Web开发与设计

图解 PHP & MySQL 服务器端 Web 开发

[美] 乔恩·达克特(Jon Duckett) 著
卢志超 译

清華大學出版社
北 京

北京市版权局著作权合同登记号 图字：01-2023-4997
Jon Duckett
PHP & MySQL
978-1-119-14922-4

图书在版编目 (CIP) 数据

图解 PHP & MySQL 服务器端 Web 开发 /(美) 乔恩 • 达克特 (Jon Duckett) 著；卢志超译. —北京：清华大学出版社，2024.3
(Web 开发与设计)
书名原文：PHP & MySQL
ISBN 978-7-302-65616-6

Ⅰ. ①图… Ⅱ. ①乔… ②卢… Ⅲ. ① PHP 语言—程序设计② SQL 语言—程序设计
Ⅳ. ① TP312.8 ② TP311.132.3

中国国家版本馆 CIP 数据核字 (2024) 第 044922 号

责任编辑：王 军
装帧设计：孔祥峰
责任校对：马遥遥
责任印制：杨 艳

出版发行：清华大学出版社
网 址：https://www.tup.com.cn，https://www.wqxuetang.com
地 址：北京清华大学学研大厦A座 邮 编：100084
社 总 机：010-83470000 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：三河市龙大印装有限公司
经 销：全国新华书店
开 本：148mm×210mm **印 张：**20.875 **字 数：**1041千字
版 次：2024年4月第1版 **印 次：**2024年4月第1次印刷
定 价：256.00元

产品编号：073227-01

目录

译者序

每一次的技术革新，都为我们开启了新的可能性。PHP和MySQL用于构建和管理动态网站，已成为Web开发中不可或缺的重要工具。

PHP是一种广泛使用的服务器端脚本语言，简单易学、功能强大，受到全球开发者的广泛欢迎。MySQL则是最流行的开源关系数据库之一，性能卓越、稳定、易用。本书以PHP和MySQL为基础，深入浅出地介绍如何进行服务器端Web开发。本书不仅介绍基本知识，还详细讲解如何使用这两种工具进行实战开发，使得你能在理论和实践之间找到平衡。

首先介绍PHP的基本指令。涵盖PHP的基础语法、数据类型、控制结构、函数等。

然后详述如何使用PHP开发动态网页。具体包括使用PHP处理表单数据、操作cookie和会话、生成动态HTML等，帮你构建出真正具有交互性的网站。

此后，深入讲解如何构建数据库驱动型网站，使得网站可存储大量用户数据。详细介绍如何使用PHP操作MySQL数据库，包括创建数据库和数据表，插入、查询、更新和删除数据，进行事务处理和错误处理等。

最后讲解如何扩展PHP应用，以满足更复杂的业务需求。重构前面创建的应用，并引入开源的第三方工具库来提升开发效率。这些知识将有助于你编写更易于阅读、维护和扩展的代码，在提升性能的同时，降低后续的维护成本。

在翻译本书的过程中，我深刻体验到学习的乐趣和战胜挑战的满足感。每次理解了一个复杂概念，每次解决了一个困扰心头的问题，我都感到无比喜悦。同时，我体验到分享知识的快乐。我希望这份热情和乐趣，能通过这本书传递给你们。

本书面向所有对服务器端Web开发感兴趣的人士，无论你是刚接触PHP和MySQL的初学者，还是有一定经验的开发者，都能在本书中找到有用的信息并受到启发，获得宝贵的经验，从而独立开发一个PHP网站。

在此感谢原书的作者撰写了这本极具实用价值的书籍。感谢我的同事和朋友们，他们在翻译过程中给予了我很大的帮助和支持。最后感谢每位读者，是你们的支持和鼓励让我有动力翻译本书。

再次感谢你选择阅读本书，我期待你在阅读过程中能收获满满乐趣，祝你的学习之路一帆风顺！同时，我期待你的反馈和建议，让我们一起进步，一起开创更美好的未来。

前言

本书将指导你使用PHP这门编程语言来搭建网站，以及将网站所用的数据保存在MySQL等类型的数据库中。

PHP 是一门用于在Web服务器上运行的语言，当用户请求服务器上的网页时，服务器将生成相应的网页并返回给发起请求的用户。这意味着网页展示的内容可以针对个人进行定制化生成。对于任何支持用户完成下列行为的网站而言，可定制化都是必须满足的要求：

- **注册或者登录**。因为每个用户的姓名、邮箱和密码都不尽相同。
- **完成购买**。因为每个客户的订单、支付以及收货详情也不尽相同。
- **搜索网站**。因为搜索结果是为不同用户定制的。

PHP语言设计用于与MySQL这类数据库协同工作，MySQL数据库可保存网站显示的页面内容、网站销售的产品或网站会员的详细信息等。通过对PHP的学习，你将掌握如何创建不同类型的网站，这些网站将允许会员更新数据库中的数据。例如：

- 内容管理系统允许网站所有者使用表单更新网站的内容；在不需要额外编写任何新代码的情况下，这些更新的内容可直接展示给访问者。
- 网店允许店主列出要出售的产品，而客户可以购买这些产品。
- 社交网站允许访问者注册和登录，创建个人信息，上传自己的资料，并访问感兴趣的定制化页面。

由于网站上展示的这些数据都保存在数据库中，因此这些网站又称为数据库驱动型网站(database-driven website)。

这是本书中将出现的不同页面类型。它们分别表示不同类型的信息。

信息页

信息页以白色为背景，介绍主题，解释上下文以及相关内容如何使用。

代码页

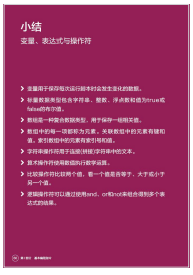
代码页以米黄色为背景，展示一些独立代码片段如何使用。

图表页

图表页以黑色为背景，使用图表和信息图来解释概念。

示例页

示例页出现在章的开头，呈现该章的主题以及相关应用。



小结页

小结页出现在每一章的末尾，对该章所涉及的关键主题进行概括。

静态网站与动态网站

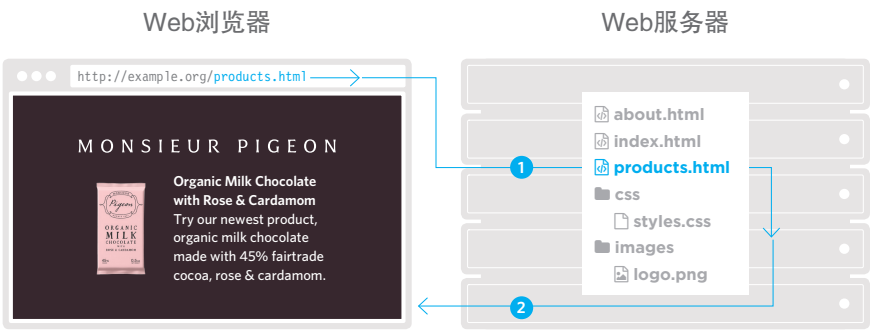
当网站仅使用HTML和CSS进行构建时，所有访问者看到的内容都是相同的，因为网站给他们发送的都是相同的HTML和CSS文件。

(1)当浏览器向网站请求使用HTML和CSS构建的页面时，这个请求被发送到托管该网站的Web服务器。

(2)Web服务器将找到浏览器所请求的HTML文件，并将其发送回浏览器。Web服务器也将发送回渲染页面样式所需的CSS文件、媒体(如图片)文件、JavaScript文件以及页面使用的其他类型的文件。

因为访问者接收到的HTML文件都是相同的，所以他们看到的内容是相同的。将这类网站称为静态网站。

对静态网站的维护者而言，需要掌握HTML和CSS的相关知识。如果想要更新页面上的文本，则必须手动更新HTML代码并将其上传到Web服务器。



本书假定你已经掌握HTML和CSS的相关知识。如果你尚不了解这些知识，请尝试访问以下网站来了解相关主题：

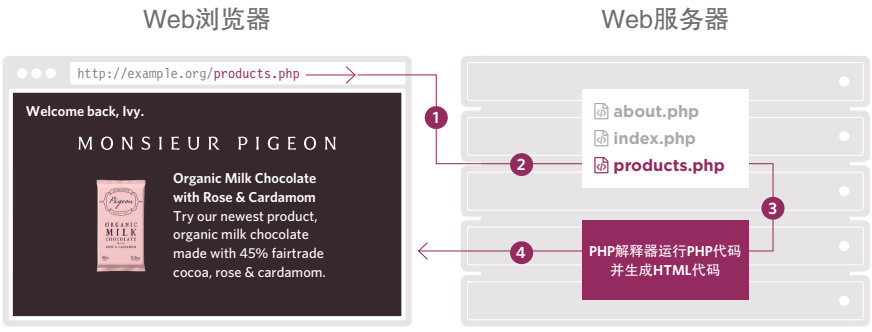
http://htmlandcssbook.com

当网站采用PHP构建时，每个访问者都能看到不同的内容。这是因为PHP页面动态创建了HTML文件并将其发送给访问者。

像eBay、Facebook和其他类似的新闻网站经常在用户每次访问时显示新的信息。如果去查看浏览器中页面的源代码，用户可看到HTML代码，但其实网站的维护者并没有在用户访问期间手动更新代码。

这种类型的网站被称为动态网站，因为返回给访问者的HTML页面是使用PHP之类的语言编写的指令创建的。

① 如果浏览器请求的网站是使用PHP搭建的，那么该请求将发送到Web服务器。
② Web服务器找到对应的PHP文件。
③ 文件中的任何PHP代码都是通过名为PHP解释器(interpreter)的软件来运行的。在解释器对代码进行解读后，将为访问者生成相应的 HTML页面。
④ Web服务器将创建的HTML页面发送到用户的浏览器(Web服务器不会保留该HTML文件的副本；在下次请求PHP文件时，会为该访问者重新创建一个新的HTML页面)。



PHP代码不会被发送回浏览器，而是直接用于创建HTML 页面，然后由Web服务器将该页面发送回浏览器。因为PHP代码在Web服务器上运行，所以被称为服务器端编程(server-side programming)。

PHP可用于创建为每个访问者量身定制或个性化的HTML页面。这可能包括显示访问者的姓名、他们感兴趣的话题或来自朋友的评论。

PHP：语言与解释器

PHP解释器是一款运行在Web服务器上的软件。你可使用PHP语言编写的代码告诉它要做什么。

通过软件可以让计算机执行特定任务，并且不必深入了解计算机如何完成任务。例如：

- 可使用电子邮件程序发送和接收电子邮件，而不必了解计算机如何保存或发送电子邮件。
- 可以使用Photoshop 来编辑图片而不必知道计算机处理图片的原理。

每当使用软件时，该软件都能执行相同的任务，但它可以用不同的数据执行这些任务：

- 电子邮件程序可以用来创建、发送、接收和保存电子邮件，但每封电子邮件的内容和收件人可以是不同的。
- Photoshop可以执行添加滤镜、调整大小或裁剪图片等任务。也可以对任何其他的图像执行相同的任务。

