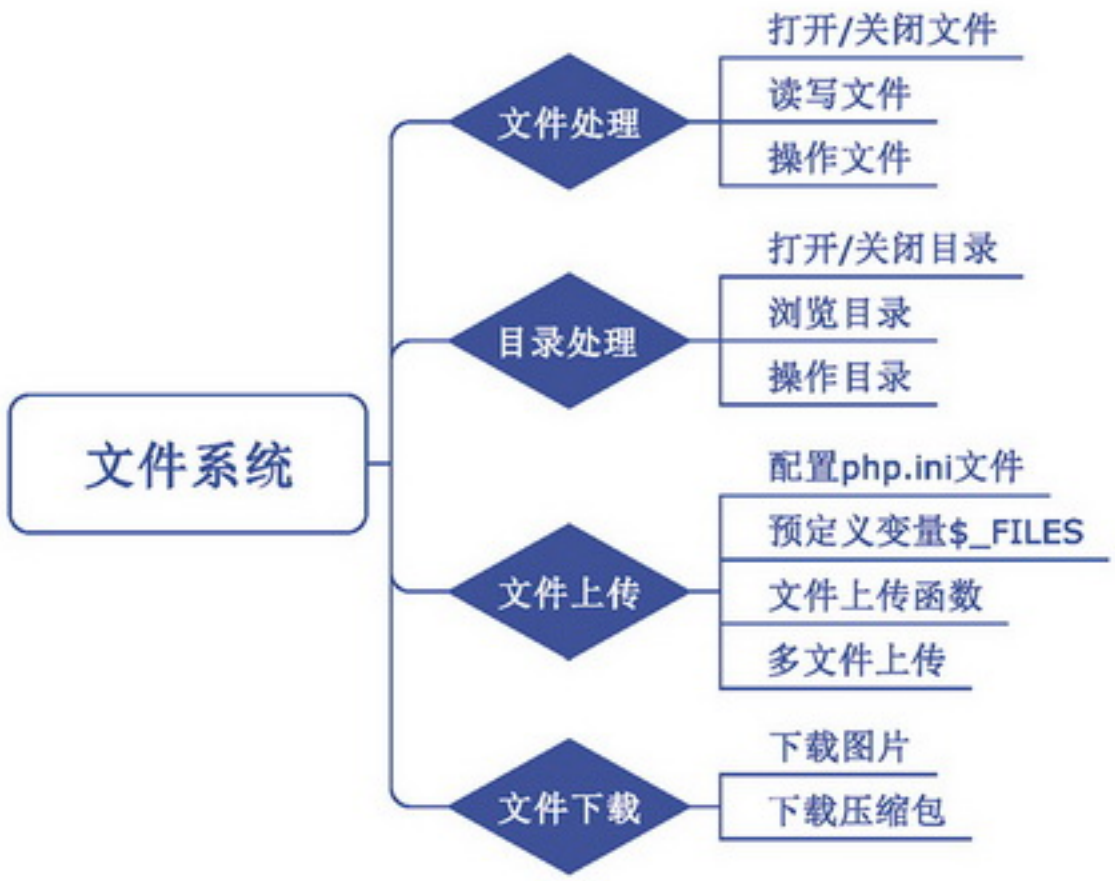
文件系统

（视频讲解：1 小时）

本章概览

文件是用来存取数据的方式之一。相对于数据库来说，文件在使用上更方便、直接。如果数据较少、较简单，使用文件无疑是最合适的方法。PHP 内置函数中提供了丰富的文件和目录读写功能以及文件上传功能，可以快速便捷地实现对文件的处理、目录的处理，以及文件的上传和下载等需求，本章将对文件系统进行详细讲解。

知识框架



13.1 文件处理

文件处理包括文件的读取、关闭和重写等，例如，访问一个文件需要 3 步：打开文件、读写文件和关闭文件。其他的操作要么是包含在读写文件中（如显示内容、写入内容等），要么与文件自身的属性有关系（如文件遍历、文件改名等）。本节将对常用的文件处理技术进行详细讲解。

13.1.1 打开 / 关闭文件

打开/关闭文件使用 fopen() 函数和 fclose() 函数。打开文件应格外认真，以免将文件内容全部删掉。

1. 打开文件

对文件进行操作时首先要打开文件，这是进行数据存取的第一步。在 PHP 中使用 fopen() 函数打开文件，fopen() 函数的语法如下：

```
resource fopen ( string $filename , string $mode [, bool $use_include_path = false [,
                 resource $context ]] )
```

第 1 个参数 filename 是要打开的包含路径的文件名，可以是相对路径，也可以是绝对路径。如果没有任何前缀则表示打开的是本地文件；第 2 个参数 mode 是打开文件的方式，可取的值如表 13.1 所示。

表 13.1　fopen() 中参数 mode 的取值列表

mode 取值	模 式 名 称	说　　明
r	只读	读模式——进行读取，文件指针位于文件的开头
r+	只读	读写模式——进行读写，文件指针位于文件的开头。在现有文件内容的末尾之前进行写入就会覆盖原有内容
w	只写	写模式——进行写入文件，文件指针指向头文件。如果该文件存在，则所有文件内容被删除；否则函数将创建这个文件
w+	只写	写模式——进行读写，文件指针指向头文件。如果该文件存在，则所有文件的内容被删除；否则函数将创建这个文件
x	谨慎写	写模式——打开文件，从文件头开始写。如果文件已经存在，则该文件将不会被打开，函数返回 false，PHP 将产生一个警告
x+	谨慎写	读 / 写模式——打开文件，从文件头开始写。如果文件已经存在，则该文件将不会被打开，函数返回 false，PHP 将产生一个警告
a	追加	追加模式——打开文件，文件指针指向尾文件。如果该文件已有内容，则将从文件末尾开始追加；如果该文件不存在，则函数将创建这个文件

续表

mode 取值	模式名称	说　明
a+	追加	追加模式——打开文件，文件指针指向头文件。如果该文件已有内容，则从文件末尾开始追加或者读取；如果该文件不存在，则函数将创建这个文件
b	二进制	二进制模式——用于与其他模式进行连接。如果文件系统能够区分二进制文件和文本文件，可能会使用它。Windows 可以区分；UNIX 则不区分。推荐使用这个选项，便于获得最大程度的可移植性。它是默认模式
t	文本	用于与其他模式的结合。这个模式只是 Windows 下的一个选项

第3个参数use_include_path是可选的，该参数在配置文件php.ini中指定一个路径，如D:\phpStudy\WWW\mess.php，如果希望服务器在这个路径下打开所指定的文件，可以将其设置为1或true。

第4个参数context是可选的，在PHP 5.0.0中增加了对上下文（Context）的支持。

2. 关闭文件

对文件的操作结束后应该关闭这个文件，否则可能引起错误。在PHP中使用fclose()函数关闭文件，该函数的语法如下：

```
bool fclose ( resource handle ) ;
```

该函数将参数handle指向的文件关闭，如果成功，返回true，否则返回false。其中的文件指针必须是有效的，并且是通过fopen()函数成功打开的文件。例如：

```
<?php
     $f_open =fopen("../file.txt.","rb");          //打开文件
     …                                             //对文件进行操作
     fclose($f_open)                               //操作完成后关闭文件
?>
```

13.1.2 从文件中读取数据

1. 读取整个文件：readfile()函数、file()函数和file_get_contents()函数

（1）readfile()函数

readfile()函数用于读入一个文件并将其写入到输出缓冲中，如果出现错误则返回false。函数语法如下：

```
int readfile ( string $filename [, bool $use_include_path = false] )
```

使用readfile()函数，不需要打开/关闭文件，不需要echo/print等输出语句，直接写出文件路径即可。

📖 **说明：** readfile()函数的第二个参数use_include_path如果设置为true，则将在include_path中搜索文件。include_path可以在php.ini文件中设置。

（2）file() 函数

file() 函数也可以读取整个文件的内容，只是 file() 函数将文件内容按行存放到数组中，包括换行符在内。如果失败则返回 false。函数语法如下：

```
array file ( string $filename [, int $flags = 0] )
```

📖 **说明：** file() 函数的第二个参数 flag 的值可以设置为以下一个或多个常量：

- ◆ FILE_USE_INCLUDE_PATH：在 include_path 中查找文件。
- ◆ FILE_IGNORE_NEW_LINES：在数组每个元素的末尾不要添加换行符。
- ◆ FILE_SKIP_EMPTY_LINES：跳过空行。

（3）file_get_contents() 函数

将整个文件读入一个字符串。函数语法如下：

```
string file_get_contents ( string $filename [, bool $use_include_path = false [, resource
                           $context [, int $offset = -1 [, int $maxlen ]]]] )
```

如果有 offset 参数和 maxlen 参数，将在参数 offset 所指定的位置开始读取长度为 maxlen 的内容。如果失败，返回 false。该函数适用于二进制对象，是将整个文件的内容读入到一个字符串中的首选方式。

实例 01　读取文件 tm.txt 的内容

实例位置：光盘\Code\SL\13\01

创建一个 tm.txt 文本文件，输入“明日科技陪伴你学习 PHP”。

在 tm.txt 文件同级目录下创建一个 index.php 文件，分别使用 readfile() 函数、file() 函数和 file_get_contents() 函数读取文件 tm.txt 的内容，代码如下：

实例01-1

```
<!DOCTYPE html>
<html lang="zh-cn">
<head>
    <meta charset="utf-8">
    <title>PHP零基础</title>
    <link   href="css/bootstrap.css" rel="stylesheet">
</head>
<body class="col-sm-6">
<h3 class="col-sm-offset-3">使用三种方法读取文件</h3>
<table class="table table-bordered">
    <thead>
    <tr class="success">
        <th>方法名</th>
        <th>内容</th>
    </tr>
    </thead>
    <tbody>
    <tr class="info">
```

```
19          <td>readfile()函数</td>
20          <!--  使用readfile()函数读取tm.txt文件的内容  -->
21          <td><?php readfile('tm.txt'); ?></td>
22      </tr>
23      <tr class="success">
24          <td>file()函数</td>
25          <!--  使用file()函数读取tm.txt文件的内容  -->
26          <td>
27              <?php
28                  $f_arr = file('tm.txt');
29                  foreach($f_arr as $cont){
30                      echo $cont."<br>";
31                  }
32              ?>
33          </td>
34      </tr>
35      <tr class="info">
36          <td>file_get_contents()函数</td>
37          <!--  使用file_get_contents()函数读取tm.txt文件的内容  -->
38          <td>
39              <?php
40                  $f_chr = file_get_contents('tm.txt');
41                  echo $f_chr;
42              ?>
43          </td>
44      </tr>
45      </tbody>
46  </table>
47  </body>
48  </html>
```

运行结果如图13.1所示。

图13.1 读取整个文件的内容

注意：tm.txt 文件和 index.php 文件之间是同级关系。读取 tm.txt 文件时，需要根据位置关系，使用相对路径或绝对路径找到 tm.txt 文件位置。

练一练：

（1）试着使用 file_get_contents() 函数获取“www.mingrisoft.com”页面资源。（光盘\Code\Try\13\01）

（2）试着使用 file() 函数获取“www.mingrisoft.com”页面资源，并输出行号，如图 13.2 所示。（光盘\Code\Try\13\02）

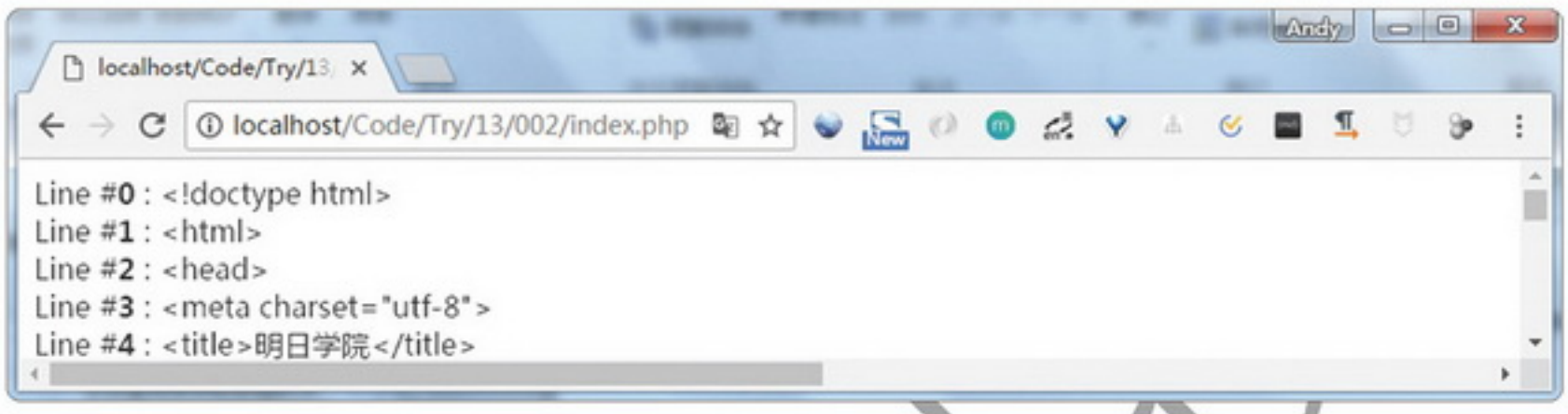


图 13.2　带行号的资源内容

2. 读取一行数据：fgets() 函数和 fgetss() 函数

（1）fgets() 函数

fgets() 函数用于一次读取一行数据。函数语法如下：

```
string fgets ( resource $handle [, int $length ] )
```

参数 handle 是被打开的文件，参数 length 是要读取的数据长度。fgets() 函数能够实现从 handle 指定文件中读取一行并返回长度最大值为 length-1 个字节的字符串。在遇到换行符、EOF 或者读取了 length-1 个字节后停止。如果忽略 length 参数，则读取数据直到行结束。

（2）fgetss() 函数

fgetss() 函数是 fgets() 函数的变体，用于读取一行数据，同时，fgetss() 函数会过滤掉 HTML 标签。函数语法如下：

```
string fgetss ( resource $handle [, int $length [, string $allowable_tags ]] )
```

使用 allowable_tags 参数来控制哪些标签不被过滤掉。如 allowable_tags 参数值为“”，则表示只保留 HTML 的 标签，其他标签会被过滤。

实例 02　使用 fgets() 函数与 fgetss() 函数读取文件　　实例位置：光盘\Code\SL\13\02

使用 fgets() 函数与 fgetss() 函数分别读取 common.php 文件并显示出来，观察它们的区别。创建一个 common.php 文件，该文件中包含 2 个 HTML 标签，分别为 <h3> 标签和 标签。代码如下：

实例02-1

```
<html>
<body>
    <h3>测试<span style="color: red">两个函数的区别</span></h3>
```

```
04  </body>
05  </html>
06  html标签以外的内容
```

在 common.php 文件同级目录下创建一个 index.php 文件，在 index.php 文件中，分别使用 fgets() 函数和 fgetss() 函数读取 common.php 文件。在 fgetss() 函数中，使用第 3 个参数 allowable_tags 指定 标签不被去掉，代码如下：

实例02-2

```
<!DOCTYPE html>
<html lang="zh-cn">
<head>
    <meta charset="utf-8">
    <title>PHP零基础</title>
    <link   href="css/bootstrap.css"rel="stylesheet">
</head>
<body class="col-sm-6">
<h3 class="col-sm-offset-3">比较fgets()函数和fgetss()函数</h3>
<table class="table table-bordered">
    <thead>
    <tr class="success">
        <th>方法名</th>
        <th>内容</th>
    </tr>
    </thead>
    <tbody>
    <tr class="info">
        <td>fgets()函数</td>
        <!--  使用fgets()函数读取common.php文件的内容  -->
        <td>
            <?php
            $fopen = fopen('common.php','rb');
            while(!feof($fopen)){                    //feof()函数测试指针是否到了文件结束的位置
                echo fgets($fopen);                  //输出当前行
            }
            fclose($fopen);
            ?>
        </td>
    </tr>
    <tr class="success">
        <td>fgetss()函数</td>
        <!--  使用fgetss()函数读取common.php文件的内容  -->
        <td>
            <?php
            $fopen = fopen('common.php','rb');
```

```
37              while(!feof($fopen)){        //使用feof()函数测试指针是否到了文件结束的位置
38                  echo fgetss($fopen,100,'<span>');        //输出当前行
39              }
40              fclose($fopen);
41              ?>
42          </td>
43      </tr>
44      </tbody>
45  </table>
46  </body>
47  </html>
```

运行结果如图 13.3 所示。

图 13.3　fgets() 函数和 fgetss() 函数的区别

练一练：

（1）试着使用 fgets() 函数输出 tm.txt 文件的内容。（光盘\Code\Try\13\03）

（2）试着使用 fgetss() 函数输出 sample.php 文件的内容，注意输出的结果中是否包含 HTML 标签。（光盘\Code\Try\13\04）

3. 读取一个字符：fgetc() 函数

在对某一个字符进行查找、替换时，需要有针对性地对某个字符进行读取，在 PHP 中可以使用 fgetc() 函数实现此功能。函数语法如下：

```
string fgetc ( resource $handle )
```

该函数返回一个字符，该字符从 handle 指向的文件中得到。遇到 EOF 则返回 false。

注意：fgetc() 函数读取的是单字节，而中文在 UTF-8 编码格式下占 3 字节，所以输出中文时会是乱码。

4. 读取任意长度的字符串：fread() 函数

fread() 函数可以从文件中读取指定长度的数据，函数语法如下：

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

参数 handle 为指向的文件资源，length 是要读取的字节数。当函数读取 length 个字节或到达 EOF 时停止执行。

例如，使用 fread() 函数读取 poem.txt 文件，poem 文件内容如下：

```
锦瑟无端五十弦，一弦一柱思华年。
```

在 poem.txt 文件同级目录下，创建一个 index.php 文件，代码如下：

```
01  <?php
02  $filename = "poem.txt";                          //要读取的文件
03  $fp = fopen($filename,"rb");                     //打开文件
04  echo fread($fp,6);                               //使用fread()函数读取文件内容的前6个字节
05  echo "<p>";
06  echo fread($fp,filesize($filename));             //输出其余的文件内容
07  ?>
```

运行结果如图 13.4 所示。

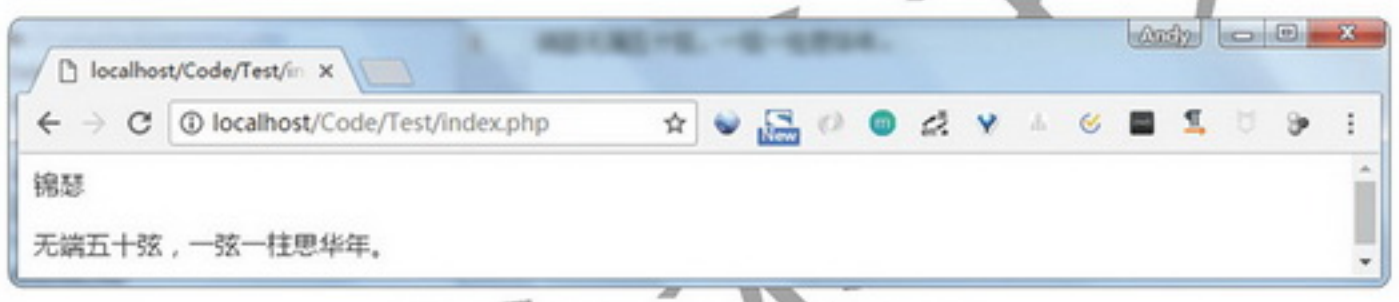


图 13.4 使用 fread() 函数读取文件

注意：poem.txt 文件的编码格式也需要设置为“UTF-8”，否则会出现乱码。

13.1.3 将数据写入文件

写入数据也是 PHP 中常用的文件操作，在 PHP 中使用 fwrite() 函数和 file_put_contents() 函数向文件中写入数据。fwrite() 函数也称为 fputs() 函数，它们的用法相同。fwrite() 函数的语法如下：

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

该函数把内容 string 写入文件指针 handle 处。如果指定了长度 length，则写入 length 个字节后停止。如果文件内容长度小于 length，则会输出全部文件内容。

file_put_contents() 函数是 PHP 5 新增的函数，其语法为：

```
int file_put_contents ( string $filename , mixed $data [, int $flags = 0 [, resource $context ]] )
```

file_put_contents() 函数中的参数说明如表 13.2 所示。

表 13.2 file_put_contents() 函数中的参数说明

参　数	说　明
filename	要被写入数据的文件名
data	要写入的数据。类型可以是 string，array 或者是 stream 资源

续表

参　数	说　明
flags	flags 的值可以是以下 flag 使用 OR(|) 运算符进行的组合： FILE_USE_INCLUDE_PATH：在 include 目录里搜索 filename FILE_APPEND：如果文件 filename 已经存在，追加数据而不是覆盖 LOCK_EX：在写入时获得一个独占锁
context	一个 context 资源

使用 file_put_contents() 函数和依次调用 fopen() 函数、fwrite() 函数、fclose() 函数的功能一样。

实例 03　分别使用 fwrite() 函数和 file_put_contents() 函数写入数据　　实例位置：光盘\Code\SL\13\03

首先使用 fwrite() 函数向 poem.txt 文件写入数据“此情可待成追忆”，然后再使用 file_put_contents() 函数写入数据“只是当时已惘然”。代码如下：

实例03-1

```
<?php
$filepath = "poem.txt";
$str1 = "此情可待成追忆<br>";
$str2 = "只是当时已惘然<br>";
echo "用fwrite函数写入文件：";
$fopen = fopen($filepath,'wb') or die('文件不存在');
fwrite($fopen,$str1);
fclose($fopen);
readfile($filepath);
echo "<p>用file_put_contents函数写入文件：";
file_put_contents($filepath,$str2);
readfile($filepath);
?>
```

运行结果如图 13.5 所示。

图 13.5　使用 fwrite() 函数和 file_put_contents() 函数写入数据

说明：在 index.php 文件的同级目录下，会生成一个 poem.txt 文本文件，内容是“只是当时已惘然”，而前一句已经被 file_put_contents() 函数写入时覆盖。

练一练：

（1）试着使用 file_get_contents() 函数读取 tm.txt 文件内容，然后使用 file_put_contents() 函数写入 new.txt 文件。（光盘\Code\Try\13\05）

（2）试着使用 file_put_contents() 函数向 tm.txt 文件中追加内容：“ ——by mrsoft”。（光盘\Code\Try\13\06）

13.1.4 操作文件

除了可以对文件内容进行读写，对文件本身同样也可以进行操作，如复制、重命名、查看修改日期等。PHP 内置了大量的文件操作函数，常用的文件操作函数如表 13.3 所示。

表 13.3 常用的文件操作函数

函数原型	函数说明	举例
bool copy(string path1, string path2)	将文件从 path1 复制到 path2。如果成功，返回 true，失败则返回 false	copy('tm.txt','../tm.txt')
bool rename(string filename1,string filename2)	把 filename1 重命名为 filename2	rename('1.txt','tm.txt')
bool unlink(string filename)	删除文件，成功返回 true，失败则返回 false	unlink('./tm.txt')
int fileatime(string filename)	返回文件最后一次被访问的时间，时间以 UNIX 时间戳的方式返回	fileatime('1.txt')
int filemtime(string filename)	返回文件最后一次被修改的时间，时间以 UNIX 时间戳的方式返回	date('Y-m-d H:i:s', filemtime('1.txt'))
int filesize(string filename)	取得文件 filename 的大小（bytes）	filesize('1.txt')

array pathinfo(string name [, int options])	返回一个数组，包含文件 name 的路径信息。有 dirname、basename 和 extension。可以通过 option 设置要返回的信息，有 PATHINFO_DIRNAME、PATHINFO_BASENAME 和 PATHINFO_EXTENSION。默认为返回全部	$arr = pathinfo('/tm/sl/12/5/1.txt');
		foreach($arr as $method => $value){
		echo $method.": ".$value."
		";
		}

string realpath (string filename)	返回文件 filename 的绝对路径。如 c:\tmp\…\1.txt	realpath('1.txt')

array stat (string filename)	返回一个数组，包括文件的相关信息，如上面提到的文件大小、最后修改时间等	$arr = stat('1.txt');
		foreach($arr as $method => $value){
		echo $method.": ".$value."
		";
		}

📖 **说明**：在读写文件时，除了 file()、readfile() 等少数几个函数外，其他操作必须要先使用 fopen() 函数打开文件，最后再用 fclose() 函数关闭文件。文件的信息函数（如 filesize、filemtime 等）则都不需要打开文件，只要文件存在即可。

13.2 目录处理

目录是一种特殊的文件。要浏览目录下的文件，首先要打开目录，浏览完毕后，同样要关闭目录。目录处理包括打开目录、浏览目录和关闭目录。

13.2.1 打开 / 关闭目录

打开 / 关闭目录和打开 / 关闭文件类似，但打开的文件如果不存在，则会自动创建一个新文件，而打开的目录如果不正确，则一定会报错。

1. 打开目录

PHP 使用 opendir() 函数来打开目录，函数语法如下：

```
resource opendir ( string $path [, resource $context ] )
```

函数 opendir() 的参数 path 是一个合法的目录路径，成功执行后返回目录的指针；如果 path 不是一个合法的目录或者因为权限或文件系统错误而不能打开目录，则返回 false，并产生一个 E_WARNING 级别的错误信息。可以在 opendir() 函数前面加上“@”符号来抑制错误信息的输出。

2. 关闭目录

关闭目录使用 closedir() 函数，函数语法如下：

```
void closedir ([ resource $dir_handle ] )
```

参数 dir_handle 为使用 opendir() 函数打开的一个目录指针。

下面为打开和关闭目录的流程代码：

```
<?php
$path = "D:\\phpStudy\\WWW\\Code";
if (is_dir($path)){                                    //检测是否是一个目录
    if ($dire = opendir($path))                        //判断打开目录是否成功
        echo $dire;                                    //输出目录指针
}else{
    echo '路径错误';
    exit();
}
…                                                      //其他操作
```

```
closedir($dire);                                    //关闭目录
?>
```

is_dir() 函数判断当前路径是否为一个合法的目录。如果合法，返回 true，否则返回 false。

13.2.2 浏览目录

在 PHP 中浏览目录中的文件使用的是 scandir() 函数，函数语法如下：

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

该函数返回一个数组，包含 directory 中的所有文件和目录。参数 sorting_order 指定排序顺序，默认按字母升序排序，如果添加了该参数，则变为降序排序。

例如，查看 D:\phpStudy\WWW\Code 目录下的所有文件。代码如下：

```
01  <?php
02  $path = 'D:\\phpStudy\\WWW\\Code';          //要浏览的目录
03  if(is_dir($path)){                          //判断文件名是否为目录
04      $dir = scandir($path);                  //使用scandir函数取得所有文件及目录
05      foreach($dir as $value){                //使用foreach循环
06          echo $value."<br>";                 //循环输出文件及目录名称
07      }
08  }else{
09      echo "目录路径错误！";
10  }
11  ?>
```

运行结果如图 13.6 所示，文件结构如图 13.7 所示。

图 13.6 浏览目录结果

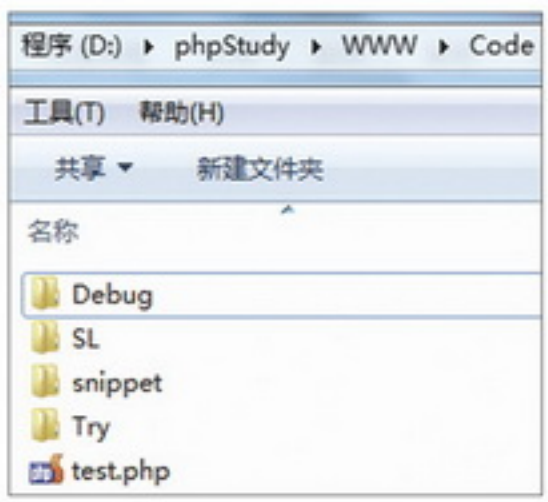


图 13.7 文件结构

13.2.3 操作目录

目录是特殊的文件，也就是说，对文件的操作处理函数（如重命名）多数同样适用于目录。但还有一些特殊的函数只针对目录进行操作，表 13.4 列举了一些常用的目录操作函数。

表 13.4　常用的目录操作函数

函 数 原 型	函 数 说 明	举　　例
bool mkdir (string pathname)	新建一个指定的目录	mkdir('temp');
bool rmdir (string dirname)	删除所指定的目录，该目录必须是空的	rmdir('tmp')
string getcwd (void)	取得当前工作的目录	getcwd()
bool chdir (string directory)	改变当前目录为 directory	echo getcwd() . " "; chdir('../'); echo getcwd() . " ";
float disk_free_space (string directory)	返回目录中的可用空间（bytes）。被检查的文件必须通过服务器的文件系统访问	disk_free_space('E:\\wamp');
float disk_total_space(string directory)	返回目录的总空间大小（bytes）	disk_total_space('E:\\wamp');

string readdir (resource handle)	返回目录中下一个文件的文件名（使用此函数时，目录必须是使用 opendir() 函数打开的）。在 PHP 5 之前，都是使用这个函数来浏览目录的	while(false!==($path=readdir($handle)))
		{
		echo $path;
		}

void rewinddir (resource handle)	将指定的目录重新指定到目录的开头	rewinddir($handle)

13.3 文件上传

文件上传可以通过 HTTP 协议来实现。要使用文件上传功能，首先要在 php.ini 配置文件中对上传功能做一些设置，然后了解预定义变量 $_FILES，并通过 $_FILES 的值对上传文件做出一些限制和判断，最后使用 move_uploaded_file() 函数实现上传的功能。

13.3.1 配置 php.ini 文件

要实现上传功能，首先要在 php.ini 中开启文件上传，并对其中的一些参数做出合理的设置。找到 File Uploads 项，可以看到以下 3 个属性值，含义如下：

◆ file_uploads：如果值是 on，说明服务器支持文件上传；如果为 off，则不支持。

◆ upload_tmp_dir：上传文件临时目录。在文件被成功上传之前，文件首先存放到服务器端的临时目录中。如果想要指定位置，可在这里设置，否则使用系统默认目录即可。

◆ upload_max_filesize：服务器允许上传的文件的最大值，以 MB 为单位，系统默认为 2MB，用

户可以自行设置。

除了 File Uploads 项，还有几个属性也会影响到上传文件的功能。

- max_execution_time：PHP 中一个指令所能执行的最大时间，单位是秒。
- memory_limit：PHP 中一个指令所分配的内存空间，单位是 MB。

📖 说明：在 phpStudy 集成环境中，上述介绍的这些配置信息默认已经配置好了。

⚡ 注意：如果要上传超大的文件，需要对 php.ini 文件进行修改。包括 upload_max_filesize 的最大值，max_execution_time 一个指令所能执行的最大时间和 memory_limit 一个指令所分配的内存空间。

13.3.2 预定义变量 $_FILES

$_FILES 变量存储的是上传文件的相关信息，这些信息对于上传功能有很大的作用。该变量是一个二维数组。保存的信息如表 13.5 所示。

表 13.5 预定义变量 $_FILES 元素

元 素 名	说 明
$_FILES[filename][name]	存储了上传文件的文件名。如 exam.txt、myDream.jpg 等
$_FILES[filename][size]	存储了文件大小。单位为字节
$_FILES[filename][tmp_name]	文件上传时，首先在临时目录中被保存成一个临时文件。该变量为临时文件名
$_FILES[filename][type]	上传文件的类型
$_FILES[filename][error]	存储了上传文件的结果。如果返回 0，则说明文件上传成功

从 PHP 4.2.0 开始，PHP 将随文件信息数组一起返回一个对应的错误代码，即生成的文件数组中的 error 字段，也就是 $_FILES[filename][error] 参数值。$_FILES[filename][error] 参数值及说明如表 13.6 所示。

表 13.6 $_FILES[filename][error] 参数值及说明

错误代码	错误常量	描 述
0	UPLOAD_ERR_OK	没有错误发生，文件上传成功
1	UPLOAD_ERR_INI_SIZE	上传的文件超过了 php.ini 中 upload_max_filesize 选项限制的值
2	UPLOAD_ERR_FORM_SIZE	上传文件的大小超过了 HTML 表单中 MAX_FILE_SIZE 选项指定的值
3	UPLOAD_ERR_PARTIAL	文件只有部分被上传
4	UPLOAD_ERR_NO_FILE	没有文件被上传
6	UPLOAD_ERR_NO_TMP_DIR	找不到临时文件夹
7	UPLOAD_ERR_CANT_WRITE	文件写入失败

例如，创建一个上传文件域，通过 $_FILES 变量输出上传文件的资料。代码如下：

```
<!DOCTYPE html>
<html lang="zh-cn">
<head>
    <meta charset="utf-8">
    <title>PHP零基础</title>
    <link   href="http://cdn.bootcss.com/bootstrap/3.3.7/css/bootstrap.css"
            rel="stylesheet">
</head>
<body class="col-sm-6 col-sm-offset-1 bg-info">
<h3 class="col-sm-offset-3">文件上传</h3>
<form action="" method="post" enctype="multipart/form-data">
    <div class="form-group">
        <label for="exampleInputEmail1">邮箱</label>
        <input type="email" class="form-control" id="exampleInputEmail1"
               placeholder="Email">
    </div>
    <div class="form-group">
        <label for="exampleInputPassword1">密码</label>
        <input type="password" class="form-control" id="exampleInputPassword1"
               placeholder="Password">
    </div>
    <div class="form-group">
        <label for="exampleInputFile">头像</label>
        <input type="file" id="exampleInputFile" name="upfile">
    </div>
    <button type="submit" class="btn btn-info">Submit</button>
</form>
</body>
</html>
<?php
    if(!empty($_FILES)){                                    //判断变量$_FILES是否为空
        foreach($_FILES['upfile'] as $name => $value)       //使用foreach循环输出上传文件信息
            echo $name.' = '.$value.'<br>';
    }
?>
```

在页面中，填写邮箱、密码，单击“选择文件”按钮，选中一张图片，单击“打开”按钮上传图片，如图 13.8 所示。

注意：使用 Form 表单上传文件时，必须设置表单的 enctype 属性值为 "multipart/form-data"，即 enctype="multipart/form-data"，否则接收不到上传的信息，$File 为空。

填写完信息后，单击 Submit 按钮，提交表单，运行结果如图 13.9 所示。

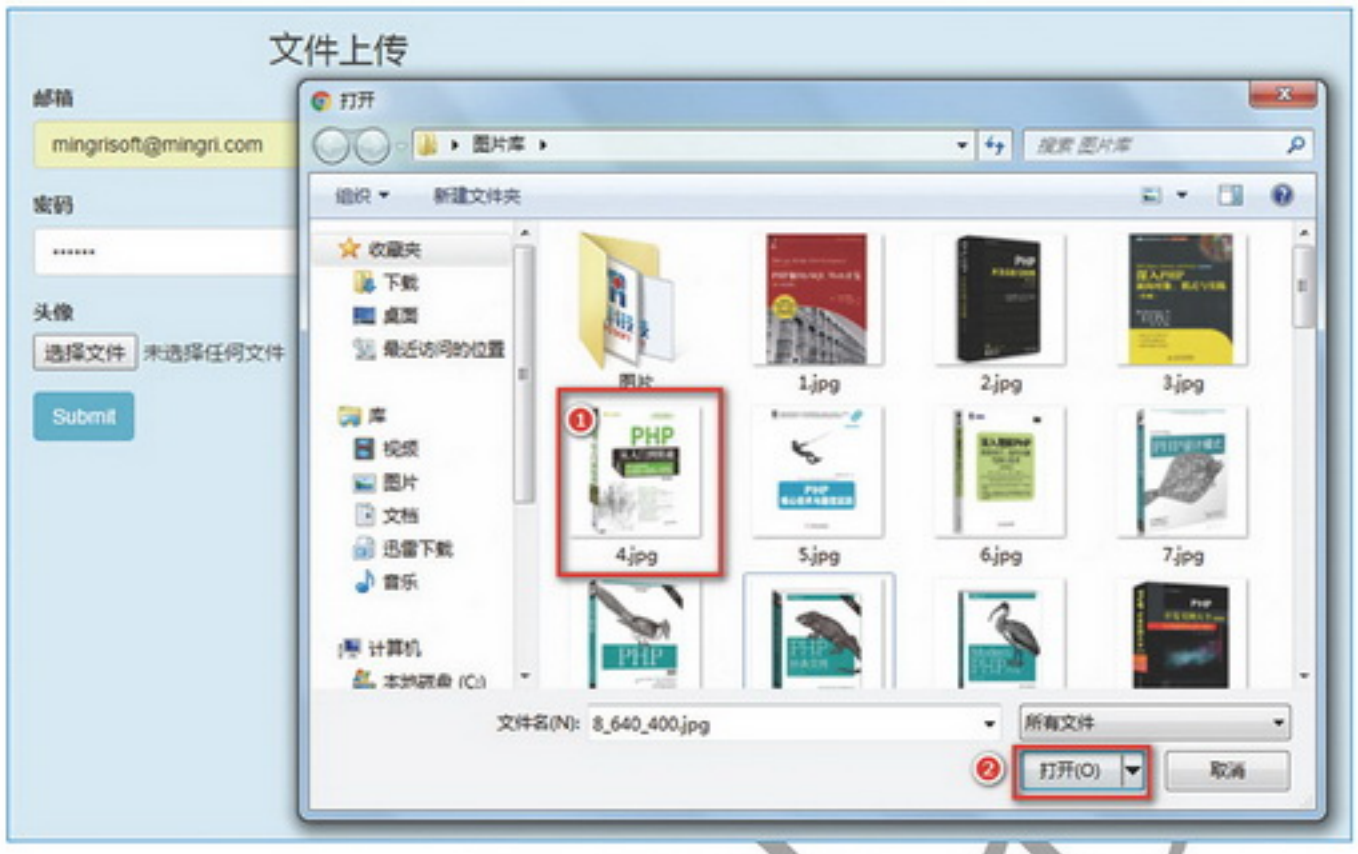


图 13.8 选择上传的文件

图 13.9 文件上传结果

13.3.3 文件上传函数

PHP 中使用 move_uploaded_file() 函数上传文件。该函数的语法如下：

```
bool move_uploaded_file ( string $filename , string $destination )
```

move_uploaded_file() 函数将上传文件存储到指定的位置。如果成功，返回 true，否则返回 false。参数 filename 是上传文件的临时文件名，即 $_FILES[filename][tmp_name]；参数 destination 是上传后保存的新的路径和名称。

