

PHP+MySQL

动态网站开发基础教程

- ◆ 安装和配置PHP运行环境
- ◆ PHP语法、函数、字符串、数组
- ◆ 正则表达式的使用
- ◆ Web页面和PHP程序的交互
- ◆ 日期和时间功能，HTTP控制
- ◆ Cookie和Session
- ◆ PHP对MySQL数据库的操纵
- ◆ PHP对文件与目录的操纵
- ◆ GD库和图像技术
- ◆ PHP面向对象编程
- ◆ JavaScript和Ajax技术
- ◆ ThinkPHP框架的使用



李颖 编著

高等学校计算机应用规划教材

PHP+MySQL 动态网站 开发基础教程

李 颖 编著

清华大学出版社

北 京

内 容 简 介

本书全面讲述了 PHP 程序设计语言的基本原理和技术。全书共分为 16 章, 深入介绍了 PHP 语言的用途与 PHP 环境的搭建、PHP 语法和函数、字符串、数组、正则表达式、PHP 与 Web 页面的交互、日期和时间、HTTP、Cookie 和 Session、数据库编程、用 PHP 操作 MySQL 数据库、文件操作、图像技术、面向对象编程、PHP 与 JavaScript 和 Ajax、ThinkPHP 等内容。

本书内容丰富、结构合理、思路清晰、语言简练流畅、示例翔实, 主要面向 PHP 语言的初学者, 适合作为各种 Web 应用开发培训机构的培训教材、高等院校的 PHP 语言教材, 还可作为 Web 应用开发人员的参考资料。

本书的电子课件、习题答案和实例源文件可以到 <http://www.tupwk.com.cn/downpage> 网站下载。

本书封面贴有清华大学出版社防伪标签, 无标签者不得销售。

版权所有, 侵权必究。侵权举报电话: 010-62782989 13701121933

图书在版编目(CIP)数据

PHP+MySQL 动态网站开发基础教程/李颖 编著. —北京: 清华大学出版社, 2018

(高等学校计算机应用规划教材)

ISBN 978-7-302-50609-6

I. ①P… II. ①李… III. ①PHP 语言—程序设计—高等学校—教材 ②SQL 语言—程序设计—高等学校—教材 IV. ①TP312.8 ②TP311.132.3

中国版本图书馆 CIP 数据核字(2018)第 151305 号

责任编辑: 胡辰浩 李维杰

封面设计: 孔祥峰

版式设计: 思创景点

责任校对: 牛艳敏

责任印制: 李红英

出版发行: 清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址: 北京清华大学学研大厦 A 座 邮 编: 100084

社 总 机: 010-62770175 邮 购: 010-62786544

投稿与读者服务: 010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈: 010-62772015, zhiliang@tup.tsinghua.edu.cn

印 装 者: 清华大学印刷厂

经 销: 全国新华书店

开 本: 185mm×260mm 印 张: 23.75 字 数: 578 千字

版 次: 2018 年 7 月第 1 版 印 次: 2018 年 7 月第 1 次印刷

定 价: 59.00 元

产品编号: 074754-01

前 言

本书将向读者介绍如何用 PHP 语言建立交互式的 Web 站点和应用程序。PHP 是当前最受欢迎的 Web 编程语言之一。作为编程语言，PHP 非常易学。PHP 提供了数以百计的内置函数，并且通过 PHP 引擎的增件，提供了数以千计的函数。本书只是把这门语言的最重要部分介绍给读者，指导读者创建可靠的、高质量的 PHP 应用程序。

本书是针对零基础编程学习者编写的 PHP 入门教程，从初学者角度出发，通过通俗易懂的语言、实用的示例，详细介绍了使用 PHP 进行程序开发所需要掌握的知识和技术。全书共分 16 章，内容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、面向对象编程、PHP 与 Web 页面的交互、MySQL 数据库基础以及综合网站开发等。书中的重要知识都结合具体实例进行讲解，对设计的程序代码也给出了详细注释，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。

下面将逐章介绍本书的内容，这有助于读者决定采用什么样的阅读方法。

第 1 章为 PHP 入门。该章介绍 Web 技术基础、PHP 基础知识、PHP 开发环境的搭建、如何编写 PHP 程序、如何运行 PHP 程序、如何调试 PHP 程序和发生错误时如何处理等。

第 2 章介绍 PHP 语法和函数，主要内容包括 PHP 语法风格、标识符和关键字、常量、变量、变量类型的转换、运算符和表达式、流程控制语句、函数等。

第 3 章介绍字符串处理，主要内容包括创建和访问字符串、求字符串的长度、搜索字符串、字符串替换、格式化字符串、大小写转换等。

第 4 章介绍数组的知识，主要内容包括数组的概念、数组的创建和访问、数组的输出和切割、统计数组个数、遍历数组、多维数组的使用、数组的常用操作等。

第 5 章介绍正则表达式，首先介绍什么是正则表达式，其次讲解正则表达式的语法规则、模式匹配、正则表达式的使用、用 `preg_match_all()` 函数实现多次匹配、用 `preg_grep()` 函数搜索数组、用正则表达式实现文本替换功能等。

第 6 章介绍如何使用 PHP 建立交互式 Web 表单，具体内容包括如何建立 HTML 表单，如何在 PHP 脚本中捕获表单数据，如何用 PHP 生成动态表单。此外，还介绍文件的上传等。

第 7 章介绍常用的日期和时间功能以及 HTTP 请求，首先详细介绍日期和时间处理，包括许多实用的内置日期函数以及 `DateTime` 类，然后介绍如何处理 HTTP 请求-响应过程中的请求头和响应头等。

第 8 章介绍 Cookie 和 Session 技术，首先介绍 Cookie 的概念，Cookie 变量的创建、读取、删除，以及 Cookie 变量的生命周期，其次介绍 Session 的概念，Session 变量的创建，设置

Session 的有效时间,使用 Session 对用户权限进行控制,删除和销毁 Session 变量;最后介绍 Session 和 Cookie 的区别等。

第 9 章讨论 MySQL 控制台下的 MySQL 数据库编程,主要内容包括 MySQL 数据库的启动和连接、MySQL 数据库操作、数据表操作、数据表记录的更新操作、数据库的备份和还原操作等。

第 10 章介绍如何使用 PHP 语言操纵 MySQL 数据库,主要内容包括 MySQL 服务器的连接、SQL 语言的执行、数据库记录的读取、如何使用 PHP 语言对数据库执行增删改查操作等。

第 11 章介绍文件操作,首先介绍文件和目录基础、获取文件相关的属性信息、打开和关闭文件、文件的读写操作、文件权限的设置,其次介绍文件的赋值、重命名和删除,最后介绍目录的操作等内容。

第 12 章介绍如何使用 PHP 创建和处理图像,首先介绍计算机图形的一些理论,其次介绍如何创建新的图像,如何修改现有的图像,最后介绍如何使用 PHP 做一些常用的图像处理操作。

第 13 章主要介绍 PHP 面向对象技术,讨论如何使用 PHP 语言实现面向对象技术中的类、对象、类中的任何成员和属性、类中的静态成员、方法重载、接口、继承等内容。

第 14 章主要介绍 JavaScript 和 Ajax 技术与 PHP 语言的交互,主要内容包括 JavaScript 语言基础和 Ajax 技术的使用。

第 15 章主要介绍流行的 ThinkPHP 框架,重点介绍的是 ThinkPHP 的安装、目录架构、MVC 概念在 ThinkPHP 中的体现、URL 和路由等内容。

第 16 章主要通过两个综合实例来对本书介绍的 PHP 知识进行巩固,第一个综合实例通过 PHP 原生语言来实现,第二个综合实例通过 ThinkPHP 框架来实现。

本书内容丰富、结构合理、思路清晰、语言简练流畅、示例翔实,每一章的引言部分概述了该章的作用和内容。在每一章的正文中,结合所讲述的关键技术和难点,穿插了大量极富实用价值的示例。每一章末尾都安排了有针对性的思考题和练习题,思考题有助于读者巩固所学的基本概念,练习题有助于培养读者的实际动手能力、增强对基本概念的理解和实际应用能力。

本书是专为 PHP 编程初学者编写的。如果读者以前使用过其他编程语言,如 Java、C# 或 Perl,则学习 PHP 会比较容易。本书适合作为高等院校 PHP 网站开发、Web 应用程序开发课程的教材,也可作为 Web 开发人员的参考资料。

全书由吉林师范大学的李颖编写并统稿。除封面署名的作者外,参加本书编写的人员还有周爱萍、屈文斌、万鑫、张春辉、梅泉滔、杨永好、郑梦成、孙红胜、何玉华、李文静、冯波、马金帅、张晓晗、张梦甜、李亮等。由于作者水平有限,本书难免有不足之处,欢迎广大读者批评指正。我们的电子邮箱是 huchenhao@263.net,电话是 010-62796045。

本书的电子课件、习题答案和实例源文件可以到 <http://www.tupwk.com.cn/downpage> 网站下载。

作者
2018年5月

目 录

第 1 章 PHP 入门	1	2.1.2 PHP 注释	27
1.1 Web 技术基础	1	2.2 PHP 标识符与关键字	28
1.1.1 Web 技术概述	1	2.2.1 PHP 标识符	28
1.1.2 主流的 Web 应用平台	3	2.2.2 关键字	28
1.1.3 Web 工作原理	3	2.3 PHP 常量	29
1.1.4 常用的 Web 技术	4	2.3.1 常量的定义	29
1.2 PHP 基础知识	7	2.3.2 预定义常量	30
1.2.1 PHP 概述	7	2.4 PHP 变量	32
1.2.2 PHP 的优势	7	2.4.1 变量的声明和使用	32
1.2.3 PHP 的应用领域	8	2.4.2 PHP 的数据类型	33
1.2.4 常用的 PHP 开发工具	8	2.4.3 检测变量的数据类型	34
1.2.5 如何学好 PHP	8	2.4.4 可变变量	35
1.2.6 PHP 学习资源	9	2.4.5 变量的作用域	35
1.3 PHP 开发环境的搭建	12	2.5 变量类型的转换	37
1.3.1 WAMP 环境的搭建	12	2.5.1 自动类型转换	38
1.3.2 LAMP 环境的搭建	17	2.5.2 强制类型转换	38
1.3.3 扩展库	19	2.6 PHP 运算符与表达式	39
1.4 第一个 PHP 程序	21	2.6.1 运算符的类型	40
1.5 调试与错误处理	22	2.6.2 运算符的优先级	44
1.5.1 使用自带的报错功能	22	2.7 流程控制语句	45
1.5.2 引进调试工具	23	2.7.1 选择结构	45
1.5.3 调试业务逻辑错误	24	2.7.2 循环结构	49
1.5.4 调试非功能性错误	24	2.8 函数	53
1.6 本章小结	24	2.8.1 定义和调用函数	53
1.7 思考和练习	25	2.8.2 在函数间传递参数	54
第 2 章 PHP 语法和函数	26	2.8.3 从函数中返回值	55
2.1 PHP 的语法风格	26	2.8.4 变量函数	56
2.1.1 PHP 标记	26	2.8.5 对函数的引用	56
		2.8.6 取消引用	57