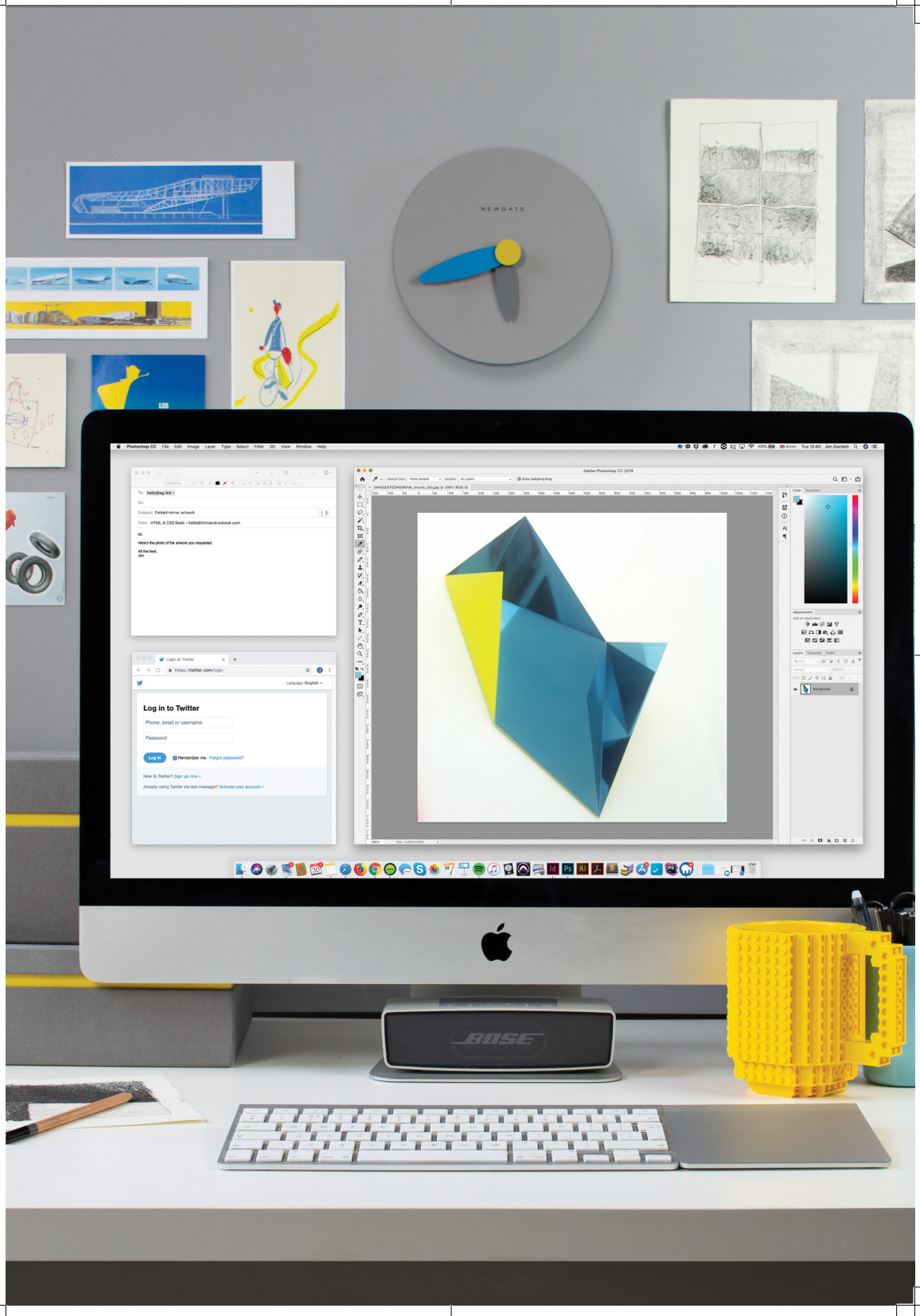
这两个软件都拥有各自的图形用户界面，你可以通过与用户界面的交互来完成这些任务。

类似地，PHP解释器也是一个软件。它作为Web服务器的一部分运行。但对于开发者而言，并不使用图形用户界面，而是使用PHP编程语言来编写代码，以告诉PHP解释器需要执行什么任务。

当使用PHP创建一个Web页面时，该页面将始终执行相同的任务，但它可以在每次请求页面时使用不同的数据来执行这些任务。例如，使用PHP编写的网站可以具有如下特性：

- 每个用户都使用相同的登录页进行登录，即使每个用户的邮箱地址和密码都不相同。
- 每个用户都可以在个人页面查看详细信息。即使同时有数百个不同的人使用这个页面，他们也只能看到自己的详细信息。

实现以上特性是可能的，这是因为虽然对于每个用户而言，执行这些任务所需遵循的规则或指令都是相同的，但他们所提供或看到的数据却可能不同。


NEWGATE
Log in to Twitter
Phone, email or username
Password
Log in
BOSE

使用不同数据执行同一任务

可使用编程语言创建规则来告诉计算机如何执行任务。每次执行任务时，程序使用的数据可能不同。

当使用任何编程语言时，都必须给计算机提供精确的指令，确切地告诉它需要做什么。这些指令与现实生活中要求某人执行某一任务的用语存在较大差异。

假设你想买5根糖果棒，则需要计算出这些糖的总成本。如果要算出总成本，则要用一根糖果棒的价格乘以想买的糖果棒的数量。可用如下方式表达这个规则：

```
total = price x quantity
```

计算糖果的成本时：

- 如果买了5根单价 1 美元的糖果棒，则总价是5美元。
- 如果一根糖果棒的价格是1.5美元，且计算规则一样，那么总价是7.5美元。
- 如果以每个2美元的单价买10根糖果棒，且计算规则一样，那么总共是20美元。

变量的值(而不是变量名称total、price和quantity)可以改变，但用于计算糖果总价的规则不变。

当使用PHP创建一个网页时，首先需要清楚的是：

- 要执行的任务
- 任务执行时每次会变化的数据

然后向PHP解释器提供关于如何完成任务的详细指令，并使用变量名表示可以变化的值。

如果你告诉PHP解释器：

```
price = 3
quantity = 5
```

然后使用以下规则：

```
total = price x quantity
```

total的计算结果为15。下次运行该页面时，可以给单价或数量赋予不同的值，它可使用相同的规则计算出新的总价。

程序员将表示值的单词称为变量，因为它们表示的值在每次程序运行时都可能发生改变。

total = price x quantity

$9 = 3 x 3

PHP页面简介

PHP页面通常混用HTML代码和PHP代码。可以通过PHP解释器将HTML页面发送回浏览器。

在下面，可以看到一个PHP页面，其中混用了HTML和PHP代码。

- 蓝色部分是HTML代码
- 紫色部分则是PHP代码

当PHP解释器打开文件时，它将会：

- 把任何HTML代码直接复制到为访问者创建的临时HTML文件中。
- 解释用PHP代码编写的任何指令(通常为HTML页面生成内容)。

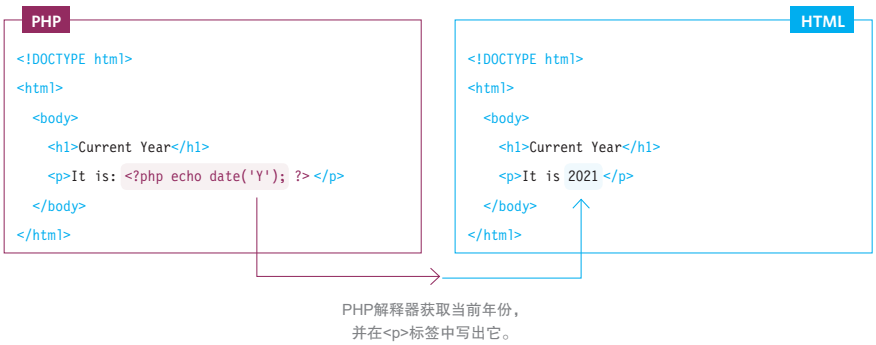
这里显示的PHP代码用于展示当前年份，并在开始标签 和结束标签 中展示该年份。

PHP代码可以执行基本任务，如计算或获取当前日期；也可以执行更复杂的任务，如使用HTML表单发送的数据来更新保存在数据库中的数据。

当PHP解释器完成PHP文件的处理后，它把为访问者创建的临时HTML页面发送回浏览器，然后删除临时HTML页面。

下面可以看到，在PHP解释器执行PHP代码后，将生成要发送回浏览器的HTML页面。

PHP解释器已经确定了当前年份，并将该年份显示在它生成的HTML页面中。



每次请求页面时，页面通常执行相同的任务，但页面也能处理网站上各个访问者的不同信息。

PHP网站由一组PHP页面组成，每个页面执行特定的任务。例如，允许会员登录的网站可能有：

- 登录页 - 允许用户登录网站
- 简介页 - 展示用户资料

每次请求其中某个页面时，都需要能够处理特定于当前会员的不同数据。因此，页面需要：

- 包含指令来完成页面要执行的任务。
- 为每次请求页面时可能更改的每个数据字段指定一个变量名。

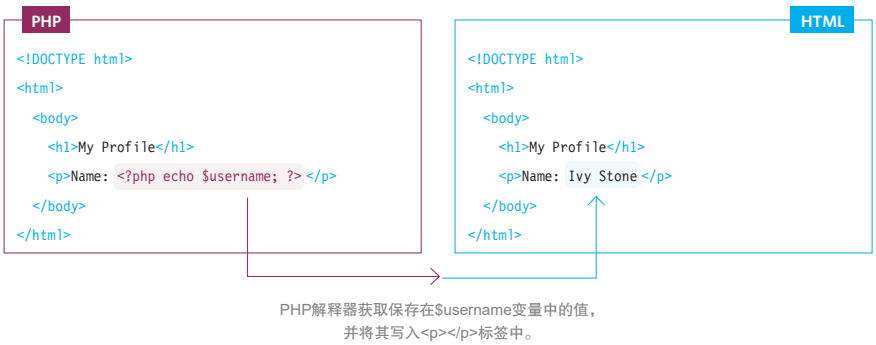
在PHP中，使用变量名表示每次请求页面时都可以更改的值。

PHP代码可以告诉PHP解释器：

- 每次请求页面时都可以更改的数据所使用的变量名。
- 本次请求页面时使用什么值。

一旦Web服务器将HTML页面发送给用户，PHP解释器就会舍弃所有保存在变量中的值，这样它就可以为下一个请求该页面的用户执行相同的任务(使用不同的值)。

如果想要将数据保存更长时间，可将变量值放在与MySQL类似的数据库中，可在下一页中看到有关MySQL的介绍。



MySQL简介

MySQL是数据库类型中的一种。数据库以结构化方式保存数据，因此可以轻易地访问和更新它们所保存的信息。

在Excel这样的电子表格软件中，信息保存在由列和行组成的网格中。然后，它可以直接对这些数据进行计算，或者将其传入公式中。

MySQL保存信息的方式与Excel类似，它将信息保存在同样由列和行组成的表(table)中。可使用PHP访问和更新数据库中保存的信息。

一个数据库可以包含多个表。每个表通常保存网站需要的同类型数据。下面是两个保存以下信息的数据库表示例：

- 网站的会员(或用户)
- 网站中显示的文章

在每个表中，列名描述了表中每一列包含的信息类型：

- member表保存每个会员的名字、姓氏、邮箱地址、密码、注册日期和个人图片。
- article表保存每篇文章的标题、摘要、内容、创建日期，以及下一页中展示的其他一些信息。