实例 04 实现上传图片功能

实例位置：光盘\Code\SL\13\04

本实例将创建一个上传表单，允许上传 1MB 以下的图片，且图片格式为 jpeg、jpg、png 或 gif。程序的开发步骤如下：

（1）文件上传表单。文件上传表单有以下特性：

◆ 在 <form> 标签中，必须设置属性 enctype="multipart/form-data"。这样，服务器可以知道上传的文件带有常规的表单信息。

◆ 可以设置上传文件最大长度的表单域。这是一个隐藏的域，如下所示：

```
<input type="hidden" name="MAX_FILE_SIZE" value="1000000" >
```

MAX_FILE_SIZE 表单域是可选的，该值可以在服务器端设置。然而，如果在这个表单中使用，表单域的名称必须是 MAX_FILE_SIZE。其值是允许上传文件的最大值（单位是字节）。这里设置为 1000000B（几乎是 1MB）。

◆ 需要指定文件类型，如下所示：

```
<input type = "file" name="upfile" id="upfile">
```

读者可以为文件选择喜欢的任何名字，但要记住，在 PHP 接收脚本时，必须使用这个名字来访问文件。

根据上面的三个特性，创建一个文件上传页面，命名为 index.php，index.php 文件代码如下：

实例04-1

```
<!DOCTYPE html>
<html lang="zh-cn">
<head>
    <meta charset="utf-8">
    <title>PHP零基础</title>
    <link   href="css/bootstrap.css" rel="stylesheet">
</head>
<body class="col-sm-6 col-sm-offset-1 bg-info">
<h3 class="col-sm-offset-3">文件上传</h3>
<form action="doAction.php" method="post" enctype="multipart/form-data">
    <input type="hidden" name="MAX_FILE_SIZE" value="1000000" >
    <div class="form-group">
        <label for="exampleInputEmail1">邮箱</label>
        <input type="email" class="form-control" id="exampleInputEmail1"
               name="email" placeholder="Email" >
    </div>
    <div class="form-group">
        <label for="exampleInputPassword1">密码</label>
        <input type="password" class="form-control" id="exampleInputPassword1"
               name="password" placeholder="Password">
    </div>
```

```
22      <div class="form-group">
23          <label for="exampleInputFile">头像</label>
24          <input type="file" id="exampleInputFile" name="upfile">
25          <p class="text-danger">需是jpeg、png或gif格式</p>
26      </div>
27      <button type="submit" class="btn btn-info">提交</button>
28  </form>
29  </body>
30  </html>
```

运行结果如图 13.10 所示。

图 13.10 单文件上传

（2）上传表单。当用户填写完信息单击“提交”按钮，将表单信息提交到 doAction.php 文件。在 index.php 文件的同级目录下，创建 doAction.php 文件。在该文件中，首先检测文件是否上传成功，然后检测上传文件是否满足设定的需求，当全部检测都通过后，使用 move_uploaded_file() 函数上传文件。doAction.php 文件代码如下：

实例04-2

```
01  <?php
02      $email    = $_POST['email'];                          //接收用户名
03      $password = $_POST['password'];                       //接收密码
04      $fileInfo = $_FILES['upfile'];                        //接收上传文件
05      /** 检测文件上传是否成功 **/
06      if(!is_null($fileInfo)){
07          if($fileInfo['error']>0){
08              switch($fileInfo['error']){
09                  case 1:
10                      echo '上传的文件超过了php.ini中upload_max_filesize选项限制的值';
11                      break;
12                  case 2:
13                      echo '上传文件的大小超过了HTML表单中 MAX_FILE_SIZE选项指定的值';
14                      break;
```

```
15                  case 3:
16                      echo '文件只有部分被上传';
17                      break;
18                  case 4:
19                      echo '没有文件被上传';
20                      break;
21                  case 6:
22                      echo '找不到临时文件夹';
23                      break;
24                  case 7:
25                      echo '文件写入失败';
26                      break;
27              }
28              exit;
29          }else{
30              /** 检测文件长度 **/
31              if($fileInfo['size'] > 1000000){
32                  echo '上传文件大于1M';
33                  exit;
34              }
35              /** 检测扩展名 **/
36              $allowExt = array('jpeg','jpg','png','gif');
37              $ext = strtolower(pathinfo($fileInfo['name'],PATHINFO_EXTENSION));
38              if(!in_array($ext,$allowExt)){
39                  echo '不允许的扩展名';
40                  exit;
41              }
42              /** 检测文件类型 **/
43              $allowMime = array('image/jpeg','image/png','image/gif');
44              if(!in_array($fileInfo['type'],$allowMime)){
45                  echo "上传文件类型错误";
46                  exit;
47              }
48              /** 检测是否为真实图片 **/
49              if(!@getimagesize($fileInfo['tmp_name'])){
50                  echo '不是真实图片';
51                  exit;
52              }
53              /** 保存图片 **/
54              $uploadPath  = 'upload';                                    //保存路径
55              if (!file_exists($uploadPath)) {
56                  $result = mkdir($uploadPath);
57              }
58              $uniName     = md5(uniqid(microtime(true),true));           //名字需要唯一
59              $destination = $uploadPath.'/'.$uniName.'.'.$ext;
```

```
60              if(@move_uploaded_file($fileInfo['tmp_name'], $destination)){
61                  echo "上传成功";
62                  //TODO:其他操作，如写入数据库等
63              }else{
64                  echo '文件移动失败';
65                  exit;
66              }
67          }
68      }else{
69          echo '文件上传出错';
70          exit;
71      }
72  ?>
```

上述代码中，首先接收表单传递的文件信息，然后根据错误码判断文件是否上传成功。如果上传失败，根据错误码输出错误信息。此外，值得注意的是，图片上传的最终路径是由三部分拼接而成。第一部分 $uploadPath（文件存储的目录）。首先判断该文件是否存在，如果不存在，使用 mkdir() 函数创建文件；第二部分 $uniName（文件名），为确保文件名唯一，同时使用 md5() 函数、uniqid() 函数和 microtime() 函数来实现；第三部分 $ext（文件扩展名），即为上传文件的扩展名。最后通过 move_uploaded_file() 函数将文件上传到指定路径。

文件上传成功运行结果如图 13.11 所示，文件存储目录如图 13.12 所示。

图 13.11 文件上传成功

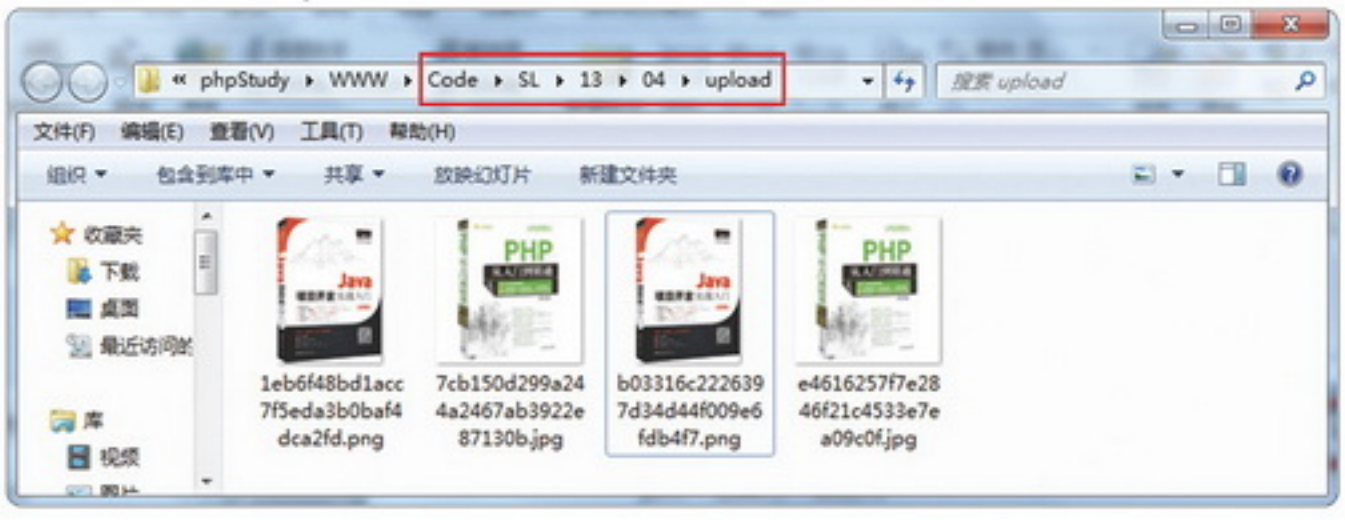


图 13.12 文件存储目录

练一练：

（1）试着在博客中上传 txt 文件，如图 13.13 所示。（光盘\Code\Try\13\07）

图 13.13　添加 txt 文件

（2）试着在博客中上传 rar 文件，且文件小于 1M。（光盘\Code\Try\13\08）

13.3.4 多文件上传

PHP 支持同时上传多个文件，只需要在表单中对文件上传域使用数组命名即可。

实例 05	实现多文件上传	实例位置：光盘\Code\SL\13\05

本实例有 3 个文件上传域，文件域的名字为 upfile[]，提交后上传的文件信息都被保存到 $_FILES[upfile] 中，生成多维数组。读取数组信息，并上传文件。程序的开发步骤如下：

（1）创建一个文件上传页面，命名为 index.php，index.php 文件代码如下：

实例05-1

```
<!DOCTYPE html>
<html lang="zh-cn">
<head>
    <meta charset="utf-8">
    <title>PHP零基础</title>
    <link   href="css/bootstrap.css" rel="stylesheet">
</head>
<body class="col-sm-6 col-sm-offset-1 bg-info">
<h3 class="col-sm-offset-3">多文件上传</h3>
<form action="doAction.php" method="post" enctype="multipart/form-data">
    <div class="form-group">
        <label for="exampleInputFile">请选择您要上传的文件</label>
        <input type="file"  name="upfile[]">
        <p class="text-danger">需是jpeg、png或gif格式</p>
    </div>
    <div class="form-group">
```

```
17          <label for="exampleInputFile">请选择您要上传的文件</label>
18          <input type="file"  name="upfile[]">
19          <p class="text-danger">需是jpeg、png或gif格式</p>
20      </div>
21      <div class="form-group">
22          <label for="exampleInputFile">请选择您要上传的文件</label>
23          <input type="file"  name="upfile[]">
24          <p class="text-danger">需是jpeg、png或gif格式</p>
25      </div>
26      <button type="submit" class="btn btn-info">提交</button>
27  </form>
28  </body>
29  </html>
```

运行结果如图 13.14 所示。

图 13.14 多文件上传

（2）上传表单。在 index.php 文件的同级目录下创建 doAction.php 文件。由于多文件上传的检测方法与单文件上传相同，在 doAction.php 文件中，就不再写验证的功能，主要使用 for 循环遍历 $_FILES['upfile'] 数组，实现对文件的依次上传。doAction.php 文件代码如下：

实例05-2

```
01  <?php
02      if(!empty($_FILES['upfile'])) {
03          $file_name      = $_FILES['upfile']['name'];      //将上传文件名另存为数组
04          $file_tmp_name  = $_FILES['upfile']['tmp_name']; //将上传的临时文件名存为数组
05          for ($i = 0; $i < count($file_name); $i++) {   //循环上传文件
06              if ($file_name[$i] != '') {                //判断上传文件名是否为空
07                  $uploadPath = 'upload';                //设置上传路径
08                  if (!file_exists($uploadPath)) {
09                      $result = mkdir($uploadPath);
10                  }
```

```
11                  $uniName    = md5(uniqid(microtime(true),true));          //名字需要唯一
12                  //获取文件类型
13                  $ext[$i]    = strtolower(pathinfo($file_name[$i],PATHINFO_EXTENSION));
14                  $destination[$i] = $uploadPath.'/'.$uniName.'.'.$ext[$i];  //生成目录
15                  move_uploaded_file($file_tmp_name[$i],$destination[$i]); //上传文件
16                  echo '文件'.$file_name[$i].'上传成功。更名为'.$uniName.'.'.$ext[$i].'<br>';
17              }
18          }
19      }
20  ?>
```

运行结果如图 13.15 所示。

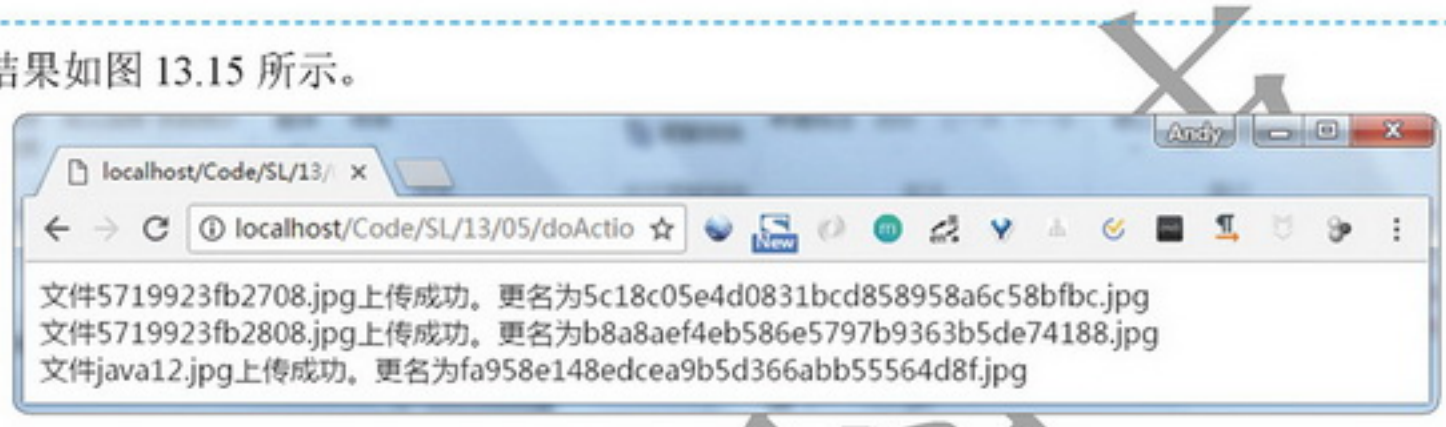


图 13.15　多文件上传

13.4 文件下载

对于文件下载，通常有两种情况。第一种情况，下载浏览器不能解析的文件，如 zip 压缩文件。这种情况的实现方式比较简单，使用 <a> 标签按如下方式即可下载：

```
<a href="upload/bootstrap-3.3.4.zip">bootstrap压缩包</a>
```

当用户单击该链接时，浏览器会显示一个文件下载的对话框，单击“保存”按钮后，就会将文件保存至用户电脑上。注意，href 属性值可以是绝对路径，也可以是相对路径。

第二种情况，下载浏览器可以解析的文件，如 jpg、png 格式等，使用第一种方法，浏览器会直接显示图片，不会弹出对话框提示用户下载，这就需要使用 header() 函数来实现文件下载，代码如下：

```
header('content-disposition:attachment;filename=somefile');
```

header() 函数发送原生 HTTP 头，使用 Content-Disposition 的报文信息来提供一个推荐的文件名，并且强制浏览器显示一个文件下载的对话框。

实例 06　实现文件下载　　实例位置：光盘\Code\SL\13\06

本实例使用两种方式实现文件下载，第一种方式下载 zip 格式的压缩包，第二种方式下载 jpg 格式的图片。首先创建 index.html 文件，index.html 文件代码如下：

实例06-1

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title></title>
</head>
<body>
    <ul>
        <li>
            <a href="upload/bootstrap-3.3.4.zip">bootstrap压缩包</a>
        </li>
        <li>
            <a href="upload/1.jpg">图片1下载</a>
        </li>
        <li>
            <a href="download.php?filename=upload/2.jpg">图片2下载</a>
        </li>
    </ul>
</body>
</html>
```

运行结果如图 13.16 所示

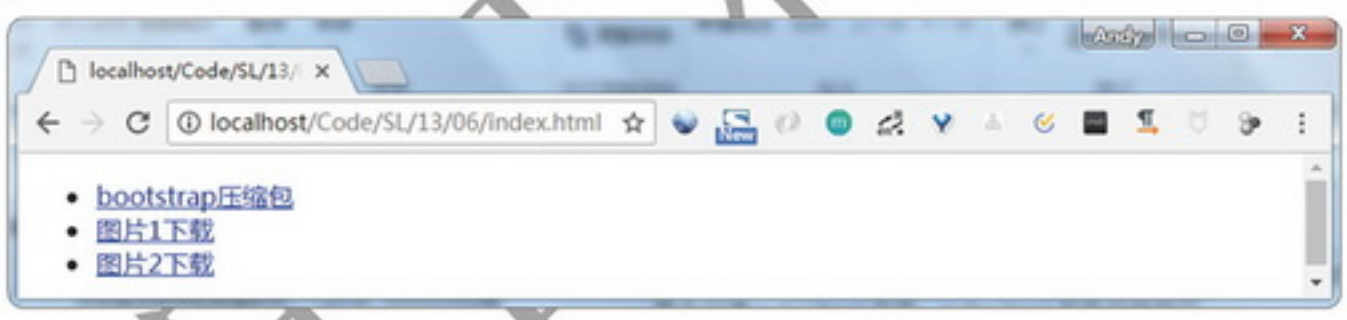


图 13.16 下载列表页面

单击“bootstrap 压缩包”时，由于浏览器不能解析 zip 文件，弹出对话框，如图 13.17 所示。

图 13.17 保存 zip 文件

在图 13.16 下载列表页面中，单击“图片 1 下载”时，由于浏览器能够识别 jpg 格式的文件，所以在浏览器中直接显示图片，如图 13.18 所示。

图 13.18　浏览器中显示图片

现在在 index.html 文件同级目录下创建 download.php 文件，在 download.php 文件中使用 header() 函数下载 jpg 格式的文件。download.php 代码如下：

实例06-2

```
<?php
$filename = $_GET['filename'];
header('content-disposition:attachment;filename='.basename($filename));
header('content-length:'.filesize($filename));
readfile($filename);
?>
```

刷新下载列表页面，单击“图片 2 下载”，运行效果如图 13.19 所示。

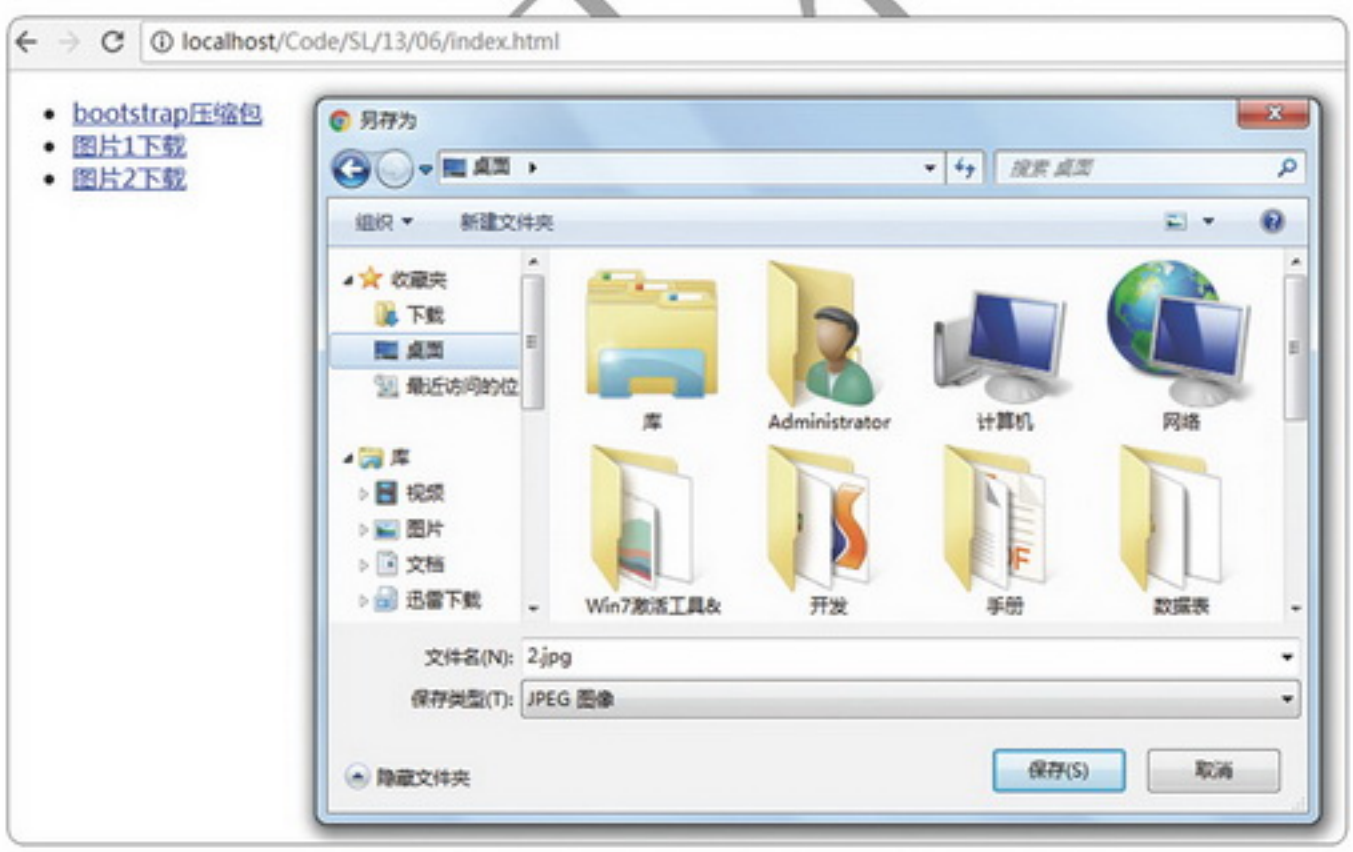


图 13.19　下载 jpg 格式的文件

练一练：

（1）使用 readdir() 函数遍历文件列表，并使用超链接实现文件的下载。（光盘\Code\Try\13\09）

（2）使用 readdir() 函数遍历图片列表，实现单击下载图片的功能，如图 13.20 所示。（光盘\Code\Try\13\10）

图 13.20 图片列表

13.5 难点解答

13.5.1 file() 函数和 file_get_contents() 函数的区别

file() 函数和 file_get_contents() 函数的作用都是将整个文件读入某个介质，其主要区别就在于这个介质的不同。file() 函数是将文件读入一个数组中，而 file_get_contents() 是将文件读入一个字符串中。file() 函数是把整个文件读入一个数组中，然后将文件作为一个数组返回。数组中的每个单元都是文件中相应的一行，包括换行符在内。如果失败，则返回 false。file_get_contents() 函数是将文件的内容读入到一个字符串中的首选方法。

13.5.2 设置表单属性 enctype

在使用表单上传文件时，必须设置 Form 表单的 enctype 属性为 "multipart/form-data"。这是因为 enctype 属性规定在发送到服务器之前，应该如何对表单数据进行编码。enctype 属性值说明如表 13.7 所示。

表 13.7 enctype 属性值的参数说明

值	描 述
application/x-www-form-urlencoded	在发送前编码所有字符（默认）
multipart/form-data	不对字符编码 在使用包含文件上传控件的表单时，必须使用该值
text/plain	空格转换为“+”加号，但不对特殊字符编码

13.6 小 结

本章首先介绍对文件的基本操作，然后学习目录的基本操作，接下来学习文件的高级处理技术，最后又学习 PHP 的文件上传技术，这是一个网站必不可少的组成部分。希望读者能够深入理解本章的

重点知识，牢固掌握常用函数，为深入学习 PHP 打好基础。

13.7 动手纠错

1．运行“光盘\Code\Debug\13\01”文件夹下的 index.php 文件，出现“Warning: file_get_contents (文件 .txt): failed to open stream: No such file or directory”的错误提示，请根据注释改正程序。

2．运行“光盘\Code\Debug\13\02”文件夹下的 index.php 文件，出现中文乱码错误，如图 13.21 所示，请根据注释改正程序。

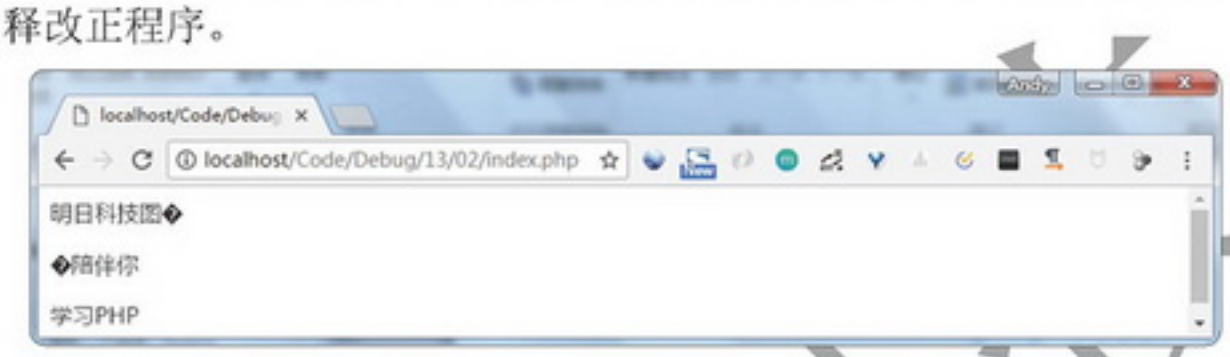


图 13.21　中文乱码

3．运行“光盘\Code\Debug\13\03”文件夹下的 index.php 文件，出现“failed to open stream: No such file or directory”的错误提示，请根据注释改正程序。

4．运行“光盘\Code\Debug\13\04”文件夹下的 index.php 文件，添加图片后单击“提交”按钮，出现“文件上传出错”的提示，请根据注释改正程序。

5．运行“光盘\Code\Debug\13\05”文件夹下的 index.php 文件，上传三张图片单击“提交”按钮，只有最后一张图片上传成功，请根据注释改正程序。

第 14 章

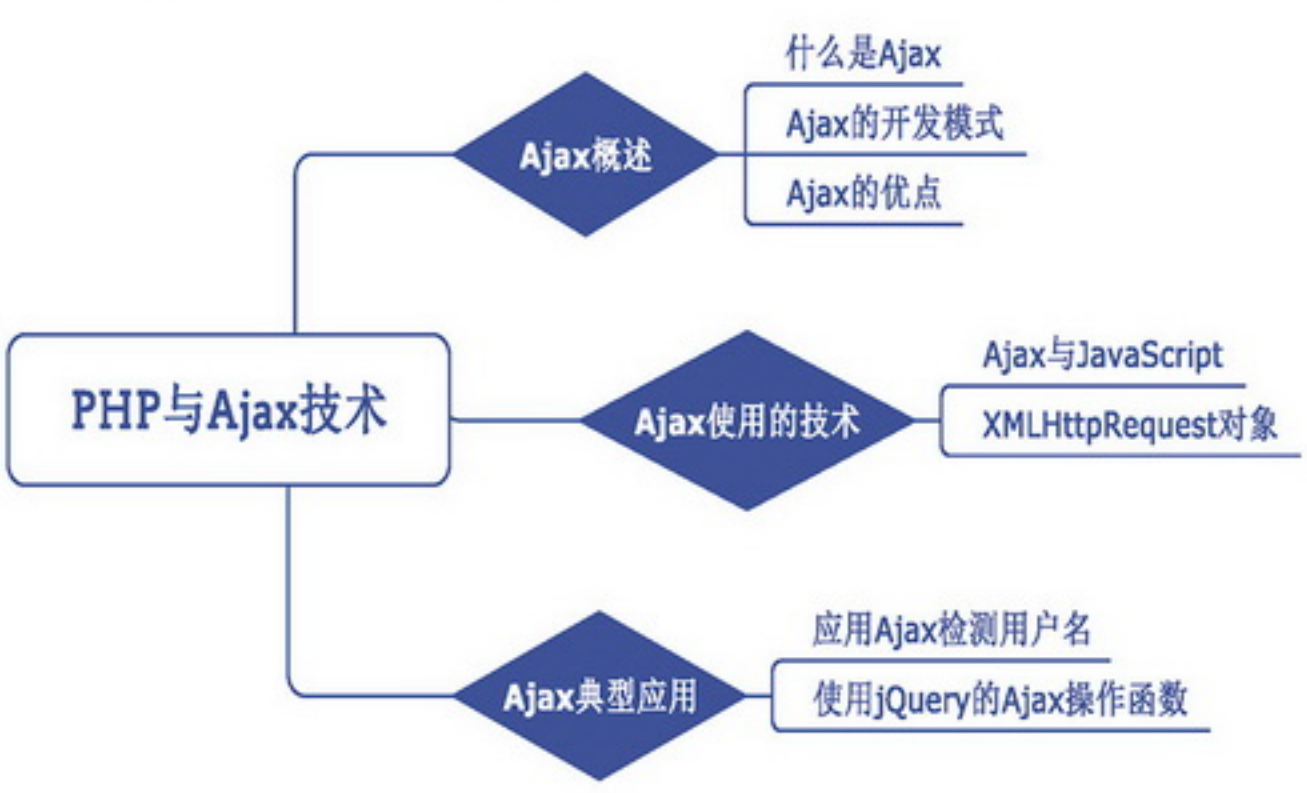
PHP 与 Ajax 技术

（视频讲解：42 分）

本章概览

Ajax 技术是 JavaScript、XML、CSS、DOM 等多种已有技术的组合，它可以实现客户端的异步请求操作，这样可以实现在不需要刷新页面的情况下与服务器进行通信，从而减少了用户的等待时间。相对于传统的 Web 应用开发，Ajax 运用的是更加先进、更加标准化、更加高效的 Web 开发技术体系。需要说明的是，Ajax 是一个客户端技术，无论使用哪种服务器端技术（如 PHP、JSP、ASP 等）都可以使用 Ajax 技术。本章主要介绍 Ajax 技术及如何在 PHP 中应用 Ajax 技术。

知识框架



14.1 Ajax 概述

Ajax 技术改善了传统 Web 应用的用户体验，发掘了 Web 浏览器的潜力，为 Web 开发开创了大量新的可能性。下面对 Ajax 技术进行详细的介绍。

14.1.1 什么是 Ajax

视频讲解

Ajax 是由 Jesse James Garrett 创造的，是 Asynchronous JavaScript And XML 的缩写，即异步 JavaScript 和 XML 技术。Ajax 并不是一门新的语言或技术，它是 JavaScript、XML、CSS、DOM 等多种已有技术的组合，可以实现客户端的异步请求操作，并可以实现在不需要刷新页面的情况下与服务器进行通信，从而减少用户的等待时间，提高使用效率。

14.1.2 Ajax 的开发模式

视频讲解

在传统的 Web 应用模式中，页面中用户的每一次操作都将触发一次返回 Web 服务器的 HTTP 请求，服务器进行相应的处理（获得数据、运行与不同的系统会话）后，返回一个 HTML 页面给客户端。如图 14.1 所示。而在 Ajax 应用中，页面中用户的操作将通过 Ajax 引擎与服务器端进行通信，然后将返回结果提交给客户端页面的 Ajax 引擎，再由 Ajax 引擎来决定将这些数据插入到页面的指定位置。如图 14.2 所示。

图 14.1　传统的 Web 开发模式

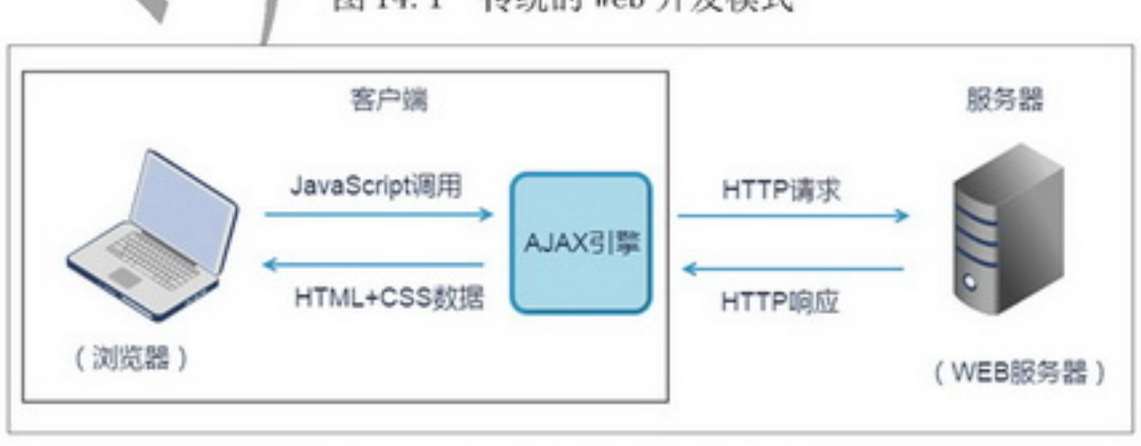


图 14.2　Ajax 应用的开发模式

从图 14.1 和图 14.2 中可以看出，对于每个用户的行为，在传统的 Web 应用模式中，将生成一次 HTTP 请求，而在 Ajax 应用开发模式中，将变成对 Ajax 引擎的一次 JavaScript 调用。在 Ajax 应用开

发模式中通过 JavaScript 实现在不刷新整个页面的情况下，对部分数据进行更新，从而降低网络流量，带来更好的用户体验。

14.1.3 Ajax 的优点

与传统的 Web 应用不同，Ajax 在用户与服务器之间引入一个中间媒介（Ajax 引擎），Web 页面不用打断交互流程进行重新加载即可动态地更新，从而消除了网络交互过程中的“处理—等待—处理—等待”的缺点。

使用 Ajax 的优点具体表现在以下几个方面：

◆ 减轻服务器的负担。Ajax 的原则是“按需求获取数据”，可以最大程度地减少由于冗余请求和响应对服务器造成的负担。

◆ 可以把一部分以前由服务器负担的工作转移到客户端，利用客户端闲置的资源进行处理，减轻服务器和带宽的负担，节约空间和宽带租用成本。

◆ 无刷新更新页面，使用户不用再像以前一样在服务器处理数据时只能在死板的白屏前焦急地等待。Ajax 使用 XMLHttpRequest 对象发送请求并得到服务器响应，在不需要重新载入整个页面的情况下，即可通过 DOM 及时将更新的内容显示在页面上。

◆ 可以调用 XML 等外部数据，进一步实现 Web 页面显示和数据的分离。

14.2 Ajax 使用的技术

14.2.1 Ajax 与 JavaScript

Ajax 利用 JavaScript 将 DOM、HTML（或 XHTML）、XML 以及 CSS 等技术综合起来，并控制它们的行为。因此，要开发一个复杂、高效的 Ajax 应用程序，就必须要对 JavaScript 有一定的了解。关于 JavaScript 脚本语言的详细讲解可参考相关书籍。

14.2.2 XMLHttpRequest 对象

Ajax 技术中，最核心的技术就是 XMLHttpRequest，它是一个具有应用程序接口的 JavaScript 对象，能够使用超文本传输协议（HTTP）连接服务器，是微软公司为了满足开发者的需要，于 1999 年在 IE 5.0 浏览器中率先推出的。现在许多浏览器都对其提供了支持，但实现方式与 IE 有所不同。

通过 XMLHttpRequest 对象，Ajax 可以像桌面应用程序一样只同服务器进行数据层面的交换，而不用每次都刷新页面，也不用每次都将数据处理的工作交给服务器来做，这样既减轻了服务器负担又加快了响应速度，从而缩短了用户等待的时间。

在使用 XMLHttpRequest 对象发送请求和处理响应之前，首先需要初始化该对象，由于 XMLHttpRequest 对象还没有标准化，所以对于不同的浏览器，初始化的方法也不同。

◆ IE 浏览器

IE 浏览器把 XMLHttpRequest 实例化为一个 ActiveX 对象。具体方法如下：