2.9 本章小结	58	4.2 数组的创建与访问	76
2.10 思考和练习	58	4.2.1 创建数组	76
第3章 字符串	59	4.2.2 访问数组的元素	76
3.1 创建和访问字符串	59	4.2.3 修改元素值	77
3.1.1 创建字符串	59	4.2.4 新增数组元素	77
3.1.2 在字符串中引用变量	60	4.2.5 输出所有数组元素	78
3.1.3 自定义分隔符	61	4.2.6 切割数组	79
3.1.4 求字符串的长度和单词 个数	62	4.2.7 统计数组中元素的个数	80
3.1.5 访问字符串中的字符	62	4.2.8 逐个访问数组的元素	80
3.2 搜索字符串	63	4.3 用 foreach 遍历数组	83
3.2.1 用 strstr()函数搜索字符串	63	4.3.1 用 foreach 遍历数组的 每个值	83
3.2.2 用 strpos()和 strrpos()函数 定位字符串位置	63	4.3.2 用 foreach 遍历数组的键 和值	83
3.2.3 用 substr_count()函数统计 字符串的出现次数	64	4.3.3 用 foreach 遍历修改数组 元素的值	84
3.2.4 用 strpbrk()函数搜索 字符集	65	4.4 多维数组	85
3.3 在字符串中进行字符替换	65	4.4.1 创建多维数组	85
3.3.1 用 str_replace()函数替换 全部搜索字符串	65	4.4.2 访问多维数组的元素	86
3.3.2 用 substr_replace()替换字符串 的部分内容	65	4.4.3 多维数组的遍历	86
3.3.3 用 strtr()函数变换字符	66	4.5 数组的操作	87
3.4 格式化字符串	66	4.5.1 数组排序	88
3.4.1 通用的格式化函数 printf()和 sprintf()	66	4.5.2 添加和删除数组元素	92
3.4.2 删除空白符的函数 trim()、 ltrim()和 rtrim()	70	4.5.3 合并数组	96
3.4.3 填充字符串函数 str_pad()	71	4.5.4 数组与字符串之间的转换	97
3.4.4 自动换行函数 wordwrap()	71	4.5.5 把数组转换为变量列表	98
3.4.5 格式化数值函数 number_format()	72	4.6 本章小结	99
3.5 字母大小写转换	73	4.7 思考和练习	99
3.6 本章小结	74	第5章 正则表达式	101
3.7 思考和练习	74	5.1 什么是正则表达式	101
第4章 数组	75	5.1.1 正则表达式的概念	101
4.1 数组概述	75	5.1.2 正则表达式的使用场景	102
		5.2 正则表达式的语法规则	103
		5.2.1 行定位符(^和\$)	103
		5.2.2 单词定界符(\b、\B)	103
		5.2.3 字符类([])	103
		5.2.4 选择字符()	104
		5.2.5 连字符(-)	104

5.2.6	排除字符([\^])	104	6.4.1	通过\$_POST[]接收表单数据	126
5.2.7	限定符(?*+{n,m})	104	6.4.2	通过\$_GET[]接收表单数据	126
5.2.8	点字符(.)	105	6.5	在网页中嵌入 PHP 脚本	127
5.2.9	转义字符(\)	105	6.5.1	在 HTML 标记中添加 PHP 脚本	127
5.2.10	反斜线(\)	106	6.5.2	对表单元素的 value 属性进行赋值	127
5.2.11	圆括号字符(())	107	6.6	在 PHP 中获取表单数据	127
5.2.12	反向引用	107	6.6.1	获取基本表单元素的值	128
5.3	PHP 中的模式匹配	107	6.6.2	获取单选按钮的值	129
5.4	正则表达式的使用	108	6.6.3	获取复选框的值	130
5.4.1	匹配字面字符	108	6.6.4	获取下拉列表框/菜单列表框的值	130
5.4.2	用字符类匹配字符类型	109	6.6.5	获取文件域的值	131
5.4.3	多次匹配同一字符	110	6.7	对 URL 传递的参数进行编码和解码	132
5.4.4	贪婪匹配法和非贪婪匹配法	110	6.7.1	对 URL 传递的参数进行编码	132
5.4.5	用子模式分组模式	110	6.7.2	对 URL 传递的参数进行解码	133
5.4.6	引用前面的子模式匹配串	111	6.8	Web 服务器端的其他数据采集方法	134
5.4.7	匹配多个模式	111	6.8.1	预定义变量\$_REQUEST	134
5.5	用 preg_match_all()函数实现多次匹配	111	6.8.2	预定义变量\$_SERVER	134
5.6	用 preg_grep()函数搜索数组	115	6.9	文件上传	136
5.7	文本替换	116	6.9.1	配置 php.ini 以实现 PHP 文件上传功能	136
5.7.1	用 preg_replace()替换文本	116	6.9.2	文件上传函数	137
5.7.2	用 preg_replace_callback()替换文本	117	6.9.3	多文件上传	139
5.8	本章小结	118	6.10	本章小结	140
5.9	思考和练习	118	6.11	思考和练习	140
第 6 章	PHP 与 Web 页面的交互	119	第 7 章	日期和时间、HTTP	141
6.1	表单	120	7.1	日期和时间的处理	141
6.1.1	创建表单	120	7.1.1	时间戳	141
6.1.2	表单元素	120	7.1.2	获取当前日期和时间	142
6.2	将表单保存为 HTML 文件	123	7.1.3	创建时间戳	142
6.3	获取表单数据的两种方法	125			
6.3.1	通过 POST 方式提交表单	125			
6.3.2	通过 GET 方式提交表单	125			
6.4	PHP 参数传递的常用方法	126			

7.1.4	转换时间戳	144	9.1.2	数据库常见术语	175
7.1.5	格式化日期字符串	146	9.2	MySQL 的启动与连接	176
7.1.6	检查日期值	148	9.2.1	启动 MySQL 服务	176
7.1.7	毫秒的使用	148	9.2.2	连接和断开 MySQL 数据库	178
7.1.8	DateTime	150	9.2.3	配置 Path 环境变量	178
7.2	HTTP 的使用	152	9.3	操作 MySQL 数据库	179
7.2.1	HTTP 请求	152	9.3.1	创建和查看数据库	180
7.2.2	HTTP 响应	153	9.3.2	删除数据库	181
7.2.3	修改 HTTP 响应方式	155	9.3.3	选择数据库	181
7.3	本章小结	156	9.3.4	综合实例——数据库的创建 和删除	181
7.4	思考和练习	156	9.4	操作数据表	184
第 8 章	Cookie 和 Session	157	9.4.1	创建数据表	184
8.1	管理 Cookie	157	9.4.2	查看表结构	186
8.1.1	了解 Cookie	157	9.4.3	修改表结构	187
8.1.2	创建 Cookie	159	9.4.4	重命名数据表	187
8.1.3	读取 Cookie	159	9.4.5	删除数据表	187
8.1.4	删除 Cookie	160	9.5	数据记录的更新操作	188
8.1.5	Cookie 的生命周期	161	9.5.1	添加数据记录	188
8.2	管理 Session	161	9.5.2	查询数据记录	188
8.2.1	了解 Session	161	9.5.3	修改数据记录	189
8.2.2	创建 Session	162	9.5.4	删除数据记录	189
8.2.3	设置 Session 的有效时间	163	9.6	MySQL 数据库的备份与还原	190
8.2.4	通过 Session 对用户操作 权限进行判断	164	9.6.1	使用 MYSQLDUMP 命令 备份数据库	190
8.2.5	删除和销毁 Session	167	9.6.2	使用 mysql 命令还原 数据库	190
8.2.6	Session 和 Cookie 的区别	168	9.7	本章小结	191
8.2.7	Session 和 Cookie 的应用	169	9.8	思考和练习	191
8.3	Session 的高级应用	169	第 10 章	用 PHP 操作 MySQL 数据库	192
8.3.1	Session 临时文件	169	10.1	PHP 访问 MySQL 数据库的基本 步骤	192
8.3.2	Session 缓存	169	10.2	PHP 操作 MySQL 数据库的 方法	194
8.3.3	Session 自动回收	170	10.2.1	连接 MySQL 服务器	194
8.3.4	php.ini 中的 Session 配置	171	10.2.2	选择 MySQL 数据库	195
8.4	本章小结	172			
8.5	思考和练习	173			
第 9 章	数据库编程	174			
9.1	MySQL 简介	174			
9.1.1	客户端/服务器(Client/Server) 软件	174			

10.2.3	执行 SQL 语句	196	11.7.2	目录对象	227
10.2.4	mysql_fetch_array() 函数	197	11.7.3	区分文件和目录	227
10.2.5	mysql_fetch_object() 函数	199	11.8	本章小结	229
10.2.6	mysql_fetch_row()函数	199	11.9	思考和练习	229
10.2.7	mysql_num_rows()函数	200	第 12 章 图像技术	230	
10.2.8	释放资源	201	12.1	计算机图形基础	230
10.2.9	关闭连接	201	12.1.1	色彩原理	230
10.3	PHP 操作数据库	202	12.1.2	坐标系	231
10.3.1	PHP 操作数据库	202	12.1.3	图像类型	231
10.3.2	动态添加用户信息	203	12.2	使用 GD 库	232
10.3.3	查询数据信息	205	12.2.1	GD&PHP 可以处理的 文件类型	232
10.3.4	修改数据	206	12.2.2	在 PHP 中启用 GD 库	232
10.3.5	删除数据	209	12.3	创建图像	233
10.4	本章小结	210	12.3.1	新建图像	234
10.5	思考和练习	210	12.3.2	颜色分配	234
第 11 章 文件操作	211		12.3.3	输出图像	235
11.1	文件与目录基础	211	12.3.4	在图像上进行绘制	236
11.2	获取文件的信息	212	12.4	处理图像	241
11.2.1	获取文件的时间属性	212	12.4.1	打开图像	241
11.2.2	从路径获取文件名	212	12.4.2	添加水印	242
11.3	打开和关闭文件	213	12.4.3	制作缩略图	245
11.3.1	用 fopen()打开文件	213	12.4.4	添加标准化文本	246
11.3.2	用 fclose()关闭文件	214	12.4.5	使用 TrueType 字体	247
11.4	文件的读写	214	12.5	本章小结	249
11.4.1	读写字符串	215	12.6	思考和练习	249
11.4.2	文件末尾的测试	217	第 13 章 面向对象编程	250	
11.4.3	一次读取一行内容	218	13.1	面向对象编程介绍	250
11.4.4	读取 CSV 文件	218	13.2	创建类和对象	252
11.4.5	读取和写入整个文件	219	13.2.1	创建类	252
11.4.6	随机存取文件数据	221	13.2.2	类的成员	252
11.5	文件的权限	222	13.2.3	实例化类	254
11.5.1	改变文件的权限	222	13.3	创建和使用属性	255
11.5.2	检查文件权限	223	13.3.1	声明属性	255
11.6	文件的复制、重命名和删除	223	13.3.2	属性的可见性	255
11.7	目录	224	13.3.3	访问属性	256
11.7.1	其他目录函数	225	13.3.4	静态属性	257

13.3.5	类常量	257	14.3.2	循环语句	288
13.4	方法	258	14.3.3	跳转语句	290
13.4.1	创建方法	258	14.4	JavaScript 事件	291
13.4.2	方法的可见性	258	14.5	调用 JavaScript 脚本	291
13.4.3	方法的调用	258	14.5.1	在 HTML 中嵌入 JavaScript 脚本	291
13.4.4	方法的参数和返回值	259	14.5.2	应用 JavaScript 事件调用自定义函数	292
13.4.5	在方法中访问对象的属性	259	14.5.3	在 PHP 动态网页中引用 JS 文件	292
13.4.6	静态方法	261	14.6	在 PHP 中调用 JavaScript	293
13.5	用 <code>__get()</code> 、 <code>__set()</code> 和 <code>__call()</code> 重载对象	261	14.6.1	使用 JavaScript 脚本验证表单元素是否为空	293
13.5.1	用 <code>__get()</code> 和 <code>__set()</code> 方法重载属性访问	262	14.6.2	使用 JavaScript 脚本制作二级导航菜单	294
13.5.2	用 <code>__call()</code> 重载方法调用	263	14.6.3	使用 JavaScript 脚本控制文本域和复选框	297
13.5.3	其他重载方法	265	14.7	Ajax 技术	298
13.6	继承与接口	266	14.7.1	Ajax 的概念	298
13.6.1	重载父类的方法	268	14.7.2	Ajax 的开发模式	298
13.6.2	保留父类的功能	270	14.7.3	Ajax 的优点	299
13.6.3	用 <code>final</code> 类和方法阻止继承和重载	270	14.7.4	XMLHttpRequest	300
13.6.4	抽象类和抽象方法	271	14.7.5	在 Ajax 开发过程中需要注意的问题	302
13.6.5	接口	273	14.7.6	用户重名检测	303
13.7	自动加载类文件	276	14.8	本章小结	305
13.8	序列化类对象	276	14.9	思考和练习	305
13.9	判断一个对象所属的类	278	第 15 章	ThinkPHP	306
13.10	本章小结	279	15.1	ThinkPHP 简介	306
13.11	思考和练习	280	15.1.1	ThinkPHP 的安装	307
第 14 章	PHP 与 JavaScript 和 Ajax	281	15.1.2	ThinkPHP 概述	308
14.1	了解 JavaScript	281	15.2	ThinkPHP 架构	309
14.1.1	什么是 JavaScript	281	15.2.1	ThinkPHP 的目录结构	310
14.1.2	JavaScript 的功能	282	15.2.2	自动生成项目目录	311
14.2	JavaScript 语言基础	282	15.2.3	命名规范	314
14.2.1	JavaScript 数据类型	282	15.2.4	资源目录	315
14.2.2	JavaScript 变量	283	15.2.5	调试模式配置	315
14.2.3	JavaScript 注释	284	15.2.6	控制器	315
14.3	JavaScript 流程控制语句	285			
14.3.1	条件语句	285			