每行数据用来表示该表所描述的一个事物：

- member表，每行代表一个会员
- article表，每行代表一篇文章

表名　列名　列

member

id	forename	surname	email	password	joined	picture
1	Ivy	Stone	ivy@eg.link	$2y$10$MAdTTCAOMiOw	2021-01-01 20:28:47	ivy.jpg
2	Luke	Wood	luke@eg.link	$2y$10$NN5HEAD3atar	2021-01-02 09:17:21	NULL
3	Emiko	Ito	emi@eg.link	$2y$10$/RpRmiUMStji	2021-01-02 10:42:36	emi.jpg

article

id	title	summary	content	created	category_id	member_id	image_id	published
1	Systemic	Brochure	<p>This	2021-01-01	1	2	1	1
2	Polite	Poster	<p>These	2021-01-02	1	1	2	1
3	Swimming	Architect	<p>This	2021-01-02	4	1	3	1

行

使用PHP，可以做到：

- 从数据库中获取数据，并在网页中显示这些信息。
- 添加新数据行。要创建新文章，需要向article 表添加一行，并提供应该保存在每个列中的数据项。
- 删除数据行。若要删除某篇文章，需要删除代表该文章的整个行。
- 更改现有行中的数据。要更新会员的邮箱地址，需要在会员表中找到代表这些会员的行，然后更新该行的email列中的值。

注意，这两个表的第一列都是名为id的列。表中的每一行在该列中都有一个唯一的值(这里，这些列中的值从1开始，且每一行增加1)。id列中的值可以告诉数据库要处理哪一行数据。例如，可获取id为2的会员或id为1的文章。

MySQL是关系型数据库，因为它可以解释保存在不同表中的数据类型之间的关系。

例如，在下面的表中，文章是由网站的不同会员撰写的。在article表中，member_id列中的值表示是哪个用户写了相应的文章，因为它包含一个数字，与会员表的id列中的某个值相匹配。

第一篇文章是由id值为2的会员(Luke Wood)写的。第二篇和第三篇文章则是由id值为1的会员(Ivy Stone)写的。

通过这些关系可以：

- 对数据进行结构化，确保每个表只保存一种特定类型的数据(会员或文章)。
- 避免数据库在多个表中重复保存相同的数据 (为数据库节省空间)。
- 使数据更容易更新。如果会员想更改自己的姓名，只需要在member表中进行更新，而不需要在所写的每篇文章中更新。

member

id	forename	surname	email	password	joined	picture
1	Ivy	Stone	ivy@eg.link	$2y$10$MAdTTCAOMiOw	2021-01-01 20:28:47	ivy.jpg
2	Luke	Wood	luke@eg.link	$2y$10$NN5HEAD3atar	2021-01-02 09:17:21	NULL
3	Emiko	Ito	emi@eg.link	$2y$10$/RpRmiUMStji	2021-01-02 10:42:36	emi.jpg

article

id	title	summary	content	created	category_id	member_id	image_id	published
1	Systemic	Brochure	<p>This	2021-01-01	1	2	1	1
2	Polite	Poster	<p>These	2021-01-02	1	1	2	1
3	Swimming	Architect	<p>This	2021-01-02	4	1	3	1

PHP的发展历史

与大多数软件一样，PHP和MySQL也有很多版本。新版本增加的新特性能使运行速度比旧版本更快。

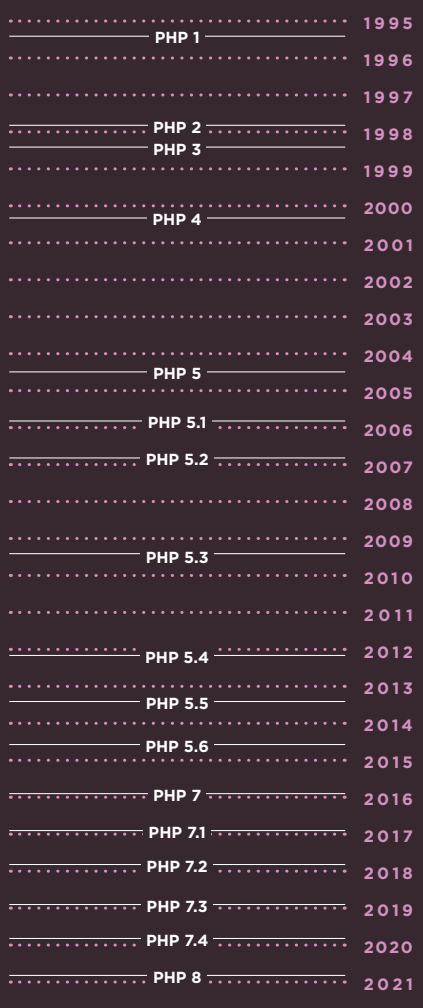
PHP是由Rasmus Lerdorf于1994年创建的。后来他于1995年向公众发布了代码，并鼓励用户改进它。当时，PHP这三个字母代表Personal Home Page(个人主页)。现在PHP代表Hypertext Processor(超文本处理器)。

现在，80%的网站在Web服务器上使用的编程语言都是PHP。

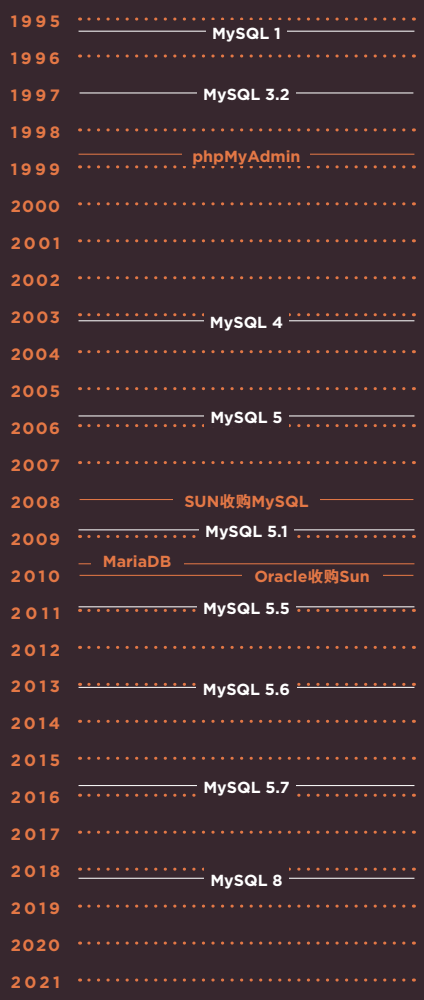
像Facebook、Etsy、Flickr和Wikipedia这样的网站最初都是用PHP开发的(尽管现在也使用其他一些技术)。

主流的开源软件，如WordPress、Drupal、Joomla和Magento都是用PHP编写的。学习PHP有助于你使用它们。

每个新版本的PHP都会添加额外特性。本书介绍2020年11月发布的PHP 8的特性。



MySQL的发展历史



MySQL于1995年首次发布。字母SQL(发音为ess-queue-el或sequel)代表结构化查询语言。SQL是一种用于在关系数据库中获取信息的语言。

MySQL是由一家名为MySQL AB的瑞典公司开发的。MySQL的创建者之一Michael Widenius以他女儿的名字My命名MySQL。

MySQL AB在2008年1月被卖给Sun Microsystems公司，之后Oracle在2010年收购了Sun。

当MySQL的开发人员得知Oracle将收购Sun(并因此拥有MySQL)时，他们担心Sun可能不再免费，所以创建了一个开源版本的数据库，名为MariaDB(以创始人的小女儿Maria命名)。

Facebook、YouTube、Facebook、YouTube、Twitter、Netflix、Spotify和Wordpress等网站都使用MySQL或MariaDB。

phpMyAdmin是一个可用来管理MySQL和MariaDB数据库的工具。它在1998年发布，是一个帮助管理MySQL数据库的免费工具(也可与MariaDB一起协同工作)。

本书中的代码适用于MySQL 5.5或MariaDB 5.5或更高版本，使用phpMyAdmin可以操作这些数据库。

MySQL的最新版本(撰写本书时)是版本8。MySQL 6从未发布，版本7在本书中未涉猎，因为它运行在集群服务器上，而非个人电脑上。

本书涉及的内容

本书内容共分四部分。以下是各部分中涉及的主题。

第I部分：基本编程指令

第Ⅰ部分将展示如何使用PHP代码编写PHP解释器可解释的指令。你将学习：

- 基本编程指令
- 在不同情况下运行不同的代码(例如，如果用户已经登录，运行一组代码；如果用户没有登录，运行另一组代码)
- 函数如何将执行单个任务需要的所有代码组合在一起
- 通过使用类和对象，组织用于表示世界上的事物的代码

第II部分：动态网页

本书第Ⅱ部分介绍PHP提供的一组工具，这些工具提供了构建动态网页的能力。你将学习：

- 收集浏览器发送的数据
- 检查用户是否提供了页面所需的数据以及数据格式是否正确
- 处理已发送的任何数据
- 处理用户上传的文件
- 在PHP中展示日期和时间
- 在cookie和session缓存中临时保存数据
- 对代码问题进行故障排除

第III部分：数据库驱动型网站

第Ⅲ部分将展示如何从数据库中获取数据并将其显示在网页中，以及如何更新保存在数据库中的数据。你将学习：

- 数据库如何保存数据
- SQL语言如何用于检索或更新数据库中保存的数据
- 数据库中的数据如何显示在PHP页面中
- 通过使用HTML表单，让访问者能够更新保存在数据库中的数据

第IV部分：扩展示例应用程序

第Ⅳ部分将展示用PHP构建网站和应用程序的实用技术。示例应用程序是一个具有社交特性的基本内容管理系统。你将学习：

- 改进代码的结构
- 合并其他程序员提供的代码
- 使用PHP发送邮件
- 允许会员注册并登录网站
- 创建个性化页面
- 使用对搜索引擎友好的url
- 添加社交功能，如点赞和评论