```
var http_request = new ActiveXObject("Msxml2.XMLHTTP");
```

或者

```
var http_request = new ActiveXObject("Microsoft.XMLHTTP");
```

在上面的代码中，Msxml2.XMLHTTP 和 Microsoft.XMLHTTP 是针对 IE 浏览器的不同版本而进行设置的，目前比较常用的是这两种。

◆ 其他浏览器

Google、Mozilla、Safari 等其他浏览器把 XMLHttpRequest 实例化为一个本地 JavaScript 对象。具体方法如下：

```
var http_request = new XMLHttpRequest();
```

为了提高程序的兼容性，可以创建一个跨浏览器的 XMLHttpRequest 对象。方法很简单，只需要判断一下不同浏览器的实现方式，如果浏览器提供了 XMLHttpRequest 类，则直接创建一个实例，否则使用 IE 的 ActiveX 控件。具体代码如下：

```
<script>
    if (window.XMLHttpRequest) {                          //Mozilla、Safari等浏览器
        http_request = new XMLHttpRequest();
    }else if (window.ActiveXObject) {                     //IE浏览器
        try {
            http_request = new ActiveXObject("Msxml2.XMLHTTP");
        }
        catch (e) {
            try {
                http_request = new ActiveXObject("Microsoft.XMLHTTP");
            }
            catch (e) {
                alert("您的浏览器不支持AJAX！");
                return false;
            }
        }
    }
</script>
```

说明：由于 JavaScript 具有动态类型特性，而且 XMLHttpRequest 对象在不同浏览器上的实例是兼容的，所以可以用同样的方式访问 XMLHttpRequest 实例的属性或方法，不需要考虑创建该实例的方法。

下面分别介绍 XMLHttpRequest 对象的常用方法和属性。

1. XMLHttpRequest 对象的常用方法

下面对 XMLHttpRequest 对象的常用方法进行详细介绍。

（1）open() 方法

open() 方法用于设置进行异步请求目标的 URL、请求方法以及其他参数信息，具体语法如下：

```
open("method","URL"[,asyncFlag[,"userName"[, "password"]]])
```

在上面的语法中的参数说明如下：

◆ method 用于指定请求的类型，一般为 get 或 post。

◆ URL 用于指定请求地址，可以使用绝对地址或者相对地址，并且可以传递查询字符串。

◆ asyncFlag 为可选参数，用于指定请求方式，异步请求为 true，同步请求为 false，默认情况下为 true；userName 为可选参数，用于指定用户名，没有时可省略。

◆ password 为可选参数，用于指定请求密码，没有时可省略。

（2）send() 方法

send() 方法用于向服务器发送请求。如果请求声明为异步，该方法将立即返回，否则将到接收到响应为止。具体语法格式如下：

```
send(content)
```

在上面的语法中，content 用于指定发送的数据，可以是 DOM 对象的实例、输入流或字符串。如果没有参数需要传递时可以设置为 null。

（3）setRequestHeader() 方法

setRequestHeader() 方法为请求的 HTTP 头设置值。具体语法格式如下：

```
setRequestHeader("label", "value")
```

在上面的语法中，label 用于指定 HTTP 头，value 用于为指定的 HTTP 头设置值。

注意： *setRequestHeader() 方法必须在调用 open() 方法之后才能调用。*

（4）abort() 方法

abort() 方法用于停止当前异步请求。

（5）getAllResponseHeaders() 方法

getAllResponseHeaders() 方法用于以字符串形式返回完整的 HTTP 头信息，当存在参数时，表示以字符串形式返回由该参数指定的 HTTP 头信息。

2. XMLHttpRequest 对象的常用属性

XMLHttpRequest 对象的常用属性如表 14.1 所示。

表 14.1　XMLHttpRequest 对象的常用属性

属　性	说　明
onreadystatechange	每个状态改变时都会触发这个事件处理器，通常会调用一个 JavaScript 函数
readyState	请求的状态。有以下 5 个取值： 0= 未初始化 1= 正在加载 2= 已加载 3= 交互中 4= 完成
responseText	服务器的响应，表示为字符串
responseXML	服务器的响应，表示为 XML。这个对象可以解析为一个 DOM 对象

续表

属　性	说　明
status	返回服务器的 HTTP 状态码，如： 200=" 成功 " 202=" 请求被接收，但尚未成功 " 400=" 错误的请求 " 404=" 文件未找到 " 500=" 内部服务器错误 "
statusText	返回 HTTP 状态码对应的文本

14.3 Ajax 技术的典型应用

14.3.1 应用 Ajax 技术检测用户名

“明日学院”用户注册时，需要检测用户名是否存在，如果存在，则需要提示该用户名已经存在，不能注册，否则可以注册。为提高用户体验，可以使用 Ajax 技术，实现不刷新页面检测用户名是否被占用的功能。

实例 01　Ajax 技术检测用户名是否被占用　　实例位置：光盘\Code\SL\14\01

本实例将使用 Ajax 技术检测用户名是否被占用，程序的开发步骤如下：

（1）创建 register.php 文件的注册页面，代码如下：

实例01-1

```
<!DOCTYPE html>
<html lang="en" class="is-centered is-bold">
<head>
    <meta charset="UTF-8">
    <title>零基础</title>
    <link href="css/main.css" rel="stylesheet">
</head>
<body>
<section style="background: transparent">
    <form class="box py-3 px-4 px-2-mobile" role="form" name="form">
        <div class="is-flex is-column is-justified-to-center">
            <h1 class="title is-3 mb-a has-text-centered">
                注册
            </h1>
            <div class="inputs-wrap py-3">
                <div class="control">
                    <input type="text" id="username" name="username" class="input"
```

```
                placeholder="用户名" value="" required>
            <a href="javascript:;" onClick="checkName();">[检测用户名]</a>
        </div>
        <div class="control">
            <input type="password" id="password" name="password" class="input"
                placeholder="密码" required>
        </div>
        <div class="control">
            <input type="password" id="password2" name="password2" class="input"
                placeholder="确认密码" required>
        </div>
            <div class="control">
                <button class="button is-submit is-primary is-outlined"
                        onClick="checkname();">
                    提交
                </button>
            </div>
        </div>
        <footer class="is-flex is-justified-space-between">
            <a href="login.html">
                已有账号，点击去登录
            </a>
        </footer>
    </div>
</form>
</section>
</body>
</html>
```

上述代码与之前创建注册页面的代码基本相同，只是在单击“检测用户名”超链接时，调用 checkName() 方法。运行结果如图 14.3 所示。

图 14.3　注册页面

（2）编写 Ajax 异步提交代码。在上一步创建的 register.php 文件中，添加如下 JavaScript 代码，实现 Ajax 的异步提交。代码如下：

实例01-2

```
<!DOCTYPE html>
<html lang="en" class="is-centered is-bold">
<head>
    <meta charset="UTF-8">
    <title>零基础</title>
    <link href="css/main.css" rel="stylesheet">
</head>
<body>
<section style="background: transparent">
        //省略部分代码
</section>
<script>
    function checkName() {
        var username = form.username.value;
        if(username=="") {
            window.alert("请添写用户名!");
            form.username.focus();
            return false;
        }
        createRequest('checkName.php',username);          //调用createRequest()方法
    }

    function createRequest(url,username) {                //初始化对象并发出XMLHttpRequest请求
        http_request = false;                             //初始化对象
        if (window.XMLHttpRequest) {                      //谷歌、火狐等浏览器
            http_request = new XMLHttpRequest();
            if (http_request.overrideMimeType) {
                http_request.overrideMimeType("text/xml");
            }
        } else if (window.ActiveXObject) {                //IE浏览器
            try {
                http_request = new ActiveXObject("Msxml2.XMLHTTP");
            } catch (e) {
                try {
                    http_request = new ActiveXObject("Microsoft.XMLHTTP");
                } catch (e) {}
            }
        }
        if (!http_request) {
            alert("不能创建XMLHTTP实例!");
            return false;
        }
```

```
43          http_request.onreadystatechange = alertContents; //指定响应方法
44          http_request.open("POST", url, true);          //设置进行异步请求目标URL和请求方法
45       http_request.setRequestHeader("Content-type","application/x-www-form-urlencoded");
46          http_request.send("username="+username);       //向服务器发送请求
47      }
48      function alertContents() {                         //处理服务器返回的信息
49          if (http_request.readyState == 4) {
50              if (http_request.status == 200) {
51                  alert(http_request.responseText);
52              } else {
53                  alert('您请求的页面发现错误');
54              }
55          }
56      }
57  </script>
58  </body>
59  </html>
```

在上述代码中，当用户单击“检测用户名”时，调用 checkName() 方法。该方法先判断用户名是否为空，如果用户名为空时，提示用户“请填写用户名！”，如果不为空，则调用 createRequest() 方法。createRequest() 方法首先根据不同浏览器实例化 XMLHttpRequest。接下来，使用 onreadystatechange 属性指定响应方法，调用 open() 方法设置进行异步请求目标 URL 和请求方法，调用 send() 方法向服务器发送请求。服务器调用指定的 alertContents() 方法。alertContents() 方法用于判断响应状态，并输出响应内容。

（3）编写检测用户名是否唯一的 PHP 处理页面 checkname.php，在该页面中使用 PDO 方式与数据库交互，使用 PHP 的 echo 语句输出检测结果，完整代码如下：

实例01-3

```
01  <?php
02  require "config.php";                          //引入配置文件
03  $username = trim($_POST['username']);          //trim()函数去除前后空格
04  try{
05      //连接数据库、选择数据库
06      $pdo = new PDO(DNS,DB_USER,DB_PWD);
07  }catch(PDOException $e){
08      //输出异常信息
09      echo $e->getMessage();
10  }
11  //users表中查找输入的用户名和密码是否匹配
12  $sql = 'select * from users where username = :username';
13  $res = $pdo->prepare($sql);
14  $res->bindParam(':username',$username);          //绑定参数
15  if($res->execute()){
16      $rows = $res->fetch(PDO::FETCH_ASSOC);      //返回一个索引为结果集列名的数组
17      if($rows){
18          echo "很抱歉!用户名[".$username."]已经被注册!";
```

```
19        }else{
20            echo "祝贺您!用户名[".$username."]没有被注册!";
21        }
22    }
23    ?>
```

上述代码中，使用 require 语句引入 config.php 文件。config.php 配置文件代码如下：

实例01-4

```
01  <?php
02  define('DB_HOST','localhost');
03  define('DB_USER','root');
04  define('DB_PWD','root');
05  define('DB_NAME','database14');
06  define('DB_PORT','3306');
07  define('DB_TYPE','mysql');
08  define('DB_CHARSET','utf8');
09  define('DNS',DB_TYPE.":host=".DB_HOST.";dbname=".DB_NAME.";charset=".DB_CHARSET);
10  ?>
```

新建 datebase14 数据库，数据库中新建 users 表，users 表数据信息如图 14.4 所示。

图 14.4　users 表数据

运行本实例，在“用户名”文本框中输入“明日科技”，单击“检测用户名”超链接，即可在不刷新页面的情况下弹出“祝贺您！用户名 [明日科技] 没有被注册！”的提示对话框，如图 14.5 所示。当“用户名”对话框中输入“mr”，单击“检测用户名”超链接，运行效果如图 14.6 所示。

图 14.5　用户名没有被注册

图 14.6　用户名已经被注册

练一练：

（1）使用 Ajax 的 get() 方法发送信息：“你好，明日科技”。（光盘\Code\Try\14\01）

（2）使用 Ajax 的 post() 方法发送信息：“你好，明日科技”。（光盘\Code\Try\14\02）

14.3.2 使用 jQuery 的 ajax() 方法

使用原始 Ajax，需要做较多的事情，比如创建 XmlHttpRequest 对象，判断请求状态，编写回调函数等，使得编写 Ajax 代码异常烦琐，并且代码的可读性很差。好在有很多 JavaScript 函数库可以解决这个问题，其中，使用 jQuery 的 ajax() 方法是个不错的选择。

实例 02　使用 jQuery 的 ajax() 方法检测用户名是否被占用　　实例位置：光盘\Code\SL\14\02

本实例主要通过 jQuery Ajax 实现与实例 01 同样的功能，用于对比 jQuery 的 ajax() 方法和原生 Ajax 的区别。程序的开发步骤如下：

（1）创建 register.php 注册页面，该文件比实例 01 步骤（1）中的 register.php 文件多一行代码，即引入 jQuery 文件。主要代码如下：

实例02-1

```
<!DOCTYPE html>
<html lang="en" class="is-centered is-bold">
<head>
    <meta charset="UTF-8">
    <title>零基础</title>
    <link href="css/main.css" rel="stylesheet">
    <script src="js/jquery.min.js"></script>
</head>
<body>
//省略其余代码
```

新增代码

（2）编写 Ajax 异步提交代码。在上一步创建的 register.php 文件中，添加如下 jQuery 代码，实现 Ajax 的异步提交。代码如下：

实例02-2

```
<!DOCTYPE html>
<html lang="en" class="is-centered is-bold">
<head>
    <meta charset="UTF-8">
    <title>零基础</title>
    <link href="css/main.css" rel="stylesheet">
    <script src="js/jquery.min.js"></script>
</head>
<body>
//省略部分代码
<script>
    function checkName() {
```

```
13          var username = $('#username').val();                    //获取用户名
14          if(username == "") {
15              window.alert("请添写用户名!");
16              $('#username').focus();
17              return false;
18          }
19          $.ajax({
20              type: "POST",                                       //提交方式
21              url:"checkName.php",                                //发送请求的地址
22              data:'username='+username,                          //传递数据
23              success:function(msg){                              //回调函数
24                  alert(msg);
25              }
26          });
27      }
28  </script>
29  </body>
30  </html>
```

上述步骤中，仅用了几行代码，就实现了和原生 Ajax 同样的功能。使用了 jQuery 的 ajax() 方法主要参数如下：

◆ type：提交方式，通常为“GET”或“POST”，默认为“GET”。

◆ url：发送请求的地址，默认为当前页地址。

◆ data：发送到服务器的数据，将自动转换为请求字符串格式。GET 请求中将附加在 URL 后。查看 processData 选项说明以禁止此自动转换。必须为 Key/Value 格式。如果为数组，jQuery 将自动为不同值对应同一个名称。如 {foo:["bar1", "bar2"]} 转换为 "&foo=bar1&foo=bar2"。

◆ success：请求成功后的回调函数。参数由服务器返回。

📖 **说明：**更多参数及 jQuery 的 ajax() 方法请参考《jQuery 开发手册》。

（3）编写检测用户名是否唯一的 PHP 处理页 checkname.php，该代码与实例 01 中的 checkname.php 文件基本相同，主要修改返回参数。checkname.php 文件具体代码如下：

实例02-3

```
01  <?php
02      require "config.php";                                       //引入配置文件
03      $username = trim($_POST['username']);                       //trim()函数去除前后空格
04      try{
05          //连接数据库、选择数据库
06          $pdo = new PDO("mysql:host=".DB_HOST.";dbname=".DB_NAME,DB_USER,DB_PWD);
07      }catch(PDOException $e){
08          //输出异常信息
09          echo $e->getMessage();
10      }
11      //user表中查找输入的用户名和密码是否匹配
12      $sql = 'select * from users where username = :username';
13      $res = $pdo->prepare($sql);
```

```
14      $res->bindParam(':username',$username);              //绑定参数
15      if($res->execute()){
16          $rows = $res->fetch(PDO::FETCH_ASSOC);           //返回一个索引为结果集列名的数组
17          if($rows){
18              $res =  "很报歉!用户名[".$username."]已经被注册!";
19          }else{
20              $res = "祝贺您!用户名[".$username."]没有被注册!";
21          }
22          echo $res;
23      }
24  ?>
```

修改后代码

上述代码中，将 $res 返回给 Ajax 的 success 回调函数，最终使用 alert() 方法输出。运行结果与实例 01 完全相同。

练一练：

（1）试着在 database14 数据库中新建一个 posts 文章表，使用 Ajax 实现无跳转添加文章功能，运行效果如图 14.7 所示。（光盘\Code\Try\14\03）

图 14.7　Ajax 无跳转添加文章

（2）试着使用 Ajax 实现无跳转删除文章功能。（光盘\Code\Try\14\04）

14.4 难点解答

14.4.1 浏览器兼容性问题

从历史来看，是微软首先在其 Internet Explorer 5 for Windows 中以一个 ActiveX 对象形式实现了 XMLHttpRequest 对象。随后，由 Mozilla 工程的工程师实现了 Mozilla 1.0（和 Netscape 7）的一种兼

容的本机版本；而稍后，苹果公司在其 Safari 1.2 上也实现了相同的工作。其实，在 W3C 标准的文档对象模型（DOM）Level 3 加载与存储规范中，也提到了类似的功能。现在，它成为一种事实上的标准，并开始在以后发行的大多数浏览器中得到实现。现代主流浏览器几乎都已经支持 XMLHttpRequest 对象，但 IE10 以下版本的浏览器是不可以直接实例化 XMLHttpRequest 的，所以，首先需要考虑浏览器兼容问题。

14.4.2 使用 jQuery 的 ajax() 方法

使用 jQuery 的 ajax() 方法能够简化代码，提高代码的可读性，所以，需要读者熟悉 jQuery 中的与 Ajax 相关的方法，以及 jQuery 的其他常用语法。比如，使用 Ajax 无刷新添加数据时，当 success 属性回调成功后，通常需要拼接成指定的 HTML 语句，并插入到指定位置。此时就需要使用 jQuery 插入节点的方法，如 append() 方法、prepend() 方法、after() 方法等等。

14.5 小 结

本章主要介绍了应用 PHP 开发动态网站时的一些高级技术，读者应该认真学习并掌握。通过这些技术可以使编程水平上升到一个新的层次。例如，使用 Ajax 技术可以实现无刷新效果，增强页面的交互性。

14.6 动手纠错

1．运行“光盘\Code\Debug\14\01”文件夹下的 index.php 文件，填写内容后单击“提交”按钮，没有新增数据，请根据 addPost.php 文件中的注释改正程序。

2．运行“光盘\Code\Debug\14\02”文件夹下的 index.php 文件，填写内容后单击“提交”按钮，没有接收到 JSON 数据，请根据 lists.html 文件中的注释改正程序。

3．运行“光盘\Code\Debug\14\03”文件夹下的 index.php 文件，填写内容后单击“提交”按钮，数据添加成功，但是显示中文乱码，如图 14.8 所示，请根据 addPost.php 文件中的注释改正程序。

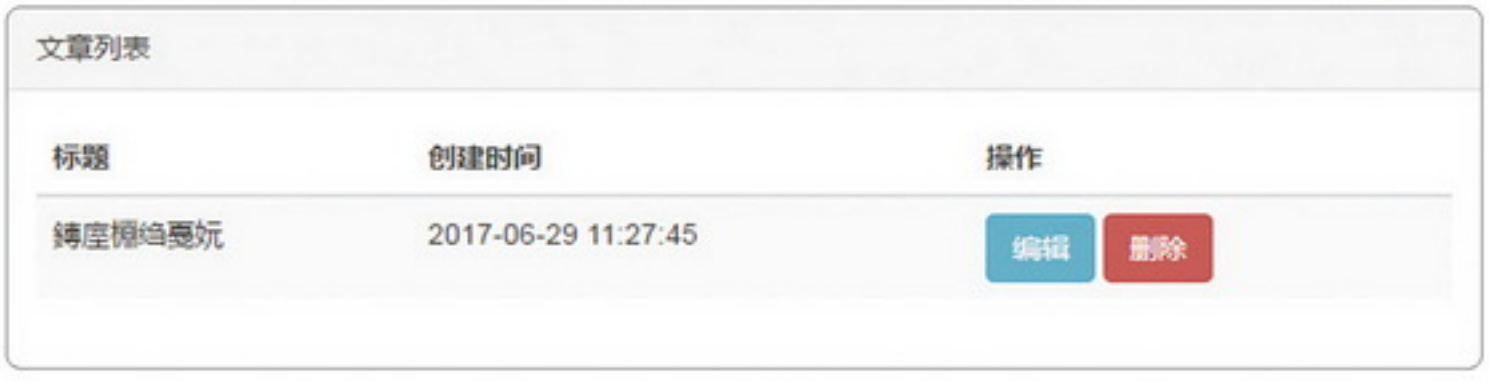


图 14.8　中文乱码

4．运行“光盘\Code\Debug\14\04”文件夹下的 index.php 文件，填写内容后单击“提交”按钮，数据添加成功，但是新添加的数据没有显示在列表顶部，而是显示在底部。如图 14.9 所示，请根据 index.html 文件中的注释，改正相应 JavaScript 代码。

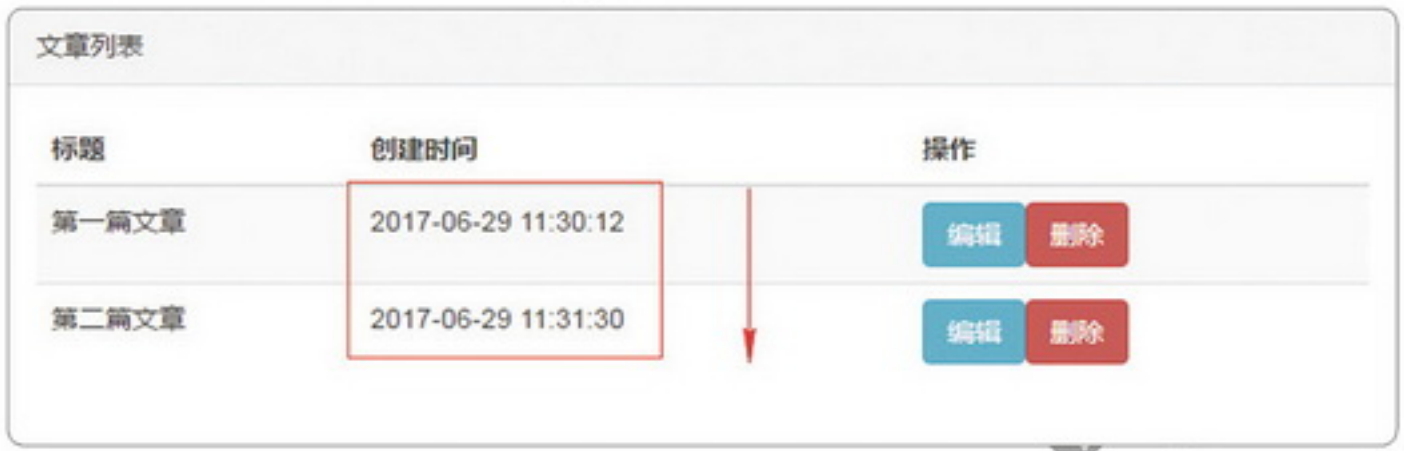
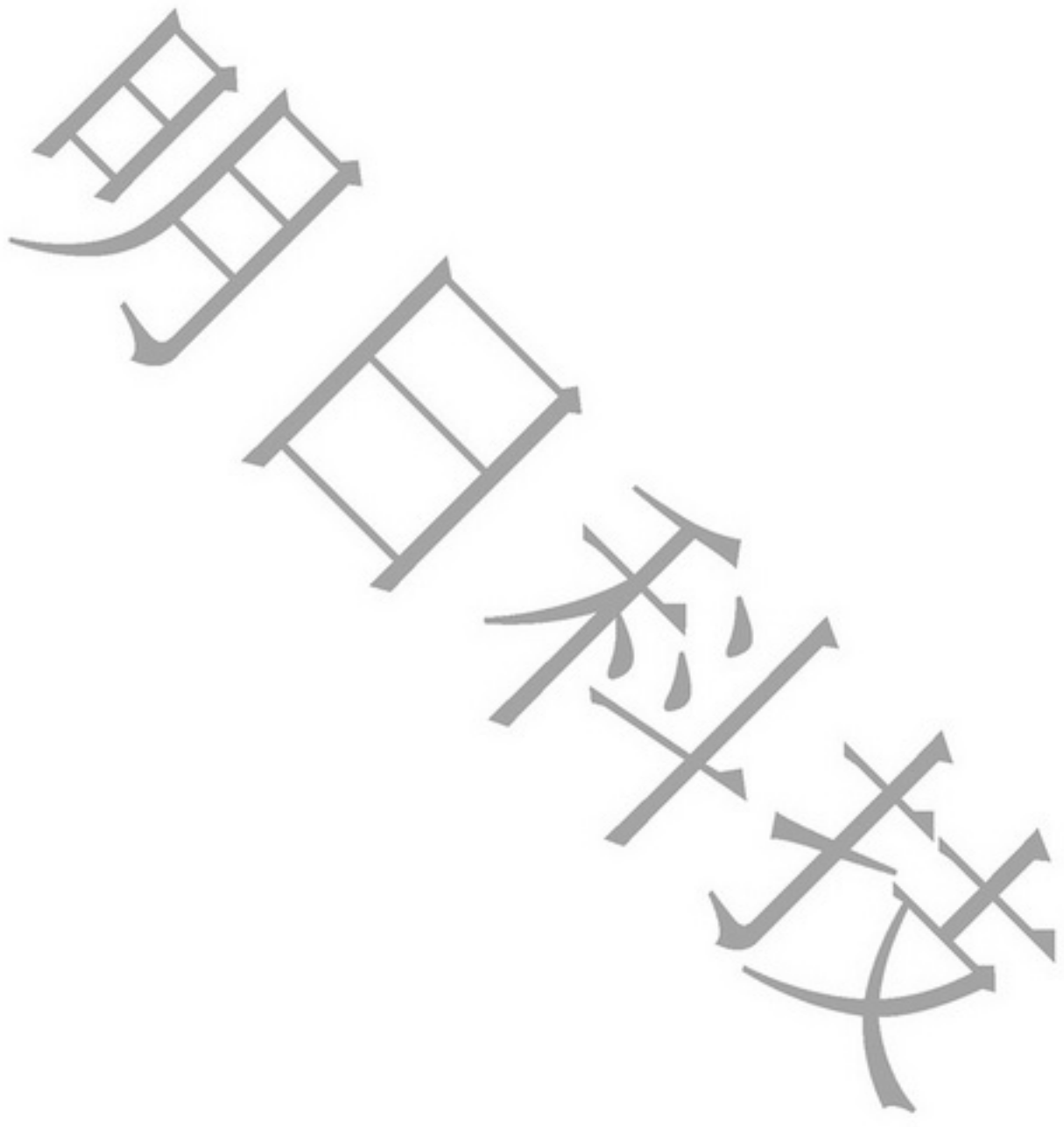


图 14.9　新增数据没有显示在顶部

5．运行“光盘\Code\Debug\14\05”文件夹下的程序：在浏览器中输入网址“localhost/Code/Debug/19/01/lists.php”，填写内容后单击“提交”按钮，数据添加成功，但是单击“删除”按钮，无法删除新增的数据，请根据 index.html 文件中的注释，改正相应 JavaScript 代码。



第 15 章

ThinkPHP 框架

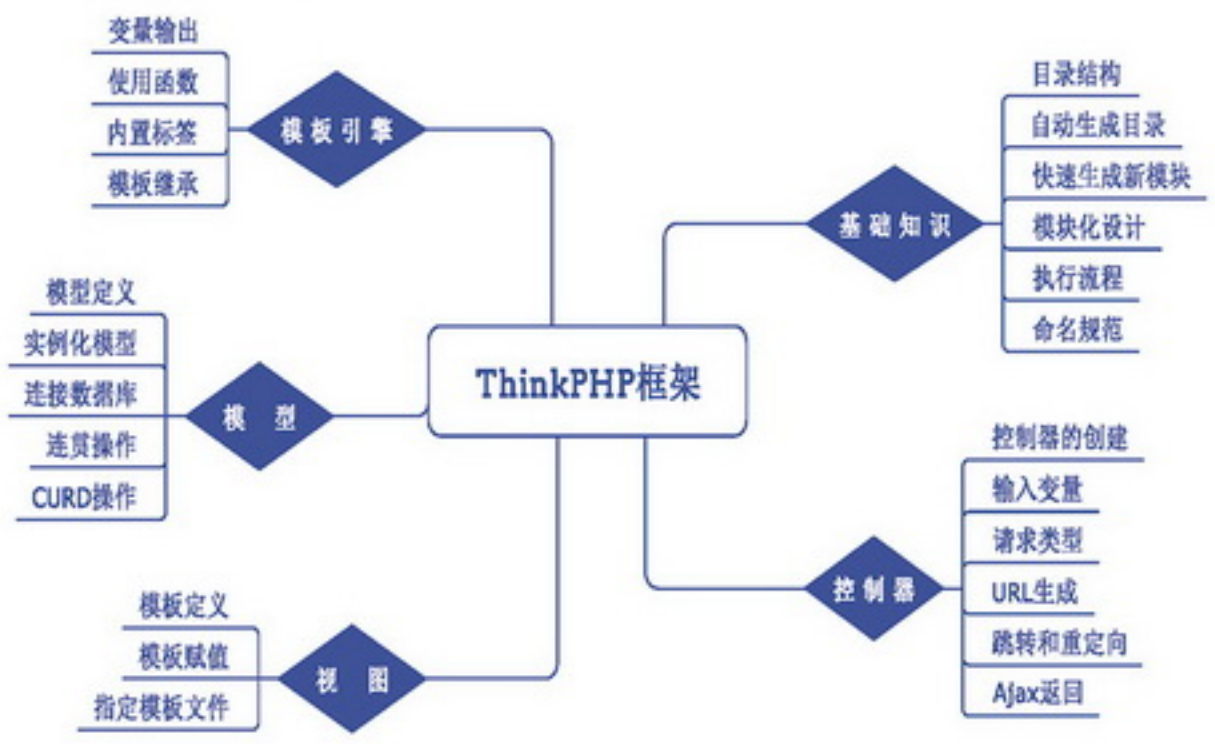
（视频讲解：1 小时 15 分）

本章概览

ThinkPHP 是一种性能卓越并且功能丰富的轻量级 PHP 开发框架，其宗旨就是让 Web 应用开发更简单、更快速。ThinkPHP 在易用性、扩展性和性能方面不断优化和改进，已经成长为国内最领先和最具影响力的 Web 应用开发框架，众多的典型案例确保其可以稳定用于商业以及门户级的开发当中。

本章不仅介绍了 ThinkPHP 框架的整体思想和架构体系，还详细讲解了 ThinkPHP 的配置、控制器、模型及视图等内容。通过本章内容的学习，读者将对 ThinkPHP 框架有深入的认识，在熟练掌握后能够将其应用在自己的项目开发中。

知识框架



15.1 ThinkPHP 简介

ThinkPHP 是一种免费开源的，快速、简单的面向对象的轻量级 PHP 开发框架，创立于 2006 年初，遵循 Apache 2 开源许可协议发布，是为了敏捷 Web 应用开发和简化企业应用开发而诞生的。ThinkPHP 从诞生以来一直秉承简洁实用的设计原则，在保持出色的性能和至简的代码的同时，既注重易用性，又拥有众多的原创功能和特性。在社区团队的积极参与下，在易用性、扩展性和性能方面不断优化和改进，目前已经成长为国内最领先和最具影响力的 Web 应用开发框架，众多的典型案例确保其可以稳定用于商业以及门户级的开发。

15.1.1 ThinkPHP 框架的特点

ThinkPHP 是一种性能卓越并且功能丰富的轻量级 PHP 开发框架。其宗旨就是让 Web 应用开发更简单、更快速。ThinkPHP 值得推荐的特性包括：

◆ MVC 支持，基于多层模型（M）、视图（V）、控制器（C）的设计模式。

◆ ORM 支持，提供了全功能和高性能的 ORM 支持，支持大部分数据库。

◆ 模板引擎支持，内置了高性能的基于标签库和 XML 标签的编译型模板引擎。

◆ RESTFul 支持，通过 REST 控制器扩展提供了 RESTFul 支持，提供全新的 URL 设计和访问体验。

◆ 云平台支持，提供了对新浪 SAE 平台和百度 BAE 平台的强力支持，具备“横跨性”和“平滑性”，支持本地化开发和调试以及部署切换，打造全新的开发体验。

◆ CLI 支持，支持基于命令行的应用开发。

◆ RPC 支持，提供包括 PHPRpc、HProse、jsonRPC 和 Yar 在内的远程调用解决方案。

◆ MongoDb 支持，提供 NoSQL 的支持。

◆ 缓存支持，提供了包括文件、数据库、Memcache、Xcache、Redis 等多种类型的缓存支持。

ThinkPHP 采用了 MVC 设计模式，此模式将应用程序分为 3 个部分：模型层（Model）、视图层（View）、控制层（Controller），MVC 是这 3 个部分英文字母的缩写。

在 PHP Web 开发中，MVC 设计模式的各自功能及相互关系如图 15.1 所示。

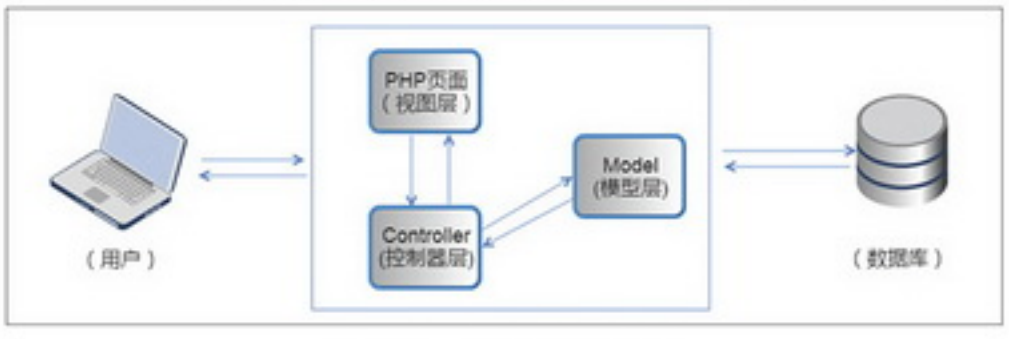


图 15.1　MVC 关系图

◆ 模型层（Model）

模型层是应用程序的核心部分，它可以是一个实体对象或一种业务逻辑，之所以称它为模型，是因为它在应用程序中有更好的重用性和扩展性。

◆ 视图层（View）

视图层提供应用程序与用户之间的交互界面，在 MVC 理论之中，这一层并不包含任何的业务逻辑，仅提供一种与用户交互的视图。

◆ 控制层（Controller）

控制层用于对程序中的请求进行控制，它可以选择调用哪些视图或者调用哪些模型。

15.1.2 环境要求

ThinkPHP 可以支持 Windows/Unix 服务器环境，可运行于包括 Apache、IIS 在内的多种 Web 服务器。需要 PHP 5.3 以上版本（注意：PHP 5.3dev 版本和 PHP 6 均不支持）及以上版本支持。同时它还支持 MySQL、MsSQL、PgSQL、Sqlite、Oracle 等数据库。

15.1.3 下载 ThinkPHP 框架

ThinkPHP 是一种免费开源、快捷、简单的轻量级 PHP 开发框架。下载地址为：http://www.thinkphp.cn/down.html。

📖 **说明：** 本章将以 ThinkPHP 3.2.3 为例来讲解 ThinkPHP 框架的使用，请读者下载 ThinkPHP 3.2.3 的完整版。

15.2 ThinkPHP 基础

ThinkPHP 遵循简洁实用的设计原则，在兼顾开发速度和执行速度的同时，也注重易用性。本节内容将对 ThinkPHP 框架的整体思想和架构体系进行详细说明。

15.2.1 目录结构

将下载的 thinkphp_3.2.3_full.zip 压缩包解压，重命名为 APP（名字只是为后面访问网址方便），将整个 APP 文件夹放在 D:\phpStudy\WWW 文件夹下（网站根目录）。解压后的文件结构如图 15.2 所示。

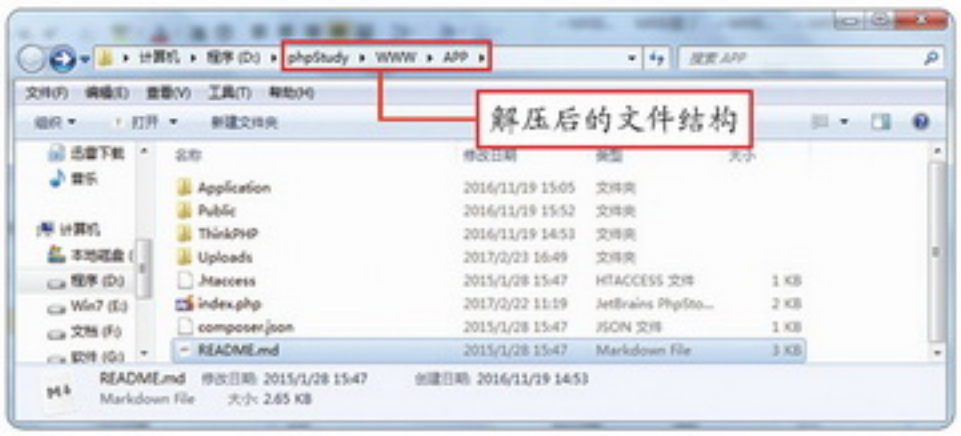


图 15.2　解压后的文件结构

使用 PhpStorm 打开项目的流程如图 15.3 所示。

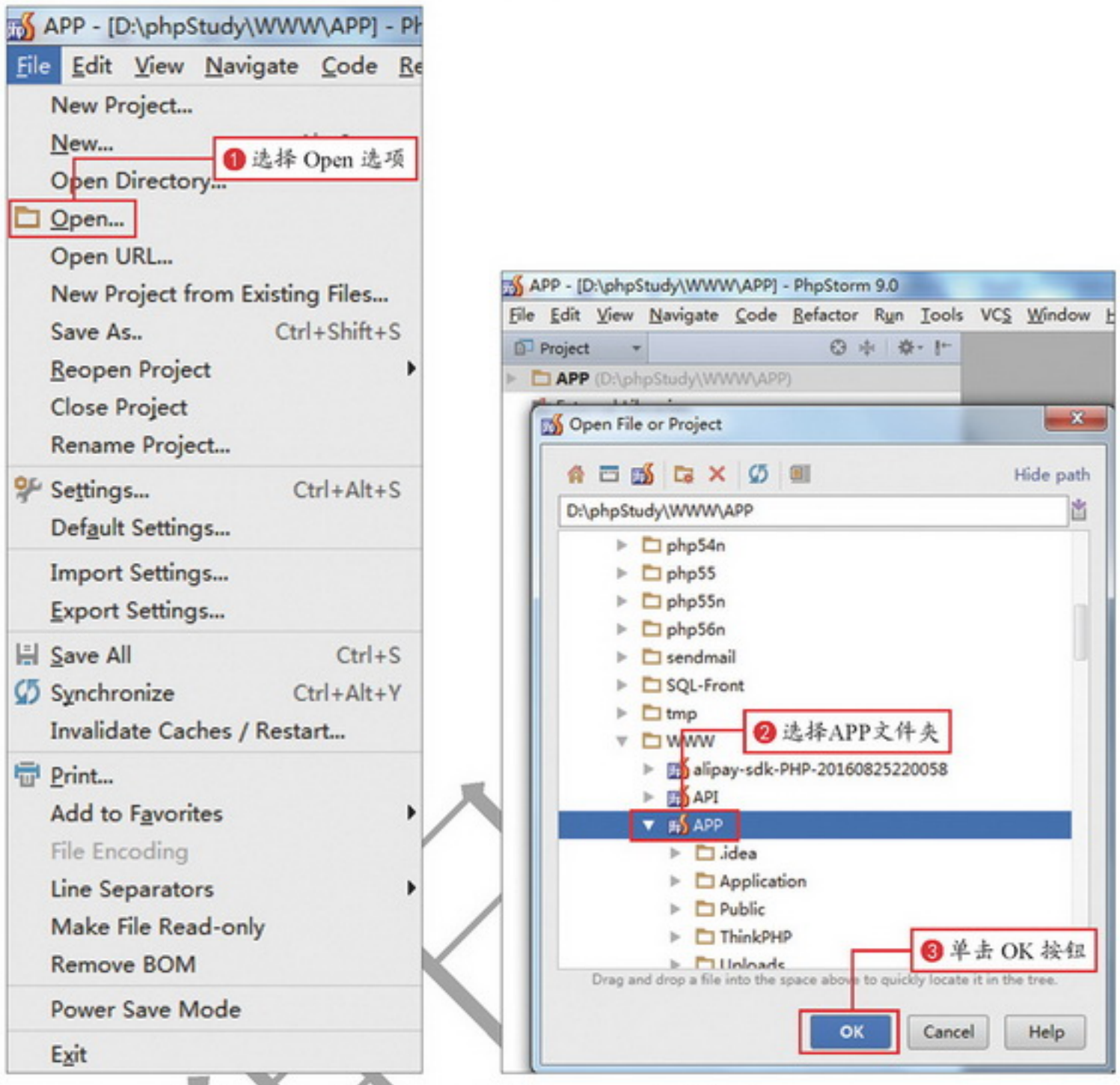


图 15.3　使用 PhpStorm 打开项目

此时，可以看到 APP 文件夹的目录结构如图 15.4 所示。

图 15.4　目录结构

15.2.2 自动生成目录

启动 phpStudy，选择“PHP 5.5 版本”，操作步骤如图 15.5 所示。

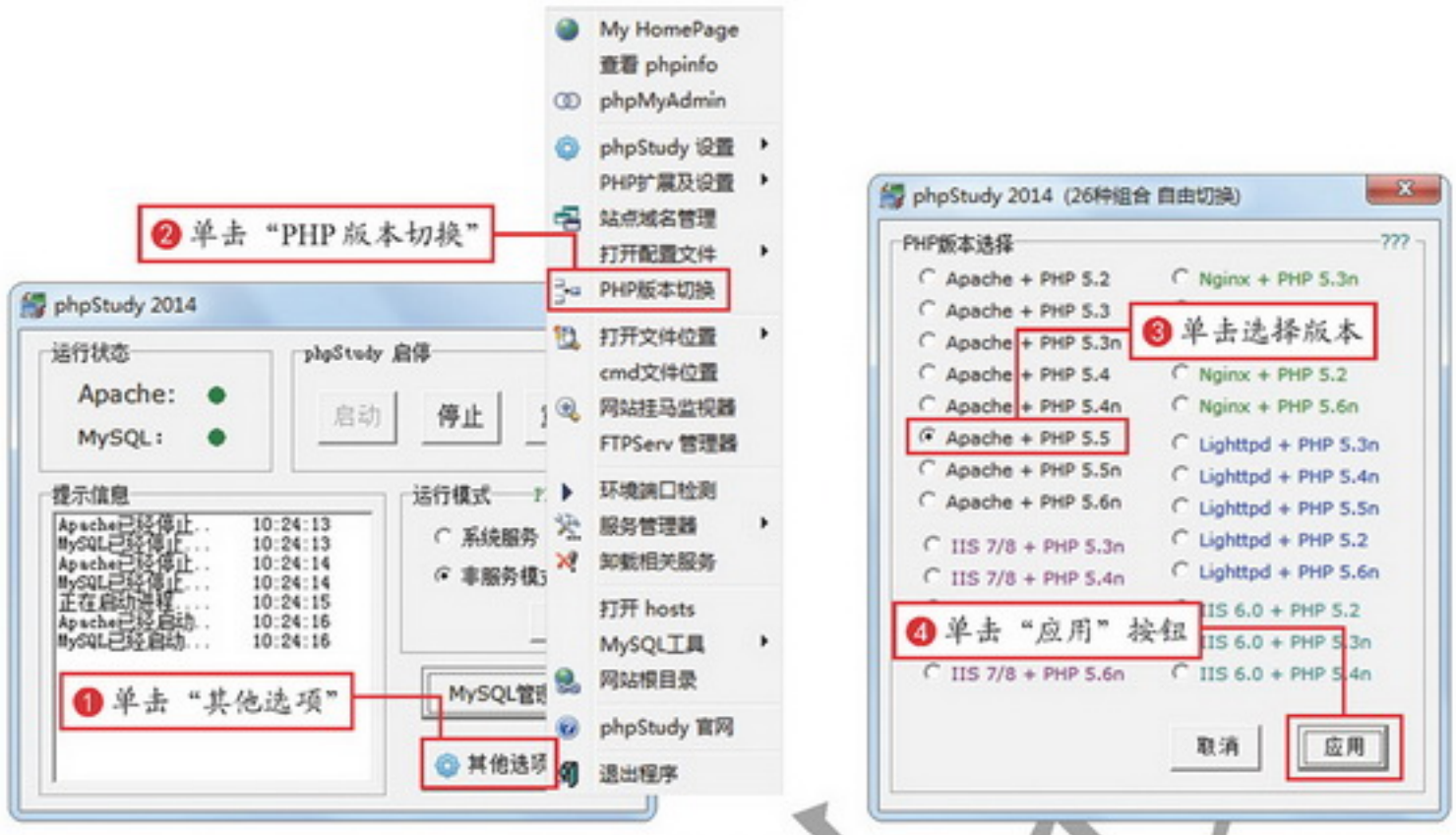


图 15.5　选择 PHP 版本

在浏览器地址栏中输入“localhost/APP/index.php”，按下 <Enter> 键后会在页面中显示“欢迎页”。运行结果如图 15.6 所示。

图 15.6　ThinkPHP 欢迎页面

再次查看 APP 文件夹的目录结构，系统已经在 Application 目录下面自动生成了公共模块 Common、默认模块 Home 和 Runtime 运行时目录，如图 15.7 所示。

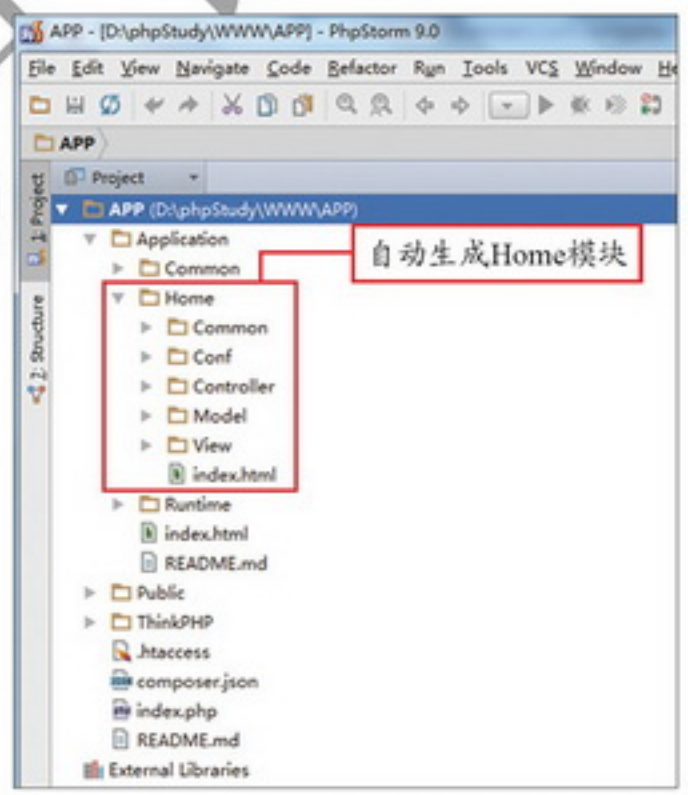


图 15.7　ThinkPHP 新增目录结构

📖 **说明：**在自动生成目录结构的同时，还在多个目录下看到了 index.html 文件，这是 ThinkPHP 自动生成的目录安全文件。为了避免某些服务器开启了目录浏览权限后可以直接在浏览器输入 URL 地址查看目录，系统默认开启了目录安全文件机制，会在自动生成目录的时候生成空白的 index.html 文件。当然安全文件的名称可以设置，也可以在入口文件里面关闭目录安全文件的生成。

15.2.3 快速生成新模块

由于采用多层的 MVC 机制，除了 Conf 和 Common 目录外，每个模块下面的目录结构可以根据需要灵活设置和添加，所以并不拘泥于上面展现的目录。

如果要添加新的模块（假设 Admin 模块），有没有快速生成模块目录结构的办法呢？ 自动生成的方式非常简单，只需要在入口文件 APP/index.php 中，新增下面指定代码即可，具体代码如下：