15.2.7	视图	318	16.1.6	程序员完成网站功能 实现	337
15.2.8	数据读取	319	16.1.7	网站上线测试	337
15.3	URL 和路由	320	16.1.8	网站推广	337
15.3.1	URL 访问	320	16.2	留言板	338
15.3.2	参数传入	321	16.2.1	留言板制作预备知识	338
15.3.3	隐藏入口	322	16.2.2	留言板功能需求分析	338
15.3.4	定义路由	322	16.2.3	留言板数据库表设计	339
15.3.5	URL 生成	323	16.2.4	留言信息的读取展示	339
15.4	请求与响应	323	16.2.5	留言表单及留言处理	341
15.4.1	请求对象	323	16.2.6	后台管理登录	344
15.4.2	请求信息	325	16.2.7	PHP 留言板系统后台 管理	345
15.4.3	响应信息	326	16.2.8	后台管理回复及留言删除 处理	346
15.5	数据库操作	327	16.3	个人博客	347
15.5.1	数据库配置	327	16.3.1	功能阐述	347
15.5.2	原生查询	327	16.3.2	功能结构	348
15.5.3	链式操作	328	16.3.3	系统预览	348
15.5.4	事务支持	328	16.3.4	数据库设计	349
15.6	模型和关联	329	16.3.5	数据表设计	349
15.6.1	模型定义	329	16.3.6	连接数据库	351
15.6.2	基础操作	330	16.3.7	自动生成项目目录	351
15.7	视图和模板	332	16.3.8	控制器的设置	352
15.7.1	模板输出	332	16.3.9	视图设置	355
15.7.2	分页输出	333	16.3.10	后台管理程序架构 分析	360
15.7.3	渲染内容	334	16.3.11	数据模型和数据校验的 设计	362
15.8	本章小结	334	16.4	本章小结	365
15.9	思考和练习	335	16.5	思考和练习	365
第 16 章	综合实例	336	参考文献	367	
16.1	网站开发流程	336			
16.1.1	确定建站目标	336			
16.1.2	进行需求分析	336			
16.1.3	绘制网站原型	337			
16.1.4	系统整理所需资料	337			
16.1.5	与网站设计美工确定布局 和风格	337			

第 1 章

PHP 入门

PHP 是一门服务器端程序设计语言。除 PHP 外，还有 JSP、ASP、ASP.NET 等重要的服务器端程序设计语言。服务器端程序语言主要运行在服务器端，用于处理来自浏览器端的客户请求；服务器端程序根据请求处理好之后，将处理结果返回到浏览器端，供用户在浏览器端查看或进行下一步交互。

PHP 语言最强大和最重要的特征就是跨平台和面向对象。本章首先介绍 Web 技术基础，其次介绍 PHP 语言的基础知识、开发环境的搭建，接着编写一个简单的 PHP 程序，最后介绍开发过程中遇到错误如何调试和处理。通过本章的学习，读者能够对 PHP 语言有一个整体上的认识，为后期学习 PHP 语言的具体内容打下良好的基础。

本章的学习目标：

- 掌握 Web 技术基础知识。
- 了解 PHP 基础知识。
- 掌握 Windows 与 Linux 操作系统平台上 PHP 开发环境的搭建。
- 掌握编写、运行和调试 PHP 程序的方法。

1.1 Web 技术基础

没有 Web 技术，就没有 PHP 的诞生。因此，在学习 PHP 之前，首先来了解 Web 技术相关的一些知识。

1.1.1 Web 技术概述

1. Web 的定义

百度百科对 Web 的定义是：Web 一般指 WWW(World Wide Web)，即全球广域网，也称为万维网，是一种基于超文本和 HTTP 的、全球性的、动态交互的、跨平台的分布式图形信息系统，是建立在 Internet 上的一种网络服务，为浏览者在 Internet 上查找和浏览信息提供了图形化的、易于访问的直观界面，其中的文档及超链接将 Internet 上的信息节点组织成一个互为关联的网状结构。简而言之，就是指互联网，人们通常说的“上网”就是访问互联网。

2. Web 的表现形式

Web 的表现形式有 3 种：超文本、超媒体、超文本传输协议。

- 超文本：百度百科将超文本定义为一种用户接口方式，用以显示文本以及与文本相关的内容。通俗易懂地说，主要是指链接到其他字段或文档的超文本链接(超链接)，允许浏览者从当前阅读位置直接跳转到超文本链接所指向的文字。一般具有这种特性的文档就是网页——用 HTML(HyperText Markup Language, 超文本标记语言)语言书写的文档。
- 超媒体：超级媒体的简称，是超文本和多媒体在信息浏览环境下的结合，使得通过网页不仅能从一段文本跳转到另一段文本，还可以播放一段声音、显示一个图形，甚至可以播放一段动画。由此可见，超媒体使网页变得丰富多彩。
- 超文本传输协议(HyperText Transfer Protocol, HTTP)是互联网上应用最为广泛的一种网络传输协议。浏览器和服务端之间的交互都是通过 HTTP 协议来实现的。

3. C/S 和 B/S 架构

C/S 和 B/S 架构是最流行的两种软件架构。

(1) C/S 架构

C/S(Client/Server)架构，即客户端/服务器架构。客户端包含一个或多个在用户计算机上运行的程序；而服务器有两种，一种是数据库服务器，客户端通过数据库连接访问数据库服务器上的数据；另一种是 Socket 服务器，服务器上的程序通过 Socket 与客户端的程序通信。

C/S 架构也可以看成胖客户端架构。因为客户端需要实现绝大多数的业务逻辑和界面展示功能。这种架构中，作为客户端的部分需要承受很大的压力，因为显示逻辑和事务处理都包含在其中，通过与数据库的交互(通常是 SQL 或存储过程的实现)来达到持久化数据，以此满足实际项目的需要。

C/S 架构的优点有：界面和操作可以很丰富；安全性能很容易得到保证；实现多层认证也不难，由于只有一层交互，因此响应速度较快。缺点是：适用面窄，通常用于局域网中；用户群固定，由于程序需要安装才可使用，因此不适合面向一些不可知的用户；维护成本高，只要升级，所有客户端的程序都需要改变。

(2) B/S 架构

B/S(Browser/Server)架构，即浏览器/服务器架构。Browser 指的是 Web 浏览器，Server 是指用某种语言编写的服务器端程序。在 B/S 架构中，业务逻辑处理一般很少在浏览器端实现，主要放在服务器端用服务器端程序语言(后端语言)实现。一般情况下，浏览器、服务器和数据库构成了网站开发的三层架构。采用 B/S 架构的系统不需要特别安装客户端组件，用浏览器执行即可。

B/S 架构中，显示逻辑(即网页)交给 Web 浏览器解释执行，业务逻辑放在服务器端，用后端语言编写程序来处理，这样减少了客户端浏览器的压力。由于客户端浏览器只需要负责页面呈现和用户交互，因此也被称为瘦客户端。

B/S 架构的优点是：客户端无须安装组件，有浏览器即可；B/S 架构可以直接放在 Internet 上，供多用户访问，交互性较强；B/S 架构无须升级多个客户端组件，更新服务器端程序即可。

B/S 架构的缺点是：在跨浏览器上，B/S 架构在呈现上不尽如人意，要达到 C/S 架构的呈现程度更难；在速度和安全性上需要花费巨大的成本，这是 B/S 架构的最大问题；客户端/服务器的交互是请求-响应模式，常常需要刷新页面。

1.1.2 主流的 Web 应用平台

动态网站服务器平台至少要包括：操作系统+Web 服务器+应用程序服务+数据库。好的动态网站服务器是由多方面因素决定的，如个人喜好、部署费用、安全机制等。目前主流的 3 种 Web 平台分别是 LAMP、J2EE 和 ASP.NET，它们的运行环境组合如下：

- LAMP: Linux+Apache+MySQL+PHP
- J2EE: UNIX+Tomcat+Oracle+JSP
- ASP.NET: Windows Server+IIS+SQL Server+ASP.NET

LAMP、J2EE 和 ASP.NET 平台各有优缺点，三者的比较如表 1-1 所示。

表 1-1 3 种 Web 平台的比较

性能比较	LAMP	J2EE	ASP.NET
运行速度	较快	快	快
开发速度	快	慢	快
运行损耗	一般	较小	较大
难易程度	简单	难	简单
运行平台	Linux/UNIX/Windows	绝大多数平台	Windows 平台
扩展性	好	好	较差
安全性	好	好	较差
应用程度	较广	较广	较广
建设成本	非常低	非常高	高

1.1.3 Web 工作原理

Web 应用程序采用的是 B/S 架构。Web 工作原理就是：B/S 架构模式下，Web 服务器如何接收用户通过浏览器发来的请求，如何处理这些请求，以及如何将处理的结果返回给浏览器呈现给用户查看(即进行下一步交互)的过程。下面以 Apache 和 PHP 为基础，详细介绍 Web 工作原理。

1. 当直接请求静态 HTML 页面时

当客户通过浏览器直接请求静态的 HTML 页面时，即请求的页面不带应用程序和数据库操作时，Web 服务器将根据访问地址，找到存放该页面的地址，然后将该页面直接返回给客户端浏览器，如图 1-1 所示。

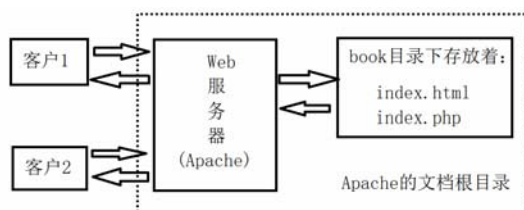


图 1-1 当直接请求静态 HTML 页面时

2. 当访问的页面带应用程序时

当客户请求一个带应用程序(如 PHP 程序)的页面时，Web 服务器将寻找到该文件，并通知

PHP 应用服务器。PHP 应用服务器(即 PHP 解释器)逐条解释程序,将其翻译成 HTML 静态页面,然后将该 HTML 静态页面返回给 Web 服务器,由 Web 服务器返回给浏览器,呈现给客户,如图 1-2 所示。

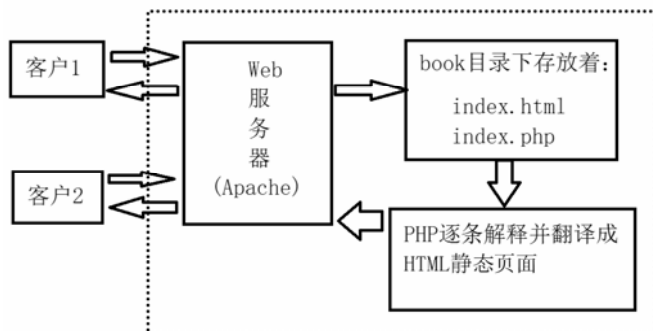


图 1-2 当访问带应用程序的页面时

3. 当访问的页面带应用程序和数据库时

当客户请求一个带应用程序(如 PHP 程序)的页面,并且该页面需要访问数据库时,PHP 应用服务器逐条解释程序时,还需要连接数据库服务器(如 MySQL 服务器),并通过标准 SQL 语句来操作数据库,得到结果,返回给 PHP 程序,翻译成 HTML 静态页面,然后将该 HTML 静态页面返回给 Web 服务器,由 Web 服务器返回给浏览器,呈现给客户,如图 1-3 所示。