第I部分

基本编程指令

本书第 I 部分将介绍有关用PHP编写代码的基础知识。

编程涉及创建一系列指令，计算机可以按照这些指令来执行特定的任务。可以将这些指令与包含制作菜肴所需步骤的菜谱进行比较。PHP中的每条指令都称为语句。

由于PHP的设计目的是构建能够为每个访问者动态创建HTML页面的网站，所以在本书第 I 部分学习的语句主要关注如何使用PHP创建HTML页面。

一个完整的网站通常由数千行代码组成，所以仔细组织代码是很重要的。本部分介绍两个将相关语句组合在一起的概念：

- 函数将执行单个任务所需的语句组合在一起。
- 对象将一组表示概念的语句组合在一起，例如，网站上显示的文章、网站销售的产品或已在网站注册的会员。

本部分中的主题构成了后续章节中学习的其他所有内容的基础。

在深入研究第1章之前，有一些基础知识需要学习，这些知识对后续学习有很大助益。

安装软件和代码示例

要在台式机或笔记本电脑上使用PHP和MySQL这样的数据库来搭建网站，需要事先安装一些软件。安装完所需的软件后，需要扫描封底二维码，从本书提供的网站下载示例代码文件。

PHP文件中混用HTML和PHP代码

因为PHP用于创建HTML页面，所以PHP页面通常混用HTML和PHP代码。你需要了解PHP解释器如何区分这两种代码。

使用PHP创建HTML

PHP解释器执行的最常见指令之一是向HTML页面添加内容，该页面将返回给访问者。本节中的每个例子都使用了此指令。

在PHP代码中添加注释

PHP解释器不会运行注释代码，但注释能够帮助你和其他人理解代码应该做什么，因此学习如何添加注释是很重要的。注释穿插在本书代码中，以帮助解释示例代码正在做什么。

安装软件和文件

在台式机或笔记本电脑上创建数据库驱动型网站时，可使用下面的工具安装需要的所有软件。

使用本书，需要安装：

- 运行PHP解释器的Web服务器。本书使用 Apache(因为在业界使用最广泛)。
- 作为数据库软件的MySQL 或MariaDB。
- 用于管理数据库的phpMyAdmin。

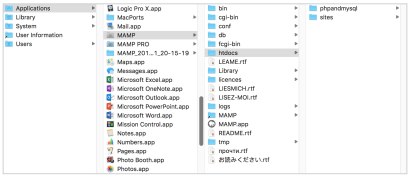
稍后介绍的工具将为你下载并安装所有这些程序，你不必逐个下载和安装它们。

推荐使用下面这款代码编辑器：

http://notes.re/php/editors

在Mac上安装

建议Mac用户使用名为MAMP的工具安装所需的软件。下载链接和使用说明可通过如下链接找到：http://notes.re/php/mamp。在Mac上安装MAMP时(使用默认设置)，会自动创建文件夹：/Applications/MAMP。这个文件夹中有个名为htdocs的文件夹，任何使用PHP编写的网页都必须放在这个文件夹中。该文件夹称为文档根目录。



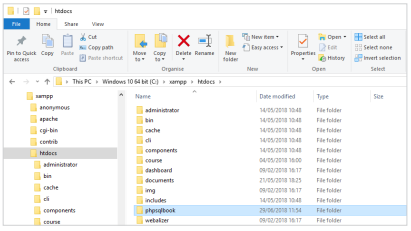
在PC / Linux上安装

建议PC和Linux用户使用名为XAMPP的工具安装所需的软件。下载链接和使用说明可通过如下链接找到：

http://notes.re/php/xampp

在PC上安装完XAMPP时(使用默认设置)，会自动创建文件夹 c:\xampp\。

其中有个名为htdocs的文件夹。使用PHP编写的任何网页都必须放在这个文件夹中。它被称为文档根目录。



下载示例代码

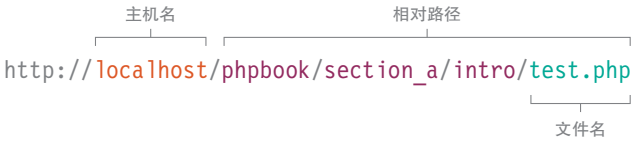
在如下网页中单击下载按钮可下载本书的示例代码(也可扫描封底二维码下载)：http://phpandmysql.com/code

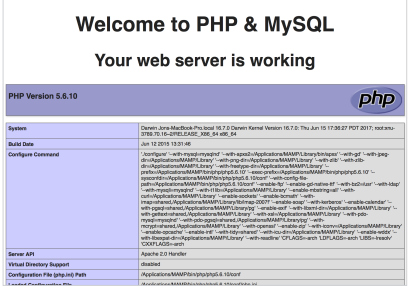
示例代码保存在名为phpbook的文件夹中。该文件夹包含本书每一部分以及每一章的子文件夹。下载完成后需要将phpbook文件夹放入htdocs文件夹中。

要在浏览器中打开PHP文件，必须在地址栏输入对应的URL。如果使用file菜单的open命令打开一个文件，双击它，或将它拖到浏览器中，PHP代码将不会运行。

尝试打开下面所示的URL，将看到测试页面显示在其下方。在计算机上打开文件时，请直接输入 localhost(而非域名)。其后跟着从htdocs文件夹到目标文件的相对路径。

下面所示的路径告诉服务器在phpbook/section_a /intro 文件夹中找到名为 test.php的文件。





错误解决方法

完成上述步骤后如果没有看到左边的页面，请查看：

http://notes.re/php/mamp for MAMP

http://notes.re/php/xampp for XAMPP

使用MAMP时，通常需要输入一个端口号。它默认使用8888端口，所以输入http://localhost: 8888/。

端口号有助于安装在一台计算机上的不同程序共享同一连接以访问互联网。这类似于办公室可能只有一个电话号码，却可以办理多个分机号码，以接入不同的个人电话。

PHP页面混用HTML和PHP代码的方式

许多PHP页面混用了HTML和PHP代码。PHP代码在PHP标签之间编写。PHP的开始和结束标签以及它们所包含的任何PHP代码被称为PHP代码块。

开始标签

<?php

开始标签指示PHP解释器在将任何内容发送回浏览器之前必须开始处理代码。

结束标签

?>

结束标签表示PHP解释器可以停止处理该段代码，直至遇到另一个<?php标签。

PHP是一种称为脚本语言的编程语言。脚本语言运行在特定环境中；PHP的创建是为了在Web服务器上使用PHP解释器。一个单独的PHP页面通常称为脚本。

尽管在PHP中有的部分是不区分大小写字母的，但为了避免意想不到的错误，应将所有PHP代码都视为区分字母大小写的代码。

PHP页面是一个文本文件(HTML文件也是如此)。它的文件扩展名是.php，这告诉服务器将该文件发送给PHP解释器，以便它能够解释用PHP编写的指令。

如下所示，在一个PHP页面中包含如下部分：

- 图中紫色部分的PHP代码放在PHP标签内。在运行过程中，PHP解释器将处理这部分代码。
- 图中白色部分是HTML代码。这部分代码会自动添加到发送给浏览器的HTML文件中(因为PHP解释器不需要对它执行任何操作)。

PHP标签中的每条指令都称为语句。大多数语句是单行语句，以分号结束。如下情况中，语句后面的分号可以省略：

- PHP代码块的最后一行
- PHP代码块只包含一条语句

推荐在每个语句的末尾添加分号，这样有助于避免错误。

该页面中计算了糖果的总价，其中糖果每袋3美元，共购买了5袋，最后将总价保存在名为$total的变量中。将在HTML页面中展示出该值。你将在第1章中学习PHP代码如何实现该功能。

```
<?php                                          开始PHP标签
   $price    = 3;                              语句
   $quantity = 5;                              语句          PHP块
   $total    = $price * $quantity;             语句
?>                                             结束PHP标签
<!DOCTYPE html>
<html>
   <head>
      <title>Cost of Candy</title>
   </head>
   <body>
      <p>Total: $ <?php echo $total; ?></p>    PHP块
   </body>
</html>
```

PHP向浏览器发送文本和HTML 页面的方式

echo命令指示PHP解释器将文本或标签发送到浏览器。

在echo命令后用引号括住的任何文本和/或HTML 代码将发送到浏览器，以便在页面中显示。echo命令后可以使用单引号或双引号，但左引号和右引号必须匹配。

下面语句中的两个引号分别告诉了PHP解释器文本的开始和结束位置。文本又称为字符串字面量。行尾的分号告诉PHP解释器这是语句的结束位置。

```
echo '<b>Hello!</b>';
```

写入浏览器

要显示的文本和标签

要在发送给浏览器的文本中显示引号，需要在引号之前添加一个反斜杠符号(\)。反斜杠告诉PHP解释器不要将\后紧随的引号视为文本内容的结束符号，而应看作文本内容的一部分。程序员称此为对引号进行转义。

下例中的echo命令使用双引号写出HTML链接。href属性中的URL必须在引号中，所以将这些引号进行了转义。下面的代码最终显示为HTML链接PHP。

```
echo "<a href=\"http://notes.re/php\">PHP</a>";
```

echo命令的起始引号

用转义后的引号来包含URL

echo命令的结束引号

也可通过将双引号放入任何想要输出的文本和HTML中，再将这些文本和HTML放入单引号中来展示双引号。

这是可行的，因为PHP解释器会寻找一个匹配的单引号来表示文本的结束。

```
echo '<a href="http://notes.re/php">PHP</a>';
```

echo命令的起始引号

双引号中的HTML属性

echo命令的结束引号

在页面中写入内容

PHP section_a/intro/echo.php

```
<!DOCTYPE html>
<html>
  <head>
    <title>echo Command</title>
    <link rel="stylesheet" href="css/styles.css">
  </head>
  <body>
    <h1>The Candy Store</h1>
①   <h2><?php echo 'Ivy\'s'; ?> page</h2>
②   <?php echo '<p class="offer">Offer: 20% off</p>' ?>
  </body>
</html>
```

结果

在左边的代码框中：

- 右上角是该文件在下载的示例代码中的文件路径。
- 图中的编号对应于代码的执行步骤。

① echo命令使用单引号写出访问者名字与其后的's；反斜杠字符用于转义访问者名字和字母s之间的引号'。

② echo命令向页面添加一个段落。<p>元素有一个class属性。

因为写入页面的文本和标签放在单引号中，所以HTML属性可以使用双引号。

虽然在echo命令后使用单引号或双引号都是允许的，但最好选择其中一种并坚持使用。本书主要使用单引号，以便文本内容可以包含这里所示的HTML 属性。

试一试：在步骤①中，将Ivy更改为你自己的名字并保存文件。当再次刷新页面时，欢迎语将展示你的名字。

注意：如果在echo命令后使用双引号，PHP解释器将检查文本是否包含变量(在第32~36页中可看到)。如果包含变量，解释器将写出变量保存的值。而在单引号中，PHP解释器会默认将引号中的内容都视为文本。参见第52页中的示例。

注释

添加描述PHP代码的注释是一个很好的习惯。
注释有助于你一段时间后仍能回想起代码所做的事情，
也帮助其他人理解你的代码。

单行注释的开始符有以下两种：

- 两个斜杠 //
- 单个磅(或哈希)符号 #

这些字符告诉PHP解释器忽略该行上的任何后续PHP代码，直至遇到结束标签?>。

```
echo "Welcome";  // Display greeting
echo "Welcome";  #  Display greeting
```