```
01  <?php
02  //应用入口文件
03  //检测PHP环境
04  if(version_compare(PHP_VERSION,'5.3.0','<'))  die('require PHP > 5.3.0 !');
05
06  //绑定入口文件到Admin模块访问
07  define('BIND_MODULE','Admin');
08
09  //开启调试模式，建议开发阶段开启，部署阶段注释或者设为false
10  define('APP_DEBUG',True);
11
12  //定义应用目录
13  define('APP_PATH','./Application/');
14
15  //引入ThinkPHP入口文件
16  require './ThinkPHP/ThinkPHP.php';
17
18  //亲^_^ 后面不需要任何代码了 就是如此简单
```

新增自动生成Admin模块的代码

BIND_MODULE 常量定义表示绑定入口文件到某个模块，由于并不存在 Admin 模块，所以会在第一次访问的时候自动生成。

在浏览器地址栏中输入“localhost/APP/index.php”，按下 <Enter> 键后，会再次看到欢迎页面。此时，项目目录结构发生变化。在 Application 文件夹下，已经自动生成了 Admin 模块及其目录结构。其目录结构如图 15.8 所示。

生成 Admin 模块以后，需要在 APP 文件夹下的入口文件 index.php 文件中，在如下代码中添加注释：

```
//define('BIND_MODULE','Admin');
```

修改后，可以正常访问 Home 模块，否则就只能访问 Admin 模块（因为应用入口中绑定了 Admin 模块）。

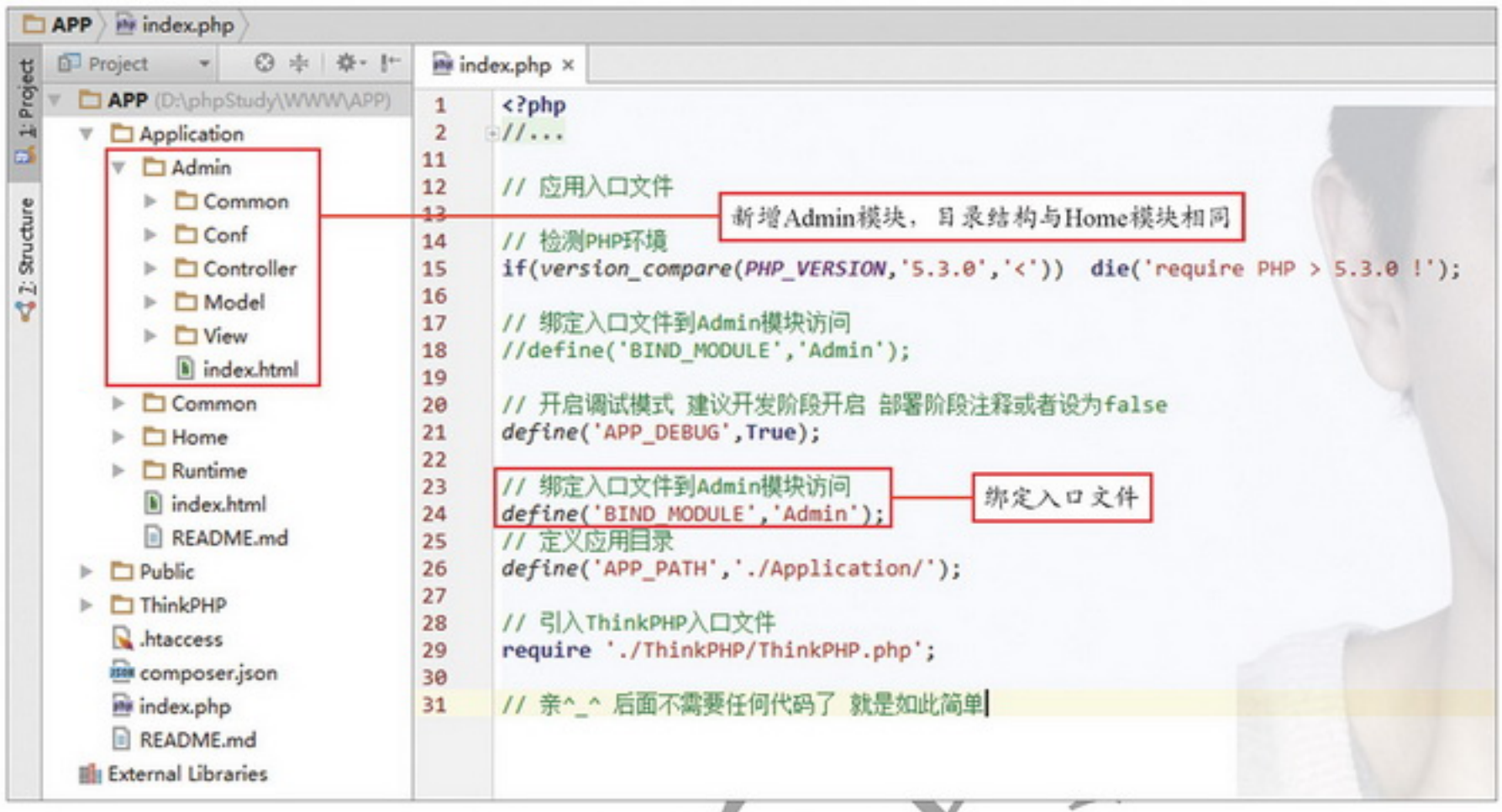


图 15.8　生成 Admin 模块后的目录结构

15.2.4 模块化设计

从图 15.7 和图 15.8 可以看出，Home 模块和 Admin 模块的目录结构相同，这是因为一个完整的 ThinkPHP 应用是基于模块 / 控制器 / 操作来设计的。在浏览器中输入网址“localhost/APP/index.php/Home/Index/index”，按下 <Enter> 键后，会再次看到欢迎页面。接下来以“/”作为分界，来分析一下这个链接地址：

◆ localhost：主机名。也可以更改为本机 IP 地址。

◆ APP：应用名称。可以自己命名，注意更改后，访问网址也要相应变化。

◆ index.php：项目的入口文件，在入口文件中定义应用目录和加载 ThinkPHP 框架，这是所有基于 ThinkPHP 开发应用的第一步。

◆ Home：模块名称。

◆ Index：控制器名称。

◆ index：方法名。使用驼峰法，首字母小写，如 firstName。

ThinkPHP 的模块 / 控制器 / 操作基本概念描述如表 15.1 所示。

表 15.1　基本概念描述

名　称	描　述
应用	基于同一个入口文件访问的项目称之为一个应用，即本项目中的 APP
模块	一个应用下面可以包含多个模块，每个模块在应用目录下面都是一个独立的子目录，如目录中的 Home

续表

名　称	描　述
控制器	每个模块可以包含多个控制器，一个控制器通常体现为一个控制器类，如 Home/Controller/IndexController.class.php
操作（也称作方法）	每个控制器类可以包含多个操作方法，也可能是绑定的某个操作类，每个操作是 URL 访问的最小单元。如：Home/Controller/IndexController.class.php 文件中的 index()

当在浏览器中输入网址“localhost/APP/index.php/Home/Index/index”，系统会自动执行 Home 模块的 Index 控制器下的 index 操作。

注意：网址中模块和控制器首字符都采用大写的方式，这是因为在 Linux 系统中区分大小写，如果是 Windows 系统则不区分。本项目中都采用区分大小写的方式。

15.2.5 执行流程

ThinkPHP 系统流程如下：

- 用户 URL 请求：用户在浏览器中输入网址，即发送 URL 请求。
- 调用应用入口文件，入口文件即根目录的 index.php 文件，路径：APP\index.php。
- 载入框架入口文件（ThinkPHP.php），路径：APP\ThinkPHP\ThinkPHP.php。
- 加载 ThinkPHP 框架内部，具体加载内容可参考《ThinkPHP 手册》。
- 获取请求的模块信息。
- 获取当前控制器和操作，以及 URL 其他参数。
- 根据请求执行控制器方法。
- 如果控制器中调用 display 或 show 方法，则说明有模版渲染。
- 获取模版内容。
- 自动识别当前主题以及定位模版文件。

当在浏览器中输入“http://localhost/APP/index.php/Home/Index/index”，系统获取到请求的模块是 Home，当前控制器是 Index，控制器方法是 index，然后会执行该方法，如果有模板渲染，就获取模板内容。

15.2.6 命名规范

ThinkPHP 框架有其自身的一定规范，要应用 ThinkPHP 框架开发项目，就要尽量遵守它的规范。下面详细介绍一下 ThinkPHP 的命名规范：

- 类文件都是以 .class.php 为后缀（这里指的是 ThinkPHP 内部使用的类库文件，不代表外部加载的类库文件），使用驼峰法命名，并且首字母大写，例如 DbMysql.class.php。
- 类的命名空间地址和所在的路径地址一致，例如 Home\Controller\UserController 类所在的路径应该是 Application\Home\Controller\UserController.class.php。
- 确保文件的命名和调用大小写一致，由于在类似 Unix 系统中，对大小写是敏感的（而 ThinkPHP 在调试模式下面，即使在 Windows 平台也会严格检查大小写格式）。

◆ 类名和文件名一致（包括上面说的大小写一致），例如 UserController 类的文件命名是 UserController.class.php，InfoModel 类的文件名是 InfoModel.class.php，不同的类库的类命名也有一定的规范。

◆ 函数、配置文件等其他类库文件之外的一般是以 .php 为后缀（第三方引入的不做要求）。

◆ 函数的命名使用小写字母和下划线的方式，例如 get_client_ip。

◆ 方法的命名使用驼峰法，并且首字母小写或者使用下划线“_”，例如 getUserName，_parseType，通常下划线开头的方法属于私有方法。

◆ 属性的命名使用驼峰法，并且首字母小写或者使用下划线“_”，例如 tableName、_instance，通常下划线开头的属性属于私有属性。

◆ 以双下划线“__”打头的函数或方法作为魔法方法，例如 __call 和 __autoload。

◆ 常量以大写字母和下划线命名，例如 HAS_ONE 和 MANY_TO_MANY。

◆ 配置参数以大写字母和下划线命名，例如 HTML_CACHE_ON。

◆ 语言变量以大写字母和下划线命名，例如 MY_LANG，以下划线打头的语言变量通常用于系统语言变量，例如 _CLASS_NOT_EXIST_。

◆ 对变量的命名没有强制的规范，可以根据团队规范来进行。

◆ ThinkPHP 的模板文件默认是以 .html 为后缀（可以通过配置修改）。

◆ 数据表和字段采用小写加下划线方式命名，并注意字段名不要以下划线开头，例如 think_user 表和 user_name 字段是正确写法，类似 _username 这样的数据表字段可能会被过滤。

15.3　ThinkPHP 的配置

配置文件是 ThinkPHP 框架程序得以运行的基础条件，框架的很多功能都需要在配置文件中配置之后，才可以生效。包括：URL 路由功能、页面伪静态和静态化等等。ThinkPHP 提供了灵活的全局配置功能，采用最有效率的 PHP 返回数组方式定义，支持惯例配置、项目配置、调试配置和模块配置，并且会自动生成配置缓存文件，无需重复解析。

ThinkPHP 在项目配置上面创造了自己独有的分层配置模式，其配置层次如图 15.9 所示。

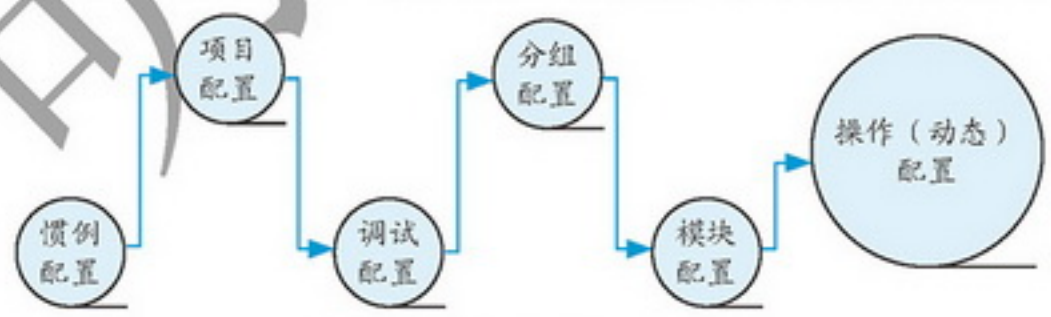


图 15.9　分层配置模式的顺序

以上是配置文件的加载顺序，但是因为后面的配置会覆盖之前的配置（在没有生效的前提下），所以遵循从右到左的优先顺序。系统的配置参数是通过静态变量全局存取的，存取方式简单、高效。

15.3.1　配置格式

ThinkPHP 框架中所有配置文件的定义格式均采用返回 PHP 数组的方式，格式为：

```php
<?php
return array(
    'APP_DEBUG' => true,
    'URL_MODEL' => 2,
    //更多的配置参数
);
?>
```

📖 **说明：**配置参数不区分大小写（因为无论使用大写还是小写定义，都会转换成小写）。但是习惯上保持大写定义的原则。另外，还可以在配置文件中使用二维数组来配置更多的信息。例如：

```php
<?php
return array(
    'APP_DEBUG' => true,
    'USER_CONFIG' => array(
        'USER_AUTH' => true,
        'USER_TYPE' => 2,
    ),
);
?>
```

⚡ **注意：**需要注意的是，二级参数配置区分大小写，要确保读取的和定义的格式相一致。

多学两招：项目配置指的是项目的全局配置，因为一个项目除了可以定义项目配置文件之外，还可以定义模块配置文件用于针对某个特定的模块进行特殊的配置。他们的定义格式都是一致的，区别只是配置文件命名的不同。系统会自动在不同的阶段读取配置文件。这里使用 .html 作为模板文件的后缀，因为 HTML 网页在互联网中更容易被搜索引擎搜索到。

15.3.2 调试配置

ThinkPHP 支持调试模式，默认情况下是运行在部署模式下面。在部署模式下，会以性能优先并且尽可能少地抛出错误信息。而调试模式则以除错方便优先，关闭所有缓存，而且尽可能多地抛出错误信息，所以对性能有一定的影响。

部署模式采用了项目编译机制，第一次运行会对核心和项目相关文件进行编译缓存，由于编译后会影响开发过程中对配置文件、函数文件和数据库修改的生效（除非修改后手动清空 Runtime 下面的缓存文件）。因此为了避免以上问题，强烈建议新手在使用 ThinkPHP 开发的过程中使用调试模式，这样可以更好地获取错误提示和避免一些不必要的问题。

开启和关闭调试的方法非常简单，在 APP\index.php 入口文件中，设置 APP_DEBUG 为 True，即可开启调试，代码如下：

```php
define('APP_DEBUG',True);                    //开启调试
```

设置 APP_DEBUG 为 False 即可关闭调试，代码如下：

```php
define('APP_DEBUG',False);                   //关闭调试
```

15.4 ThinkPHP 的控制器

15.4.1 控制器的创建

一般来说，ThinkPHP 的控制器是一个类，而操作则是控制器类的一个公共方法。创建控制器需要为每个控制器定义一个控制器类，控制器类的命名规范是：控制器名 +Controller.class.php（采用驼峰法并且首字母大写）。

例如，为后台 Admin 模块下创建一个 Test 控制器，在控制器中创建 test1 和 test2 方法，操作步骤如下：

（1）创建控制器。使用 PhpStorm 在 APP/Application/Admin/Controller 文件夹下创建 TestController.class.php 文件，操作过程如图 15.10、图 15.11 和图 15.12 所示。

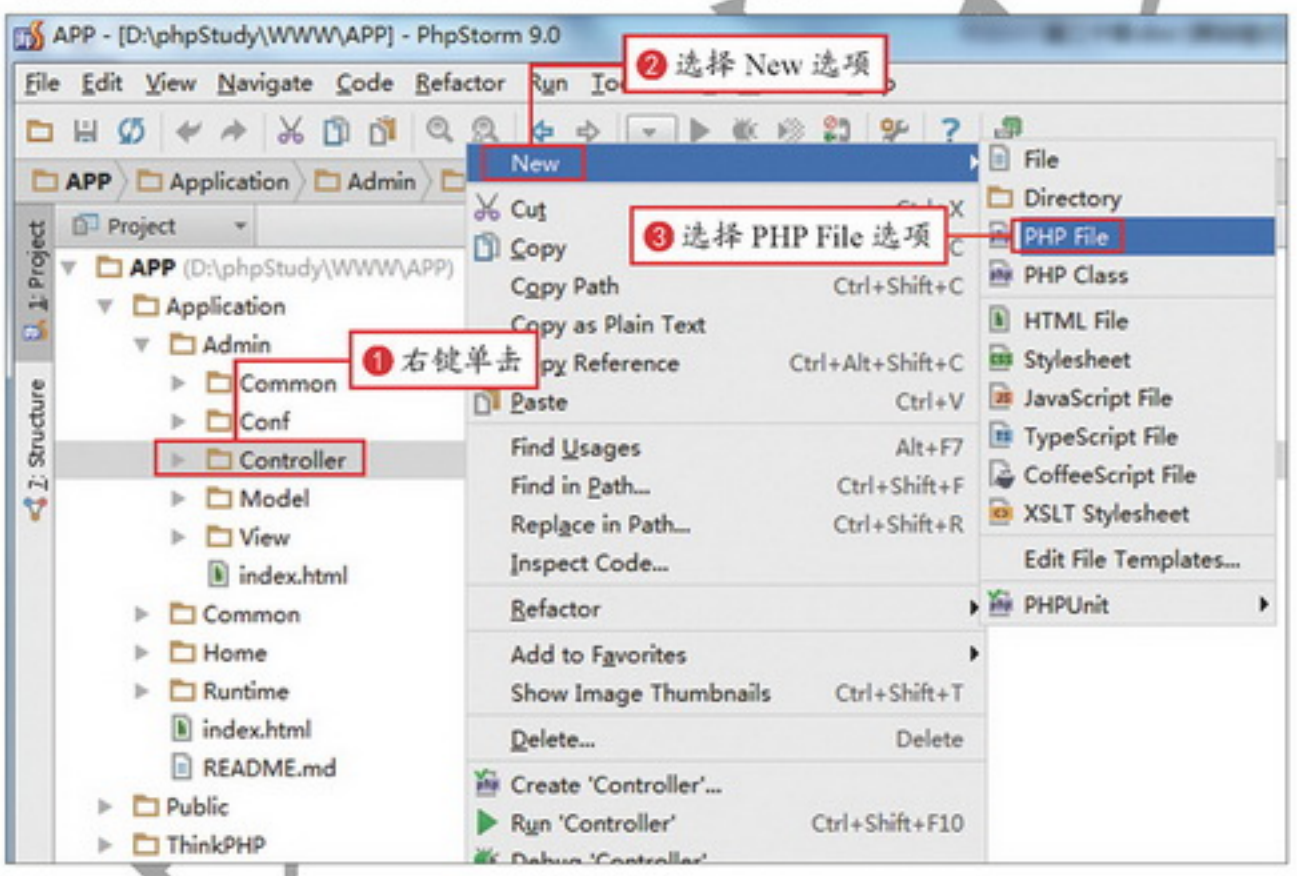


图 15.10　使用 PhpStorm 创建 PHP 文件

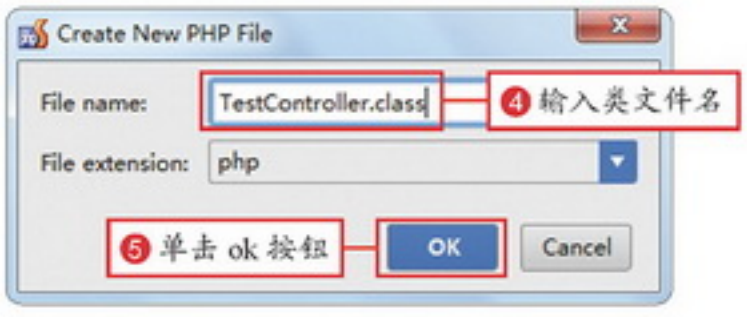


图 15.11　输入类文件名

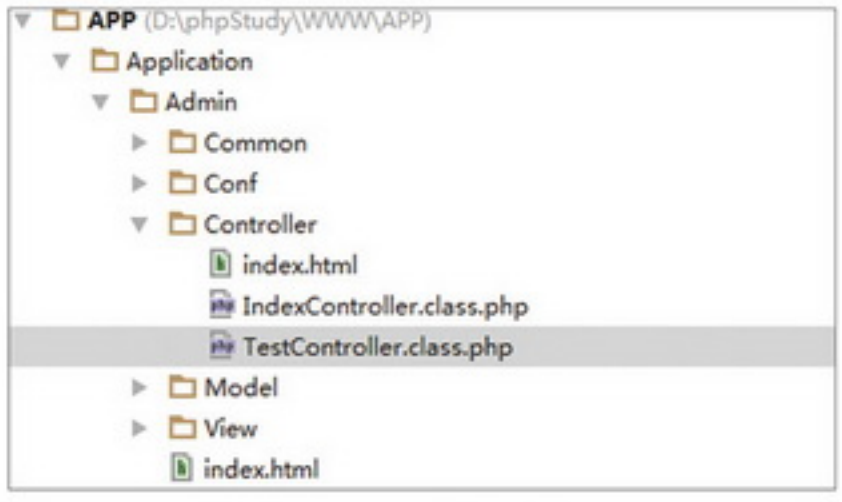


图 15.12　创建后的目录

TestController.class.php 文件具体代码如下：

```
01  <?php
02  namespace Admin\Controller;                    //命名空间
03  use Think\Controller;                          //命名空间引用
04
05  class TestController extends Controller {
06
07  }
```

注意：“<?php”要写在第一行，否则会出现错误信息：Namespace declaration statement has to be the very first statement in the script。

（2）创建方法。在 TestController.class.php 文件中创建两个测试方法。具体代码如下：

```
01  <?php
02  namespace Admin\Controller;                    //命名空间
03  use Think\Controller;                          //命名空间引用
04
05  class TestController extends Controller {
06
07      public function test1(){
08          echo "我是测试一";
09      }
10      public function test2(){
11          echo "我是测试二";
12      }
13
14  }
```

在浏览器地址栏中输入“localhost/APP/index.php/Admin/Test/test1”，按下 <Enter> 键后，运行结果如图 15.13 所示。

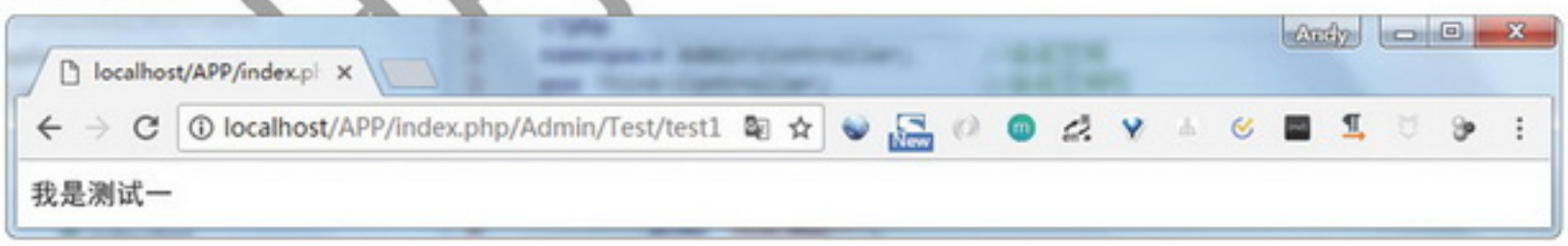


图 15.13　test1 方法

在浏览器地址栏中输入“localhost/APP/index.php/Admin/Test/test2”，按下 <Enter> 键后，运行结果如图 15.14 所示。

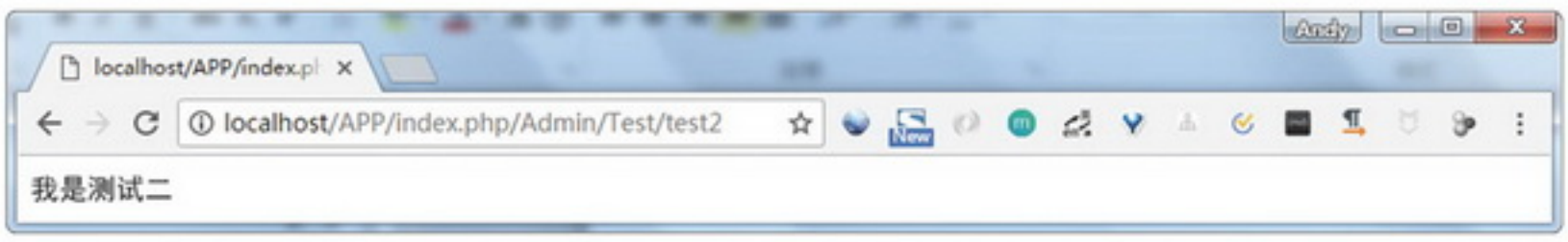


图 15.14　test2 方法

15.4.2 输入变量

ThinkPHP 可以使用 I 方法更加方便和安全地获取系统输入变量，语法格式如下：

```
I('变量类型.变量名/修饰符',['默认值'],['过滤方法'],['额外数据源'])
```

变量类型是指请求方式或者输入类型，如表 15.2 所示。

表 15.2　变量类型

名　称	含　义
get	获取 GET 参数
post	获取 POST 参数
request	获取 REQUEST 参数
session	获取 $_SESSION 参数
cookie	获取 $_COOKIE 参数
server	获取 $_SERVER 参数
path	获取 PATHINFO 模式的 URL 参数（ThinkPHP 3.2.2 新增）

注意：变量类型不区分大小写。变量名则严格区分大小写。默认值和过滤方法均属于可选参数。

以 GET 变量类型为例，说明一下 I 方法的使用：

```
echo I('get.id');                          //相当于$_GET['id']
echo I('get.name');                        //相当于$_GET['name']
```

支持默认值：