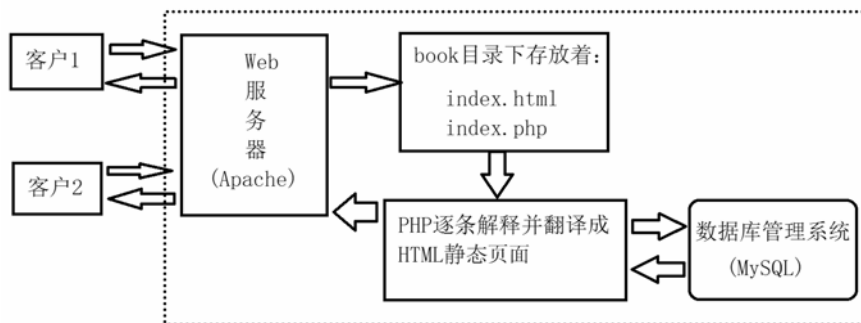


图 1-3 当访问的页面带应用程序和数据库时

1.1.4 常用的 Web 技术

Web 技术基础,也就是构成网页的技术基础。一个网页,首先需要有构成这个网页的结构。表示网页构成的语言是 HTML。HTML 语言可以用来描述网页的构成元素。为了使页面的布局或元素的样式符合人们的视觉习惯和更具艺术感,需要使用 CSS 语言来描述。如果网页还要能和用户进行交互,则需要通过 JavaScript 语言来实现。

1. HTML

HTML(HyperText Markup Language, 超文本标记语言)是用标记来描述网页的一种语言,因此 HTML 不是编程语言,而是标记语言。网站由一个个网页组成,因此可以使用 HTML 来建立网站。HTML 网页运行在浏览器上,由浏览器解析。

HTML 文件的扩展名为.html 或.htm。文件结构如下：

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>HTML 文档标题</title>
</head>
<body>
  <h1>第一个标题</h1>
  <p>第一个段落</p>
</body>
</html>
```

其中，各组成部分的含义如下：

- <!DOCTYPE html>声明为 HTML 文档。
- <html>元素是 HTML 页面的根元素。
- <head>元素包含文档的元(meta)数据。
- <title>元素描述文档的标题。
- <body>元素包含可见的页面内容。
- <h1>元素定义一个大标题。
- <p>元素定义一个段落。

由此可见，HTML 语言主要由一个个标记来描述网页的组成及结构。标记格式为：<标记名></标记名>。由于本书主要讲解 PHP 语言，因此不对 HTML 作详细介绍，后面章节仅对使用到的标记进行介绍。若要深入了解 HTML 语言，可参考其他 HTML 教程。

2. CSS

HTML 标记主要用于定义网页结构，通过使用<h1>、<p>、<table> 这样的标记，HTML 表达的是“这是标题”“这是段落”“这是表格”之类的信息。同时，网页布局由浏览器来完成，而不使用任何的格式化标记。

由于浏览器不断地将新的 HTML 标记和属性(比如字体标签和颜色属性)添加到 HTML 规范中，创建文档内容清晰地独立于文档表现层的站点变得越来越困难。

为了解决这个问题，万维网联盟(W3C)肩负起了 HTML 标准化的使命，并在 HTML 4.0 之外创造出层叠样式表(CSS)。由此可见，CSS 主要是为了描述网页元素的呈现效果而诞生的，有了 CSS，HTML 可以专注于定义网页内容，CSS 则专注于定义 HTML 标记的呈现效果及页面布局。

样式通常保存在外部的.css 文件中。样式表允许以多种方式定义样式信息。样式可以定义在单个 HTML 标记中，也可以定义在 HTML 页面的头元素<head></head>中，还可以单独在一个外部的 CSS 文件中，甚至可以在同一个 HTML 文档内部引用多个外部样式表。

CSS 规则由两个主要部分构成：选择器，以及一条或多条声明。格式如下：

```
selector {declaration1; declaration2; ... declarationN }
```

其中，选择器(selector)通常是需要改变样式的 HTML 元素。每条声明(declaration)由一个属性(property)和一个值(value)组成。属性是希望设置的样式属性。每个属性有一个值。属性和值被冒号分开，格式如下：

```
selector {property: value}
```

下面这行代码的作用是将 h1 元素内的文字颜色定义为红色，同时将字体大小设置为 14 像素。在这个例子中，h1 是选择器，color 和 font-size 是属性，red 和 14px 是值。

```
h1 {color:red; font-size:14px;}
```

图 1-4 展示了上面这行代码的结构。

由于本书主要讲解 PHP 语言，因此不对 CSS 作详细介绍，后面章节仅对用到的样式进行介绍。若要深入了解 CSS，请参考专门的 CSS 教程。



图 1-4 代码结构示意图

3. JavaScript

JavaScript 是互联网上最流行的脚本语言，被数百万计的网页用来改进设计、验证表单、检测浏览器、创建 Cookies 以及进行其他更多的应用。JavaScript 是一种轻量级的编程语言，可插入 HTML 页面，由浏览器执行。

在使用 JavaScript 语言时，可以将 JavaScript 脚本通过 <script> 标记插入 HTML 的 <head> 或 <body> 中，或是单独保存为一个 .js 文件，然后在 HTML 文档中通过 <script> 标签引用这个 .js 文件。

(1) <head> 中的 JavaScript 函数

把一个 JavaScript 函数放置到 HTML 页面的 <head> 部分，例如：

```
<!DOCTYPE html>
<html>
<head>
<script>
function myFunction()
{
    document.getElementById("demo").innerHTML="我的第一个 JavaScript 程序";
}
</script>
</head>
<body>
<h1>个人页面</h1>
<p id="demo">一个段落</p>
<button type="button" onclick="myFunction()">试一试</button>
</body>
</html>
```

该函数会在单击按钮时被调用。

(2) <body> 中的 JavaScript 函数

把一个 JavaScript 函数放置到 HTML 页面的 <body> 部分，例如：

```
<!DOCTYPE html>
<html>
<body>
<h1>个人页面</h1>
<p id="demo">一个段落</p>
<button type="button" onclick="myFunction()">试一试</button>
<script>
```

```
function myFunction()
{
    document.getElementById("demo").innerHTML="我的第一个 JavaScript 方法";
}
</script>
</body>
</html>
```

该函数会在单击按钮时被调用。这里需要注意的是，这里把 JavaScript 函数放到了页面代码的底部，这样就可以确保在<p>元素创建之后再执行脚本。

(3) 外部的 JavaScript

可以把脚本保存到外部文件中。外部文件通常包含被多个网页使用的代码。外部 JavaScript 文件的文件扩展名是.js。如果需要使用外部文件，可在<script>标记中通过 src 属性引用 .js 文件，例如定义了一个 index.js 文件，则在 HTML 页面中这样引用：