单行注释

多行注释是指在PHP文件中跨越多行的注释。这样就提供了向代码添加更详细描述或注释的能力。

斜杠和星号/*告诉PHP解释器忽略所有内容，直至遇到星号*和斜杠/。

```
echo "Welcome";
/*
After welcome message:
- Add profile image next to member's name
- Make both a link to the member's profile page
*/
```

多行注释

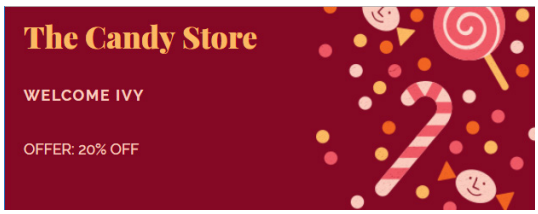
在代码中添加注释

PHP section_a/intro/comments.php

```php
<?php
  /*
  This page displays the member's name
  and details of a current offer
  */
?>
<!DOCTYPE html>
<html>
  <head>
    <title>Adding Comments to Your Code</title>
    <link rel="stylesheet" href="css/styles.css">
  </head>
  <body>
    <h1>The Candy Store</h1>
    <h2><?php echo 'Welcome Ivy'; // Show name ?></h2>
    <?php echo '<p class="offer">Offer: 20% off</p>' ?>
  </body>
</html>
```

① ②

结果



这个示例与前一个很相似，区别是这里向代码中添加了注释。

① 页面以描述代码功能的多行注释开始。

② 在欢迎消息之后，一条单行注释指示将要显示的内容。

注释部分的内容不会被添加到发送给浏览器的HTML文件中；它们仅在PHP代码中可见。

试一试：在步骤①中，向注释中添加另一行文本。

试一试：在步骤②中，将双斜杠字符更改为#(磅/哈希符号)。

注意：本书使用了大量注释来帮助描述示例中每行代码的作用。有经验的程序员很少像本书中这样一行一行地使用这么多注释。

第I部分

基本编程指令

1 **变量、表达式与操作符**

每次PHP页面执行任务时，都可使用不同的值来完成任务，因此学习如何使用变量在代码中表示数据是很重要的。你还将学习如何使用表达式和操作符来处理这些值。

2 **流程控制**

PHP页面并非总以相同的顺序运行相同的代码行。通过流程控制，能够编写PHP 解释器使用的规则，以确定接下来应该运行哪一段代码。

3 **函数**

使用函数可将执行任务需要的所有单条语句组合在一起。这不仅有助于组织代码，而且如果页面需要多次执行某个相同的任务，还可以避免重复执行相同的指令。

4 **类和对象**

代码用于表示网站的会员、销售的产品和显示的商品等概念。程序员使用类和对象将表示这些不同概念的代码组合在一起。

第1章

变量、表达式与操作符

本章将展示变量如何保存那些每次请求PHP页面时都发生变化的数据，以及表达式和操作符如何处理变量中的值

变量使用名称来表示每次PHP页面请求时都可以发生变化的值：

- 变量名描述了变量所保存的数据类型
- 变量值是本次请求该页面时应保存的值

一旦该页面运行完毕，HTML文本信息送往浏览器时，PHP解释器将忘掉该变量(以便下次页面运行时，它可以存入不同的值)。

PHP能够辨别存在于同一变量中的不同数据类型(如文本和数值)；以下是一些常用的数据类型：

- 字符串表示文本
- 整型表示整数
- 浮点型表示小数
- 布尔型仅有两个值，分别是true和false
- 数组可以保存一系列关联的名称和值

了解变量后，你将学习如何在表达式中使用多个值来创建单个值。例如，保存在两个变量中的文本可以拼接在一起形成一个句子，或者保存在一个变量中的数值乘以另一个变量中的数值可得到一个新值。

表达式依赖于操作符创建单个值。例如，+操作符用于将两个值相加，而-操作符用于将一个值减去另一个值。

变量

变量能保存每次请求PHP页面时都可改变的数据。这些变量使用变量名来表示可改变的值。

创建变量并给它赋值，需要遵循以下几条规则：

- 变量名必须以$开头，后跟一个或多个单词，用来描述变量可以保存的数据类型。
- 使用=为变量赋值，因此=也被称为赋值操作符。
- 确定变量值。

如果变量保存的是文本，则这些文本需要写在一对引号中。使用单引号或双引号均可，但它们必须成对出现。(例如，请勿以单引号开始，而以双引号结束)。如果变量保存的是数值或布尔值(true或false)，则不需要将变量值放在引号中。

当创建变量时，程序员们称之为声明变量。当该变量获得1个值时，则称为给变量赋值。

声明变量并为其赋值后，通过变量名就可在PHP代码中访问该变量当前保存的值。

当PHP解释器遇到一个变量名时，它会用该变量保存的值替换变量名。在如下的例子中，echo命令用于展示保存在$name中的变量值。

创建和访问变量

PHP section_a/c01/variables.php

```php
<?php
① $name  = 'Ivy';
② $price = 5;
?>
<!DOCTYPE html>
<html>
  <head>
    <title>Variables</title>
    <link rel="stylesheet" href="css/styles.css">
  </head>
  <body>
③   <h1>The Candy Store</h1>
    <h2>Welcome <?php echo $name; ?></h2>
④   <p>The cost of your candy is
      $<?php echo $price; ?> per pack.</p>
  </body>
</html>
```

结果

本章其余部分将通过PHP代码演示如何为变量赋值。后续章节中，为变量赋予的值将来自访问者提交的HTML表单、URL中的数据以及数据库。

在本示例中，可看到在页面的顶部创建了两个变量，并分别对它们进行了赋值。

① $name 保存了该网站当前访问者的名字。因为名字是文本信息，所以将名字放在一对引号之间。

② $price 保存了购买一份糖果的价格。由于价格是数值，所以它的值没有放在引号中。

接下来，在发送回访问者浏览器的HTML中可以看到：

③ 使用echo命令将访问者的名字展示在页面中。

④ 在页面中展示糖果的价格。

试一试：在步骤①中，更改$name变量的值以保存你的名字。然后保存文件并在浏览器中刷新页面。你会看到页面中展示了你的名字。

试一试：在步骤②中，将糖果价格改为2。然后保存文件并在浏览器中刷新页面。你会看到页面中展示了新的价格。

为变量命名

变量的名称应该描述它所保存的数据内容。推荐使用以下规则创建变量名。

1

以美元($)符号开头。

```
$greeting
greeting
```

2

然后加上一个字母或下画线(不要使用数值)。

```
$greeting
$2_greeting
```

3

再后使用字母A~z(大写和小写)、数值和下画线的任意组合。不允许使用破折号或句点。

```
$greeting_2
$greeting-2
$greeting.2
```

注意：$this有特殊的含义。请勿用它作为变量名。

```
$this
```

使用变量名来描述变量保存的数据使代码更容易理解。

如果使用多个单词来描述变量保存的数据，那么通常使用下画线分隔每个单词。

变量名是区分大小写的，所以$Score和$score是两个不同的变量。但通常情况下，变量名应该避免使用相同的单词(即使由不同大小写字母组成)，因为这样很可能让阅读代码的其他人感到困惑。

从技术角度看，使用来自不同字符集的字符(如中文或西里尔字母)都是允许的，但通常认为只使用字母A~z、数值和下画线(因为在支持其他字符时存在一些复杂问题)是更好的做法。

标量(基本)数据类型

PHP包含三种保存文本、数值和布尔值的标量数据类型。

字符串数据类型

程序员称一段文本为字符串。字符串数据类型可以由字母、数值和其他字符组成，但它们都只用于表示文本。

```
$name = 'Ivy';
```

字符串总是用单引号或双引号括起来。但左引号必须与右引号匹配。

```
$name = 'Ivy';
$name = "Ivy";
$name = "Ivy';
$name = 'Ivy";
```

数值数据类型

数值数据类型允许使用它们所保存的值执行数学操作，如加法或乘法。

```
$price = 5;
```

数值不需要使用引号括住。如果将数值放在引号中，它们将被视为字符串而非数值。

PHP有两种数值数据类型：

- int表示整数，表示没有小数位的数值(如275)。
- float则保存浮点数，表示小数(如2.75)。

null数据类型

PHP还有一种名为null的数据类型。只具有null值，表示没有为变量指定值。

布尔数据类型

布尔数据类型只能取两个值中的一个：true或false。这些值在大多数编程语言中都很常见。

```
$logged_in = true;
```

true和false应该用小写字母书写，并且不要放在引号中。刚接触到布尔值概念时，会觉得这一概念很抽象，但很多东西都可用true或false表示，例如：

- 访问者是否已登录?
- 他们同意条款和条件吗?
- 产品符合免运费条件吗?

类型转换

在第60~61页中，将看到PHP解释器如何将值从一种数据类型转换为另一种数据类型(例如，一个字符串转换为一个数值)。

更新变量中的值

可通过给变量赋新值来更改或覆盖保存在变量中的值。这与在创建变量时为其赋值的方式相同。

① $name变量完成了初始化。这意味着声明了变量并赋给它一个初始值，如果未在页面的后面更新变量，将一直使用这个初始值。

$name的初始值为Guest；因为它是一串文本，所以需要写在引号中。

② 然后为$name变量赋一个新值Ivy。

③ $price变量表示一包糖果的价格。

接下来，可看到将被发送回访问者浏览器的HTML。

④ 使用echo命令将变量写入页面。它显示了在步骤②中分配给$name变量的新值。

⑤ 在页面中展示糖果的价格。

section_a/c01/updating-variables.php PHP

```php
<?php
① $name  = 'Guest';
② $name  = 'Ivy';
③ $price = 5;
?>
<!DOCTYPE html>
<html>
  <head>
    <title>Updating Variables</title>
    <link rel="stylesheet" href="css/styles.css">
  </head>
  <body>
    <h1>The Candy Store</h1>
④   <h2>Welcome <?php echo $name; ?></h2>
    <p>The cost of your candy is
⑤      $<?php echo $price; ?> per pack.</p>
  </body>
</html>
```