```
echo I('get.id',0);                        //如果不存在$_GET['id']，则返回0
echo I('get.name','');                     //如果不存在$_GET['name']，则返回空字符串
```

采用方法过滤：

```
//采用htmlspecialchars方法对$_GET['name'] 进行过滤，如果不存在，则返回空字符串
echo I('get.name','','htmlspecialchars');
```

支持直接获取整个变量类型，例如：

```
I('get.');                                 //获取整个$_GET数组
```

用同样的方式，可以获取 post 或者其他输入类型的变量，例如：

```
//采用htmlspecialchars方法对$_POST['name'] 进行过滤，如果不存在，则返回空字符串
I('post.name','','htmlspecialchars');
I('session.user_id',0);                    //获取$_SESSION['user_id']，如果不存在，则默认为0
I('cookie.');                              //获取整个$_COOKIE数组
I('server.REQUEST_METHOD');                //获取$_SERVER['REQUEST_METHOD']
```

15.4.3 请求类型

在很多情况下，需要判断当前操作的请求类型是 GET、POST、PUT 或 DELETE，一方面可以针对请求类型做出不同的逻辑处理，另外一方面有些情况下需要验证安全性，过滤掉不安全的请求。 系统内置了一些常量用于判断请求类型，如表 15.3 所示。

表 15.3　用于判断请求类型的常量

常　量	说　明
IS_GET	判断是否是 GET 方式提交
IS_POST	判断是否是 POST 方式提交
IS_PUT	判断是否是 PUT 方式提交
IS_DELETE	判断是否是 DELETE 方式提交
IS_AJAX	判断是否是 AJAX 提交
REQUEST_METHOD	当前提交类型

例如，判断是否是POST提交，如果是POST提交，保存数据，否则会提示“非法请求”。代码如下：

```
01  public function update(){
02      if (IS_POST){
03          $User = M('User');
04          $User->create();
05          $User->save();
06          $this->success('保存完成');
07      }else{
08          $this->error('非法请求');
09      }
10  }
```

注意：如果使用的是 ThinkAjax 或者自己写的 Ajax 类库的话，需要在表单里面添加一个隐藏域，告诉后台属于 Ajax 方式提交，默认的隐藏域名称是 Ajax（可以通过 VAR_AJAX_SUBMIT 配置），如果是 jQuery 类库的话，则无需添加任何隐藏域即可自动判断。

15.4.4 URL 生成

为了配合所使用的URL 模式，需要能够动态地根据当前的URL 设置生成对应的URL 地址，为此，ThinkPHP 内置提供了 U 方法，用于 URL 的动态生成，并且可以确保项目在移植过程中不受环境的影响。

U 方法的定义规则如下（方括号内参数根据实际应用决定）：

```
U('地址表达式',['参数'],['伪静态后缀'],['显示域名'])
```

U 方法的参数说明如下：

◆ 地址表达式

地址表达式的格式定义如下：

```
[模块/控制器/操作#锚点@域名]?参数1=值1&参数2=值2…
```

如果不定义模块的话，就表示当前模块名称，下面是一些简单的例子：

```
U('User/add')                    //生成User控制器的add操作的URL地址
U('Blog/read?id=1')              //生成Blog控制器的read操作，并且id为1的URL地址
U('Admin/User/select')           //生成Admin模块的User控制器的select操作的URL地址
```

◆ 参数

U 方法的第二个参数支持数组和字符串两种定义方式，如果只是字符串方式的参数可以在第一个参数中定义，例如：

```
U('Blog/cate',array('cate_id'=>1,'status'=>1))
U('Blog/cate','cate_id=1&status=1')
U('Blog/cate?cate_id=1&status=1')
```

以上三种方式是等效的，都是生成 Blog 控制器 cate 操作的 URL 地址，并且传递参数 cate_id 和 status 以及参数值。但是不允许使用下面的定义方式来传递参数：

```
U('Blog/cate/cate_id/1/status/1');
```

◆ 伪静态后缀

U 方法会自动识别当前配置的伪静态后缀，如果需要指定后缀生成 URL 地址的话，可以显式传入，例如：

```
U('Blog/cate','cate_id=1&status=1','xml');
```

15.4.5 跳转和重定向

1. 页面跳转

在开发应用中，经常会遇到一些带有提示信息的跳转页面，例如操作成功或者操作错误页面，并且自动跳转到另外一个目标页面。系统的 \Think\Controller 类内置了两个跳转方法 success() 和 error()，不仅可以用于页面跳转提示，而且可以支持 Ajax 提交，例如：

```
01  public function add(){
02      $User = M('User');              //实例化User对象
03      $result = $User->add($data);
04      if($result){
05          //设置成功后跳转页面的地址，默认的返回页面是$_SERVER['HTTP_REFERER']
06          $this->success('新增成功', 'User/list');
07      } else {
```

```
08            //错误页面的默认跳转页面是返回前一页，通常不需要设置
09            $this->error('新增失败');
10        }
11    }
```

success() 方法和 error() 方法的第一个参数表示提示信息，第二个参数表示跳转地址，第三个参数是跳转时间（单位为秒），例如：

```
//操作完成3秒后跳转到/Article/index
$this->success('操作完成','/Article/index',3);
//操作失败5秒后跳转到/Article/error
$this->error('操作失败','/Article/error',5);
```

跳转地址是可选的，success() 方法的默认跳转地址是 $_SERVER["HTTP_REFERER"]，error() 方法的默认跳转地址是javascript:history.back(-1);。success() 方法默认的等待时间是 1 秒，error() 方法是 3 秒。success() 方法和 error() 方法都有对应的模板，默认的设置中两个方法对应的模板都是：

```
//默认错误跳转对应的模板文件
'TMPL_ACTION_ERROR' => THINK_PATH . 'Tpl/dispatch_jump.tpl',
//默认成功跳转对应的模板文件
'TMPL_ACTION_SUCCESS' => THINK_PATH . 'Tpl/dispatch_jump.tpl',
```

也可以使用项目内部的模板文件：

```
//默认错误跳转对应的模板文件
'TMPL_ACTION_ERROR' => 'Public:error';
//默认成功跳转对应的模板文件
'TMPL_ACTION_SUCCESS' => 'Public:success';
```

success() 方法和 error() 方法会自动判断当前请求是否属于 Ajax 请求，如果属于 Ajax 请求，则会调用 ajaxReturn() 方法返回信息。 对于 Ajax 方式的请求，success() 方法和 error() 方法会封装下面的数据返回：

```
$data['info']   =   $message;          //提示信息内容
$data['status'] =   $status;           //如果success返回1，如果error返回0
$data['url']    =   $jumpUrl;          //成功或者错误的跳转地址
```

2. 重定向

Controller 类的 redirect() 方法可以实现页面的重定向功能。redirect() 方法的参数用法和 U 函数的用法一致（参考 URL 生成部分），例如：

```
//重定向到New模块的Category操作
$this->redirect('New/category', array('cate_id' => 2), 5, '页面跳转中...');
```

上面的用法是停留 5 秒后跳转到 New 模块的 category 操作，并且显示页面跳转中字样，重定向后会改变当前的 URL 地址。如果仅仅是想重定向到一个指定的 URL 地址，而不是到某个模块的操作方法，也可以直接使用 redirect() 函数重定向，例如：

```
//重定向到指定的URL地址
redirect('/New/category/cate_id/2', 5, '页面跳转中...')
```

redirect 函数的第一个参数是一个 URL 地址。

📖 **说明**：控制器的 redirect() 方法和 redirect() 函数的区别在于前者是用 URL 规则定义跳转地址，后者是一个纯粹的 URL 地址。

15.4.6 Ajax 返回

ThinkPHP 可以很好地支持 Ajax 请求，系统的\Think\Controller 类提供了 ajaxReturn() 方法用于 Ajax 调用后返回数据给客户端。并且支持 JSON、JSONP、XML 和 EVAL 四种方式给客户端接收数据，并且支持配置其他方式的数据格式返回。

ajaxReturn() 方法调用示例：

```
$data = 'ok';
$this->ajaxReturn($data);
```

支持返回数组数据：

```
$data['status']  = 1;
$data['content'] = 'content';
$this->ajaxReturn($data);
```

默认配置采用 JSON 格式返回数据（通过配置 DEFAULT_AJAX_RETURN 进行设置），可以指定格式返回，例如：

```
//指定XML格式返回数据
$data['status']  = 1;
$data['content'] = 'content';
$this->ajaxReturn($data,'xml');
```

返回数据 data 可以支持字符串、数字、数组和对象，返回客户端的时候根据不同的返回格式进行编码后传输。如果是 JSON/JSONP 格式，会自动编码成 JSON 字符串，如果是 XML 方式，会自动编码成 XML 字符串，如果是 EVAL 方式的话，则只会输出字符串 data 数据。

📖 **说明**：JSON 是一种数据交换格式，而 JSONP 是一种非官方跨域数据交互协议。一个是描述信息的格式，一个是信息传递的约定方法。

默认的 JSONP 格式返回的处理方法是 jsonpReturn()，如果采用不同的方法，可以设置如下：

```
'DEFAULT_JSONP_HANDLER' =>  'myJsonpReturn',          //设置JSONP格式返回的处理方法
```

或者直接在页面中使用 callback 参数来指定。

📖 **说明**：除了上面四种返回类型外，还可以通过行为扩展来增加其他类型的支持，只需要对 ajax_return 标签位进行行为绑定即可。

15.5 ThinkPHP 的模型

顾名思义，模型就是按照某一个形状进行操作的代名词。模型的主要作用是封装数据库的相关逻辑。也就是说，每执行一次数据库操作，都要遵循定义的数据模型规则来完成。

15.5.1 模型定义

模型类通常需要继承系统的 \Think\Model 类或其子类，下面是一个 Home\Model\UserModel 类的定义：

```
namespace Home\Model;
use Think\Model;
class UserModel extends Model {
}
```

模型类主要用于操作数据表，如果按照系统的规范来命名模型类的话，大多数情况下可以自动对应数据表。模型类的命名规则是除去表前缀的数据表名称，采用驼峰法命名，并且首字母大写，然后加上模型层的名称（默认定义是 Model）。例如，定义一个 User 模型，则模型名称为 UserModel，约定对应数据表（假设数据表的前缀定义是 think_）为 think_user。

📖 **说明：**如果实际规则和上面的系统约定不符合，则需要设置 Model 类的数据表名称属性，以确保能够找到对应的数据表。

在 ThinkPHP 的模型里面，有几个关于数据表名称的属性定义，如表 15.4 所示。

表 15.4　Model 类的数据表名称属性

属　性	说　明
tablePrefix	定义模型对应数据表的前缀，如果未定义则获取配置文件中的 DB_PREFIX 参数
tableName	不包含表前缀的数据表名称，一般情况下默认和模型名称相同，只有当实际表名和当前的模型类的名称不同的时候才需要定义
trueTableName	包含前缀的数据表名称，也就是数据库中的实际表名，该名称无需设置，只有当上面的规则都不适用的情况或者特殊情况下才需要设置
dbName	定义模型当前对应的数据库名称，只有当前的模型类对应的数据库名称和配置文件不同的时候才需要定义

例如，在数据库里面有一个 think_categories 表，而实际定义的模型类名称是 CategoryModel，按照系统的约定，这个模型的名称是 Category，对应的数据表名称应该是 think_category（全部小写），但是现在的数据表名称是 think_categories，此时就需要设置 tableName 属性来改变默认的规则（假设已经在配置文件里面定义了 DB_PREFIX 为 think_），具体代码如下：

```
01  namespace Home\Model;
02  use Think\Model;
03  class CategoryModel extends Model {
04      protected $tableName = 'categories';
05  }
```

注意：tableName 属性的定义不需要加表的前缀 think_。

15.5.2 实例化模型

在 ThinkPHP 中无需进行任何模型定义。只有在需要封装单独的业务逻辑时，模型类才是必须被定义的，因此 ThinkPHP 在模型有很强的灵活性和方便性。

根据不同的模型定义，有以下几种实例化模型的方法，可以根据需要采用不同的方式：

1. 直接实例化

可以和实例化其他类库一样实例化模型类，例如：

```
$User = new \Home\Model\UserModel();
$Info = new \Admin\Model\InfoModel();
$New  = new \Home\Model\NewModel('blog','think_',$connection);      //带参数实例化
```

模型类通常都是继承系统的 \Think\Model 类，大多数情况下，根本无需传入任何参数即可实例化。

2. D 方法实例化

直接实例化的时候需要传入完整的类名，系统提供了一个快捷方法 D 用于数据模型的实例化操作。使用 D 方法实例化自定义模型类，可以使用下面的方式：

```
<?php
$User = D('User');                                          //实例化模型
//相当于 $User = new \Home\Model\UserModel();
//执行具体的数据操作
$User->select();
```

说明：当 \Home\Model\UserModel 类不存在的时候，D 方法会尝试实例化公共模块下面的 \Common\Model\UserModel 类。如果在 Linux 环境下面，一定要注意 D 方法实例化时模型名称的大小写。

D 方法还可以支持跨模块调用，如当前模块为 Home 模块，则实例化 Admin 模块的 User 模型代码如下：

```
D('Admin/User');
```

3. M 方法实例化

D 方法实例化模型类的时候通常是实例化某个具体的模型类，如果仅仅是对数据表进行基本的 CURD 操作的话，使用不需要加载具体模型类的 M 方法，性能会更高。例如：

```
$User = M('User');
//和用法$User = new \Think\Model('User'); 等效
//执行其他的数据操作
$User->select();
```

M 方法也可以支持跨库操作，例如：

```
$User = M('db_name.User','ot_');                    //使用M方法实例化，操作db_name数据库的ot_user表
$User->select();                                    //执行其他的数据操作
```

4. 实例化空模型

如果仅仅是使用原生 SQL 查询的话，则不需要使用额外的模型类，实例化一个空模型类即可进行操作，例如：

```
//实例化空模型
$Model = new Model();
//或者使用M快捷方法是等效的
$Model = M();
$Model->query('SELECT * FROM think_user WHERE status = 1'); //进行原生的SQL查询
```

实例化空模型类后还可以用 table() 方法切换到具体的数据表进行操作。

📖 **说明：** 在实例化的过程中，经常使用 D 方法和 M 方法，这两个方法的区别在于 M 方法实例化模型无需用户为每个数据表定义模型类，如果 D 方法没有找到定义的模型类，则会自动调用 M 方法。

15.5.3 连接数据库

ThinkPHP 内置了抽象数据库访问层，把不同的数据库操作封装起来，我们只需要使用公共的 Db 类进行操作，而无需针对不同的数据库写不同的代码和底层实现，Db 类会自动调用相应的数据库驱动来处理。目前的数据库包括 Mysql、SqlServer、PgSQL、Sqlite、Oracle、Ibase、Mongo，也包括对 PDO 的支持。

如果应用需要使用数据库，必须配置数据库连接信息，数据库的配置文件有多种定义方式，通常使用全局配置定义方式。在 APP\Application\Common\Conf\config.php 文件中配置数据库信息，代码如下：

```
<?php
return array(
    //'配置项'=>'配置值'
    'DB_TYPE' => 'mysql',                   //数据库类型
    'DB_HOST' => '127.0.0.1',               //host地址，也可以填写localhost
    'DB_USER' => 'root',                    //数据库用户名
    'DB_PWD' =>  'root',                    //数据库密码
    'DB_NAME' => 'test',                    //数据库名
```

```
09      'DB_PREFIX' => 'mr_',                    //表前缀
10  );
```

15.5.4 连贯操作

ThinkPHP 模型基础类提供的连贯操作方法（也有些框架称之为链式操作），可以有效地提高数据存取的代码清晰度和开发效率，并且还支持所有的 CURD 操作。

例如，现在要查询一个 User 表的满足状态为 1 的前 10 条记录，并希望按照用户的创建时间排序，代码如下：

```
$User->where('status=1')->order('create_time')->limit(10)->select();
```

上述代码中的 where、order 和 limit 方法就被称之为连贯操作方法（select() 方法必须放到最后，因为select方法并不是连贯操作方法），连贯操作方法的调用顺序没有先后，下面的代码和上面的等效：

```
$User->order('create_time')->limit(10)->where('status=1')->select();
```

如果不习惯使用连贯操作的话，还可以直接使用参数进行查询。例如上面的代码可以改写为：

```
$User->select(array('order'=>'create_time','where'=>'status=1','limit'=>'10'));
```

如果使用数组参数方式，索引的名称就是连贯操作的方法名称。其实不仅仅是查询方法可以使用连贯操作，所有的 CURD 方法都可以使用，例如：

```
$User->where('id=1')->field('id,name,email')->find();
$User->where('status=1 and id=1')->delete();
```

系统支持的连贯操作方法如表 15.5 所示。

表 15.5　系统支持的连贯操作方法

字　段　名	默认值或绑定	描　　述
where*	用于查询或者更新条件的定义	字符串、数组和对象
table	用于定义要操作的数据表名称	字符串和数组
alias	用于给当前数据表定义别名	字符串
data	用于新增或者更新数据之前的数据对象赋值	数组和对象
field	用于定义要查询的字段（支持字段排除）	字符串和数组
order	用于对结果排序	字符串和数组
limit	用于限制查询结果数量	字符串和数字
page	用于查询分页（内部会转换成 limit）	字符串和数字
group	用于对查询的 group 支持	字符串

续表

字 段 名	默认值或绑定	描 述
having	用于对查询的 having 支持	字符串
join*	用于对查询的 join 支持	字符串和数组
union*	用于对查询的 union 支持	字符串、数组和对象
distinct	用于对查询的 distinct 支持	布尔值
relation	用于关联查询（需要关联模型支持）	字符串
validate	用于数据自动验证	数组

说明：所有的连贯操作都返回当前的模型实例对象（this），其中带 * 标识的表示支持多次调用。

15.5.5 CURD 操作

ThinkPHP 提供了灵活、方便的数据操作方法，CURD 的创建、更新、读取和删除是四种最基本的数据库操作。CURD 操作通常与连贯操作配合使用。下面将对各种操作的使用方法进行分析（在执行类的实例化操作时，统一使用 M 方法）。

1. 数据创建

在进行数据操作之前，往往需要手动创建需要的数据，例如对于提交的表单数据：

```
//获取表单的POST数据
$data['name'] = $_POST['name'];
$data['email'] = $_POST['email'];
```

ThinkPHP 可以快速地创建数据对象，最典型的应用就是自动根据表单数据创建数据对象，这个优势在数据表的字段非常多的情况下尤其明显。例如：

```
//实例化User模型
$User = M('User');
//根据表单提交的POST数据创建数据对象
$User->create();
```

create 方法支持从其他方式创建数据对象，例如，从其他的数据对象或者数组等，例如：

```
$data['name'] = 'ThinkPHP';
$data['email'] = 'ThinkPHP@gmail.com';
$User->create($data);
```

甚至还可以支持从对象创建新的数据对象，例如：

```
//从User数据对象创建新的Member数据对象
$User = stdClass();
```

```
$User->name = 'ThinkPHP';
$User->email = 'ThinkPHP@gmail.com';
$Member = M("Member");
$Member->create($User);
```

创建完成的数据可以直接读取和修改，例如：

```
$data['name'] = 'ThinkPHP';
$data['email'] = 'ThinkPHP@gmail.com';
$User->create($data);
//创建完成数据对象后可以直接读取数据
echo $User->name;
echo $User->email;
//也可以直接修改创建完成的数据
$User->name = 'onethink';                    //修改name字段数据
$User->status = 1;                           //增加新的字段数据
```

2. 数据写入

ThinkPHP 的数据写入操作使用 add() 方法，例如：

```
$User = M("User");                           //实例化User对象
$data['name']  = 'ThinkPHP';
$data['email'] = 'ThinkPHP@gmail.com';
$User->add($data);
```

如果在 add() 方法之前已经创建了数据对象（例如使用了 create() 方法或者 data() 方法），add() 方法就不需要再传入数据了。例如：

```
$User = M("User");                           //实例化User对象
//根据表单提交的POST数据创建数据对象
if($User->create()){
    $result = $User->add();                  //写入数据到数据库
    if($result){
        //如果主键是自动增长型 成功后返回值就是最新插入的值
        $insertId = $result;
    }
}
```

说明：create() 方法并不算是连贯操作，因为其返回值可能是布尔值，所以必须要进行严格判断。

3. 数据读取

在 ThinkPHP 中读取数据的方式很多，通常分为读取数据、读取数据集和读取字段值。数据读取方法支持连贯操作。

◆ 读取数据

读取数据是指读取数据表中的一行数据（或者关联数据），主要通过 find() 方法完成，例如：

```
$User = M("User");                                    //实例化User对象
//查找status值为1，name值为thinkphp的用户数据
$data = $User->where('status=1 AND name="thinkphp"')->find();
dump($data);
```

find() 方法查询数据的时候可以配合相关的连贯操作方法，其中最关键的则是 where() 方法，如果查询出错，find() 方法则返回 false，如果查询结果为空则返回 NULL，查询成功则返回一个关联数组（键值是字段名或者别名）。如果上面的查询成功的话，输出结果如下：

```
array (size=3)
  'name' => string 'thinkphp' (length=8)
  'email' => string 'thinkphp@gmail.com' (length=18)
  'status'=> int 1
```

📖 **说明**：即使满足条件的数据不止一个，find() 方法也只会返回第一条记录（可以通过 order() 方法排序后查询）。

◆ 读取数据集

读取数据集其实就是获取数据表中的多行记录（以及关联数据），使用 select() 方法，例如：

```
$User = M("User");                                    //实例化User对象
//查找status值为1的用户数据，以创建时间排序，返回10条数据
$list = $User->where('status=1')->order('create_time')->limit(10)->select();
```

如果查询出错，select 的返回值是 false，如果查询结果为空，则返回 NULL，否则返回二维数组。

◆ 读取字段值

读取字段值其实就是获取数据表中的某个列的多个或者单个数据，最常用的方法是 getField() 方法。例如：

```
$User = M("User");                                    //实例化User对象
//获取ID为3的用户的昵称
$nickname = $User->where('id=3')->getField('nickname');
```

默认情况下，当只有一个字段的时候，返回满足条件的数据表中的该字段第一行的值。如果需要返回整个列的数据，可以使用如下代码：

```
$User->getField('id',true);                           //获取id数组
//返回数据格式如array(1,2,3,4,5)一维数组，其中value就是id列每行的值
```

如果传入多个字段的话，默认返回一个关联数组，代码如下：

```
$User = M("User");                                    //实例化User对象
//获取所有用户的id和昵称列表
$list = $User->getField('id,nickname');
//两个字段的情况下返回的是array(`id`=>`nickname`)的关联数组，以id的值为key，nickname字段值为value
```

那么返回的是一个数组 list，键名是用户的 id 字段的值，键值是用户的昵称 nickname。如果传入多个字段的名称，例如：

```
$list = $User->getField('id,nickname,email');
//返回的数组格式是array(`id`=>array(`id`=>value,`nickname`=>value,`email`=>value))是一个二维数组，key还是id字段的值，但value是整行的array数组，类似于select()方法的结果遍历将id的值设为数组key
```

那么返回的是一个二维数组，类似 select() 方法的返回结果，不同的是这个二维数组的键名是用户的 id（准确的说是 getField 方法的第一个字段名）。如果传入一个字符串分隔符，代码如下：

```
$list = $User->getField('id,nickname,email',':');
```

那么返回的结果就是一个数组，键名是用户 id，键值是 nickname:email 的输出字符串。getField() 方法还可以支持限制数量，例如：

```
$this->getField('id,name',5);                    //限制返回5条记录
$this->getField('id',3);                         //获取id数组，限制3条记录
```

4. 数据更新

ThinkPHP 的数据更新操作包括更新数据和更新字段方法。更新数据使用 save() 方法，例如：

```
01  $User = M("User");                           //实例化User对象
02  //要修改的数据对象属性赋值
03  $data['name'] = 'ThinkPHP';
04  $data['email'] = 'ThinkPHP@gmail.com';
05  $User->where('id=5')->save($data);           //根据条件更新记录
```

也可以改成对象方式来操作，例如：

```
01  $User = M("User");                           //实例化User对象
02  //要修改的数据对象属性赋值
03  $User->name = 'ThinkPHP';
04  $User->email = 'ThinkPHP@gmail.com';
05  $User->where('id=5')->save();                //根据条件更新记录
```

数据对象赋值的方式，save() 方法无需传入数据，会自动识别。

注意： save() 方法的返回值是影响的记录数，如果返回 false 则表示更新出错，因此一定要用恒等来判断是否更新失败。

为了保证数据库的安全、避免出错而更新整个数据表时，在没有任何更新条件并且数据对象本身也不包含主键字段的情况下，save() 方法不会更新任何数据库的记录。因此下面的代码不会更改数据库的任何记录：

```
$User->save($data);
```

除非使用下面的方式：

```
01  $User = M("User");                           //实例化User对象
02  //要修改的数据对象属性赋值
```

```
$data['id'] = 5;
$data['name'] = 'ThinkPHP';
$data['email'] = 'ThinkPHP@gmail.com';
$User->save($data);                                    //根据条件保存修改的数据
```

如果 id 是数据表的主键，系统会自动把主键的值作为更新条件来更新其他字段的值。数据更新方法也支持连贯操作方法。

◆ 更新字段

如果只是更新个别字段的值，可以使用 setField() 方法。例如：

```
$User = M("User");                                     //实例化User对象
//更改用户的name值
$User-> where('id=5')->setField('name','ThinkPHP');
```

setField() 方法支持同时更新多个字段，只需要传入数组即可，例如：

```
$User = M("User");                                     //实例化User对象
//更改用户的name和email的值
$data = array('name'=>'ThinkPHP','email'=>'ThinkPHP@gmail.com');
$User-> where('id=5')->setField($data);
```

而对于统计字段（通常指的是数字类型）的更新，系统还提供了 setInc() 方法和 setDec() 方法。例如：

```
$User = M("User");                                     //实例化User对象
$User->where('id=5')->setInc('score',3);               //用户的积分加3
$User->where('id=5')->setInc('score');                 //用户的积分加1
$User->where('id=5')->setDec('score',5);               //用户的积分减5
$User->where('id=5')->setDec('score');                 //用户的积分减1
```

setInc() 方法和 setDec() 方法支持延迟更新，用法如下：

```
$Article = M("Article");                               //实例化Article对象
$Article->where('id=5')->setInc('view',1);             //文章阅读数加1
$Article->where('id=5')->setInc('view',1,60);          //文章阅读数加1，并且延迟60秒更新（写入）
```

5. 数据删除

ThinkPHP 删除数据使用 delete() 方法，例如：

```
$Form = M('Form');
$Form->delete(5);
```

表示删除主键为 5 的数据。delete() 方法既可以删除单个数据，也可以删除多个数据，这取决于删除条件，例如：

```
$User = M("User");                                     //实例化User对象
$User->where('id=5')->delete();                        //删除id为5的用户数据
$User->delete('1,2,5');                                //删除主键为1,2和5的用户数据
```

```
$User->where('status=0')->delete();                    //删除所有状态为0的用户数据
```

delete 方法的返回值是删除的记录数，如果返回值是 false，则表示 SQL 出错，返回值如果为 0，则表示没有删除任何数据。也可以用 order() 方法和 limit() 方法来限制要删除的个数，例如：

```
//删除所有状态为0的5个用户数据，按照创建时间排序
$User->where('status=0')->order('create_time')->limit('5')->delete();
```

为了避免错删数据，如果没有传入任何条件进行删除操作的话，则不会执行删除操作，例如：

```
$User = M("User");                                     //实例化User对象
$User->delete();
```

如果确实要删除所有的记录，则可以使用下面的方式：

```
$User = M("User");                                     //实例化User对象
$User->where('1')->delete();
```

数据删除方法也支持连贯操作。

15.6 ThinkPHP 的视图

在 ThinkPHP 里面，视图由两个部分组成：View 类和模板文件。Controller 控制器直接与 View 视图类进行交互，把要输出的数据通过模板变量赋值的方式传递到视图类，而具体的输出工作则交由 View 视图类来进行，同时视图类还完成了一些辅助的工作，包括调用模板引擎、布局渲染、输出替换、页面 Trace 等功能。为了方便使用，在 Controller 类中封装了 View 类的一些输出方法，例如 display()、fetch()、assign()、trace() 和 buildHtml() 等方法，这些方法的原型都在 View 视图类里面。

15.6.1 模板定义

每个模块的模板文件都是独立的，为了对模板文件进行更加有效的管理，ThinkPHP 对模板文件进行目录划分，默认的模板文件定义规则是：

```
视图目录/[模板主题/]控制器名/操作名+模板后缀
```

默认的视图目录是模块的 View 目录（模块可以有多个视图文件目录，这取决于应用需要），框架的默认视图文件后缀是 .html。 新版模板主题默认是空（表示不启用模板主题功能）。

在每个模板主题下，都是以模块下面的控制器名为目录，然后是每个控制器的具体操作模板文件，例如，User 控制器的 add 操作对应的模板文件就应该是 .\Application\Home\View\User\add.html，如果默认视图层不是 View，例如：

```
'DEFAULT_V_LAYER'       =>  'Template',                //设置默认的视图层名称
```

那么，对应的模板文件就变成了 .\Application\Home\Template\User\add.html。模板文件的默认后缀

是 .html，也可以通过“TMPL_TEMPLATE_SUFFIX”参数更改为其他的后缀名。例如：

```
'TMPL_TEMPLATE_SUFFIX'=>'.tpl'
```

定义后，User 控制器的 add 操作对应的模板文件就变成“.\Application\Home\View\User\add.tpl”。如果觉得目录结构太深，可以通过设置“TMPL_FILE_DEPR”参数来配置简化模板的目录层次，例如设置：

```
'TMPL_FILE_DEPR'=>'_'
```

默认的模板文件就变成了“.\Application\Home\View\User_add.html”。

15.6.2 模板赋值

如果要在模板中输出变量，必须在控制器中把变量传递给模板，系统提供了 assign() 方法对模板变量赋值，无论何种变量类型都统一使用 assign 赋值，例如：

```
$this->assign('name',$value);
//下面的写法是等效的
$this->name = $value;
```

assign() 方法必须在 display() 方法和 show() 方法之前调用，并且系统只会输出设定的变量，其他变量不会输出（系统变量例外），一定程度上保证了变量的安全性。赋值后，就可以在模板文件中输出变量了，如果使用的是内置模板的话，可以这样输出“{$name}”。

如果要同时输出多个模板变量，可以使用下面的方式：

```
$array['name']    =    'thinkphp';
$array['email']   =    'liu21st@gmail.com';
$array['phone']   =    '12335678';
$this->assign($array);
```

这样就可以在模板文件中同时输出 name、email 和 phone 三个变量。模板变量的输出根据不同的模板引擎有不同的方法，后面会专门讲解内置模板引擎的用法。如果使用 PHP 本身作为模板引擎的话，就可以直接在模板文件里面输出，代码如下：

```
<?php echo $name.'['.$email.''.$phone.']';?>
```

如果采用内置的模板引擎，可以使用如下方式输出同样的内容：

```
{$name} [ {$email} {$phone} ]
```

15.6.3 指定模板文件

模板定义后就可以渲染模板输出，系统也支持直接渲染内容输出，模板赋值必须在模板渲染之前操作。渲染模板输出最常用的是使用 display() 方法，调用格式：

```
display('[模板文件]'[,'字符编码'][,'输出类型'])
```

display 模板文件的用法如表 15.6 所示。

表 15.6　display 模板文件的用法

用　法	描　述
不带任何参数	自动定位当前操作的模板文件
[模块 @][控制器 :][操作]	常用写法，支持跨模块模板主题，可以和 theme() 方法配合
完整的模板文件名	直接使用完整的模板文件名（包括模板后缀）

下面是一个最典型的用法，不带任何参数：

```
//不带任何参数 自动定位当前操作的模板文件
$this->display();
```

不带任何参数表示系统会按照默认规则自动定位模板文件，其规则是：

（1）如果当前没有启用模板主题则定位到：

```
当前模块/默认视图目录/当前控制器/当前操作.html
```

（2）如果有启用模板主题则定位到：

```
当前模块/默认视图目录/当前主题/当前控制器/当前操作.html
```

（3）如果有更改 TMPL_FILE_DEPR 设置（假设 'TMPL_FILE_DEPR'=>'_'）的话，则上面的自动定位规则变成：

```
当前模块/默认视图目录/当前控制器_当前操作.html
```

和

```
当前模块/默认视图目录/当前主题/当前控制器_当前操作.html
```

所以通常 display() 方法无需带任何参数即可输出对应的模板，是模板输出的最简单的用法。

📖 **说明**：通常默认的视图目录是 View。

如果没有按照模板定义规则来定义模板文件（或者需要调用其他控制器下面的某个模板），可以使用：

```
//指定模板输出
$this->display('edit');
```

调用当前控制器下面的 edit 模板使用：

```
$this->display('Member:read');
```

表示调用 Member 控制器下面的 read 模板。

例如，为后台 Admin 模块的 Test 控制器的 test1 方法创建模板文件。创建过程如下所示：

（1）创建目录。test1 方法对应的模板文件默认是 APP\Application\Admin\View\Test\test1.html，在 APP\Application\Admin\View 文件夹下创建 Test 文件夹，创建方法如图 15.15 所示。

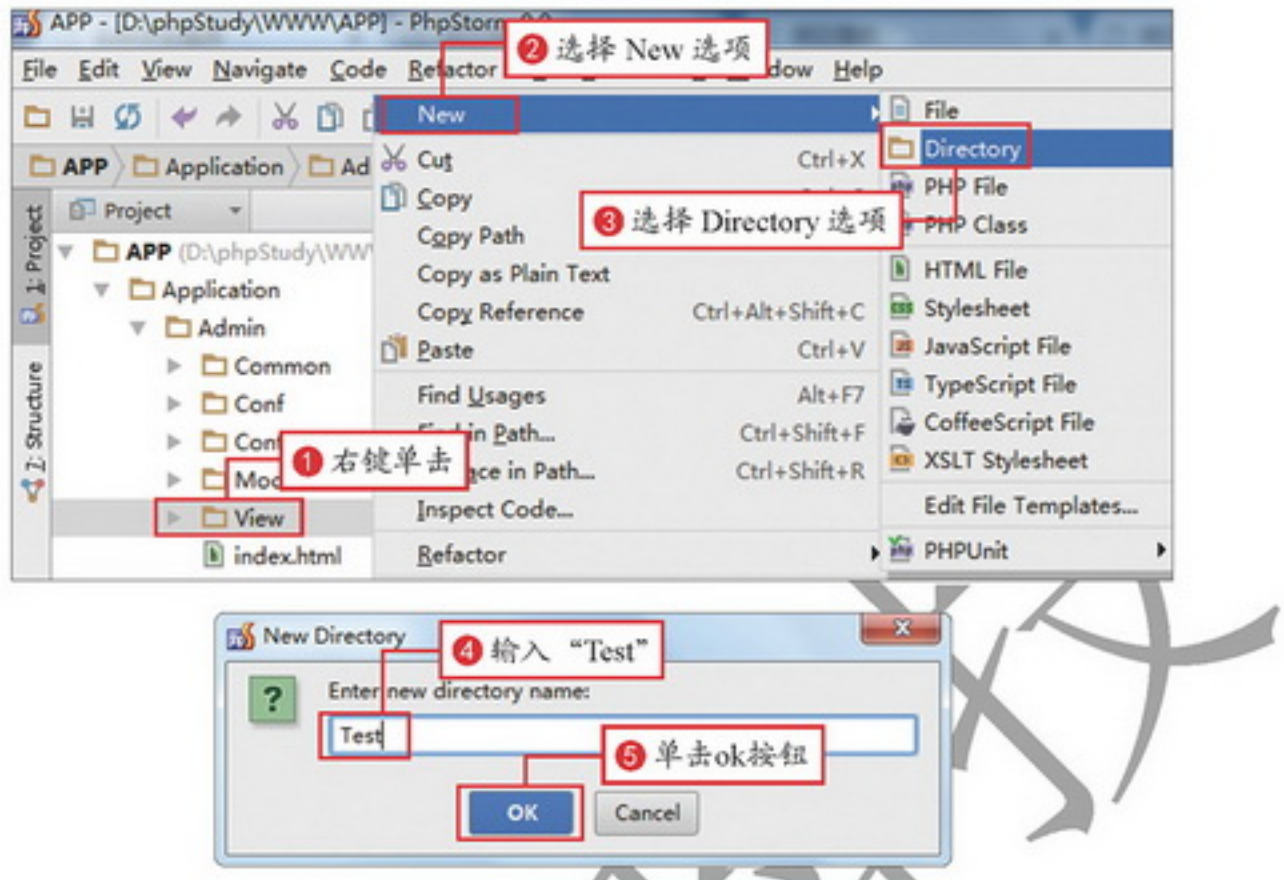


图 15.15　创建目录

（2）创建视图文件。在 Test 文件夹下创建 test1.html 文件，如图 15.16 所示。

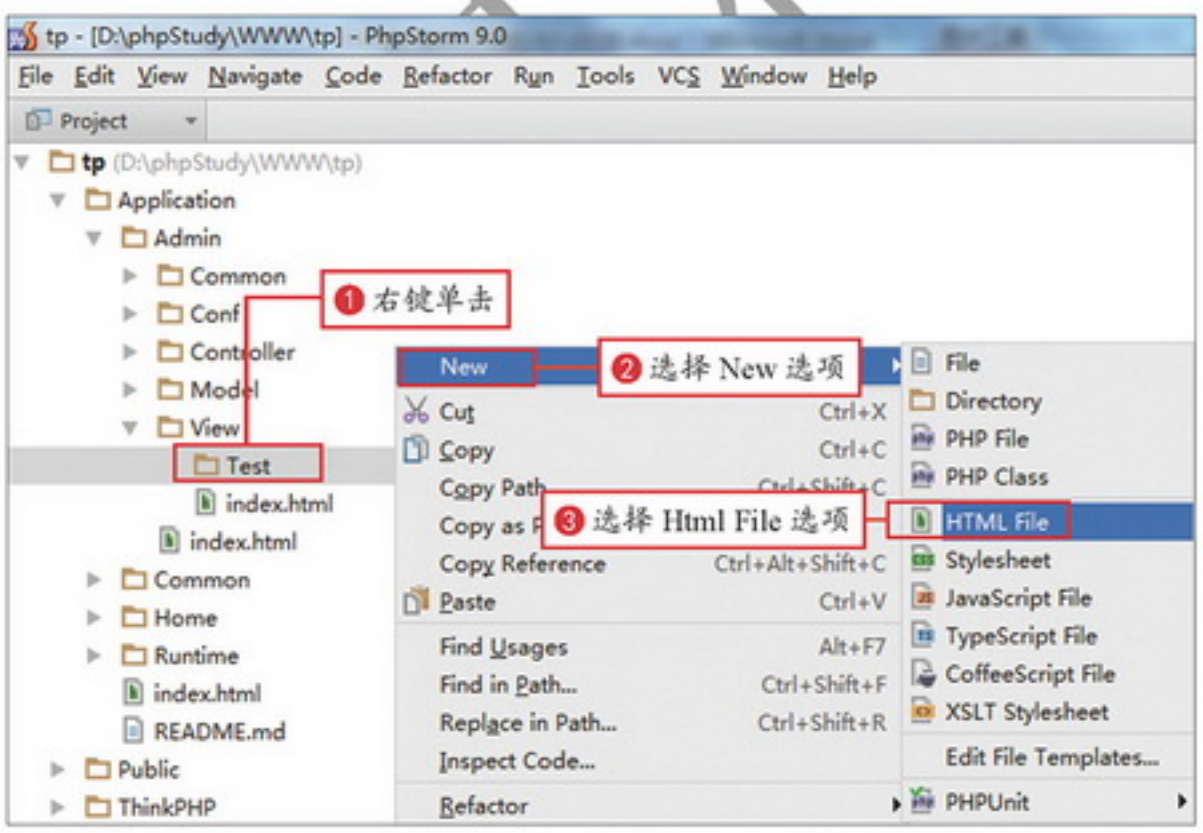


图 15.16　创建视图文件

创建完成后，打开 test1.html 文件，会发现 PhpStorm 已经自动生成 HTML 基本结构，在 test1.html 代码基础上，创建一个简单的表单提交页面，具体代码如下：