```
<!DOCTYPE html>
<html>
<body>
<h1>个人页面</h1>
<p id="demo">一个段落</p>
<button type="button" onclick="myFunction()">单击这里</button>
<p><b>注释：</b>myFunction 保存在名为 "myScript.js" 的外部文件中。</p>
<script type="text/javascript" src="/js/index.js"></script>
</body>
</html>
```

以上 HTML 代码中，将引用保存在当前路径下 js 文件夹中的 index.js 脚本文件。

由于本书主要讲解 PHP 语言，因此不对 JavaScript 语言作详细介绍，只在后续章节对 JavaScript 和 PHP 之间的交互作介绍。若要深入了解，请参考专门的 JavaScript 教程。

1.2 PHP 基础知识

1.2.1 PHP 概述

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

PHP 采用 B/S 体系架构，PHP 程序在 Web 服务器启动后，用户可以不使用客户端软件，使用浏览器即可访问，既保持了图形化的用户界面，又大大减少了应用的维护量。

1.2.2 PHP 的优势

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：

- 跨平台特性：PHP 几乎支持所有的操作系统平台，如 Windows、UNIX、Linux、Macintosh、FreeBSD、OS2 等，并且支持 Apache、IIS 等多种 Web 服务器。

- 对主流数据库的良好支持：可操纵多种主流数据库，如 MySQL、Access、SQL Server、Oracle、DB2 等，其中 PHP 和 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，方便开发人员学习掌握。
- 免费：在流行的企业级应用 LAMP 平台中，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。
- 模板化：实现程序逻辑与用户界面的分离。
- 支持面向对象与过程：支持面向对象和过程的两种开发风格，并可向下兼容。
- 内嵌 Zend 加速引擎，性能稳定快速。

1.2.3 PHP 的应用领域

PHP 在互联网高速发展的今天，应用范围非常广泛，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，企业级应用开发。

PHP 吸引着越来越多的 Web 开发人员。PHP 可应用于任何地方、任何领域，并且已拥有数百万用户，拥有良好的生态社区。

1.2.4 常用的 PHP 开发工具

PHP 的开发工具很多，常用的开发工具通常分为三类：一类是简单文本编辑器类型，如 EditPlus、Notepad++；一类是专门的 PHP 开发编辑器，如 phpDesigner 8、PHP Coder、Zend Studio、PHP Editor；还有一类是在通用开发编辑器上通过嵌入 PHP 插件支持形成的 PHP 开发编辑器，如 NetBeans IDE、Eclipse PDT 等。每种开发工具各有优势，好的开发工具可以提升开发效率，开发人员可以根据需要进行选择。这里不对编辑器作过多介绍，读者可搜索相应的开发工具，查看工具介绍，找一款适合自己需要的开发编辑器。

1.2.5 如何学好 PHP

如何学好 PHP 语言？这是所有初学者共同面临的问题。其实，每种程序设计语言的学习方法都大同小异，需要注意的有以下几点：

- 明确自己的学习目标和方向，选择并锁定一门语言，按照自己的学习方向努力学习，认真研究。
- 学会配置 PHP 的开发环境，选择一种适合自己的开发工具。
- 扎实的基础对程序员来说尤为重要。因此，建议读者多阅读一些程序设计基础教材，了解基本的编程知识，掌握常用的函数。
- 了解设计模式，这几乎是学习任何一种编程语言都必须掌握的高级技能。开发人员编写的程序代码应该具有良好的可读性，这样才能使编写的程序具有调试、维护和升级的价值，学习一些设计模式，就能更好地把握项目的整体结构。
- 多实践，多思考，多请教。不要死记语法，在刚接触一门编程语言时，要掌握好基本语法，反复实践，如果有实际项目进行操练，效果最佳。仅读懂书本中的内容和技术

是不行的，必须动手编写程序代码，并运行程序、分析运行结构，以便对学习内容有整体的认识。平时可以多借鉴网上一些好的功能模块，培养自己的编程思想。多向他人请教，学习他人的编程思想。多和他人沟通技术问题，提高自己的技术和见识。

- 学习最忌讳急躁，遇到技术问题，必须冷静对待，分析出现问题的原因大致有哪些，然后学会使用 Google、百度等搜索引擎工具，寻找此类问题都有哪些解决方案，逐一尝试，找到针对当前问题的最佳解决方案。
- PHP 函数有几千个，需要下载 PHP 帮助手册和 MySQL 手册，或者查看 PHP 函数类的相关书籍，以解决程序中出现的的问题。
- 现在很多 PHP 案例书籍都配有视频教程，可以看一些视频录像来领悟他人的编程思想。只有掌握了整体的开发思路，才能够系统地学习编程。
- 养成良好的编程习惯。遇到问题不要放弃，要有坚持不懈、持之以恒的精神。

1.2.6 PHP 学习资源

1. PHP 帮助手册的下载和使用

学习任何编程语言都不可能一一记住每一个方法和参数，因此，在学习 PHP 的时候，手边最好有一份 PHP 帮助手册，以方便查阅需要用到 PHP 方法。PHP 官方网站提供了电子版的 PHP 帮助手册，使用前需要到官方网站进行下载。

(1) 打开浏览器，在地址栏中输入网址 <http://php.net/>，回车后进入 PHP 官网，如图 1-5 所示。在导航栏中找到 Documentation，然后单击。

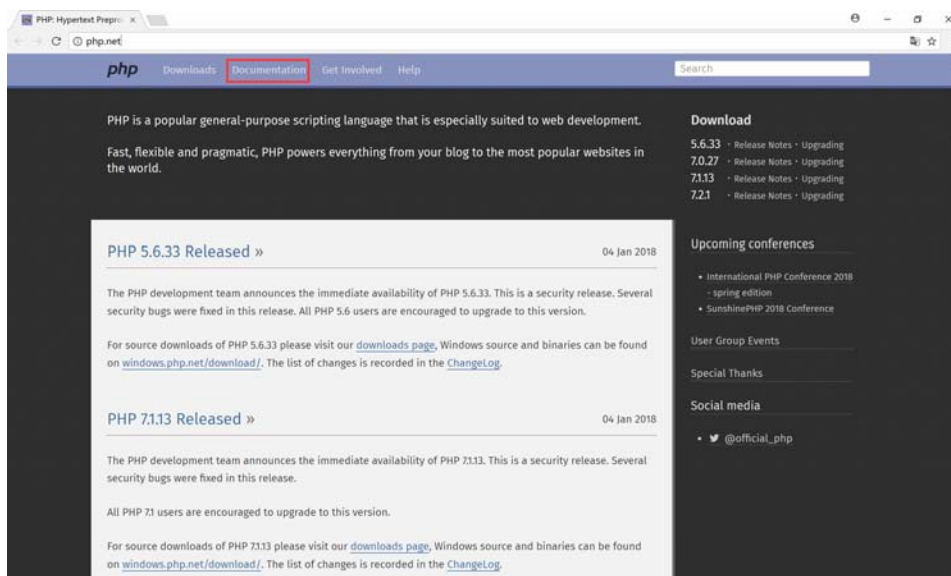


图 1-5 PHP 官网

(2) 进入 Documentation 页面，找到 documentation downloads 链接，如图 1-6 所示。

(3) 单击链接进入多语言下载页面，如图 1-7 所示。在该页面中找到 Chinese(Simplified)，单击该栏最右侧的 chm 链接。

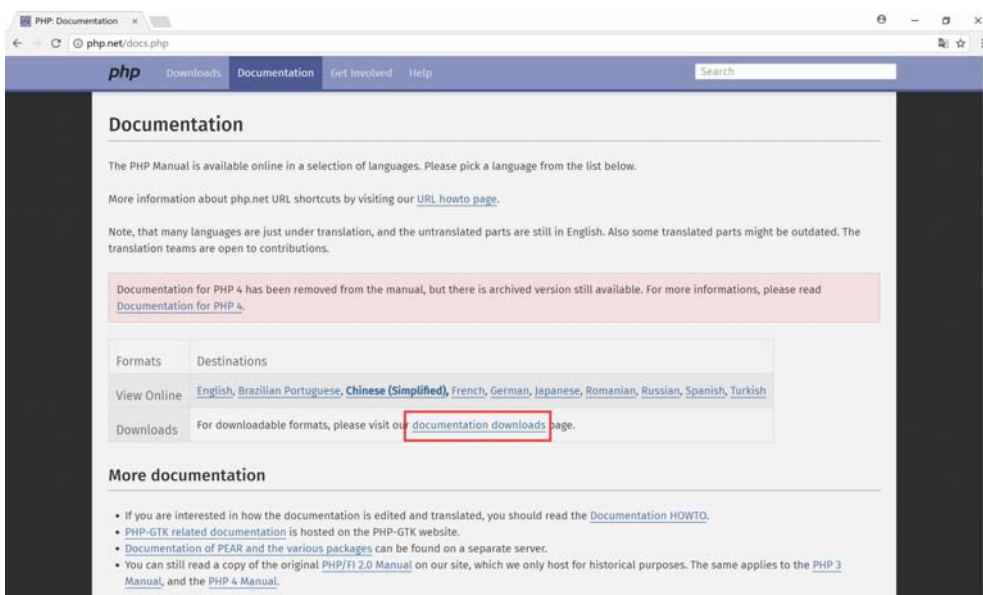


图 1-6 documentation downloads 链接

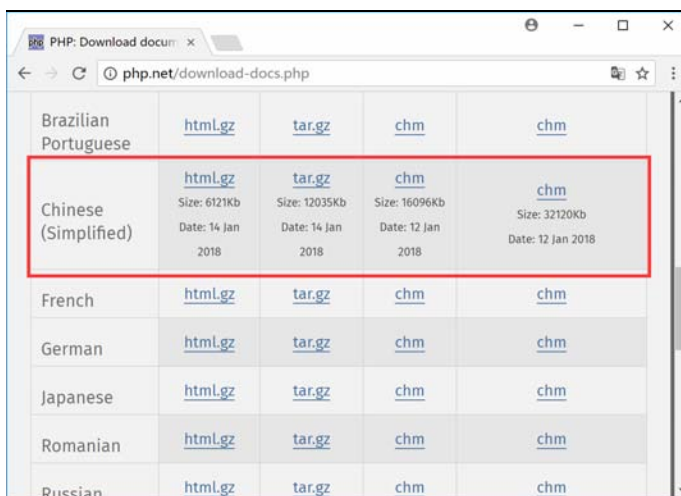


图 1-7 多语言下载页面

(4) 打开下载页面，单击该页面中 China 下的下载链接，即可开始下载帮助手册，如图 1-8 所示。

(5) 打开已下载的 PHP 帮助文档，如图 1-9 所示。左侧默认按【目录】显示帮助文档的内容。

如果要查找某个方法，例如查找 `in_array` 方法，可以单击左侧窗格上方的【搜索】标签，切换到搜索页。在【键入关键字进行查找】文本框中输入 `in_array`，手册将会自动匹配。若找到 `in_array` 项，将高亮显示(蓝色背景词条)，双击该项，即可在右侧显示 `in_array` 方法的功能、使用格式、使用示例等，如图 1-10 所示。另外，还可以通过索引进行查找，以及对经常用到的词条进行收藏，方便以后查阅。

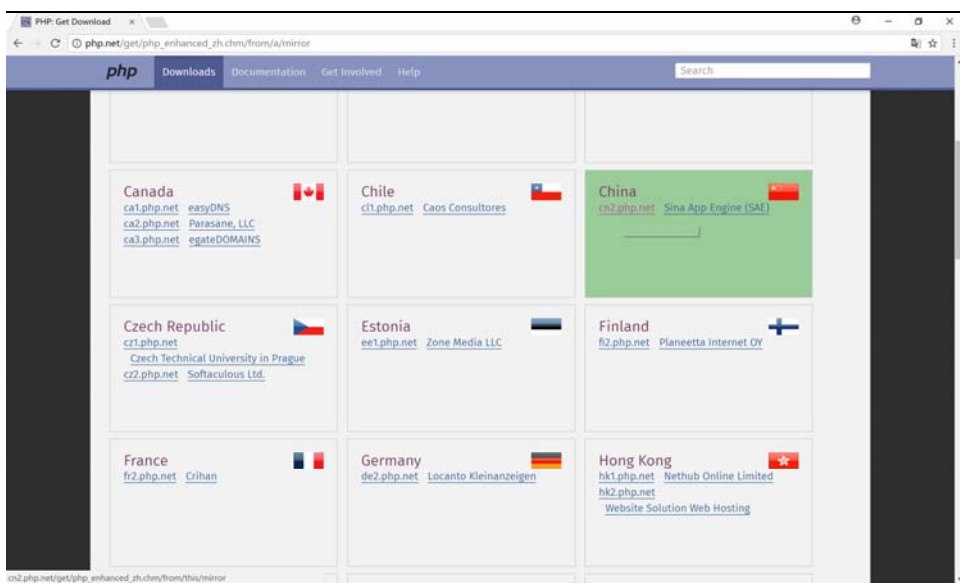


图 1-8 下载页面

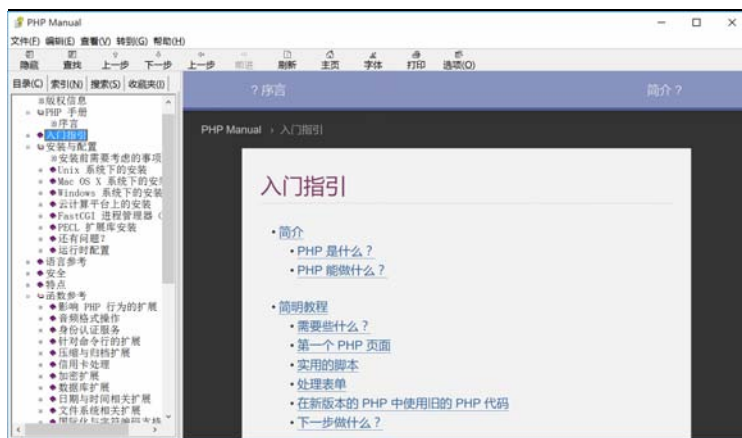


图 1-9 PHP 帮助手册



图 1-10 查找 in_array 方法

2. 网上学习资源

要想学好一门编程语言，除了教材和帮助手册，还需要多上一些技术论坛和社区，与同行多交流，多看他人发布的一些问题和解决方法。这些资源不仅可以提高程序员的技术水平，也是程序员学习和工作的好帮手。

常用的 PHP 技术论坛有 PHP100(<http://www.php100.com>)、PHP 中国(<http://www.phpchina.com>)、PHP 论坛(<http://www.php.cn>)等。另外，学习或工作中遇到问题时，百度和谷歌等搜索引擎永远是学习的好助手。

3. 图书网站

要学好 PHP，网上资源和图书教程都必不可少。网上资源有助于解决遇到的问题以及将所学应用到实际项目中，而图书教程则可以让我们对 PHP 进行系统的学习。国内比较大的图书网站有：当当网图书频道(<http://book.dangdang.com/>)、亚马逊中国(<https://www.amazon.cn/>)、京东图书(<http://book.jd.com>)等。

1.3 PHP 开发环境的搭建

PHP 是运行在服务器端的语言，要想快速在自己的计算机上测试 PHP，需要让自己的计算机也能运行服务器软件。服务器软件(即 Web 服务器)主要运行在 Windows 和 Linux 操作系统上，对应的搭建环境是 WAMP(Windows+Apache+MySQL+PHP)和 LAMP(Linux+Apache+MySQL+PHP)。在 Windows 上，一般常常安装 WAMP、XAMPP、AppServ 等集成套件，其中包含了 Apache、PHP、MySQL 软件；在 Linux 系统上，一般采用 CentOS 操作系统，采用 yum 方式安装 Apache、MySQL 和 PHP 组件。下面分别详细介绍。

1.3.1 WAMP 环境的搭建

1. 安装与配置 WAMP