结果

试一试：在步骤②中，更改$name变量的值并写入你的名字。然后保存文件，并在浏览器中刷新页面。你会看到你所写入的名字。

试一试：在步骤②之后添加新行，并给$name变量赋予一个别的名字。然后保存文件，并在浏览器中刷新页面。将看到页面显示的是你写入的名字。

数组

变量还可以保存数组，数组用于保存一系列相关值。数组又被称为复合数据类型，因为它们可以保存多个值。

数组就像一个容器，它保存一组相关的变量。数组中的每一项都称为元素。就像变量用变量名来表示值一样，数组中的每个元素都具有：

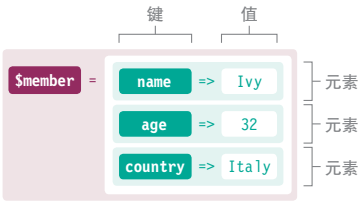
- 键，它的作用类似于变量名
- 值，它是键名所代表的数据

PHP有两种类型的数组：

- 在关联数组中，每个元素的键是描述它所代表的数据的名称。
- 在索引数组中，每个元素的键是一个称为索引号的数值。

关联数组

以下数组所保存的数据代表网站会员。每次使用数组时，键(描述保存在数组每个元素中的数据)中使用的名称将保持不变。

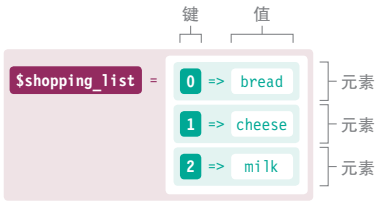


在这两个示例中，保存在数组中的每个值都是标量数据类型(单个数据片段)。

在第44页中，可以看到数组的例子，其中一个元素保存着另一个数组。

索引数组

以下数组用于保存购物列表。每次使用这样的列表时，它们可以保存不同数量的元素。键不使用名称来描述列表中的每个项，而是使用索引值(这是一个从0开始的整数)。



注意：索引值从0(而不是1)开始。列表中的第一个元素的索引号为0。第二个元素由索引号1标识，以此类推。索引号通常用来描述列表中条目的顺序。

关联数组

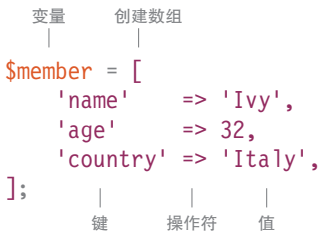
要创建关联数组，需要给数组中的每个元素(或项)指定一个键来描述它所保存的数据。

要将关联数组保存在变量中，请使用：

- 一个变量名，用于描述数组所要保存的值
- 操作符，用于进行赋值
- 方括号，用于创建数组

在方括号或圆括号内，请使用：

- 键名(加引号)
- 双箭头操作符=>
- 元素值(字符串加引号；数值和布尔值不加引号)
- 逗号(跟在每个元素后)



还可使用如下语法创建关联数组，即单词array后跟圆括号(而不是方括号)。

```
$member = array(
  'name'    => 'Ivy',
  'age'     => 32,
  'country' => 'Italy',
);
```

要访问关联数组中的元素，请使用：

- 保存数组的变量
- 方括号和引号
- 需要检索的元素的键

创建和访问关联数组

PHP section_a/c01/variables.php

```php
<?php
$nutrition = [
    'fat'   => 16,
    'sugar' => 51,
    'salt'  => 6.3,
];
?>
<!DOCTYPE html>
<html>
  <head> ... </head>
  <body>
    <h1>The Candy Store</h1>
    <h2>Nutrition (per 100g)</h2>
    <p>Fat:   <?php echo $nutrition['fat']; ?>%</p>
    <p>Sugar: <?php echo $nutrition['sugar']; ?>%</p>
    <p>Salt:  <?php echo $nutrition['salt']; ?>%</p>
  </body>
</html>
```

① ②

结果

① 在本示例中，创建了一个关联数组，并将其保存在名为$nutrition的变量中。

数组是在方括号内创建的。它有三个元素(每个元素对应一个键/值对)。使用=>操作符为每个键赋值。

② 要展示保存在数组中的数据，请按如下步骤操作：

- 使用echo命令将跟随其后的值写入页面。
- 后面跟着保存数组的变量的键。
- 加上方括号和引号，表示要访问的键的名称。

例如，要将糖的内容写入页面，可使用echo $nutrition['sugar']。

试一试：在步骤①中，修改数组的值。分别给以下键赋值：

- 为fat赋值42
- 为suger赋值60
- 为salt赋值3.5

刷新页面以查看更新后的值。

试一试：在步骤①中，将另一个元素添加到数组中。使用protein作为键名为并将其赋值为2.6。然后在步骤②中，在页面中显示protein的值。

索引数组

在创建数组时，如果没有为其中的元素提供键，PHP解释器将为它分配一个称为索引值的数值。索引值从0(而非1)开始。

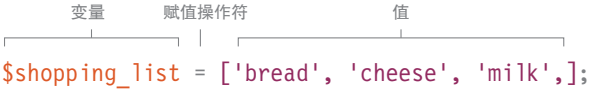
要将索引数组保存在变量中，请使用：

- 保存数组的变量，该变量的名称能够描述所保存的值
- 操作符，用于进行赋值
- 方括号，用于创建数组

在方括号或圆括号内，请遵循：

- 在数组中写明要保存的值(字符串需要放在引号中，而数值和布尔值则不用)
- 每个值后面跟一个逗号

每个元素都会被分配一个索引值。



在上例中，bread的索引值是0，cheese的是1，milk的是2。索引值通常用于表示数组中列出项的顺序。

也可以使用如下所示的语法创建索引数组，用的是圆括号(而不是方括号)。

```
$shopping_list = array('bread',
                       'cheese',
                       'milk');
```

添加到数组中的每个值可在同一行上，也可在新行上(如上所示)。

要访问索引数组中的项，请使用：

- 保存数组的变量的名称
- 后跟方括号(无需引号)
- 写入需要检索的元素的索引值(在方括号中)

下面的代码获取数组中的第3项，因此在本例中它将获取值milk。

创建并访问索引数组

PHP section_a/c01/indexed-arrays.php

```php
<?php
$best_sellers = ['Chocolate', 'Mints', 'Fudge',
    'Bubble gum', 'Toffee', 'Jelly beans',];
?>
<!DOCTYPE html>
<html>
  <head> ... </head>
  <body>
    <h1>The Candy Store</h1>
    <h2>Best Sellers</h2>
    <ul>
      <li><?php echo $best_sellers[0]; ?></li>
      <li><?php echo $best_sellers[1]; ?></li>
      <li><?php echo $best_sellers[2]; ?></li>
    </ul>
  </body>
</html>
```

结果

① 在本示例中，首先创建名为$best_sellers的变量。它的值是索引数组，保存了网站上最畅销的商品。

这个数组是用方括号创建的，数组项被添加到方括号内的数组中。因为数组中的项是文本，所以将它们放在引号中(数值和布尔值不会加引号)。每一项后面都有一个逗号。

② 页面中展示了最畅销的三个数组项：

- echo命令表示需要在页面展示其后的值。
- 然后是保存数组的变量名。
- 用方括号括住要检索的项的索引号。需要记住，索引号从0(而非1)开始。

试一试：在步骤①中，把Licorice添加到数组中的Fudge元素之后。在步骤②中，添加数组中的第4项和第5项。

更新数组

一旦创建了数组，就可以向其中添加新项或更新其中任何元素的值。

要更新保存在关联数组中的值，请使用如下语法：

- 保存数组的变量
- 其后跟随方括号
- 括号中放置键名
- 赋值操作符
- 要保存的新值

```
$member['name'] = 'Tom';
```

变量　键　新值

要向关联数组中添加新的数组项，需要执行与上面步骤完全相同的操作，但使用的是一个新的键名 (不能使用数组中已有的键名)。

当键名是字符串时，需要用引号包裹键名，因为引号表示字符串数据类型。

要更新保存在索引数组中的值，请使用如下语法：

- 保存数组的变量
- 其后跟随方括号
- 索引值 (无需引号)
- 赋值操作符
- 要保存的新值

```
$shopping_list[2] = 'butter';
```

变量　索引值　新值

在第220页可以看到如何将新项添加到索引数组中。本页的执行过程与之不同，因为这里可以指定新项在数组中的位置。

索引数值周围不用加引号，因为数值数据类型无需引号。

两种数组的适用场景

如下情况更适合使用关联数组：

- 确切知道数组将保存哪些信息。这对于为每个元素提供一个键名是必要的。
- 需要使用键名获取单条数据。

如下情况更适合使用索引数组：

- 不知道有多少数据要保存在数组中(随着更多项目被添加到列表中，索引值也会增加)。
- 希望按特定顺序保存一系列值。

更改数组中保存的值

PHP section_a/c01/indexed-arrays.php

```php
<?php
$nutrition = [
    'fat'   => 38,
    'sugar' => 51,
    'salt'  => 0.25,
];
$nutrition['fat']   = 36;
$nutrition['fiber'] = 2.1;
?>
<!DOCTYPE html>
<html>
  <head> ... </head>
  <body>
    <h1>The Candy Store</h1>
    <h2>Nutrition (per 100g)</h2>
    <p>Fat:   <?php echo $nutrition['fat']; ?>%</p>
    <p>Sugar: <?php echo $nutrition['sugar']; ?>%</p>
    <p>Salt:  <?php echo $nutrition['salt']; ?>%</p>
    <p>Fiber: <?php echo $nutrition['fiber']; ?>%</p>
  </body>
</html>
```

① 本例首先将一个数组保存在变量$nutrition中。

组成数组的每个元素的键和值并不需要另起一行(如这里所示)，但如果它们各占一行，则更便于阅读。

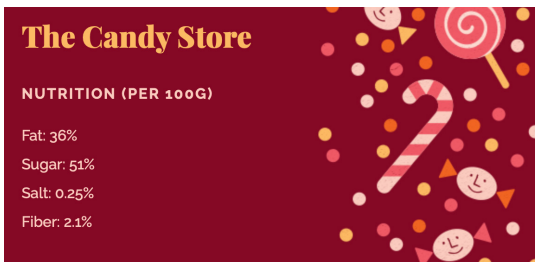
② 将fat中保存的值从38更新为36。

③ 新元素被添加到数组中。键名为fiber，值为2.1。

④ 数组中的值被将写入页面。

试一试：在步骤③之后，添加一个名为protein的键，并将其赋值为7.3。

结果



在数组中保存数组

数组中的每个元素都可以保存为另一个数组。当数组的每个元素都保存着另一个数组时，称该数组为多维数组。该类型的数组可用于表示表中展示的数据。

很多时候都需要在数组的元素中保存一组相关的值(例如，在传统的表中看到的数据)。在右边的表格中，上面有三个会员，表中还记录了他们的Age和Country。

该表的每一行(每个会员)都可以使用索引数组的一个元素表示。然后，每个元素可以保存一个关联数组，该数组保存每个会员的姓名、年龄和国籍。

Name	Age	Country
Ivy	32	UK
Emi	24	Japan
Luke	47	USA

索引数组的索引值由PHP解释器自动分配。每个关联数组后面的逗号表示该元素值的结束。

```
$members = [
    ['name' => 'Ivy',  'age' => 32, 'country' => 'UK',],
    ['name' => 'Emi',  'age' => 24, 'country' => 'Japan',],
    ['name' => 'Luke', 'age' => 47, 'country' => 'USA',],
];
```