```
<!Doctype html>
<html>
<head>
    <meta charset="utf-8">
    <title>test1页面测试</title>
</head>
<body>
<form action="" method="post" >
    <div>
        <input type="text"     name="username" /> 用户名
    </div>
    <div>
        <input type="password" name="password" /> 密码
    </div>
    <div>
        <input type="submit"   value="提交" />
    </div>
</form>
</body>
</html>
```

（3）渲染模板。所谓模板渲染就是将控制器和对应的模板通过某种方式关联起来，并展示模板页的内容。控制器和模板的关系如图 15.17 所示。

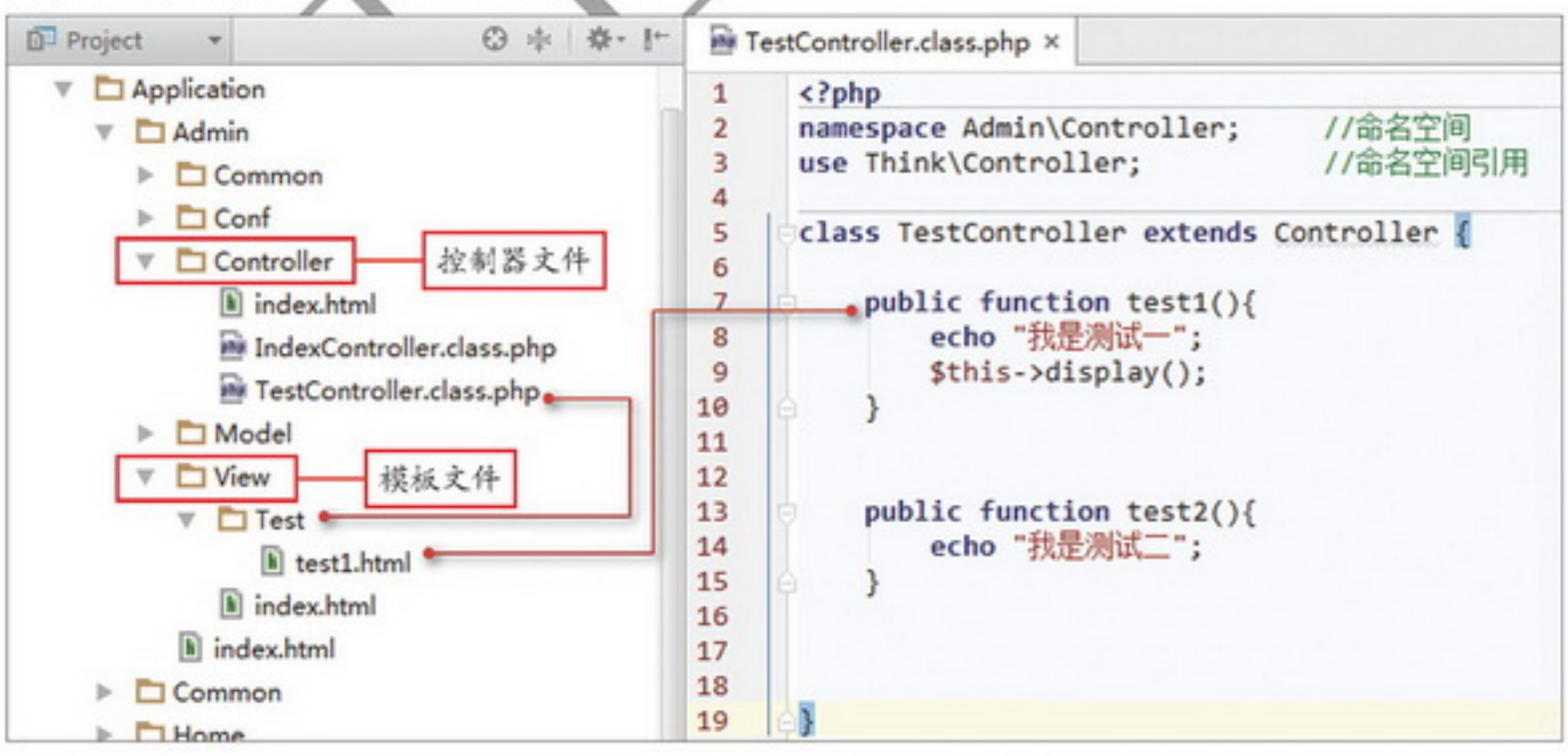


图 15.17　控制器和模板的关系图

在浏览器地址栏中输入“localhost/APP/index.php/Admin/Test/test1”，按下 <Enter> 键后，运行结果如图 15.18 所示。

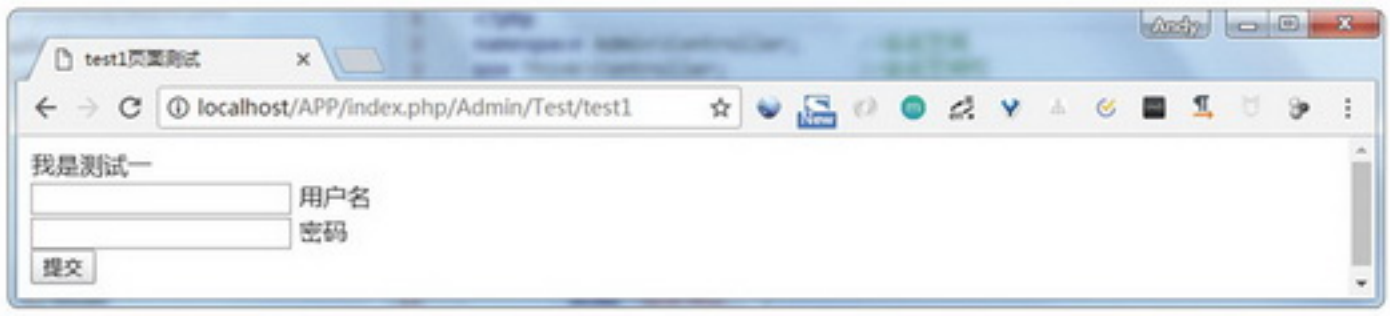


图 15.18　模板渲染运行效果

15.7　内置 ThinkTemplate 模板引擎

15.7.1　变量输出

在模板中输出变量的方法很简单，例如，在控制器中给模板变量赋值的代码如下：

```
$name = 'ThinkPHP';
$this->assign('name',$name);
$this->display();
```

然后就可以在模板中使用：

```
Hello,{$name}!
```

模板编译后的结果就是：

```
Hello,<?php echo($name);?>!
```

这样，运行的时候就会在模板中显示：

```
Hello,ThinkPHP!
```

注意模板标签的“{”和“$”之间不能有任何的空格，否则标签无效，将不会正常输出 name 变量，而是直接保持不变输出，输出结果如下：

```
Hello,{ $name}!
```

普通标签默认开始标记是“{”，结束标记是“}”。也可以通过设置 TMPL_L_DELIM 和 TMPL_R_DELIM 进行更改。例如，在项目配置文件中定义：

```
'TMPL_L_DELIM'=>'<{',
'TMPL_R_DELIM'=>'}>',
```

那么，上面的变量输出标签就应该改成：

```
Hello,<{$name}>!
```

后面的内容都以默认的标签定义来说明。

模板标签的变量输出根据变量类型有所区别，刚才输出的是字符串变量，如果是数组变量，例如：

```
$data['name'] = 'ThinkPHP';
$data['email'] = 'thinkphp@qq.com';
$this->assign('data',$data);
```

那么，在模板中可以用下面的方式输出：

```
Name: {$data.name}
Email: {$data.email}
```

或者用下面的方式也有效：

```
Name: {$data['name']}
Email: {$data['email']}
```

📖 **说明：**当输出多维数组的时候，往往采用后面的方式。

如果 data 变量是一个对象（并且包含有 name 和 email 两个属性），则可以用下面的方式输出：

```
Name: {$data:name}
Email: {$data:email}
```

或者

```
Name: {$data->name}
Email: {$data->email}
```

15.7.2 使用函数

如果需要对模板中的变量使用函数，例如，对模板中输出的变量使用 md5 加密，可以使用如下代码：

```
{$data.name|md5}
```

编译后的结果是：

```
<?php echo (md5($data['name'])); ?>
```

如果函数有多个参数需要调用，则可以使用如下代码：

```
{$create_time|date="y-m-d",###}
```

上述代码表示 date() 函数传入两个参数，参数间用逗号分隔，这里第一个参数是 y-m-d，第二个参数是前面要输出的 create_time 变量，因为该变量是第二个参数，因此需要用 ### 标识变量位置，编译后的结果是：

```
<?php echo (date("y-m-d",$create_time)); ?>
```

15.7.3 内置标签

变量输出使用普通标签就足够了，但是要完成控制、循环和判断等其他功能，就需要借助模板引擎的标签库，系统内置标签库的所有标签无需引入标签库即可直接使用。ThinkPHP 常用内置标签如表 15.7 所示。

表 15.7　ThinkPHP 常用内置标签

标 签 名	作　　用	包 含 属 性
include	包含外部模板文件（闭合）	file
volist	循环数组数据输出	name,id,offset,length,key,mod
foreach	数组或对象遍历输出	name,item,key
for	for 循环数据输出	name,from,to,before,step
switch	分支判断输出	name
case	分支判断输出（必须和 switch 配套使用）	value,break
compare	比较输出（包括 eq neq lt gt egt elt heq nheq 等别名）	name,value,type
empty	判断数据是否为空	name
assign	变量赋值（闭合）	name,value
if	条件判断输出	condition

15.7.4 模板继承

模板继承是一项更加灵活的模板布局方式，模板继承不同于模板布局，甚至可以说是在模板布局的上层。模板继承其实并不难理解，就好比类的继承一样，模板也可以定义一个基础模板（或者是布局），并在其中定义相关的区块（block），然后在继承（extend）该基础模板的子模板中就可以对基础模板中定义的区块进行重载。

因此，模板继承的优势其实是设计基础模板中的区块（block），然后用子模板替换这些区块。每个区块由 <block></block> 标签组成。下面就是基础模板中的一个典型的区块设计（用于设计网站标题）：

```
<block name="title"><title>网站标题</title></block>
```

<block> 标签必须指定 name 属性来标识当前区块的名称，这个标识在当前模板中应该是唯一的，<block> 标签中可以包含任何模板内容，包括其他标签和变量，例如：

```
<block name="title"><title>{$web_title}</title></block>
```

甚至还可以在区块中加载外部文件：

```
<block name="include"><include file="Public:header" /></block>
```

15.8 难点解答

15.8.1 什么是单一入口？

单一入口通常是指一个项目或者应用具有一个统一（但并不一定是唯一）的入口文件，也就是说项目的所有功能操作都是通过这个入口文件进行的，并且入口文件往往是第一步被执行的。单一入口的好处是项目整体比较规范，因为是同一个入口，其不同操作之间往往具有相同的规则。另外一个方面就是单一入口控制较为灵活，因为拦截方便，类似一些权限控制、用户登录方面的判断和操作就可以统一处理。

15.8.2 为什么要使用 MVC 设计模式？

应用程序中用来完成任务的代码——模型层（也叫“业务逻辑”），通常是程序中相对稳定的部分，重用率高；而与用户交互界面——视图层，却经常改变。如果因需求变动而不得不对业务逻辑代码进行修改，或者要在不同的模块中应用到相同的功能而重复的编写业务逻辑代码，不仅降低整体程序开发的进度，也会使未来的维护变得非常困难。因此将业务逻辑代码与外观分离，将会更方便根据需求改进程序，所以通常使用 MVC 设计模式。

15.9 小　结

本章主要对 ThinkPHP 框架的下载、架构、配置、控制器、模型以及视图进行了详细的讲解。并举例对 ThinkPHP 的各种应用进行了介绍，以此来增加读者对 ThinkPHP 的理解。希望通过本章的学习，读者能够掌握 ThinkPHP 技术，并能够将其灵活地运用到实际的网站开发中。

第 4 篇

项目实战

- 第 16 章　51 购商城

第 16 章

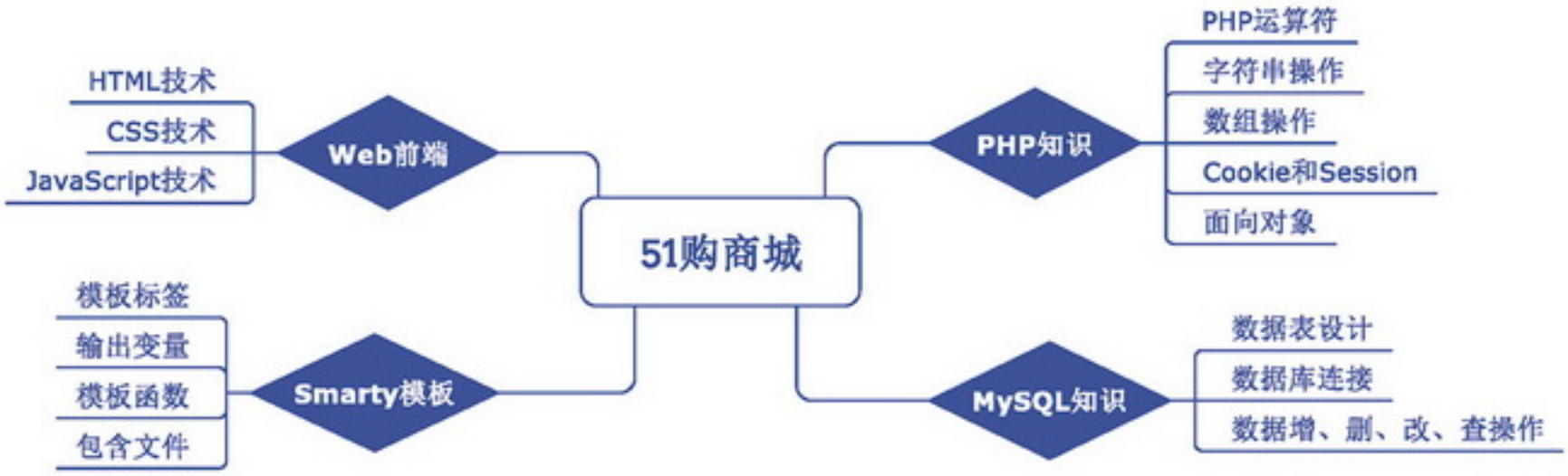
51 购商城

（视频讲解：2 小时）

本章概览

上一章介绍了 ThinkPHP 框架的基本使用方法，本章应用 ThinkPHP 3.2.3 版本实现一个模拟京东的 B2C（Business-to-Customer）商城——51 购商城。51 购商城包括前台和后台，后台用于实现商品和用户的管理，前台用于实现商品分类、展示以及用户注册、登录和购物的流程。

知识框架



16.1　系统功能设计

16.1.1　系统功能结构

51 购商城共分为前台和后台两个部分，前台主要实现商品展示及销售，后台主要是对商城中的商品信息、会员信息，以及订单信息进行有效的管理等。

51 购商城前台的功能设计如图 16.1 所示。

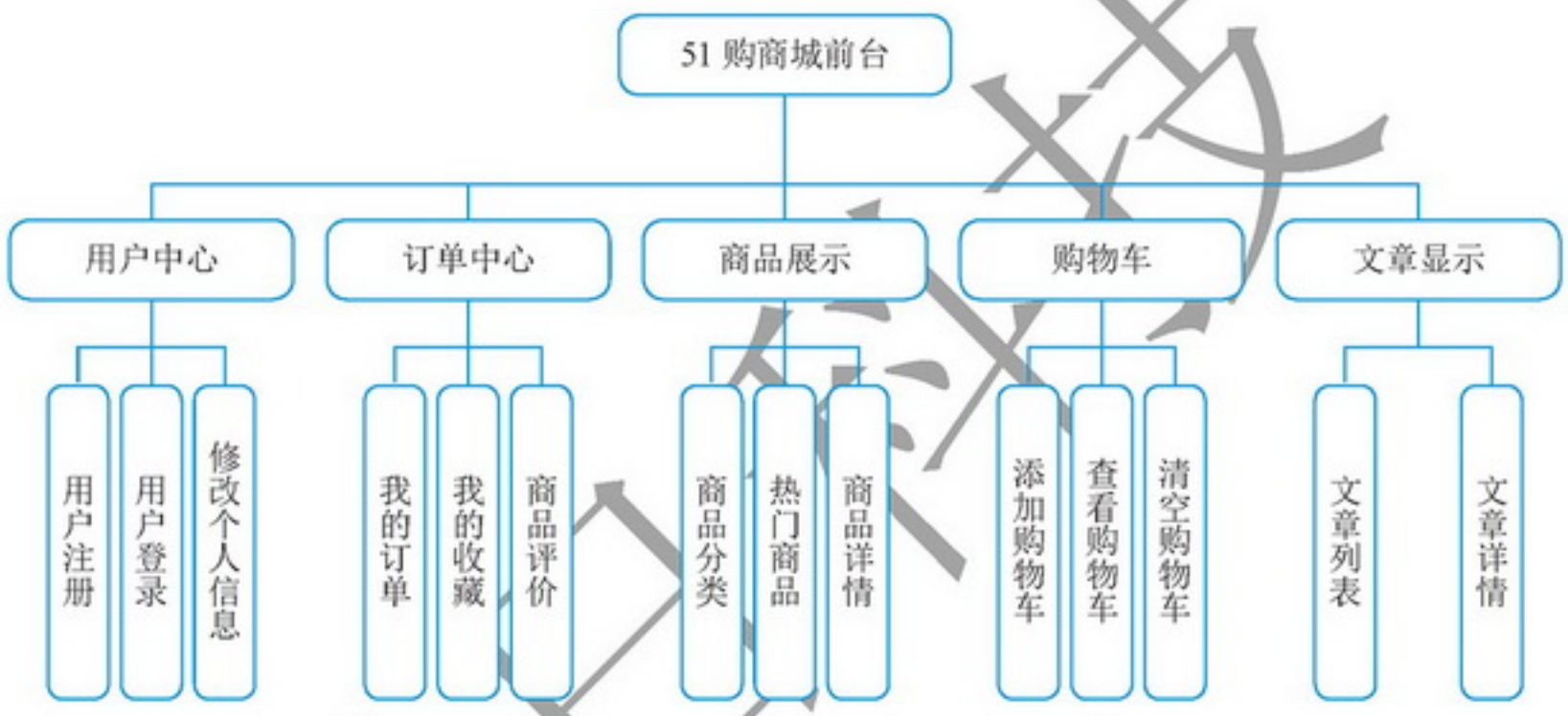


图 16.1　前台功能模块结构图

51 购商城后台管理系统的功能设计如图 16.2 所示。

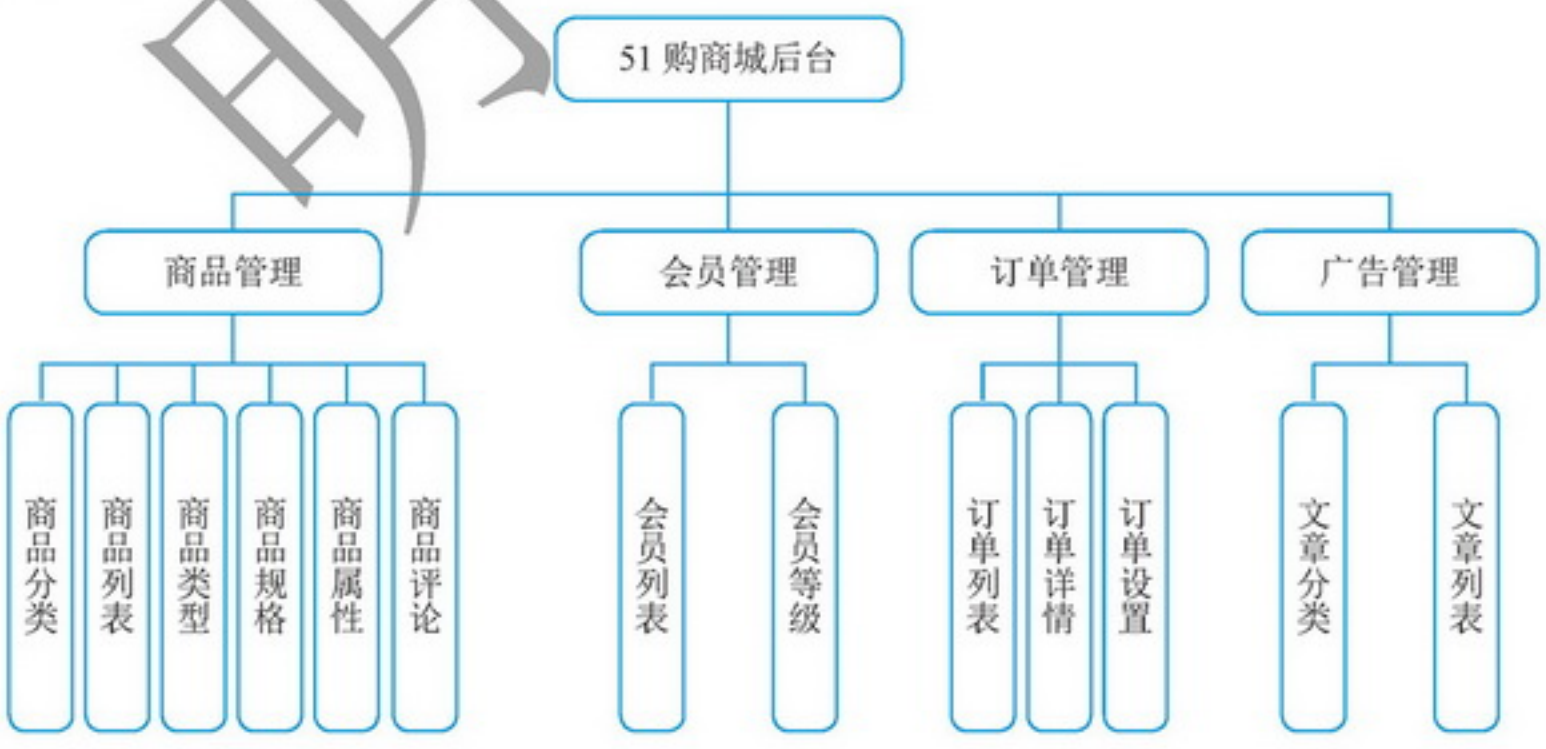


图 16.2　后台功能模块结构图

16.1.2 系统业务流程

51 购商城涉及很多业务流程，其中最重要的就是用户购物流程。该流程包括用户登录、选择商品、加入购物车、结算订单等，具体流程如图 16.3 所示。

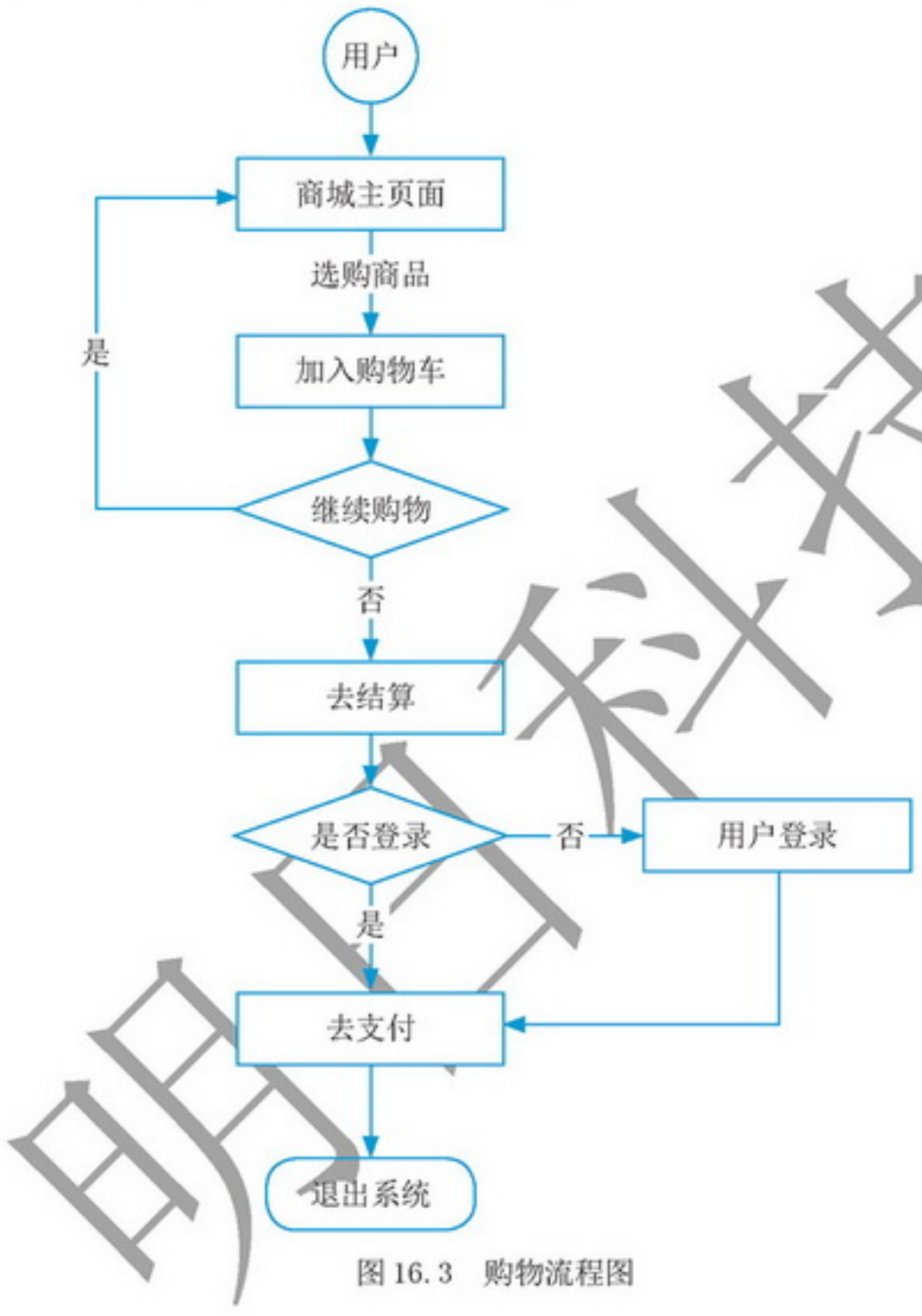


图 16.3　购物流程图

16.2 系统开发必备

16.2.1 系统开发环境

本系统的软件开发及运行环境具体如下。

- 操作系统：Windows 7 及以上版本
- 集成开发环境：phpStudy

- MySQL 图形化管理软件：Navicat for MySQL
- 开发工具：PhpStorm 9
- ThinkPHP 版本：ThinkPHP 3.2.3
- 浏览器：谷歌浏览器

16.2.2 文件夹组织结构

在进行网站开发前，首先要规划网站的架构。即通过建立多个文件夹对各个功能模块进行划分，并实现统一管理，这样做有利于网站的开发、管理和维护。在本项目中，使用默认的 ThinkPHP 目录结构，将 Home 文件夹作为前台模块，Admin 文件夹作为后台模块，如图 16.4 所示。

图 16.4　文件夹组织结构

16.3 数据库设计

16.3.1 数据库概要说明

本项目采用 MySQL 作为数据库，数据库名称为 shop，其数据表名称及作用如表 16.1 所示。

表 16.1　数据库表名称及作用

表　名	含　义	作　用
ad	广告表	用于存储广告信息
ad_position	广告位置表	用于存储广告分布的位置信息
admin	管理员表	用于存储管理员用户信息
article	文章表	用于存储商城中的文章信息
article_cat	文章分类表	用于存储文章的分类信息
brand	品牌表	用于存储商品品牌信息
cart	购物车表	用于存储购物车信息，包括未登录用户的购物车信息

续表

表　名	含　义	作　用
comment	评论表	用于存储商品评论信息
config	网站配置表	用于存储网站配置信息
goods	商品表	用于存储商品信息
goods_attr	属性映射表	用于存储商品和属性的对应关系信息
goods_attribute	商品属性表	用于存储商品属性信息
goods_category	商品分类表	用于存储商品分类信息
goods_collect	商品收藏表	用于存储用户收藏的商品信息
goods_images	商品图片表	用于存储商品图片信息
goods_type	商品类型表	用于存储商品类型信息
order	订单表	用于存储用户订单信息
order_action	订单操作表	用于存储订单操作信息，包括下单、取消订单等
order_goods	订单商品表	用于存储订单商品信息
region	地区表	用于存储地区信息
spec	商品规格表	用于存储商品规格信息
spec_goods_price	规格价钱表	用于存储商品规格对应的价钱信息
spec_image	规格图片表	用于存储商品规格图片信息
spec_item	规格选项表	用于存储商品规格选项信息
user_address	用户地址表	用于存储用户地址信息
user_level	用户等级表	用于存储用户等级信息
users	用户表	用于存储用户信息

16.3.2 数据库逻辑设计

1. 创建数据表

由于篇幅所限，这里只给出较重要的数据表的部分字段，完整数据表请参见本书光盘资源。

◆ admin（后台管理员表）

admin 表用于保存后台管理员数据信息，其结构如表 16.2 所示。

表 16.2　后台管理员表

字　段　名	数 据 类 型	默　认　值	允 许 为 空	自 动 增 加	备　　注
admin_id	smallint(5) unsigned		NO	是	用户 id
user_name	varchar(60)		NO		用户名
email	varchar(60)		NO		e-mail
password	varchar(32)		NO		密码
add_time	int(11)	0	NO		添加时间
last_login	int(11)	0	NO		最后登录时间
last_ip	varchar(15)		NO		最后登录 IP

◆ users（用户表）

表 users 用于存储用户数据信息，其结构如表 16.3 所示。

表 16.3　用户表

字　段　名	数 据 类 型	默　认　值	允 许 为 空	自 动 增 加	备　　注
user_id	mediumint(8) unsigned		NO	是	表 id
email	varchar(60)		NO		邮件
password	varchar(32)		NO		密码
sex	tinyint(1) unsigned	0	NO		0 保密，1 男，2 女
birthday	int(11)	0	NO		生日
pay_points	int(10) unsigned	0	NO		消费积分
address_id	mediumint(8) unsigned	0	NO		默认收货地址
reg_time	int(10) unsigned	0	NO		注册时间
last_login	int(11) unsigned	0	NO		最后登录时间
last_ip	varchar(15)		NO		最后登录 IP
qq	varchar(20)		NO		QQ
mobile	varchar(20)		NO		手机号码
head_pic	varchar(255)		YES		头像
province	int(6)	0	YES		省份
city	int(6)	0	YES		市、区

续表

字 段 名	数 据 类 型	默 认 值	允 许 为 空	自 动 增 加	备 注
district	int(6)	0	YES		县
nickname	varchar(50)		YES		第三方返回昵称
level	tinyint(1)	1	YES		会员等级

◆ goods（商品表）

表 goods 用于存储商品信息，其结构如表 16.4 所示。

表 16.4 商品表

字 段 名	数 据 类 型	默 认 值	允 许 为 空	自 动 增 加	备 注
goods_id	mediumint(8) unsigned		NO	是	商品 id
cat_id	int(11) unsigned	0	NO		分类 id
goods_sn	varchar(60)		NO		商品编号
goods_name	varchar(120)		NO		商品名称
click_count	int(10) unsigned	0	NO		点击数
brand_id	smallint(5) unsigned	0	NO		品牌 id
store_count	smallint(5) unsigned	10	NO		库存数量
comment_count	smallint(5)	0	YES		商品评论数
market_price	decimal(10,2) unsigned	0.00	NO		市场价
shop_price	decimal(10,2) unsigned	0.00	NO		本店价
cost_price	decimal(10,2)	0.00	YES		商品成本价
keywords	varchar(255)		NO		商品关键词
goods_remark	varchar(255)		NO		商品简单描述
goods_content	text		YES		商品详细描述
original_img	varchar(255)		NO		商品上传原始图
is_on_sale	tinyint(1) unsigned	1	NO		是否上架
on_time	int(10) unsigned	0	NO		商品上架时间
sort	smallint(4) unsigned	50	NO		商品排序
is_recommend	tinyint(1) unsigned	0	NO		是否推荐

续表

字　段　名	数 据 类 型	默　认　值	允 许 为 空	自 动 增 加	备　　注
is_new	tinyint(1) unsigned	0	NO		是否新品
is_hot	tinyint(1)	0	YES		是否热卖
last_update	int(10) unsigned	0	NO		最后更新时间
goods_type	smallint(5) unsigned	0	NO		商品所属类型 id
spec_type	smallint(5)	0	YES		商品规格类型 id

2. 数据库连接相关配置

在 ThinkPHP 全局配置文件中配置数据库信息，具体配置如下：

<代码位置：光盘\Code\51购商城\Shop\Application\Common\Conf\config.php>

```
<?php
return array(
    //数据库配置
    'DB_TYPE' => 'mysql',                    //数据库类型
    'DB_HOST' => '127.0.0.1',                //host地址
    'DB_USER' => 'root',                     //数据库用户名
    'DB_PWD' =>  'root',                     //数据库密码
    'DB_NAME' => 'shop',                     //数据库名称
```

16.4 前台用户模块设计

16.4.1 会员注册模块

会员注册模块主要用于实现新用户注册成为网站会员的功能。在会员注册页面中，用户需要填写会员信息，并且需要勾选“同意《账号服务条款、隐私政策》”选项，单击“注册”按钮，程序将验证输入的账号是否唯一。如果唯一，就把填写的会员信息保存到数据库中，否则给出错误提示，并需要修改成唯一的账号后，方可完成注册。

会员注册页面中，主要包括 Form 表单的提交和表单数据的验证。Form 表单主要代码如下：

<代码位置：光盘\Code\51购商城\Shop\Application\Home\View\User\register.html >

```
<form id="register">
    <div class="user-email">
        <label for="email"><i class="mr-icon-envelope-o"></i></label>
        <input type="email" name="email" id="email" placeholder="请输入邮箱">
    </div>
```