WAMP 环境的搭建一般采用 WampServer 集成安装包，即在 Windows 系统上一键安装 Apache 服务器、MySQL 数据库、PHP 解析器，对于快速学习 PHP 来说非常合适。

WAMP 安装必备的工具具有：VC11 支持(vcredist_x86/x64)和 WampServer 3.0.6 32/64 位集成(Apache 2.4.23、PHP 5.6.25/7.0.10、MySQL 5.7.14、phpMyAdmin 4.6.4、Adminer 4.2.5、phpSysInfo 3.2.5)。

具体安装步骤如下：

(1) 查看系统类型。因为 WAMP 需要根据操作系统的类型是 32 位还是 64 位来选择不同的安装程序，所以要先了解操作系统是 32 位还是 64 位。

在桌面上右击【计算机】系统图标，选择【属性】命令，打开【系统】窗口，可以查看操作系统的位数，如图 1-11 所示。

(2) 下载和安装组件。安装 WAMP 时必须先安装 VC11(Visual C++ Redistributable for Visual Studio 2012)支持，否则安装 WAMP 时系统会提示找不到 MSVCR110.dll。为了方便，这里提供了两个文件最新的安装包(于 2018 年 1 月 15 日从官网获取)。

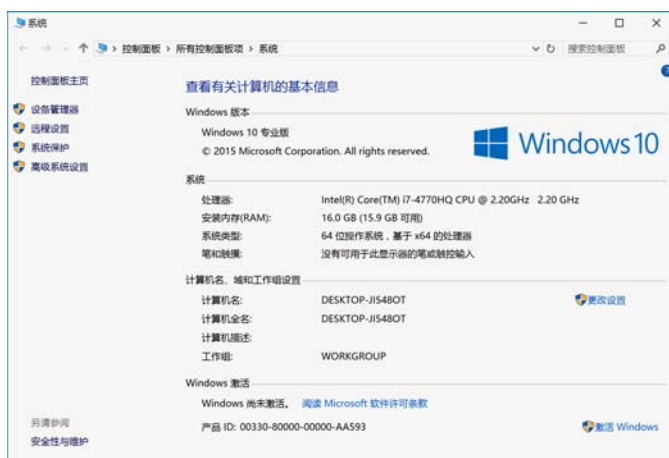


图 1-11 查看 Windows 系统的位数

VC11 官网下载地址: <https://www.microsoft.com/en-us/download/details.aspx?id=30679>。

WAMP 下载地址: <http://www.wampserver.com/en/download-wampserver-64bits/>。

(3) 运行安装 VC11 支持包(vcredist_x64.exe 或 vcredist_x86.exe), 如图 1-12 所示。另外, 还要确保有最新版的 VC9、VC10、VC13 和 VC14 支持包。

(4) 运行安装已下载的 WampServer 安装包(WampServer 3.0.6), 根据安装向导进行软件的安装。首先选择语言, 如图 1-13 所示, 只有两种选择, 这里选择 English 选项。



图 1-12 安装 VC11 支持包

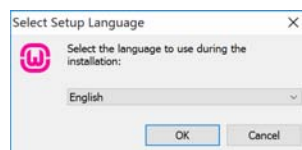


图 1-13 选择语言

(5) 然后选择是否接受协议和安装信息, 如图 1-14 所示。

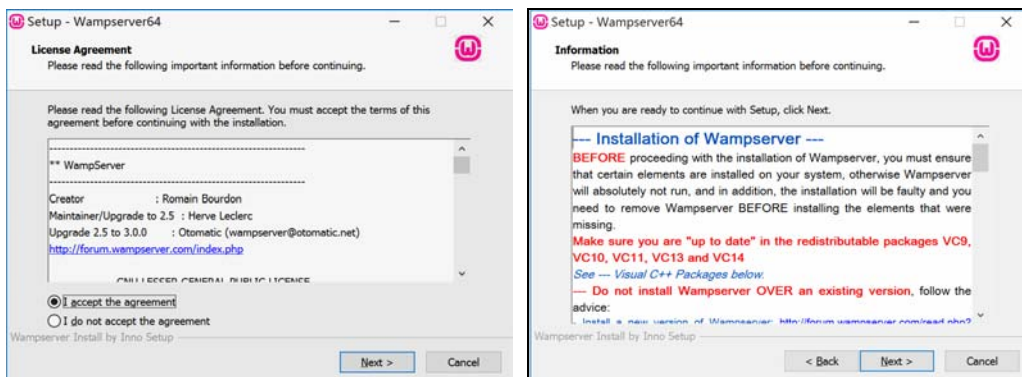



图 1-14 选择是否接受协议和安装信息

(6) 接下来选择安装目录, 默认为 C:\wamp64。根据安装向导一步步进行安装即可。

需要注意的是，安装过程中会弹出提示，让选择默认启动 localhost 地址的浏览器，找不到路径跳过也没有关系。只是想帮用户关联 localhost 快捷访问，默认是 IE。其实以后通过打开 Chrome 浏览器，输入 localhost 或 127.0.0.1(本机地址)进行访问也是一样的。

(7) 安装完毕后，桌面上出现一个图标，双击即可启动 WampServer 64。Windows 桌面右下角会出现 WampServer 的运行状态图标(图标多的话可能会被隐藏，仔细找找)，如图 1-15 所示。如果这时图标是绿色的，说明 Apache、PHP、MySQL 都正常运行，服务器可以投入使用。如果是橙色或红色，说明 Apache、MySQL 或 PHP 组件启动失败。

(8) 配置与调试。在服务器状态图标上右击，在弹出的菜单中展开 Language 选项，选择 chinese(中文)，如图 1-16 所示。

在服务器状态图标上单击，打开控制管理面板，如图 1-17 所示，其中各选项如下：

- Localhost: 调用浏览器访问本机地址，默认优先读取 www 目录下的 index.php 文件。



图 1-15 启动图标

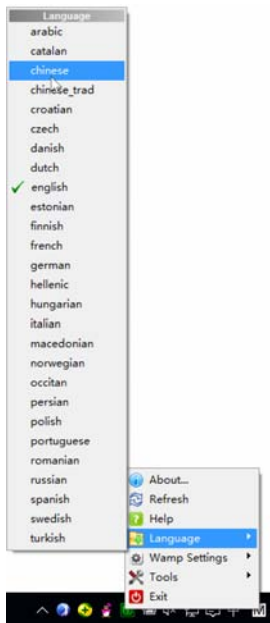


图 1-16 设置语言



图 1-17 WAMP 的控制管理面板

- phpMyAdmin: 一款可视化的数据库管理工具，这里操作的是 MySQL 数据库。
- www 目录: 网站文件根目录，PHP 工程文件都要放在这个目录下，也可以通过 Apache 配置文件 httpd.conf 指定为其他目录。

访问上面的 localhost(127.0.0.1)或者打开任意一个浏览器后输入 localhost，可以看到如图 1-18 所示的页面。因为服务器默认优先打开 index.php，即首页，所以如果有需要，可以用自己做好的首页文件来替换。如果目录下不存在 index.php，服务器会显示文件列表。要访问自己的 xxx.php 文件，在浏览器的地址栏中输入 127.0.0.1/xxx.php 并回车即可。

在服务器图标上单击，打开控制管理面板，选择 MySQL，如图 1-19 所示，选择进入 MySQL 控制台。如果 MySQL 服务器没问题，将出现黑色命令行窗口(这里将背景设置为白色)。提示输入密码，直接按回车键，成功登录数据库服务器，如图 1-20 所示，这时可以使用各种数据库操作命令了。

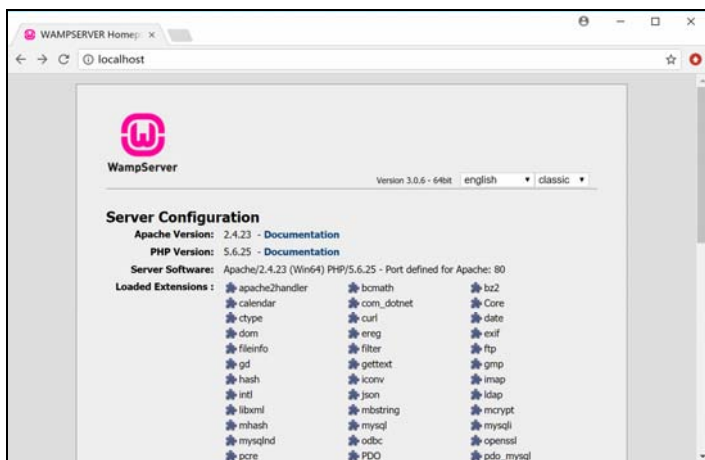


图 1-18 默认的 localhost 页面

PHP 连接时默认的数据库用户名是 root，密码为空(留空)。黑色界面是原生的数据库命令行操作界面，所以上面的 phpMyAdmin 是这个界面之上的可视化界面，类似的客户端软件还有 Navicat。

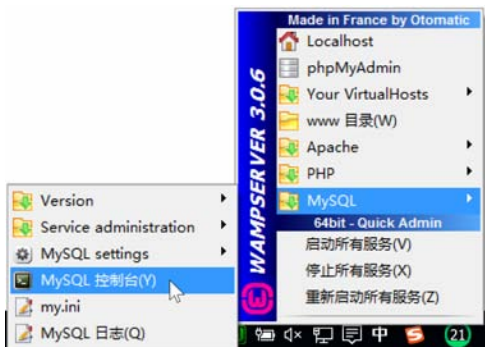


图 1-19 打开 MySQL 控制台

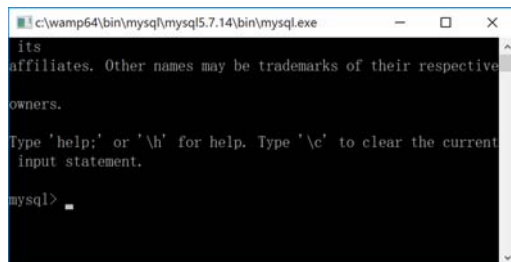


图 1-20 MySQL 命令行窗口

2. 将 Apache 和 MySQL 注册为系统服务

WAMP 环境已安装完毕。当在本机上使用 WAMP 环境浏览网站时，首先要双击 WAMP 图标以启动程序。但是，当网站已被部署到服务器上时，如果也这样启动 WAMP 程序，只要服务器崩溃重启，就又要到服务器桌面手动重启 WAMP 程序，显然很不方便。更方便的办法是，将 Apache 和 MySQL 注册为 Windows 系统服务，设置为自动运行，这样，只要服务多次无法响应或服务器重启，服务组件就会自动重启，而不需要手工干预。

事实上，当运行 WAMP 时，程序已经将 Apache 和 MySQL 注册为系统服务。

(1) 在桌面上右击【计算机】(或【此电脑】)，从弹出的快捷菜单中选择【管理】命令，如图 1-21 所示。这时将打开【计算机管理】窗口，如图 1-22 所示。

(2) 在【计算机管理】窗口的左侧窗格中找到【服务和应用程序】，展开后单击【服务】选项，右侧窗格将显示当前计算机中所有已安装的服务。找到 wampapache64 和 wampmysqld64 选项。



图 1-21 选择【管理】命令

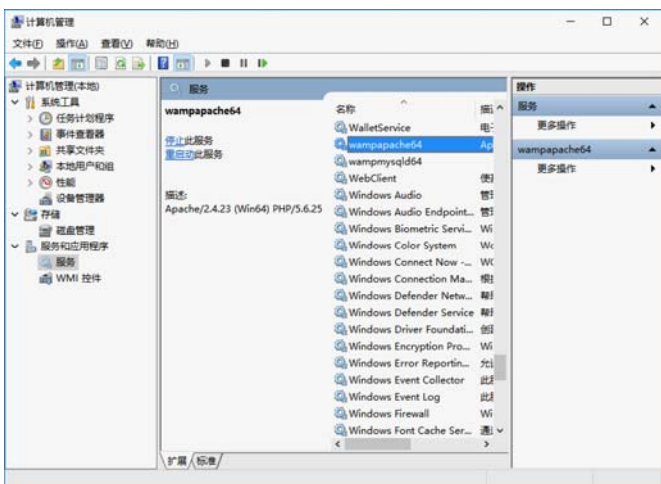


图 1-22 【计算机管理】窗口

(3) 右击 wampapache64 选项，从弹出的快捷菜单中选择【属性】命令，打开【wampapache64 的属性(本地计算机)】对话框，如图 1-23 所示。在【常规】选项卡的【启动类型】中，选择【自动】，然后单击【应用】按钮；切换到【恢复】选项卡，在【第一次失败】中选择【重新启动服务】，在【第二次失败】中选择【重新启动服务】，在【后续失败】中选择【重新启动计算机】，如图 1-24 所示。单击【确定】按钮保存设置。

使用同样的方法设置 wampmysqld64 服务。

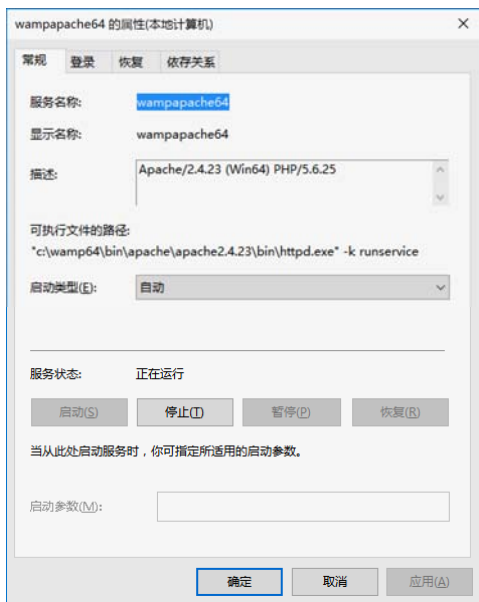


图 1-23 设置【启动类型】

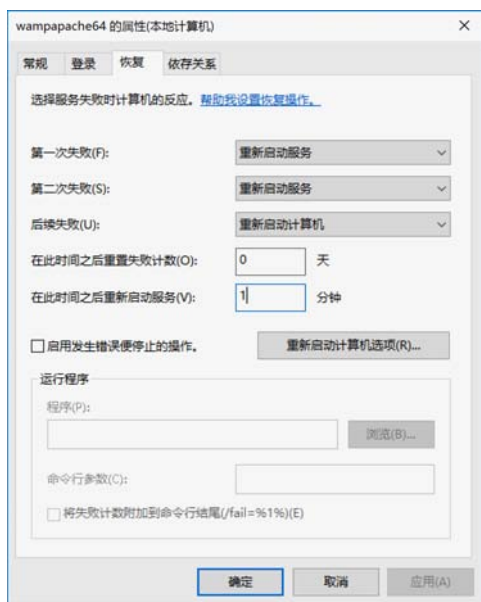


图 1-24 设置服务失败后的处理

经过上面这些设置后，每当计算机重启时，WAMP 会自动跟随启动；当 WAMP 发生故障无法正常接收和响应客户端请求时，WAMP 服务会自动重启，如果故障非常严重，将重新启动计算机以重置服务器。

注意，也可以分别单独将安装目录下的 bin 目录下的 Apache 和 MySQL 手动注册为系统服

务, 这里不作过多介绍, 感兴趣的读者可以自行搜寻将 Apache 或 MySQL 分别注册为 Windows 系统服务的方法。

1.3.2 LAMP 环境的搭建

在 Linux 操作系统下搭建 PHP 开发环境相比 Windows 操作系统下要复杂得多, 除了 Apache、MySQL、PHP 等软件外, 还要安装一些相关工具, 并设置必要参数。此外, 如果要使用 PHP 扩展库, 还要进行编译。

安装之前需要准备的安装包如下:

- httpd-2.4.23.tar.gz
- mysql-5.7.16-linux-glibc2.5-i686.tar.gz
- php-5.6.25.tar.gz
- libxml2-2.9.7.tar.gz

1. 安装 Apache 服务器

为了安装 Apache 服务器, 首先需要打开 Linux 终端。例如, 以 Red Hat 为例, 选择【主菜单】|【系统工具】命令, 在弹出的子菜单中选择【终端】命令。下面介绍 Apache 的安装步骤。

- (1) 进入 Apache 安装文件所在的目录, 如/usr/local/work。