如果要获取保存Emi数据的数组，请使用：

- 保存该索引数组的变量。
- 要访问的元素的索引号放在方括号中(请记住，索引数组从0开始，且数值没有放在引号中)。

```
$members[1];
```

如果要获取Luke的Age数值，请使用：

- 保存该索引数组的变量。
- 保存Luke数据数组的元素索引值(放在方括号中)。
- 想要访问的Luke数据数组中的元素的键(放在第二组方括号中；因为键是一个字符串，所以把它放在引号中)。

```
$members[2]['age'];
```

多维数组

PHP section_a/c01/multidimensional-arrays.php

```php
<?php
$offers = [
  ['name' => 'Toffee', 'price' => 5, 'stock' => 120,],
  ['name' => 'Mints',  'price' => 3, 'stock' => 66,],
  ['name' => 'Fudge',  'price' => 4, 'stock' => 97,],
];
?>
<!DOCTYPE html>
<html>
  <head> ... </head>
  <body>
    <h1>The Candy Store</h1>
    <h2>Offers</h2>
    <p><?php echo $offers[0]['name']; ?> -
      $<?php echo $offers[0]['price']; ?> </p>
    <p><?php echo $offers[1]['name']; ?> -
      $<?php echo $offers[1]['price']; ?> </p>
    <p><?php echo $offers[2]['name']; ?> -
      $<?php echo $offers[2]['price']; ?> </p>
  </body>
</html>
```

结果

① 在本示例中，首先在名为$offers的变量中保存一个索引数组。

数组中的每个元素保存着一个关联数组，这些关联数组中保存着正在出售的商品名称、价格和库存数量。

② 该行中展示了第1个产品的name值(第1个产品的索引号为0)。

③ 该行展示了第1个产品的price值。

④ 这里显示了第2个产品的name和price值。

⑤ 这里显示了第3个产品的name和price值。

试一试：在步骤①中，将另一个名为Chocolate的产品添加到数组中。将其price设置为2，将其stock设置为83。然后，在步骤⑤之后，写出刚添加的新产品的名称和价格。

在下一章中，将可看到如何使用循环写出$offers数组中每个产品的name和price，无论该数组包含多少个产品。

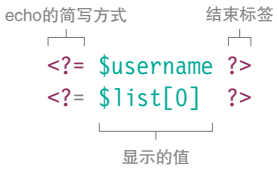
echo的简写方式

当PHP块只用于向浏览器写入一个值时，可以使用echo的简写来代替<?php echo ?>。

可以使用简写<?= $name ?>来代替<?php echo $name; ?>，这是唯一不需要把开始标签<?php写全的场景。

你不需要：

- 在开始标签中写php
- 写入echo命令
- 在结束标签前写入分号;



在本书前几章的许多例子中，可以看到每个PHP文件都分为两个部分：

- 首先，PHP代码将值保存在变量或数组中(也可用数组包含的数据执行任务)。
- 然后是返回给浏览器的HTML代码。下一页将使用上面所示的简写语法显示保存在变量中的值。

如果每个页面在开始时都创建需要显示的值并将这些值保存在变量中，则有助于将服务器上运行的PHP代码和访问者最终将看到的HTML代码进行明确的分隔。

创建HTML页面的部分应该使用尽可能少的PHP代码。在前面的示例中，页面这一部分的PHP代码只将保存在变量中的值写入HTML。

echo简写的使用

PHP section_a/c01/echo-shorthand.php

```php
<?php
① $name      = 'Ivy';
② $favorites = ['Chocolate', 'Toffee', 'Fudge',];
?>
<!DOCTYPE html>
<html>
  <head>
    <title>Echo Shorthand</title>
    <link rel="stylesheet" href="css/styles.css">
  </head>
  <body>
    <h1>The Candy Store</h1>
③   <h2>Welcome <?= $name ?></h2>
    <p>Your favorite type of candy is:
④      <?= $favorites[0] ?>.</p>
  </body>
</html>
```

结果

在这个例子中，可以看到已经在页面顶部创建了两个不同的变量，并放置在HTML代码的开始位置之前。

① $name变量保存站点会员的名称。由于这是一串文本，所以它被包裹在引号中。

② $favorites保存的数组表示会员最喜欢的糖果类型。

③ 用echo命令的简写将名字写入页面。

④ 用echo命令的简写将会员最喜欢的糖果类型写入页面。

试一试：在步骤①中，将保存在$name变量中的值更改为你的名字。在步骤②中，将你最喜欢的糖果类型添加到数组的开头。保存文件，并在浏览器中刷新页面。然后观察页面内容的变化。

表达式和操作符

当要创建一个新值时，通常使用两个(或更多)已存在的值进行运算后得到。表达式由一个或多个运算结构组成，但最终其计算结果应为单个值。在这个过程中，表达式使用操作符得到所需的结果。

两个值使用基本数学运算(加、减、乘和除)创建一个新值。下面的表达式将数值3乘以数值5，得到的值为15：

```
3 * 5
```

程序员认为通过表达式计算可得到一个新值。如下，新创建的值保存在一个名为$total的变量中：

```
$total = 3 * 5;
```

+、-、*、/、=等符号被称为操作符。

可以使用称为连接操作符的字符串操作符将两个或多个字符串连接在一起以创建一个较长的文本。下面的表达式将值'Hi'和'Ivy'连接起来，创建一个字符串。

```
$greeting = 'Hi ' . 'Ivy';
```

这两个字符串的连接计算结果为单个值HiIvy，该值保存在名为$greeting的变量中。

在本章的其余部分，将用到下一页介绍的操作符。

算术操作符

第50~51页

算术操作符提供处理数值，执行加法、减法、乘法和除法等任务的能力。

例如，如果某人购买3包糖果，每包5美元，你可以使用乘法操作符计算出这3包糖果的总价。

字符串操作符

第52~53页

字符串操作符提供处理文本的能力。共有两个字符串操作符用于将不同的文本片段组合成一个字符串。

例如，如果将会员的名字保存在一个变量中，而将会员的姓氏保存在第二个变量中，则可将两个变量连接起来以创建其全名。

比较操作符

第54~55页和第58页

顾名思义，比较操作符将比较两个值并返回布尔值true或false。

例如，如果取数值3和5，可以尝试比较它们，并观察结果：

- 3 大于 5 (false)
- 3 等于 5 (false)
- 3 小于 5 (true)

也可以对字符串进行比较，看看一个值是大于还是小于另一个值：

- 'Apple' 大于 'Banana' (false)
- 'A' 等于 'B' (false)
- 'A' 小于 'B' (true)

逻辑操作符

第56~57页和第59页

三个逻辑操作符and、or和not的运算结果为true或false。为了理解它们是如何使用的，请考虑以下两个问题；两者都可以用true或false来回答：

今天的温度高吗？天气是否晴朗？

- and操作符可以判断温度高并且天气晴朗。
- or操作符可以判断温度高或天气晴朗。
- not操作符可以判断这些问题的答案中是否有一个不成立(false)。例如，天气不晴朗吗？

以上这些结果值都是true或false。

算术操作符

PHP提供以下数学操作符的运算能力。
它们可以与数值或保存数值的变量一起使用。

名称	操作符	用途	示例	结果
加	+	两数相加	`10 + 5`	15
减	-	两数相减	`10 - 5`	5
乘	*	两数相乘 (注意：这是一个星号，不是字母x)	`10 * 5`	50
除	/	两数相除	`10 / 5`	2
模	%	两数求余数	`10 % 3`	1
幂	**	一个数以另一个数为幂	`10 ** 5`	100000
递增	++	在原有值上加1并返回新的值	`$i = 10;` `$i++;`	11
递减	--	在原有值上减1并返回新的值	`$i = 10;` `$i--;`	9

执行顺序

在一个表达式中可以执行多个算术运算，但是理解计算的顺序是很重要的：乘除在加减之前执行。

操作符的执行顺序可能导致结果与你的预期不符。例如，这里的数值是从左到右计算的。结果是16：

```
$total = 2 + 4 + 10;
```

然而，在下例中，结果是 42(不是60)：

```
$total = 2 + 4 * 10;
```

圆括号包裹的运算将优先执行，所以下面的结果总共是60：

```
$total = (2 + 4) * 10;
```

这里圆括号指明先计算2加4，再用其结果乘10。

算术操作符的使用

PHP section_a/c01/arithmetic-operators.php

```php
<?php
① $items    = 3;
② $cost     = 5;
③ $subtotal = $cost * $items;
④ $tax      = ($subtotal / 100) * 20;
⑤ $total    = $subtotal + $tax;
?>
<!DOCTYPE html>
  <html>
  <head> ... </head>
  <body>
    <h1>The Candy Store</h1>
    <h2>Shopping Cart</h2>
    <p>Items: <?= $items ?></p>
    <p>Cost per pack: $<?= $cost ?></p>
⑥   <p>Subtotal: $<?= $subtotal ?></p>
    <p>Tax: $<?= $tax ?></p>
    <p>Total: $<?= $total ?></p>
  </body>
</html>
```

结果

本示例展示了算术操作符如何与数值一起使用来计算一个订单的总价。

首先，创建两个变量：

① 购买的糖果数量($items)。

② 每袋糖果的价格($cost)。

接下来执行计算，并在创建HTML之前将结果保存在变量中。这有助于将PHP代码与HTML代码分离开来。

③ 订单的总价是用商品数量($items)乘以一包糖果的价格($cost)来计算的。

④ 因为需要按20%的税率计算缴纳的税费。为此，要将缴税前的总价($subtotal)先除以100(这是在括号中完成的，以确保优先计算)，再乘以20。

⑤ 最后，将应缴税费($tax)加到税前的总价中，得到总价($total)。

⑥ 在HTML页面中展示总价的最终结果。

试一试：修改步骤①中的糖果数量和步骤②中的价格。

字符串操作符

有时需要连接两个或多个字符串来创建单个值。连接两个或多个字符串的过程称为连接。

. 连接操作符

连接操作符是一个句点符号。它将一个字符串中的值与另一个字符串中的值连接起来。在下面的例子中，变量$name将运算得到字符串'Ivy Stone':

```
$forename = 'Ivy';
$surname  = 'Stone';
$name     = $forename . ' ' . $surname;
```

$forename和$surname变量之间加了一个空格；如果没有空格，$name变量保存的值将变为IvyStone。

只要在每个字符串之间使用连接操作符，就可以在一条语句中连接任意多个字符串。

可通过另一种方式连接保存在变量中的字符串，而不需要连接操作符。如果用双引号(而不是单引号)赋值，PHP解释器将双引号中的变量名替换为它们包含的值。例如，

```
$name = "$forename $surname";
```

.= 连接赋值符

如果要向现有变量添加一些文本，可以使用连接赋值操作符。可以把该操作符看作创建一个更新字符串操作的简写：

```
$greeting = 'Hello ';
$greeting .= 'Ivy';
```

这里的字符串'Hello'保存在一个名为$greeting的变量中。在下一行，连接赋值操作符将字符串'Ivy'添加到名为$greeting的变量所保存的值的末尾。