```
06      <div class="user-pass">
07          <label for="password"><i class="mr-icon-lock"></i></label>
08          <input type="password" name="" id="password" placeholder="设置密码">
09      </div>
10      <div class="user-pass">
11          <label for="password2"><i class="mr-icon-lock"></i></label>
12          <input type="password" name="" id="password2" placeholder="确认密码">
13      </div>
14
15      <div class="user-pass">
16          <label for="mobile"><i class="mr-icon-mobile"></i></label>
17          <input type="text" name="mobile" id="mobile" placeholder="请输入手机号">
18      </div>
19      <div class="mr-cf">
20          <input type="button"  value="注册" onClick="checkSubmit()"
21                 class="mr-btn mr-btn-primary mr-btn-sm mr-fl">
22      </div>
23  </form>
```

会员注册页面的运行结果如图 16.5 所示。

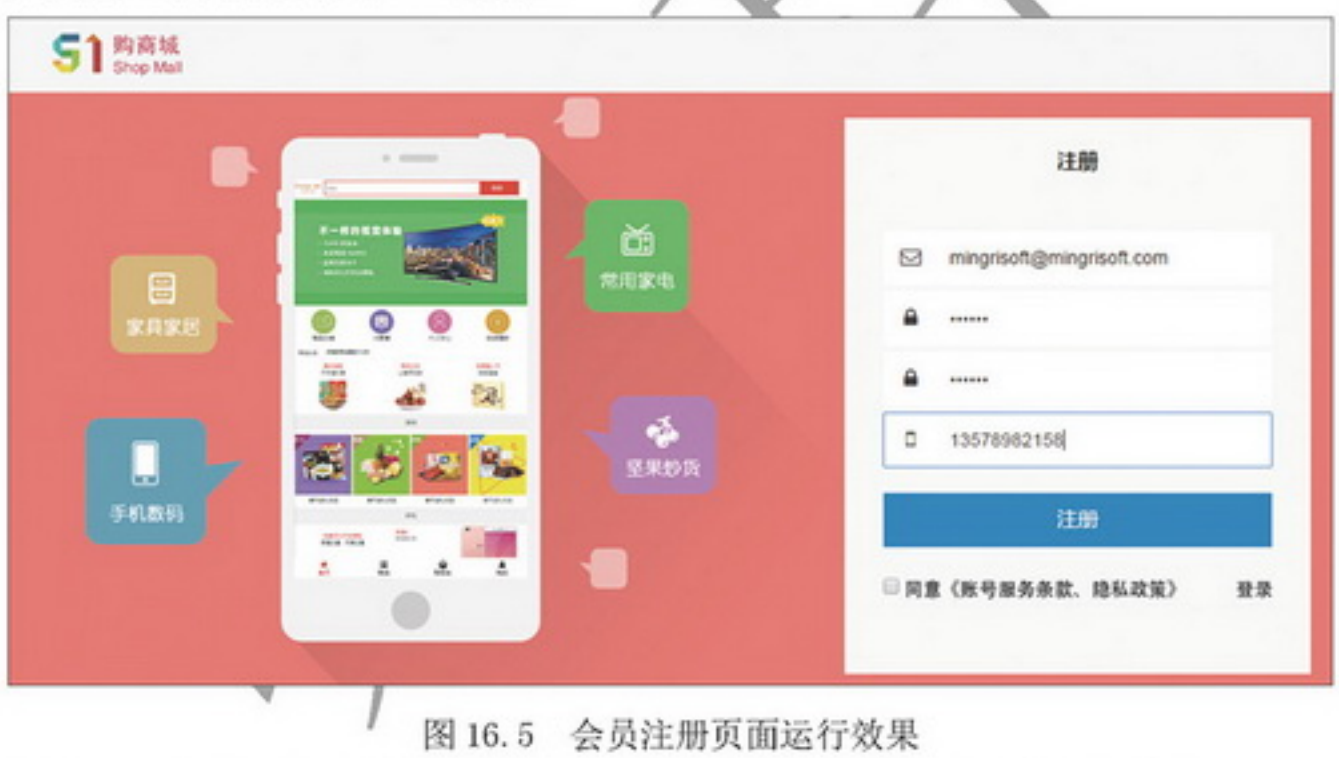


图 16.5　会员注册页面运行效果

在用户注册时，必须同意服务条款，否则提示错误信息。该验证功能是通过 JavaScript 代码实现的。当用户单击“同意《账号服务条款、隐私政策》”超链接时，页面会弹出“服务条款”的具体内容。该功能是通过 Layer.js（弹层插件）实现的。关键代码如下：

<代码位置：光盘\Code\51购商城\Shop\Application\Home\View\User\register.html >

```
<script>
    //单击提交
    function checkSubmit(){
        //省略其余代码
        var agree = $('input[type="checkbox"]:checked').val(); //获取账号服务条款
```

```
        //检测是否勾选注册协议
        if(!agree){
            showErrorMsg('您没有同意注册协议!');
            return false;
        }
        //Ajax异步提交到后台验证
        $.ajax({
            type : 'post',
            url : "{:U('Home/User/register')}",
            data : {email:email,password:password,mobile:mobile},
            dataType : 'json',
            success : function(res){
                if(res.status == 1){
                    layer.msg(res.msg,{icon:1,time:2000},function(){
                        window.location.href = "{:U('Home/Index/index')}"; //跳转到首页
                    });
                }else{
                    showErrorMsg(res.msg);                              //显示错误信息
                }
            },
            error : function(XMLHttpRequest, textStatus, errorThrown) {
                showErrorMsg('网络失败，请刷新页面后重试');
            }
        })

    }
    //显示错误信息的方法
    function showErrorMsg(msg){
        layer.msg(msg,{icon:2,time:2000});
    }
    //显示协议内容
    function showProtocol(){
        var protocol = '<p style="padding: 10px">      欢迎来到明日学院，为了
            保障您的权益，请在使用明日学院服务之前，详细阅读此服务协议（以下简称"本协议"）所有
            内容，如您不同意本协议任何条款，请勿注册账号或使用本平台。本协议内容包括协议正文、
            本协议下述协议明确援引的其他协议、明日科技公司已经发布的或将来可能发布的各类规则。
            所有规则为本协议不可分割的组成部分，与协议正文具有同等法律效力。除另行明确声明外，
            您使用明日学院服务均受本协议约束。</p>';
        layer.open({
            title:'协议内容',                                           //弹层标题
            type: 1,                                                    //弹层类型
            skin: 'layui-layer-rim',                                    //加上边框
            area: ['520px', '240px'],                                   //弹层宽和高
            content: protocol                                           //弹层内容
        });
    }
</script>
```

运行效果如图 16.6 所示。

图 16.6　弹出服务协议内容

从上述代码中可以看出，当满足验证条件后，单击“注册”按钮，会通过 Ajax 异步提交的方式，将 Form 表单中的内容提交到 Home 模块下的 User 控制器中的 register() 方法。register() 方法关键代码如下：

<代码位置：光盘\Code\51购商城\Shop\Application\Home\Controller\UserController.class.php >

```
01  public function register(){
02      //如果已经登录，直接跳转到首页
03      if($this->user_id > 0) header("Location: ".U('Home/Index/index'));
04      /** 表单提交操作 **/
05      if(IS_POST){
06          $logic = new UsersLogic();                          //实例化逻辑层UsersLogic类
07          $email = I('post.email','');                        //接收传递的邮箱
08          $password = I('post.password','');                  //接收传递的密码
09          $mobile    = I('post.mobile','');                   //接收传递的手机号
10          $data = $logic->register($email,$password,$mobile);
11          //省略部分代码
12          $this->ajaxReturn($data);                           //返回数据
13      }
14      /** 非表单提交，直接显示页面 **/
15      $this->display();
16  }
```

上述代码中，首先实例化逻辑层 UsersLogic 类，该类文件路径为：Shop\Application\Home\Logic\UsersLogic.class.php。然后，调用 UsersLogic 类的 register() 方法，实现对表单提交内容和用户是否已注册等信息进行验证。如果全部验证通过，则将用户信息写入 users 表，页面跳转到商城首页，否则提示相应的错误信息。

16.4.2 会员登录模块

会员登录模块主要用于实现网站的会员登录功能，在该页面中，填写会员账号和密码，单击“登录”按钮，即可实现会员登录。如果没有输入账号、密码或者账号密码不匹配，则都将给予提示。登

录模块功能主要是由 User 控制器下的 do_login() 方法实现的，关键代码如下：

<代码位置：光盘\Code\51购商城\Shop\Application\Home\Controller\UserController.class.php >

```
01  public function do_login(){
02      $username = trim(I('post.username'));                    //获取用户名并去除首尾空格
03      $password = trim(I('post.password'));                    //获取密码并去除首尾空格
04
05      $logic = new UsersLogic();                               //实例化UsersLogic类
06      $res = $logic->login($username,$password);               //调用UsersLogic类的login()方法
07      //省略其余代码
08  }
```

上述代码与注册模块代码相似，都是先实例化逻辑层 UsersLogic 类，然后调用登录验证的方法 login()。在 login() 方法中，先判断手机号或者邮箱是否存在，如果存在，继续判断输入的密码是否正确；否则，返回错误信息。

16.5 前台首页模块设计

当用户访问 51 购商城时，首先进入的便是前台首页。前台首页是对整个网站总体内容的概述。在 51 购商城的前台首页中，主要包含以下内容：

◆ 商品分类模块：主要包括“图书、音像、电子书”等一级分类、“科技、音像”等二级分类和“计算机与互联网、建筑”等三级分类。如图 16.7 所示。

◆ 网站菜单导航模块：主要包括“首页”以及“计算机编程”“手机”“平板电脑”等热门的商品类别。如图 16.7 所示。

◆ 搜索模块：主要用于商品信息的快速搜索。如图 16.7 所示。

◆ 幻灯片模块：主要用于以幻灯片的方式演示商品。如图 16.7 所示。

◆ 热卖商品模块：主要用于展示商城重点推荐的商品及其详细信息的查看。如图 16.8 所示。

◆ 商品列表模块：主要用于展示每个分类下的推荐商品。如图 16.9 所示。

◆ 文章分类和信息模块：主要显示分类的文章，为用户提供相应的帮助和支持。如图 16.10 所示。

图 16.7　前台分类、搜索、导航、幻灯片模块效果图

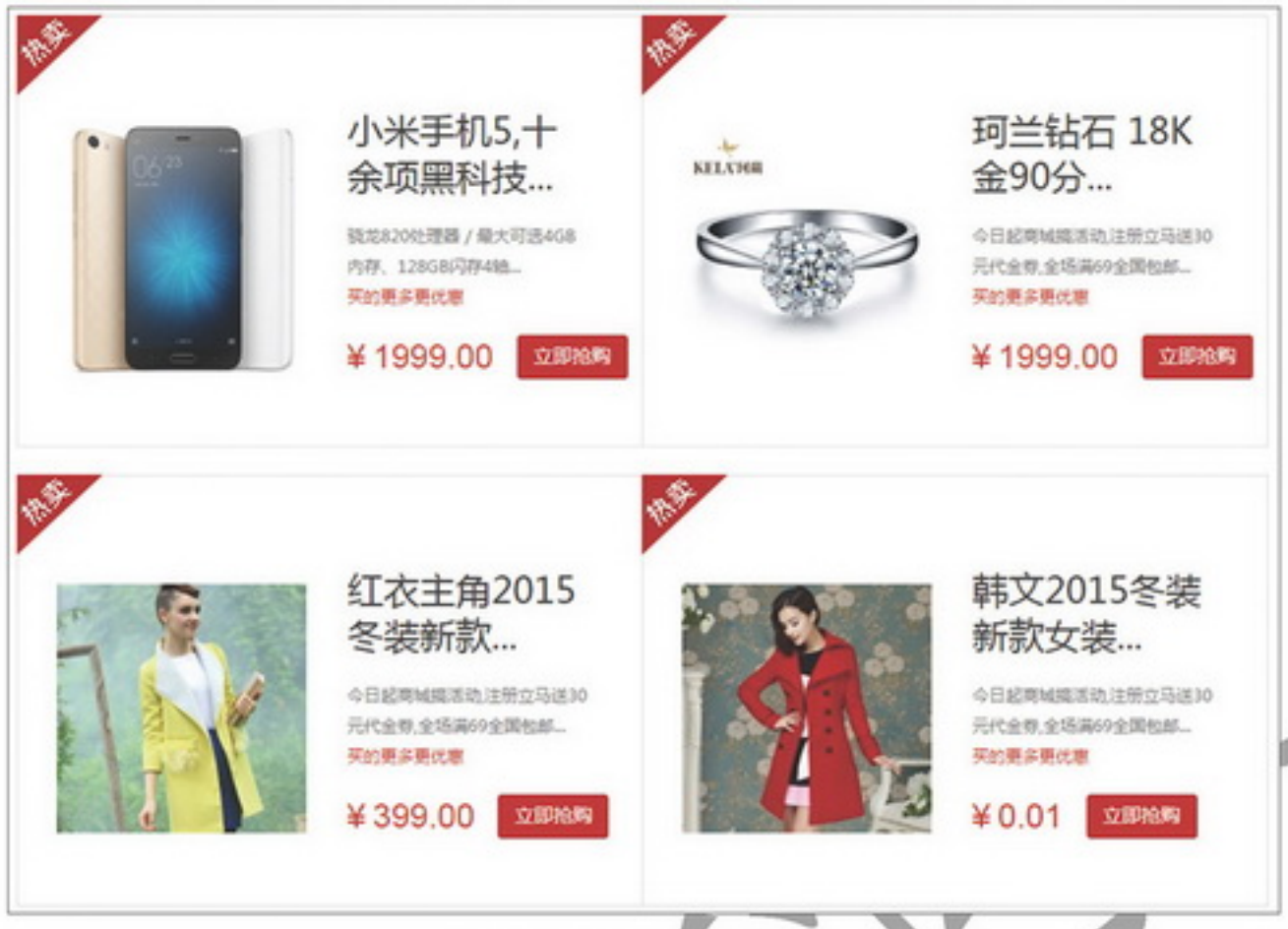


图 16.8　热卖商品模块

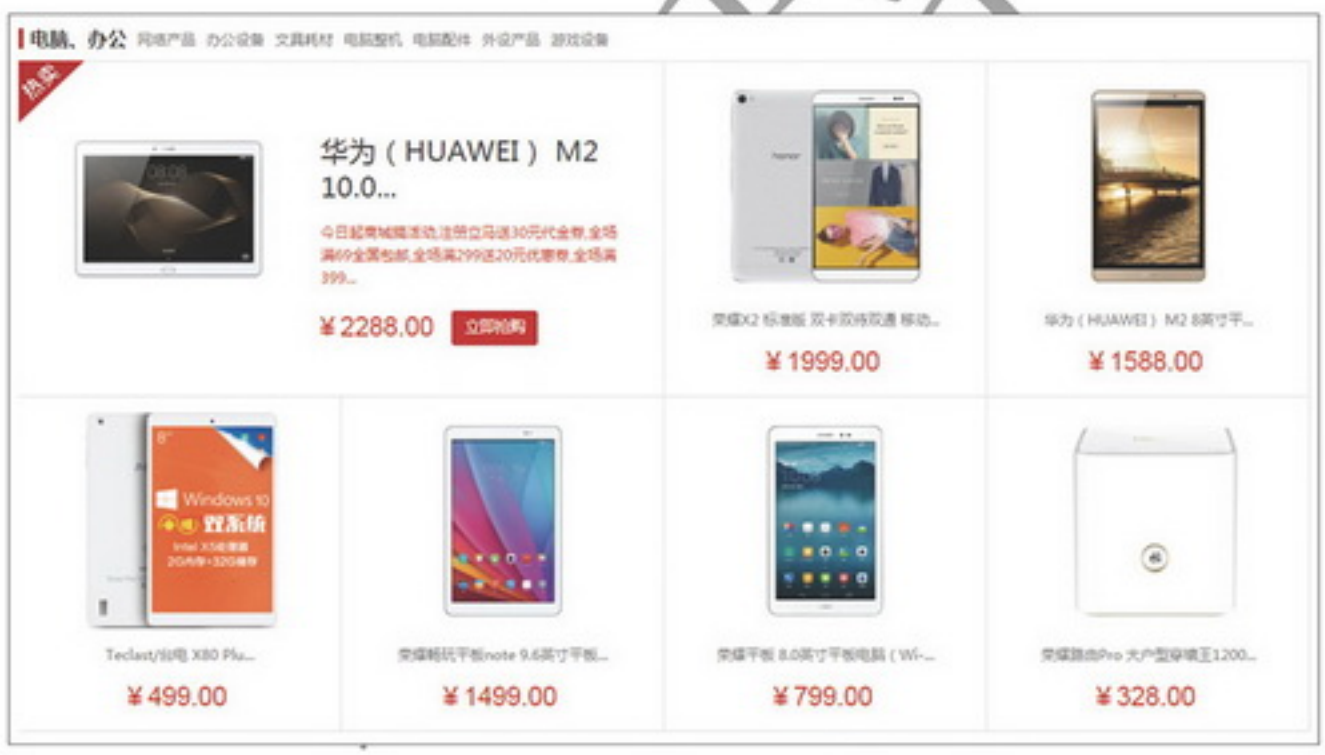


图 16.9　商品列表模块

图 16.10　文章模块

由于篇幅有限，在前台首页模块中，只介绍商品分类模块和商品列表模块。

16.5.1 商品分类模块

1. 获取分类数据

在本项目中，不止前台首页包含商品分类数据，在其他页面，当鼠标悬浮在“全部商品”内容上时，也会显示商品分类数据。所以，可以将商品分类作为通用数据，写在 BaseController.class.php 父类文件中，然后令相应的控制器类通过 extends 继承 BaseController.class.php 父类。关键代码如下：

<代码位置：光盘\Code\51购商城\Shop\Application\Home\Controller\IndexController.class.php >

```
<?php

class IndexController extends BaseController {
    public function index(){
        //省略其余代码
}
```

当访问前台首页，即 Home 模块的 Index 控制器的 index 方法时，程序会优先执行其父类的 _initialize() 初始化方法。关键代码如下：

<代码位置：光盘\Code\51购商城\Shop\Application\Home\Controller\BaseController.class.php >

```
<?php

namespace Home\Controller;
use Think\Controller;

class BaseController extends Controller {
    /*
     * 初始化操作
     */
    public function _initialize() {
        //省略其余代码
        $this->public_assign();                                    //调用public_assign()方法
    }
    /**
     * 保存公告变量到smarty模板中
     */
    public function public_assign()
    {
        //省略其余代码
        $goods_category_tree = get_goods_category_tree();  //获取商品一、二、三级分类
        $this->cateTrre = $goods_category_tree;                    //分类赋值
        $this->assign('goods_category_tree', $goods_category_tree); //模板赋值
    }
```

上述代码中，调用了 get_goods_category_tree() 函数，该函数主要用于将所有分类按照一、二、三级分类的数据格式保存在数组中。具体代码如下：

<代码位置：光盘\Code\51购商城\Shop\Application\Home\Common\function.php >

```php
/**
 * 获取商品一、二、三级分类
 * @return type
 */
function get_goods_category_tree(){
    $result = array();
    //查询所有显示的分类
    $cat_list = M('goods_category')->where("is_show = 1")->order('sort_order')->select();
    foreach ($cat_list as $val){
        if($val['level'] == 2){                    //如果是二级分类
            $arr[$val['parent_id']][] = $val;      //将该分类赋值给$arr数组的元素
        }
        if($val['level'] == 3){                    //如果是三级分类
            $crr[$val['parent_id']][] = $val;      //将该分类赋值给$crr数组的元素
        }
        if($val['level'] == 1){                    //如果是一级分类
            $tree[] = $val;                        //将该分类赋值给$tree数组的元素
        }
    }

    //遍历二级分类，将三级分类的内容赋值给二级分类
    foreach ($arr as $k=>$v){
        foreach ($v as $kk=>$vv){
            //三级分类内容作为其二级分类的sub_menu
            $arr[$k][$kk]['sub_menu'] = empty($crr[$vv['id']]) ? array() : $crr[$vv['id']];
        }
    }

    //遍历一级分类，将二级分类的内容赋值给一级分类
    foreach ($tree as $val){
        //二级分类内容作为其一级分类的tmenu
        $val['tmenu'] = empty($arr[$val['id']]) ? array() : $arr[$val['id']];
        $result[$val['id']] = $val;
    }
    return $result;
}
```

get_goods_category_tree() 函数返回值示例数据如下：

```
Array(
    [11] => Array(
        [id] => 11
            [name] => 图书、音像、电子书
```

```
            [parent_id] => 0
            [parent_id_path] => 0_11
            [level] => 1
            [tmenu] => Array(
                        [0] => Array(
                            [id] => 94
                            [name] => 科技
                            [parent_id] => 11
                            [parent_id_path] => 0_11_94
                            [level] => 2
                        [sub_menu] => Array(
                            [0] => Array(
                                    [id] => 842
                                    [name] => 计算机与互联网
                                    [parent_id] => 94
                                    [parent_id_path] => 0_11_94_842
                                    [level] => 3
                            )
                            [1] => Array(
                                [id] => 836
                                [name] => 建筑
                                [parent_id] => 94
                                [parent_id_path] => 0_11_94_836
                                [level] => 3
                            )
```

从上面的示例中可以看出，get_goods_category_tree() 函数返回值为多维数组，其中 id 为 11 的记录是一级分类，分类名称是“图书、音像、电子书”。id 为 94 的记录是一级分类下的二级分类，分类名称为“科技”。id 为 842 和 836 的记录为二级分类下的三级分类，分类名称分别是“计算机与互联网”和“建筑”。

2. 渲染模板

获取完分类数据后，接下来就需要渲染模板显示数据了。由于分类数据是一个多维数组，所以使用 <foreach> 标签循环嵌套获取各级分类数据即可。关键代码如下：

<代码位置：光盘\Code\51购商城\Shop\Application\Home\View\Public\header.html >

```
<!-- 遍历一、二、三级分类 -->
<foreach name="goods_category_tree" key="k" item='v'>
    <if condition="$v['level'] eq 1">    <!-- if标签判断是否为一级分类 -->
        <li class="list-li">
            <div class="list_a">
                <!-- 输出一级分类内容 -->
                <h3><a href="{:U('Home/Goods/goodsList',array('id'=>$v['id']))}">
                    <span>{$v['name']}</span>
```

```
                </a></h3>
                <p>
                    <!-- 选出3个二级标题 -->
                   <assign name="index" value="1" />
                   <foreach name="v['tmenu']" item="v2" key="k2" >
                     <if condition="$v2['parent_id'] eq $v['id']">
                        <?php if($index++ > 3) break; ?>
                        <a href="{:U('Home/Goods/goodsList',array('id'=>$v2['id']))}">
                           {$v2['name']}
                         </a>
                     </if>
                   </foreach>
                </p>
            </div>
            <div class="list_b">
                <div class="list_bigfl">
                    <!-- 选出6个二级标题 -->
                   <assign name="index" value="1" />
                   <foreach name="v['tmenu']" item="v2" key="k2" >
                       <if condition="$v2[parent_id] eq $v['id']">
                        <?php if($index++ > 6) break; ?>
                            <a class="list_big_o ma-le-30"
                              href="{:U('Home/Goods/goodsList',array('id'=>$v2['id']))}">
                              {$v2['name']} <i>></i>
                            </a>
                       </if>
                   </foreach>
                </div>
                <div class="subitems">
                    <!-- 遍历二级标题 -->
                   <foreach name="v['tmenu']" item="v2" key="k2" >
                       <if condition="$v2['parent_id'] eq $v['id']">
                            <dl class="ma-to-20 cl-bo">
                                <dt class="bigheader wh-sp">
                                    <a href="{:U('Home/Goods/goodsList',
                                            array('id'=>$v2['id']))}"> {$v2['name']}
                                    </a><i>></i>
                                </dt>
                                <dd class="ma-le-100">
                                    <!-- 遍历二级标题下的三级标题 -->
                                   <foreach name="v2['sub_menu']" item="v3" key="k3" >
                                     <if condition="$v3['parent_id'] eq $v2['id']">
                                        <a class="hover-r ma-le-10 "
                                            href="{:U('Home/Goods/goodsList',
                                                array('id'=>$v3['id']))}"> {$v3['name']}
```

```
                                        </a>
                                    </if>
                                </foreach>
                            </dd>
                        </dl>
                    </if>
                </foreach>
            </div>
        </div>
    </li>
</if>
</foreach>
```

运行结果如图 16.11 所示。

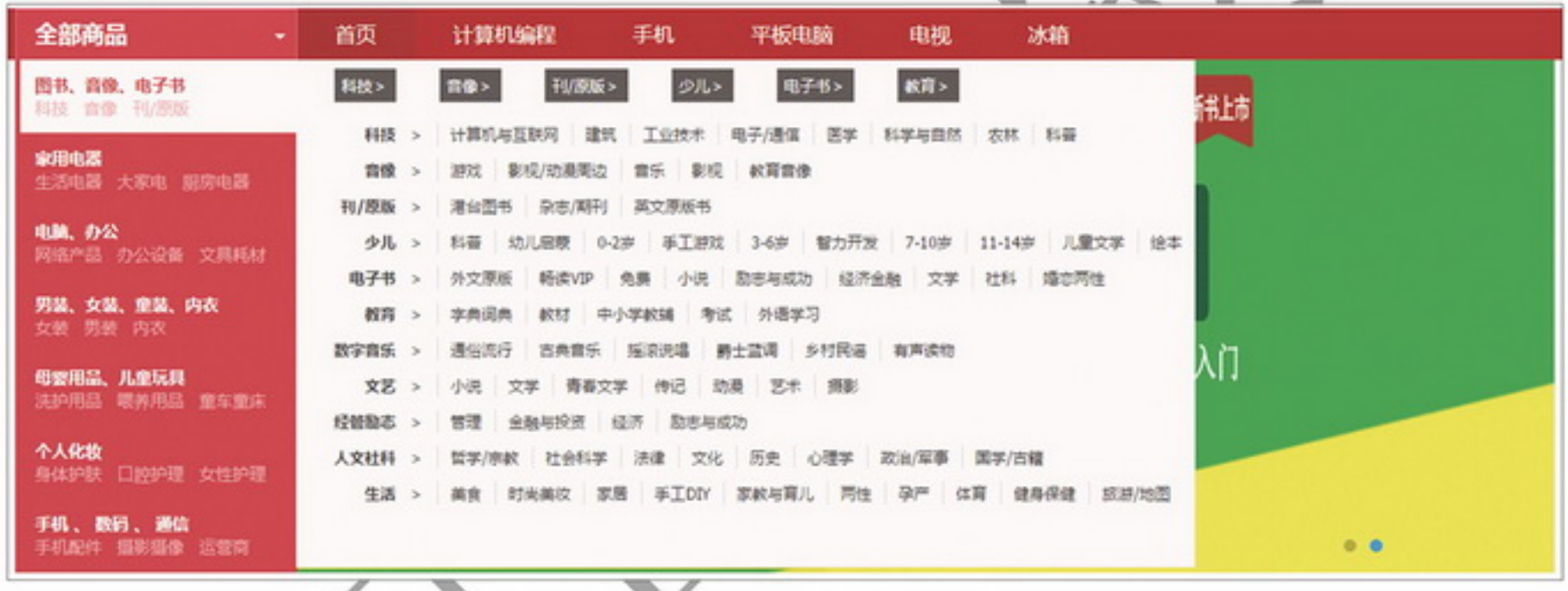


图 16.11　三级分类效果图

16.5.2 商品列表模块

由于商品数量较多，在前台首页只能展示一部分商品数据，所以从 goods 商品表中筛选数据时，只筛选出满足以下条件的数据：

◆ goods_category 表商品分类 is_show 字段值为 1 的分类，且最多只筛选 7 个分类。

◆ goods 表 is_on_sale 字段值为 1，即在售的商品，且最多只筛选 7 个商品。

具体代码如下：

<代码位置：光盘\Code\51购商城\Shop\Application\Home\Controller\IndexController.class.php >

```
$category1 = M('goods_category')->where(array('is_show'=>1,'level'=>1))
                                ->limit(7)->select();                    //筛选一级分类
foreach($category1 as $key=>$v ){
    $category2 = M('goods_category')->where(array('is_show'=>1,'parent_id'=>$v['id']))
                                    ->field('id,name')->select();        //筛选二级分类
```

```
06      $category[$v['name']]['sub_category'] = $category2;
07      $cat_id_arr = getCatGrandson($v['id']);              //找到一级下面的所有子分类id
08      $sub_id_str = implode(',',$cat_id_arr);              //将子分类id拼接成字符串
09      $map['cat_id']      = array('in',$sub_id_str);       //搜索条件：商品分类id在子类id中
10      $map['is_on_sale'] = 1;                              //搜索条件：商品在售
11      //从商品表中，筛选7条满足以上2个条件的记录
12      $category[$v['name']]['goods'] = M('goods')->where($map)->limit(7)
13                          ->field('goods_id,goods_name,keywords,goods_remark,shop_price')
14                          ->order('goods_id')->select();
15  }
16  $this->assign('category',$category);                     //模板赋值
```

运行结果如图 16.9 所示。

16.6 购物车模块设计

购物车是前台用户端程序中非常关键的一个功能模块，它帮助用户完成商品的选购，并把商品交给服务器进行结算。下面介绍购物车模块的设计方法。

16.6.1 添加商品至购物车

在前台首页选择商品后，进入商品详情页。商品详情页包含商品的名称、简介、属性、价格等信息，如图 16.12 所示。当选择完商品属性后，单击“加入购物车”按钮，即可将商品加入购物车中。运行效果如图 16.13 所示。

图 16.12　商品详情页

图 16.13　加入购物车

将商品加入购物车的步骤如下：

（1）单击“加入购物车”按钮，执行 JavaScript 的 onclick 点击事件，调用 AjaxAddCart() 方法。关键代码如下：

<代码位置：光盘\Code\51购商城\Shop\Application\Home\View\Goods\goodsInfo.html >

```
<a class="jrgwc-shopping-img2"
    onClick="javascript:AjaxAddCart({$goods.goods_id},1,0);">
    <span>加入购物车</span>
</a>
```

（2）AjaxAddCart() 是 common.js 文件中的方法，关键代码如下：

<代码位置：光盘\Code\51购商城\Shop\Public\js\common.js >

```
$.ajax({
    type : "POST",
    url:"/index.php?m=Home&c=Cart&a=ajaxAddCart",          //请求地址
    data : $('#buy_goods_form').serialize(),               //搜索表单，序列化提交
    dataType:'json',                                       //数据格式
    success: function(data){                               //请求成功后的响应
        if(data.status < 0){
            layer.alert(data.msg, {icon: 2});
            return false;
        }
        //加入购物车后再跳转到购物车页面
        if(to_catr == 1){                                  //直接购买
            location.href = "/index.php?m=Home&c=Cart&a=cart"; //跳转到购物车页面
        }else{
            cart_num = parseInt($('#cart_quantity').html())+
```

```
                    parseInt($('input[name="goods_num"]').val()); //获取商品数量
            $('#cart_quantity').html(cart_num);                    //获取商品数量写入DOM中
            //使用layer.js 弹出加入成功框
            layer.open({
                type: 2,                                           //弹层类型
                title: '温馨提示',                                  //标题
                skin: 'layui-layer-rim',                           //弹层样式
                area: ['490px', '386px'],                          //弹层宽度、高度
                content:["/index.php?m=Home&c=Goods&a=open_add_cart","no"], //页面内容
                success: function(layero, index) {                 //成功响应
                    layer.iframeAuto(index);                       //指定iframe层自适应
                }
            });
        }
    }
});
```

上述代码中，使用 Ajax 将购物车中商品数据异步提交到 Home 模块的 Cart 控制器的 ajaxAddCart() 方法，该方法主要用于将商品信息写入 cart 购物车表。成功加入购物车后，使用 Layer 弹层弹出加入成功页面，并且在该页面展示热卖商品。其中，热卖商品数据来源于 Goods 控制器的 open_add_cart() 方法。

16.6.2 查看购物车商品

将商品添加至购物车后，单击“去结算”按钮，即可跳转至购物车列表页。或者，在前台首页“我的购物车”按钮处显示购物车商品数量，当鼠标悬停在“我的购物车”按钮时，将展示所有添加到购物车的商品。如图 16.14 所示。

图 16.14 前台首页“我的购物车”效果

在购物车列表页，使用 JavaScript 调用 ajax_cart_list() 方法，该方法使用 Ajax 异步提交的方式获取购物车列表数据，并将获取到的内容追加到当前页面。具体代码如下：

<代码位置：光盘\Code\51购商城\Shop\Home\View\Cart\cart.html >

```
$(document).ready(function(){
    ajax_cart_list();                    //Ajax请求获取购物车列表
});

//Ajax提交购物车
var before_request = 1;                  //判断上一次请求是否已经返回，只有返回才可以进行下一次请求
function ajax_cart_list(){
    if(before_request == 0)              //如果上一次请求没返回，则不进行下一次请求
        return false;
    before_request = 0;
    $.ajax({
        type : "POST",
        url:"{:U('Home/Cart/ajaxCartList')}",        //提交地址
        data : $('#cart_form').serialize(),          //Form表单序列化
        success: function(data){
            $("#ajax_return").html('');
            $("#ajax_return").append(data);
            before_request = 1;
        }
    });
}
```

在上述代码中，将 Ajax 异步提交到 Cart 控制器下的 ajaxCartList() 方法，该方法主要用于修改购物车中商品数量和更改商品选中状态，并且调用逻辑层的 cartLogic 类的 cartList() 方法，从购物车表中筛选数据并计算总价。运行效果如图 16.15 所示。

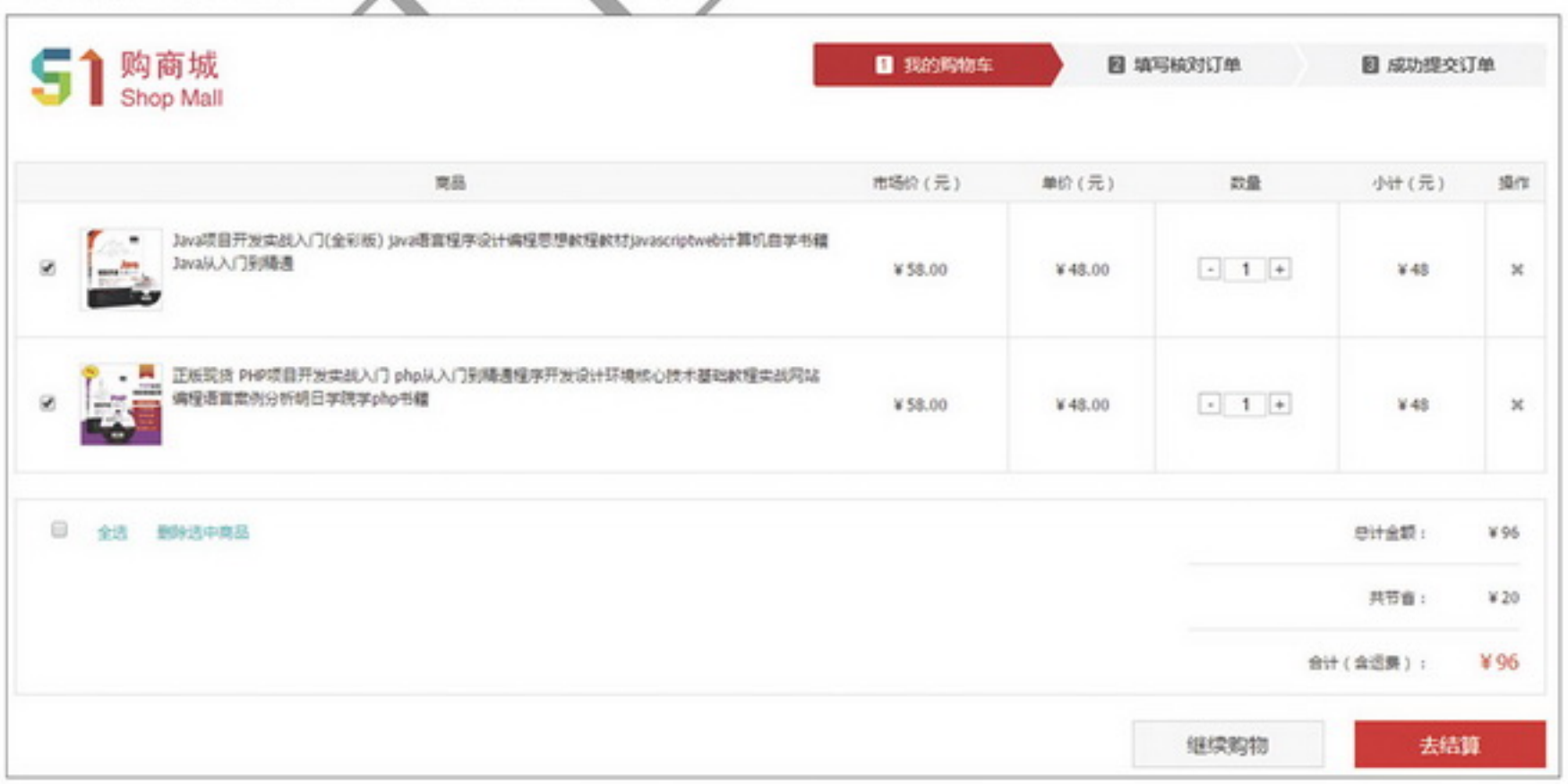


图 16.15　查看购物车效果

16.6.3 清空购物车

当用户想要重新选购商品时，可以删除购物车中的某一个商品或是清空所有商品。当单击购物车中某个商品后的删除图标，弹出“确定删除吗？”确认框，当用户单击“确定”按钮时，即可删除该商品。使用 Ajax 异步提交实现该功能，具体代码如下：

<代码位置：光盘\Code\51购商城\Shop\Home\View\Cart\cart.html >