```
cd /usr/local/work/
```

- (2) 解压安装包。解压完成后, 进入 httpd2.4.23 目录。

```
tar xzf httpd-2.4.23.tar.gz
cd httpd-2.4.23
```

- (3) 建立 makefile, 将 Apache 服务器安装到 usr/local/Apache2 下。

```
./configure --prefix=/usr/local/Apache2 --enable-module=so
```

- (4) 编译文件。

```
make
```

- (5) 开始安装。

```
make install
```

- (6) 安装完成后, 将 Apache 服务器添加到系统启动项中, 最后重启服务器。

```
/usr/local/Apache2/bin/Apachectl start >> /etc/rc.d/rc.local
/usr/local/Apache2/bin/Apachectl restart
```

- (7) 打开 Chrome 浏览器, 在地址栏中输入 http://localhost/, 按 Enter 键查看安装是否成功。

2. 安装 MySQL 数据库

安装 MySQL 比安装 Apache 复杂一些, 因为需要创建 MySQL 账号, 并将新建的账号加入到组群中。安装步骤如下:

- (1) 创建 MySQL 账号, 并加入到组群中。

```
groupadd mysql
useradd -g mysql mysql
```

- (2) 进入 MySQL 的安装目录, 将安装包解压, 例如目录为/usr/local/mysql。

```
cd /usr/local/mysql
tar xzf /usr/local/work/mysql-5.7.16-linux-glibc2.5-i686.tar.gz
```

- (3) 考虑到 MySQL 数据库升级的需要, 通常以链接的方式建立/usr/local/mysql 目录。

```
ln -s mysql-5.7.16-linux-glibc2.5-i686.tar.gz mysql
```

- (4) 进入 MySQL 目录, 在/usr/local/mysql/data 中建立 MySQL 数据库。

```
cd mysql
scripts/mysql_install_db --user=mysql
```

(5) 修改文件权限如下:

```
chown -R root
chown -R mysql data
chgrp -R mysql
```

(6) 至此, MySQL 安装成功。用户可以通过在终端输入命令来启动 MySQL 服务。

```
/usr/local/mysql/bin/mysqld_safe --user=mysql &
```

启动后输入命令, 进入 MySQL:

```
/usr/local/mysql/bin/mysql -u root
```

3. 安装 PHP

在安装 PHP 之前, 首先需要查看 libxml 的版本号, 其版本号不宜过低。安装 libxml 和 PHP 的步骤如下:

(1) 将 libxml 和 PHP 复制到 /usr/local/work 目录下, 并进入该目录。

```
cp php-5.6.25.tar.gz libxml2-2.9.7.tar.gz /usr/local/work
cd /usr/local/work
```

(2) 分别将 libxml 和 PHP 安装包解压。

```
tar xzf libxml2-2.9.7.tar.gz
tar php-5.6.25.tar.gz
```

(3) 进入 libxml2 目录, 建立 makefile, 将 libxml 安装到 /usr/local/libxml2 下。

```
cd libxml2-2.9.7
./configure --prefix=/usr/local/libxml2
```

(4) 编译文件。

```
makefile
```

(5) 开始安装。

```
make install
```

(6) libxml2 安装完毕后, 开始安装 PHP。进入 PHP 目录。

```
cd ./php-5.6.25
```

(7) 建立 makefile。

```
./configure --with-apxs2=/usr/local/Apache2/bin/apxs
--with-mysql=/usr/local/mysql
--with-libxml-dir=/usr/local/libxml2
```

(8) 开始编译。

```
make
```

(9) 开始安装。

```
make install
```

(10) 复制 php.ini-dist 或 php.ini-recommended 到 /usr/local/lib 目录中, 并命名为 php.ini。

```
cp php.ini-dist /usr/local/lib/php.ini
```

(11) 更改 httpd.conf 文件设置, 该文件位于 /usr/local/Apache2/conf 目录。找到该文件中的以下指令行:

```
AddType application/x-gzip .gz .tgz
```

在该行后加入如下指令:

```
AddType application/x-httpd-php .php
```

重启 Apache, 在 Apache 主目录下建立文件 phpinfo.php。

```
<?php phpinfo(); ?>
```

在浏览器中输入 <http://localhost/phpinfo.php>, 按 Enter 键测试是否成功显示 PHP 信息页。如果成功显示, 则说明 PHP 安装成功。

1.3.3 扩展库

PHP 一直在不断升级和更新，总体上围绕着性能、安全和新特性，不断为开发者提供新的动力。PHP 提供了一些扩展库，这些扩展库使 PHP 如虎添翼，更加灵活方便。如网上社区、BBS 论坛等，如果没有扩展库的支持，它们都可能无法使用。因此，在安装 PHP 时，要根据各自用途选择安装扩展库。

从 PHP5 开始，PHP 新增了内置的标准扩展库，如表 1-2 所示，包括 XML 扩展库-DOM、SimpleXML、SPL、SQLite 等，而 MySQL、MySQLi、Overload、GD2 这些库则被放在 PECL 外部扩展库中，需要在 `php.ini` 配置文件中选择加载。

在 Windows 下加载扩展库，是通过修改 `php.ini` 文件来完成的。用户也可以在脚本中通过使用 `dl()` 函数来动态加载。PHP 扩展库的 DLL 文件都具有 `php_` 前缀。

很多扩展库都内置于 Windows 版本的 PHP 中，加载这些扩展库时不需要额外的 DLL 文件和 `extension` 配置命令。Windows 下的 PHP 扩展库列表列出了需要或曾经需要额外 DLL 文件的扩展库。

在编辑 `php.ini` 文件时，需要注意以下几点：

- 需要修改 `extension_dir` 设置，使其指向用户放置扩展库的目录或者放置 `php_*.dll` 文件的位置，例如 `extension_dir=d:\php\extensions`。
- 为了在 `php.ini` 文件中启用某扩展库，需要去掉 `extension=php_*.dll` 前的注释符号，即需要将加载的扩展库前的分号“；”删除。例如，为了启用 Bzip2 扩展库，需要将下面这行代码的注释分号去掉：

```
;extension=php_bz2.dll
```

改成：

```
extension=php_bz2.dll
```

- 有些扩展库需要额外的 DLL 文件才能工作。其中一部分 DLL 文件绑定在发行包中，但有一些需要的 DLL 文件并没有绑定在发行包中。如果安装 PHP5，需要将绑定的 DLL 文件从 `D:\php\dlls` 复制到主目录 `D:\php` 中。值得注意的是，必须将 `C:\php5` 放到系统路径 `PATH` 中。
- 某些 DLL 文件没有绑定在 PHP 发行包中。PECL 中有庞大的 PHP 扩展库，这些扩展库需要单独下载。

表 1-2 PHP 扩展库

扩展库	说明	注解
<code>php_bz2.dll</code>	Bzip2 压缩函数库	无
<code>php_calendar.dll</code>	历法转换函数库	自 PHP 4.0.3 起内置
<code>php_cpdf.dll</code>	ClibPDF 函数库	无
<code>php_crack.dll</code>	密码破解函数库	无
<code>php_ctype.dll</code>	ctype 家族函数库	自 PHP 4.3.0 起内置
<code>php_curl.dll</code>	CURL，客户端 URL 函数库	需要 <code>libeay32.dll</code> 和 <code>ssleay32.dll</code> (已附带)
<code>php_cyrcrash.dll</code>	网络现金支付函数库	PHP<=4.2.0 版本之前内置
<code>php_dbase.dll</code>	dBase 函数库	无
<code>php_dba.dll</code>	DBA，数据库(dbm 风格)抽象层函数库	无

(续表)

扩展库	说明	注解
php_dbx.dll	dbx 函数库	无
php_domxml.dll	DOM XML 函数库	PHP<=4.2.0, 需要 libxml2.dll(已附带) PHP>=4.3.0, 需要 icon.dll(已附带)
php_dotnet.dll	.NET 函数库	PHP<=4.1.1
php_exif.dll	EXIF 函数库	需要 php_mbstring.dll, 并且在 php.ini 中, php_exif.dll 必须在 php_mbstring.dll 之后加载
php_fbsql.dll	frontBase 函数库	PHP<=4.2.0
php_fdf.dll	PDF:表单数据格式化函数库	需要 fdfk.dll(已附带)
php_filepro.dll	filePro 函数库	只读访问
php_ftp.dll	FTP 函数库	自 PHP 4.0.3 起内置
php_gd.dll	GD 库图像函数库	自 PHP 4.0.3 中删除。此外注意, 在 GD1 中 不能用真彩色函数, 应用 php_gd2.dll 替代
php_gd2.dll	GD2 函数库	GD2
php_gettext.dll	Gettext 函数库	PHP<=4.2.0, 需要 gun_gettext.dll(已附带) PHP>=4.2.3, 需要 libintl-1.dll 和 iconv.dll(已 附带)
php_hyperwave.dll	HyperWave 函数库	无
php_iconv.dll	ICONV 字符集转换	需要 iconv-1.3.dll(已附带)
php_ifx.dll	Informix 函数库	需要 Informix 库
php_iisfunc.dll	IIS 管理函数库	无
php_imap.dll	IMAP、POP3 和 NNTP 函数库	无
php_ingres.dll	Ingres II 函数库	需要 Ingres II 库
php_interbase.dll	InterBase 函数库	需要 gds32.dll(已附带)
php_java.dll	Java 函数库	PHP<=4.0.6, 需要 jvm.dll(已附带)
php_ldap.dll	LDAP 函数库	PHP<=4.2.0, 需要 libsasl.dll(已附带) PHP>=4.3.0, 需要 libeasy32.dll 和 ssleay32.dll (已附带)
php_mbstring.dll	多字节字符串函数库	无
php_mcrypt.dll	Mcrypt 加密函数库	需要 libmcrypt.dll
php_mhash.dll	Mhash 函数库	PHP>=4.3.0, 需要 libmhash.dll(已附带)
php_mime_magic.dll	Mimetype 函数库	需要 libmcrypt.dll
php_mhash.dll	Mhash 函数库	PHP>=4.3.0, 需要 libmhash.dll(已附带)
php_mime_magic.dll	Mimetype 函数库	需要 magic.mime(已附带)
php_ming.dll	Ming 函数库(Flash)	无
php_msql.dll	MSQL 函数库	需要 msql.dll(已附带)
php_mssql.dll	MSSQL 函数库	需要 ntwdblib.dll

(续表)

扩展库	说明	注解
php_mysql.dll	MySQL 函数库	PHP>=5.0.0, 需要 libmysql.dll
php_oci8.dll	Oracle 8 函数库	需要 Oracle 8.1+客户端库
php_openssl.dll	OpenSSL 函数库	需要 libeasys 32.dll
php_oracle.dll	Oracle 函数库	需要 Oracle 7 客户端库
php_overload.dll	对象重载函数库	自 PHP 4.3.0 起内置
php_pdf.dll	PDF 函数库	无
php_pgsql.dll	PostgreSQL 函数库	无
php_printer.dll	打印机函数库	无
php_shmop.dll	共享内存函数库	无
php_snmp.dll	SNMP 函数库	仅用于 Windows NT
php_soap.dll	SOAP 函数库	PHP>=5.0.0
php_sockets.dll	Socket 函数库	无
php_sybase_ct.dll	Sybase 函数库	无
php_tidy.dll	Tidy 函数库	PHP>=5.0.0
php_tokenizer.dll	Tokenizer 函数库	自 PHP 4.3.0 起内置
php_w32api.dll	W32api 函数库	无
php_xmlrpc.dll	XML_RPC 函数库	PHP>=4.2.1 需要 iconv.dll
php_xslt.dll	XSLT 函数库	PHP<4.2.0, 需要 sablot.dll 和 expat.dll PHP>=4.2.1, 需要 sablot.dll、expat.dll 和 iconv.dll
php_yaz.dll	YAZ 函数库	需要 yaz.dll
php_zip.dll	Zip 文件函数库	只读访问
php_zlib.dll	ZLib 压缩函数库	自 PHP 4.3.0 起内置

1.4 第一个 PHP 程序

下面以 NotePad++作为编辑器编写一个简单的 PHP 程序。

【例 1-1】 本例的目的是熟悉 PHP 的书写规则和程序的执行。本例将输出一行“Hello World”。

(1) 找到 WAMP 的安装目录，本书为 C:\wamp64。打开 www 文件夹，新建一个文件夹并命名为 book，用于存放本书的示例程序。

(2) 打开 book 文件夹，再新建 ch01 文件夹，右击，从弹出的菜单中选择【新建】|【文本文档】命令，新建一个文本文档，命名为 index.php。

(3) 右击 index.php，选择用 NotePad++打开。

(4) 输入以下代码：

```
<?php
echo "hello world";
?>
```

启动 WAMP 服务程序，打开浏览器，输入网址，回车后页面输出如图 1-25 所示。

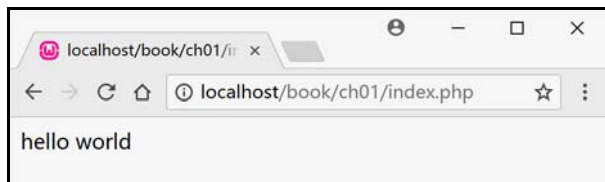


图 1-25 第一个 PHP 程序的输出

在这个程序中，有以下几点需要注意：

- “<?php”和“?>”是 PHP 的标记对。在这对标记中，所有的代码都被当成 PHP 处理。除了这种表示方法外，PHP 还可以使用 ASP 风格的“<%...%>”和 SGML 风格的“<?...?>”等。
- echo 是 PHP 中的输出语句，与 ASP 中的 response.write、JSP 中的 out.print 的含义相同，用于将紧跟其后的字符串或变量值显示在页面中。每行代码都以分号“;”结尾。
- www 目录是 WAMP 服务器默认用于存放网站的位置，在这个示例中，PHP 将会默认到这个位置的 book 目录的 ch01 文件夹中查找 index.php 文件，然后进行解析、输出。

1.5 调试与错误处理

程序的调试与错误处理，就是通过一定的方法，在程序中找到并减少缺陷的数量，从而使程序能正常工作。这里介绍一些调试 PHP 程序的经验。

1.5.1 使用自带的报错功能

1. 环境配置

实际项目中，一般在网站的开发阶段和上线之后，采用两种不同的环境方案：开发环境和生产环境。开发环境是开发人员进行开发和调试的环境，生产环境是最终客户使用的线上环境。一般情况下，开发环境和生产环境要分开设置报错功能。

(1) 开发环境

开发环境需要打开报错功能，以下是 php.ini 中的配置项及其说明：

```
; This directive sets the error reporting level.
; Development Value: E_ALL | E_STRICT (Show all errors, warnings and notices including coding standards.)
error_reporting = E_ALL | E_STRICT
; This directive controls whether or not and where PHP will output errors,
; notices and warnings too. Error output is very useful during development.
; Development Value: On
display_errors = On
```

这样在开发过程中，能第一时间发现错误，即使是低等级的报错“Notice: Undefined

variable: a in E:\phpspace\test.php on line 14”，但未定义变量的使用往往暗藏着 bug。

有的读者会问，如果引入了开源的类库，运行时抛出一堆低级的错误怎么办？一般代码质量好的类库，是没有 Notice 级别的报错的。所以这也是一种鉴别类库的质量的方法。

(2) 生产环境

生产环境下不能直接将错误输出，而是记入日志，以下是 php.ini 中的配置项及其说明：

```
; It could be very dangerous in production environments.
; It's recommended that errors be logged on production servers rather than
; having the errors sent to STDOUT.
display_errors = Off
; Besides displaying errors, PHP can also log errors to locations such as a
; server-specific log, STDERR, or a location specified by the error_log
; directive found below. While errors should not be displayed on productions
; servers they should still be monitored and logging is a great way to do that.
; Production Value: On
log_errors = On
; Log errors to specified file.
error_log = /path/to/php_error.log
```

生产环境是给客户提供服务的，不能在上面进行设置断点、打印输出等操作，所以日志是不错的选择。当然，将日志写到文件里只是一个选择，还有其他配置，可参考手册。

2. 其他 PHP 语言特性、功能的使用

(1) 少用错误控制运算符 “@”

将 “@” 放置在一个 PHP 表达式之前，该表达式可能产生的任何错误信息都被忽略掉。如果在这个表达式中发生缺陷，从 PHP 的输出中看不到任何错误，这增加了调试的难度。所以能不用则不用。

(2) 有些函数自带调试功能

比如下面这行代码：

```
$fp = fsockopen("www.example.com", 80, $errno, $errstr, 30);
```

调试时已经确定，\$fp 为空，连接失败，是这一行有问题，但是为什么连接失败？函数是 PHP 自带的，无法进行更深入的调试。所以一般这样的函数(主要是网络通信类的函数)，会自己提供调试参数：\$errno 和 \$errstr。可以加一句：

```
if (!$fp) echo "$errstr ($errno)<br />\n";
```

这样就能看到连接失败的原因了。这类函数有 fsockopen、pfsockopen、stream_socket_server、stream_socket_client 等。

还有些函数是调试功能用的，比如 mysql_errno、socket_last_error、socket_strerror 等。

1.5.2 引进调试工具

在遇到复杂问题时，可以借助调试工具。比较成熟的有 Xdebug、ZendDebugger。

以 Xdebug 为例，它能够控制打印输出的样式和数组层级、堆栈式的追踪错误、追踪函数调用、对代码执行覆盖分析、程序的概要分析(Profiling)、远程调试。Xdebug 的前两个功能对 PHP 原有的调试功能做了改进，更方便调试，详见 <http://xdebug.org/docs/>。

1.5.3 调试业务逻辑错误

当 PHP 程序执行没有任何错误时，说明程序此时没有语法上的错误，但并不能说业务逻辑上没有错误。很多业务逻辑上的错误并不会反映在语法错误上，但调试的思路和 PHP 自带调试功能差不多。下面是一些常用的调试方法。

1. 最基本的调试方法

首先要确定程序预期的结果，以及程序现在的不符合预期的结果。然后，寻找与两种结果相关的代码片段。首先阅读代码，尝试以“肉眼”找出错误；如果找不出，则需要输出一些关键变量，通过检查它们的值是否正确来判断是哪里发生了错误；若干次尝试后，最终一般可以确定错误发生在哪个点。

另外，也可以借助 Xdebug 等工具，查看变量值的变化，或者设置断点进行调试。

2. 记录运行日志

有些复杂或特殊的业务，用上面的方法不合适，比如不能被打断运行的后台脚本。这些情况下记录运行日志比较合适。记日志的点要有所选择，除了业务上比较重要的点，通常容易出错的地方有：网络连接和通信、系统权限问题等。

3. 单元测试

以代码测试代码，而不是调试完就把测试代码丢掉。以测试驱动开发为例，这个话题比较大，但适合放这里提一下。有兴趣的同学可以去了解。

1.5.4 调试非功能性错误

非功能性错误，如内存溢出导致程序终止，效率有问题等导致程序非常慢、死循环等。这些问题，无法通过“肉眼”检查代码来发现，这样效率太低，这时可以借助调试工具进行程序的概要分析(Profiling)，从中检查出程序的瓶颈所在。

1.6 本章小结

本章主要从概要上介绍了 Web 技术和 PHP 在 Web 中的位置。Web 应用程序是以 B/S 架构为基础的应用程序，主要以浏览器(Browser)为客户端。网站就是 Web 应用程序的一种。用户通过浏览器发送请求到服务器(Server)，服务器处理完毕后，再把结果 HTML 页面发送回客户端。

网站开发过程中，客户端需要用到 HTML、CSS、JavaScript 语言，而服务器端程序语言一般采用 PHP、JSP、ASP 等。PHP 以简单、高效著称。

PHP 是一种服务器端、跨平台、HTML 嵌入式的脚本语言，其独特的语法混合了 C、Java 和 Perl 语言的特点，是一种被广泛应用的、开源的多用途脚本语言，尤其适合 Web 开发。PHP 语言的优点有：安全性高、跨平台、易学、执行速度快、免费、模板化、支持面向对象与过程、内嵌 Zend 加速引擎、性能稳定快速。

PHP 适合用来开发任何应用类型的服务器端程序，可应用于各行各业。在编写 PHP 程序时，如果程序不多，简单的记事本即可满足编辑要求；如果是大型项目，也有许多开源免费的 IDE 平台可供选择。另外，在学习 PHP 的过程中，有大量的 PHP 图书、网站、技术论坛等辅助渠道。

本书的最后还介绍了如何在 Windows 和 Linux 系统上搭建 PHP 环境，以及如何编写一个简单的 PHP 程序，如何调试 PHP 程序以及出现错误时如何处理。接下来就开始 PHP 语言学习之旅吧！

1.7 思考和练习

1. 简述 Web 工作原理，并画出交互图。
2. 常用的 PHP 开发工具有哪些？
3. 尝试在 Linux 上部署 PHP 运行环境。
4. 尝试在 Windows 上部署 PHP 运行环境。
5. 编写一个简单的 PHP 程序，并在浏览器中执行。