```
//Ajax删除购物车的商品
function ajax_del_cart(ids){
    layer.confirm('确定要删除吗？',function(){            //Layer弹层确认框
        $.ajax({
        type : "POST",                                    //请求方式
        url:"{:U('Home/Cart/ajaxDelCart')}",              //请求地址
        data:{ids:ids},                                   //提交数据
        dataType:'json',
        success: function(data){
            if(data.status == 1){
                    layer.msg(data.msg, {icon: 1, time: 1000}, function () {
                        ajax_cart_list();                 //Ajax请求获取购物车列表
                        layer.closeAll();                 //关闭弹层
                    });
                }
        }
    });
  });
}

//批量删除购物车的商品
function del_cart_more(){
      //循环获取复选框选中的值
      var chk_value = [];
      $('input[name^="cart_select"]:checked').each(function(){
          var s_name = $(this).attr('name');
          var id = s_name.replace('cart_select[',").replace(']',");
          chk_value.push(id);
      });
      //Ajax调用删除
      if(chk_value.length > 0)
          ajax_del_cart(chk_value.join(','));
}
```

在上述代码中，ajax_del_cart() 方法为单选删除的方法，del_cart_more() 方法为多选删除的方法。接下来，看一下 Ajax 异步提交到 Cart 控制器的 ajaxDelCart() 方法，在该方法中实现从 Cart 数据表中删除数据的操作。具体代码如下：

<代码位置：光盘\Code\51购商城\Shop\Home\Controller\CartController.class.php >

```
public function ajaxDelCart()
```

```
02  {
03      $ids = I("ids");                                          //商品ids
04      $result = M("Cart")->where(" id in ($ids)")->delete();    //删除ids数组中的数据
05      $return_arr = array('status'=>1,'msg'=>'删除成功','result'=>'');  //返回结果状态
06      $this->ajaxReturn($return_arr);
07  }
```

运行效果如图 16.16 和图 16.17 所示。

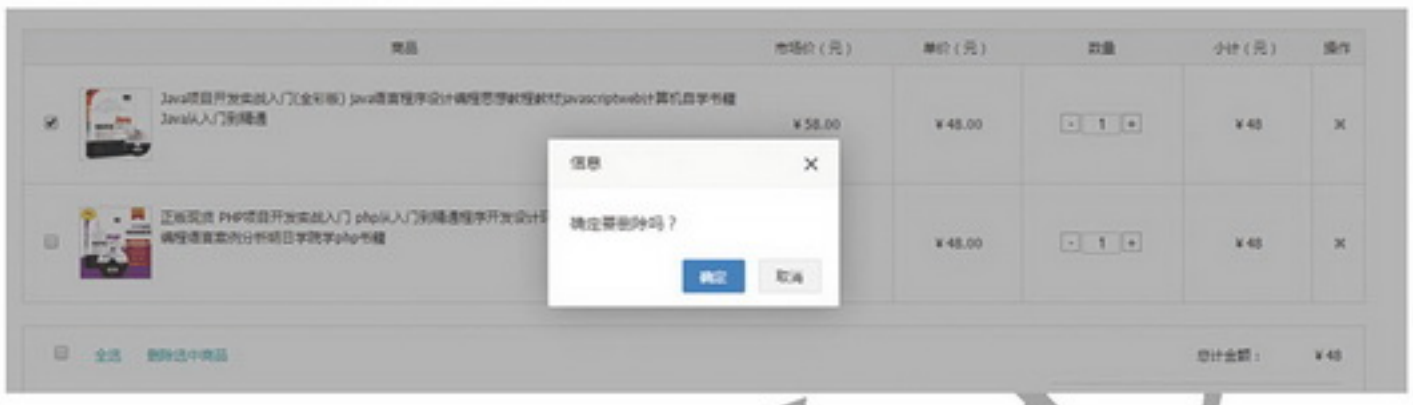


图 16.16　弹层删除提示框效果

图 16.17　删除成功效果

16.6.4 添加收货地址

用户在购物车列表页面选择商品后，单击“去结算”按钮，跳转至核对订单页面，该页面包括收货人信息和订单详细信息。如果用户首次购买商品，单击“提交订单”按钮，程序会提示“请先填写收货人信息”。运行效果如图 16.18 所示。

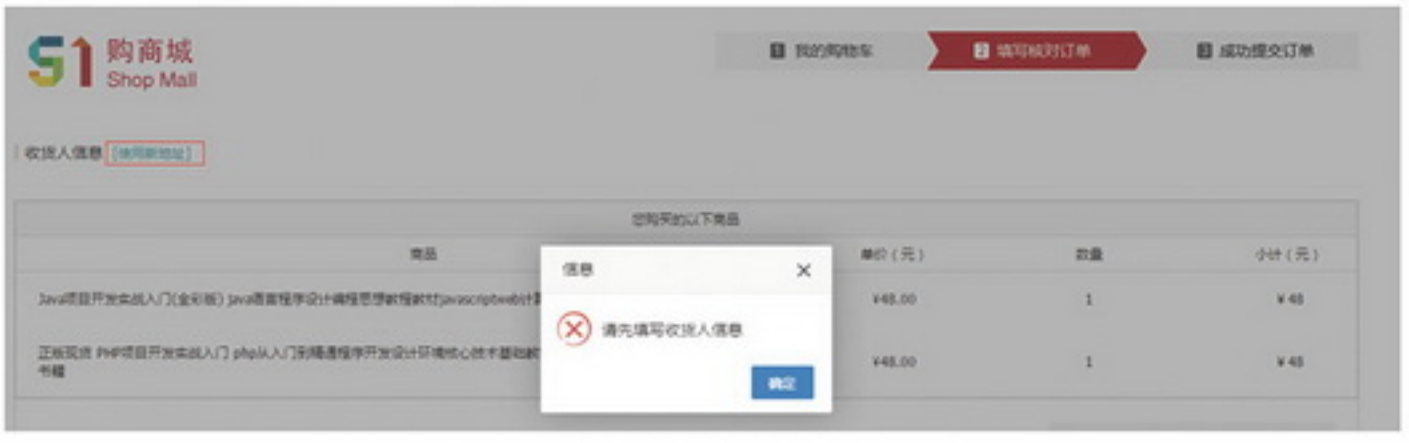


图 16.18　提示“请先填写收货人信息”

单击图 16.18 所示的“使用新地址”超链接，弹出弹层，显示添加地址的表单。该功能主要是通过 Layer.js 插件来实现的。关键代码如下：

<代码位置：光盘\Code\51购商城\Shop\Home\View\Cart\step2.html >

```
<div class="con-y-info ma-bo-35">
        <h3 style="margin-top:30px">收货人信息<b>[<a href="javascript:void(0);"
            onClick="add_edit_address(0);">使用新地址</a>]</b></h3>
        <div id="ajax_address"><!--Ajax 返回收货地址--></div>
 </div>

/**
 * 新增或修改收货地址
 * id为0，为新增,否则是修改
 */
function add_edit_address(id)
{
   if(id > 0){
      var url = "/index.php?m=Home&c=User&a=edit_address&scene=1
                &call_back=call_back_fun&id="+id;          //修改地址
    }else {
        var url = "/index.php?m=Home&c=User&a=add_address&scene=1
                  &call_back=call_back_fun";               //新增地址
    }
    layer.open({
      type: 2,                                            //弹出层类型
      title: '添加收货地址',                               //标题
      shadeClose: true,                                   //是否有遮罩层
      shade: 0.3,                                         //阴影比例
      area: ['880px', '580px'],                           //弹层宽和高
      content: url,                                       //弹层内容
   });
}
```

运行效果如图 16.19 所示。

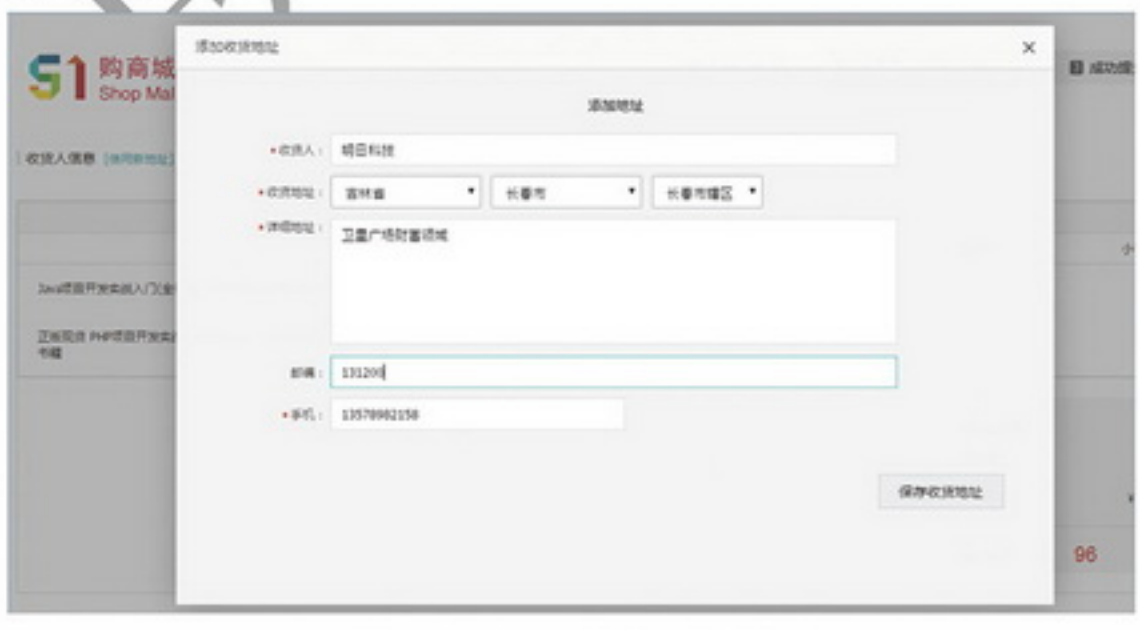


图 16.19　收货地址表单弹层

填写完收货信息后，单击“保存收货地址”按钮，将调用 User 控制器的 add_address() 方法，该方法通过实例化逻辑层的 UsersLogic 类，调用 UsersLogic 类的 add_address() 方法，实现新增或编辑收货信息的功能。保存成功后，运行效果如图 16.20 所示。

图 16.20　保存收货信息

16.6.5 提交订单

在核对订单页面，添加完用户收货信息后，单击“提交订单”按钮，程序会将订单信息写入 order 订单表。此时，order 表中的 pay_status（订单支付状态）为 0，表示该订单未支付。选择“我的商城”→“我的订单”可以查看该订单，如图 16.21 所示。

图 16.21　订单状态

单击“立即支付”按钮，跳转至订单支付页面。在该页面中，列举了第三方支付方式和网银支付方式。由于本地测试无法实现支付功能，且每种支付方式的接口并不相同，读者需要自行编写支付代码。为保证项目流程的完整性，本项目使用 Layer 弹层模拟支付过程。当单击“确认支付方式”按钮时，弹出支付弹层。单击“支付”按钮，表示支付成功，单击“取消”按钮，表示支付失败。如图 16.22 所示。

图 16.22　支付页面

支付成功后，选择“我的商城”→“我的订单”可以查看该订单，此时，订单状态已经由“未支付”更改为“待发货”，如图 16.23 所示。

图 16.23　订单状态变更

16.7 后台模块设计

对于动态网站而言，网站后台起着至关重要的作用，因为需要在后台对数据实现增、删、改、查等操作，从而管理所有前台显示的动态数据。51 购商城后台包括登录模块、后台首页、商品模块和订单模块等。由于篇幅有限，本节只对以上模块做简单介绍和效果展示。

16.7.1 管理员登录模块

设计 51 购商城时，使用了双入口模式，即访问“www.***.com/index.php”进入前台首页，访问“www.***.com/admin.php”进入后台首页（如果管理员未登录，则跳转到后台登录页）。在后台登录页面中，填写管理员账号、密码和验证码（如果验证码看不清楚，可以单击验证码图片刷新该验证码），单击“登录”按钮，即可实现管理员登录。如果没有输入账号、密码或者验证码，则都将给予提示。另外，验证码输入错误也将给予提示。运行效果如图 16.24 所示。

图 16.24　后台登录页面

在后台登录代码中，使用 ThinkPHP 自身封装的 Verify 类实现验证码的生成和检测，调用 Verify 类的 entry() 方法可以生成验证码，调用 check() 方法可以检测验证码。生成验证码的代码如下：

<代码位置：光盘\Code\51购商城\Shop\Application\Admin\Controller\AdminController.class.php >

```
01  public function verify(){
02      $config =    array(
03          'fontSize' => 15,                    //验证码字号
04          'length'   => 4,                     //验证码位数
05          'useNoise' => false,                 //关闭验证码杂点
```

```
06          'imageW'    => 120,                    //图片宽度
07          'imageH'    => 34,                     //图片高度
08          'codeSet'   => '0123456789',           //随机产生0~9中的数字
09      );
10      $Verify = new \Think\Verify($config);
11      $Verify->entry();                          //调用entry()方法生成验证码
12  }
```

在后台登录页模板中，需要在 <img> 标签内调用 verify() 方法，并且使用 JavaScript 的 onClick 点击事件实现单击图片生成新验证码的功能。关键代码如下：

<代码位置：光盘\Code\51购商城\Shop\Application\Admin\View\Admin\login.html >

```
01  <div class="form-group login">
02      <span>验证码</span>
03      <input name="code" class="code" type="text" id="code" />
04      <a> <img class="reloadverify" src="{:U('Admin/Admin/verify')}"   id="imgcode"
05              onClick="this.src=this.src+'?'+Math.random()"></a>
06  </div>
```

16.7.2 后台首页

管理员登录成功后，进入到后台首页。后台首页是对网站重要数据的一个综合概述，它包括“全年营业额”“全年订单数量”“全部商品数量”“本月会员增长数量”“月销售额”等信息。为直观显示，本项目用卡片样式和柱状图形式展现以上数据。如图 16.25 所示。

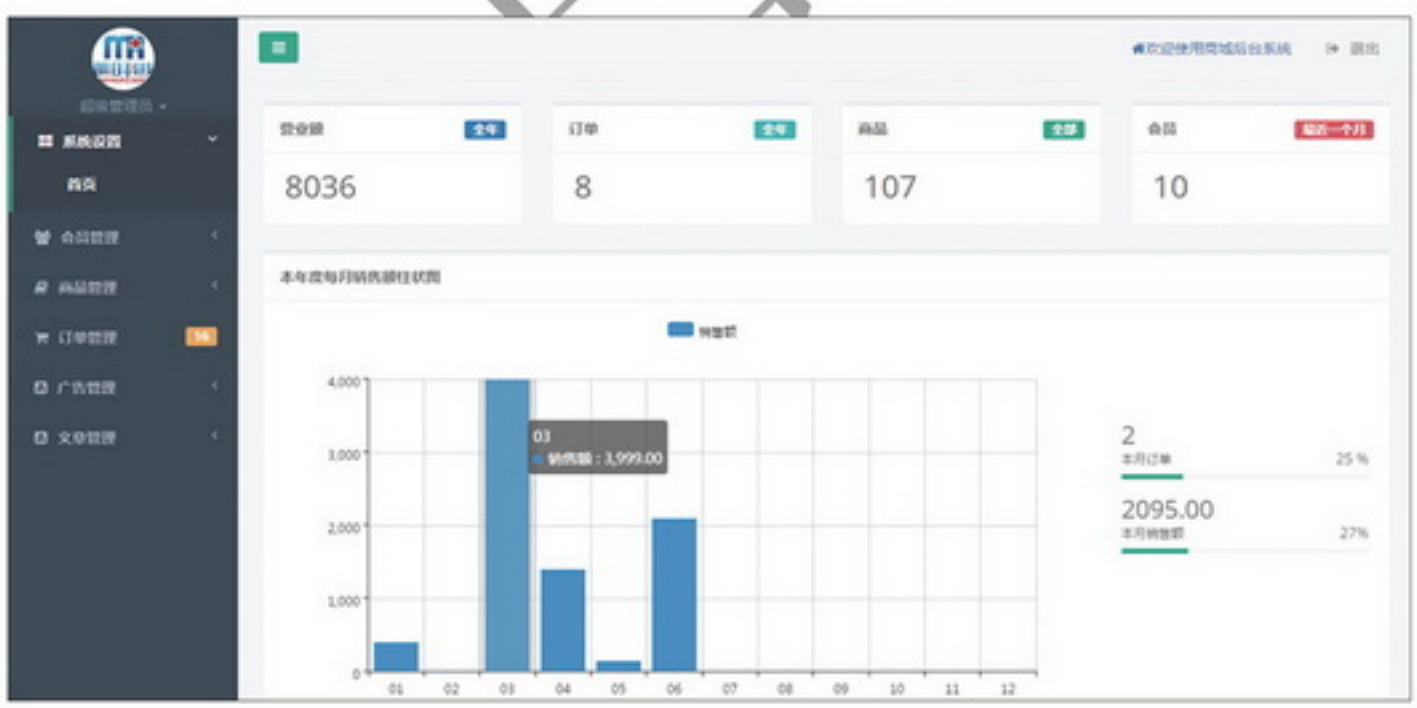


图 16.25　后台首页

16.7.3 商品模块

商品模块是后台最重要的模块之一，包括“商品分类”“商品列表”“商品类型”“商品规格”“商

品属性”“商品评论”。在“商品列表”中，可以实现对商品的增、删、改、查功能。如图 16.26 所示。

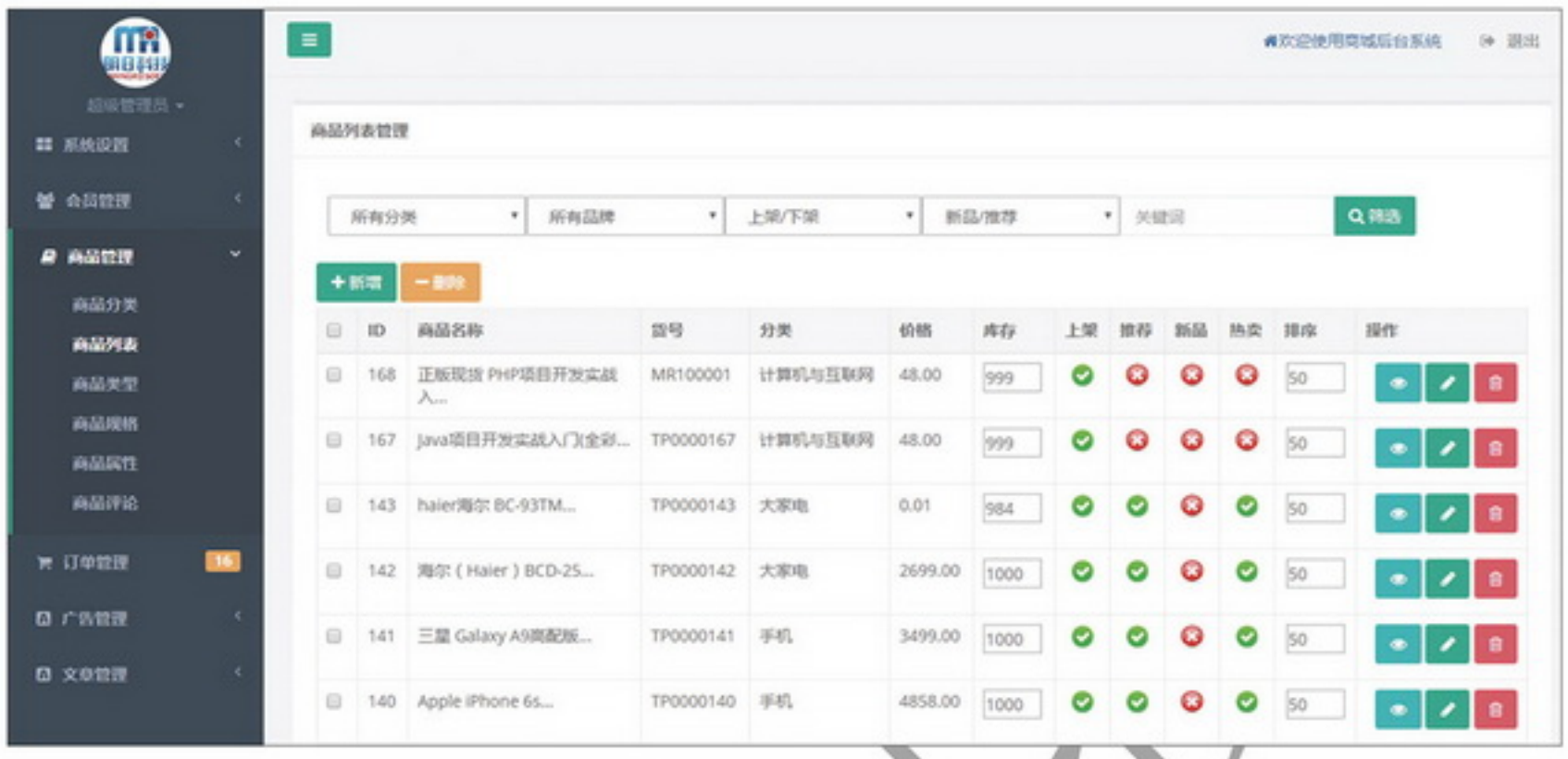


图 16.26　商品列表

在添加商品时，需要从“商品类型”“商品规格”“商品属性”中选择相应的选项。为提高用户体验，使用 Ajax 异步提交的方式来获取相应选项。此外，在添加“商品相册”时，使用了 Plupload 插件来实现多图上传功能。运行效果如图 16.27 所示。

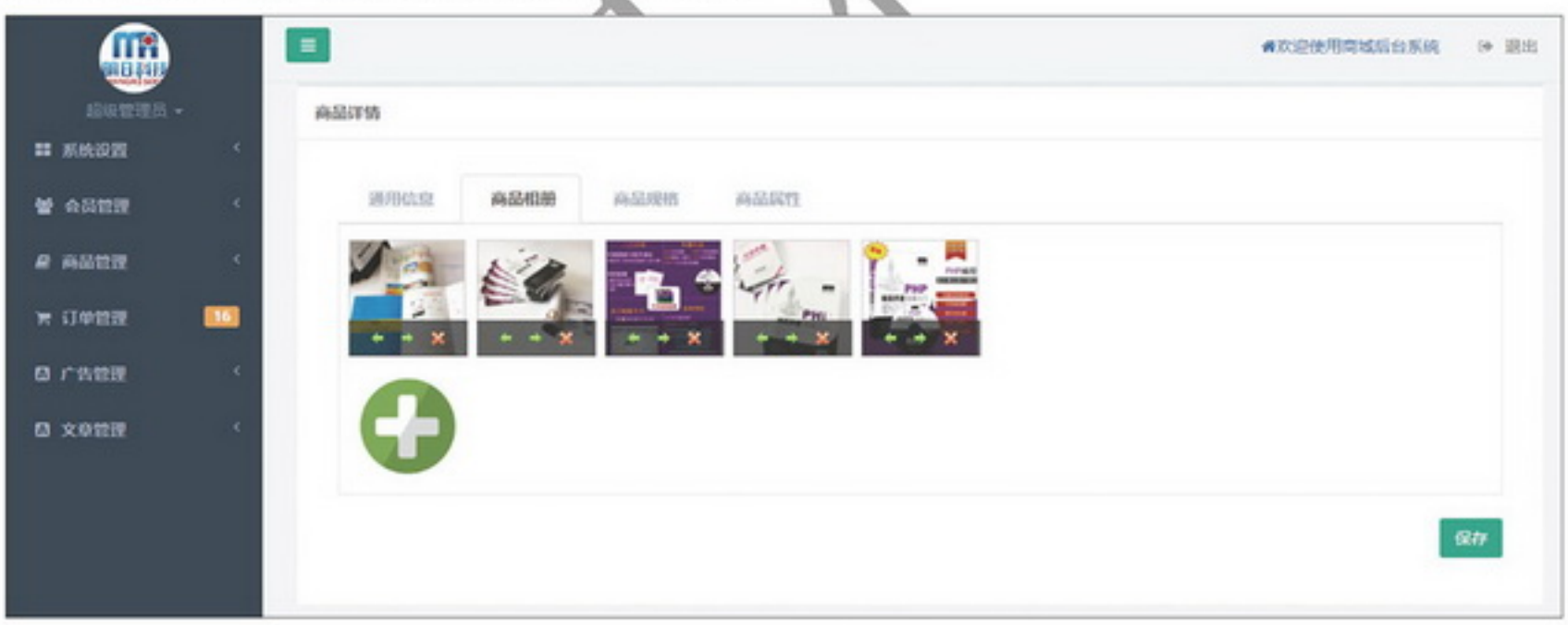


图 16.27　商品相册

16.7.4 订单模块

当用户提交订单后，在后台选择“订单管理”→“订单列表”选项，即可查看该订单，此时的订单状态可能是“未支付”“已支付”或者是“已作废”等，如图 16.28 所示。管理员需要单击“查看”按钮，查看订单详情，并单击“确认”按钮确认订单。确认无误后，在订单列表页单击“发货”按钮，弹出确认框，如图 16.29 所示。当单击“确定”按钮时，使用 Ajax 异步更改订单状态为“已发货”。如图 16.30 所示。

图 16.28　未发货状态

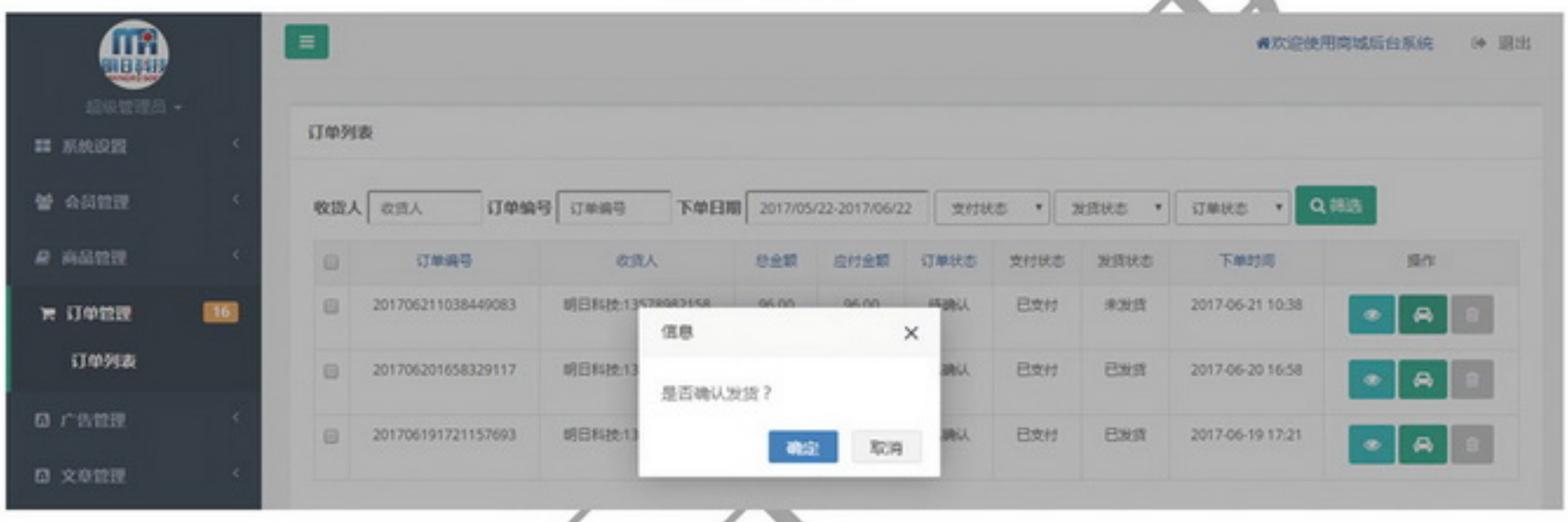


图 16.29　确认发货

图 16.30　发货成功

16.7.5 其他模块

1. 修改管理员密码

单击“超级管理员”右侧的下拉图标，将弹出“修改密码”和“安全退出”选项框。单击“修改密码”，进入修改密码页面。输入“原始密码”“新密码”和“确认密码”后，单击“提交”按钮，即

可更改密码。如图 16.31 所示。

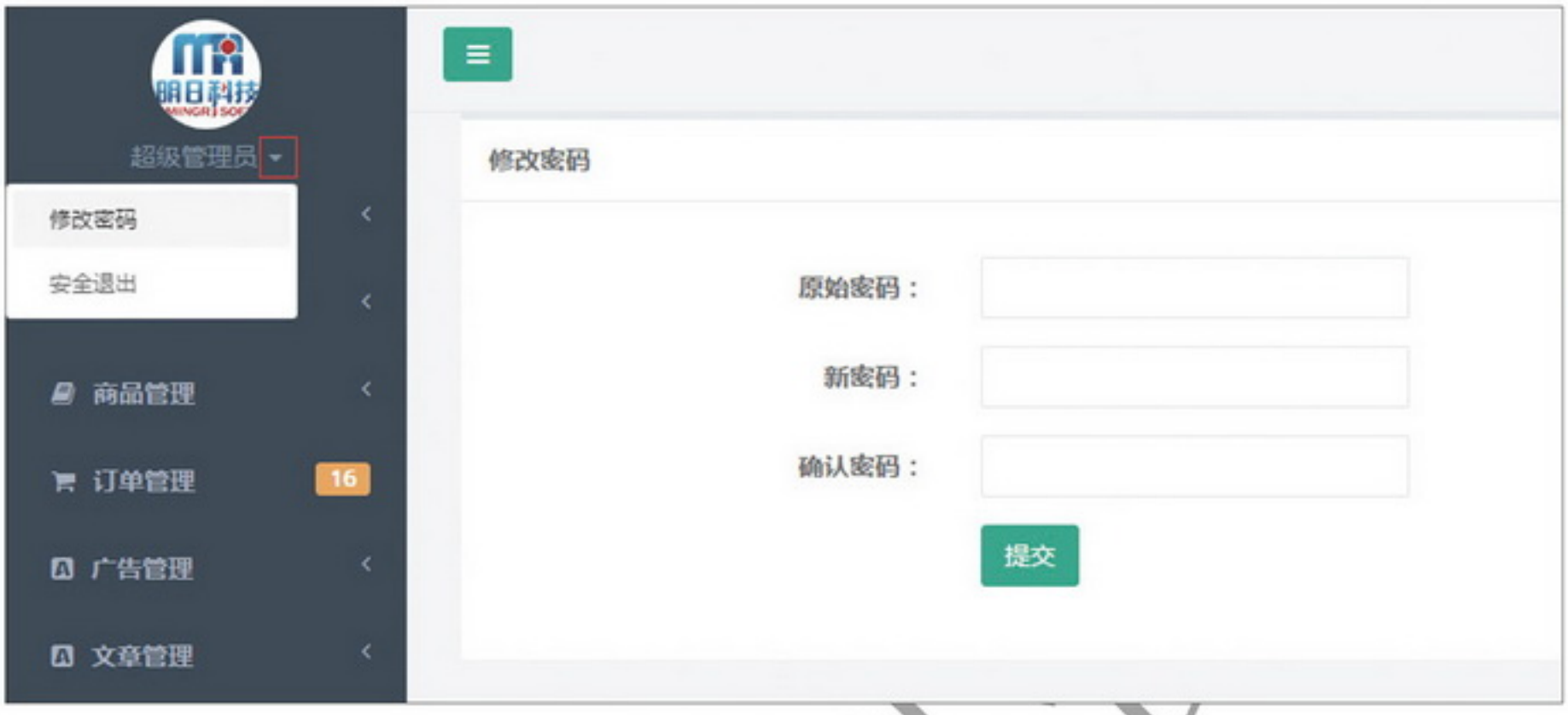


图 16.31　修改密码

2. 会员模块

会员模块包括“会员列表”和“会员等级”。当消费额度满足设置的等级时，即可成为该等级的会员。会员等级如图 16.32 所示。

图 16.32　会员等级

3. 广告模块

广告模块包括“广告列表”和“广告位置”。添加广告时，需要先选择广告位置。该模块可以设置在不同的前台位置显示不同的广告内容。广告位列表如图 16.33 所示。

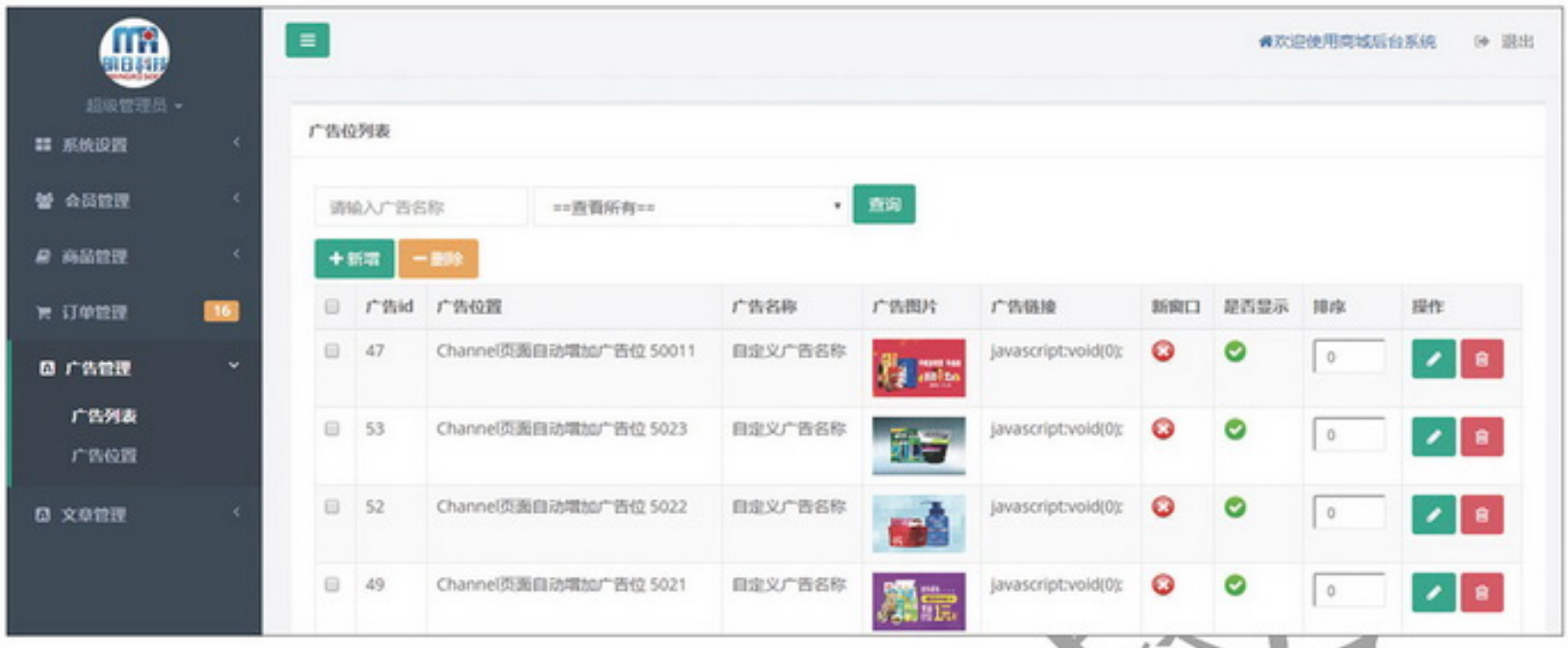


图 16.33　广告列表

16.8 小　结

本章主要介绍如何使用 ThinkPHP 框架实现 51 购商城项目，包括网站的系统功能设计、数据库设计以及前台和后台的主要功能模块设计。希望通过本章的学习，读者能够将前面章节所学知识融会贯通，了解项目开发流程，并掌握 PHP 网站开发技术，为今后的项目开发积累经验。

附录

实例索引

和外星人一起学

基础·入门

分类建议：计算机/程序设计

ISBN 978-7-5692-0868-9

9 787569 208689 >

定价：79.80元

（附1DVD，含视频讲解，程序源码，资源文件及代码查错器）

扫描后输入学习码，
获得在线课程资源

激活学习码网址：
http://www.mingrisoft.com/key.html

责任编辑：张宏亮
美术设计：明日科技