现在，$greeting变量保存的值为'Hello Ivy'。可以看到，它比左边的示例省去了一行代码。

拼接字符串

PHP section_a/c01/string-operator.php

```php
<?php
① $prefix  = 'Thank you';
② $name    = 'Ivy';
③ $message = $prefix . ', ' . $name;
?>
<!DOCTYPE html>
<html>
  <head>
    <title>String Operator</title>
    <link rel="stylesheet" href="css/styles.css">
  </head>
  <body>
    <h1>The Candy Store</h1>
    <h2><?= $name ?>'s Order</h2>
    <p><?= $message ?></p>
  </body>
</html>
```

结果

本示例中将展示为用户定制的信息。

① 首先，创建名为$prefix的变量为访问者保存消息的开始部分。该变量保存的字符串为'Thank you'。

② 创建第二个变量来保存访问者的名字。变量名为$name，访问者名字为Ivy。

③ 访问者的个人消息是通过连接(或拼接)三个值并将新值保存在一个名为$message的变量中创建的：

- 首先，将保存在$prefix中的值添加到$message中。
- 接下来，添加逗号和空格符。
- 最后，添加保存在$name中的值。

试一试：在步骤②中，将保存在$name中的值更改为你的名字。

试一试：在步骤③中，使用双引号(而非连接操作符)赋值的方式为$message变量赋值，例如：

```
$message = "$prefix $name";
```

比较操作符

比较操作符提供比较两个或多个值的能力。结果是布尔值true或false。

==

等于

该操作符用来比较两个值，看它们是否相同。

`'Hello' == 'Hello'`的结果为`true`

因为它们是相同的字符串。

`'Hello' == 'Goodbye'`的结果为`false`

因为它们是不同的字符串。

!= 或 <>

不等于

这两个操作符用来比较两个值，看它们是否不同。

`'Hello' != 'Hello'`的结果为`false`

因为它们是相同的字符串。

`'Hello' != 'Goodbye'`的结果为`true`

因为它们不是相同的字符串。

上面的操作符允许PHP解释器确定这两个值是否相等。下面的操作符更严格，因为它们同时检查值和数据类型。

上面的操作符会将数值3(整数)视为与数值3.0(浮点数)相等。而下面的操作符则将这两者判断为不相等(第60~61页将会说明：0将视为布尔值false，而1将视为true)。

===

等于

该操作符用来比较两个值，以检查值和数据类型是否相同。

`'3' === 3`的结果为`false`

因为它们类型不同。

`'3' === '3'`的结果为`true`

因为它们类型和值都相同。

!==

不等于

该操作符用于比较两个值，以检查值和数据类型是否不相同。

`3.0 !== 3`的结果为`true`

因为它们类型不同。

`3.0 !== 3.0`的结果为`false`

因为它们的类型和值都相同。

如果使用echo将布尔值写入页面，true将显示1，false将不显示任何内容。

< 和 >

小于和大于

< 检查左侧值是否小于右侧值。

4 < 3 的结果为 false

3 < 4 的结果为 true

> 检查左侧值是否大于右侧值。

z > a 的结果为 true

a > z 的结果为 false

<= 和 >=

小于或等于，大于或等于

<= 检查左侧的值是否小于或等于右侧的值。

4 <= 3 的结果为 false

3 <= 4 的结果为 true

>= 检查左侧的值是否大于或等于右侧的值。

z >= a 的结果为 true

z >= z 的结果为 true

<=>

宇宙飞船操作符

宇宙飞船操作符将其左值和右值进行比较，所得结果为：

0，如果两个值相等

1，如果左边的值更大

-1，如果右边的值更大

该操作符是在PHP 7中引入的(不能用于较早版本的PHP)。

1 <=> 1 的结果为：0

2 <=> 1 的结果为：1

2 <=> 3 的结果为：-1

逻辑操作符

比较操作符只得到一个单独的值：true或false。逻辑操作符可以与多个比较操作符一起使用，以比较多个表达式的结果。

在这行代码中，有三个表达式，其中每个表达式都将解析得到一个单独的值：true或 false。

表达式1(左边的)和表达式2(右边的)都使用比较操作符，并且两个表达式的结果都是false。

表达式3使用了逻辑操作符(而非比较操作符)。

逻辑与操作符(&&)验证两个表达式(两边)是否都返回true。在本例中，它们并非都返回true，因此整个表达式将计算为false值。

从运算顺序看，表达式1和2在表达式3之前运算。

推荐将每个表达式都放在相应的圆括号中。这有助于说明每一组括号中的代码应该计算为单个值。不使用圆括号也是可以的，但这样代码可读性较差。

&&

逻辑操作符与

该操作符验证多个条件：

```
((2 < 5) && (3 >= 2))
```

的返回值是 true

如果两个表达式的返回值都是true，则整个表达式返回true。如果其中一个表达式的返回值为false，则整个表达式返回false。

true && true 返回 true

true && false 返回 false

false && true 返回 false

false && false 返回 false

可用单词 and代替&&符号。

||

逻辑操作符或

该操作符至少验证一个条件：

```
((2 < 5) || (2 < 1))
```

的返回值是 true

如果其中一个表达式的返回值为 true，则整个表达式返回true。如果两个表达式的返回值为false，则整个表达式返回false。

true || true返回true

true || false返回true

false|| true返回true

false|| false返回false

可用单词or代替||符号。

!

逻辑操作符非

该操作符接受一个布尔值并对其求反：

```
!(2 < 1)
```

的返回值是 true

!表示对表达式的结果求反。如果表达式的计算结果为false(表达式前没有!)，则最终返回true。如果表达式的返回值是true，则最终返回false。

!true 返回 false

!false 返回 true

不能使用not代替!符号。

短路运算

逻辑表达式从左到右进行计算求值。一旦算出第一个表达式的返回值，并且PHP解释器知道了逻辑操作符，可能就不需要求第二个条件了，正如右边的示例中展示的那样。

```
((5 < 2) && (2 >= 2))
```

↑

计算结果为false。

继续测试第二个条件是没有意义的，因为第二个表达式的计算结果不影响整个表达式返回false。

```
((2 < 5) || (2 >= 2))
```

↑

计算结果为true。

继续测试第二个条件是没有意义的，因为第二个表达式的计算结果不影响整个表达式返回true。

比较操作符的使用

① 这里创建了三个变量。

- 第1个变量保存了客户想购买的糖果类型。
- 第2个变量表示商店库存共有5袋糖果。
- 第3个变量表示客户想购买8 袋糖果。

② 比较操作符检查所需的数量是否小于或等于库存数量。并将比较结果保存在一个名为$can_buy的变量中。

③ 页面中需要展示布尔值的情况很少。大多数情况下，布尔值将用于条件逻辑，该情况将在下一章中遇到。当尝试在页面上分别写下不同布尔值，页面展示情况如下：

- 布尔值为true，页面将展示1。
- 布尔值为false，页面将不展示。

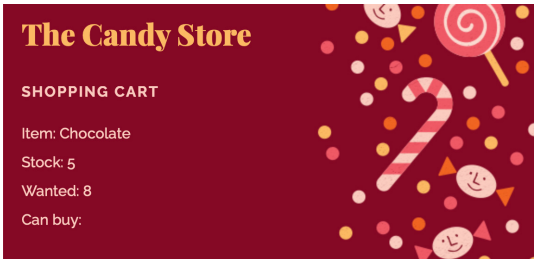
试一试：在步骤①中，交换$stock和$wanted的值。$can_buy中的值会改变。

在第75页中，将能学习到如何在使用比较操作符的表达式的结果为true或false时显示不同的消息。

section_a/c01/comparison-operators.php PHP

```php
<?php
$item    = 'Chocolate';
$stock   = 5;
$wanted  = 8;
$can_buy = ($wanted <= $stock);
?>
<!DOCTYPE html>
<html>
  <head> ... </head>
  <body>
    <h1>The Candy Store</h1>
    <h2>Shopping Cart</h2>
    <p>Item:    <?= $item ?></p>
    <p>Stock:   <?= $stock ?></p>
    <p>Wanted:  <?= $wanted ?></p>
    <p>Can buy: <?= $can_buy ?></p>
  </body>
</html>
```

结果



逻辑操作符的使用

PHP section_a/c01/string-operator.php

```php
<?php
$item    = 'Chocolate';
$stock   = 5;
① $wanted  = 3;
② $deliver = true;
③ $can_buy = (($wanted <= $stock) && ($deliver ==
true));
?>
<!DOCTYPE html>
<html>
  <head> ... </head>
  <body>
    <h1>The Candy Store</h1>
    <h2>Shopping Cart</h2>
    <p>Item:    <?= $item ?></p>
    <p>Stock:   <?= $stock ?></p>
    <p>Ordered: <?= $wanted ?></p>
    <p>Can buy: <?= $can_buy ?></p>
  </body>
</html>
```

结果

如下示例基于上一页面的代码。

① 客户想购买3袋糖果。

② 这里添加了一个名为$deliver的变量；用于保存一个布尔值来表示是否可以发货。

③ 这个表达式使用了两个比较操作符，它们分别用于：

- 首先检查是否有足够的糖果库存。
- 之后再检查是否可发货。

逻辑操作符&&检查两个操作符是否都返回true。如果经检查返回值均为true，那么$can_buy的值将为true，页面将显示数值1。

如果两个操作符并不都返回true，$can_buy将得到false，且不会显示任何内容。

试一试：在步骤①中，交换$stock和$wanted的值。$can_buy的返回值会改变。

在第75页上，将学习如何在表达式返回true或false时显示不同的消息。

数据类型转换

PHP解释器可以将一个值从一种数据类型转换为另一种数据类型。这称为类型转换，这种转换可能得到意想不到的结果。

PHP被认为是一种松散类型语言，因为在创建变量时，不需要指定变量所包含的值的数据类型。下面先给$title变量赋值为一个字符串值，然后赋值为整数：

```
$title = 'Ten';  // 字符串
$title = 10;     // 整数
```

可将PHP的方法与严格类型的编程语言(如C++或C #)进行比较，后者要求程序员在声明每个变量时指定数据类型。

当PHP解释器遇到与预期的数据类型不符的值时，它可以尝试将该值转换为预期的数据类型。这个过程叫作类型转换。

类型转换可能令人感到困惑，因为通过类型转换后，PHP解释器可能得到预期外的结果或错误。例如，下面的加法操作符将两个值相加。1是一个整数，但2是一个字符串，因为它是用引号包裹起来的。

```
$total =  1 + '2';
```

这种情况下，PHP解释器将自动尝试将字符串转换为数值，以便执行算术运算。因此，$total变量将保存数值3。

在下一页，可看到如何指定将值从一种数据类型转换为另一种数据类型的规则。下面的一些示例演示了类型转换：http://notes.re/php/type-juggling。

当一个值的数据类型更改为不同的数据类型时，程序员称之为该值的数据类型从一种类型转换为另一种类型。类型转换称为隐式转换，因为这是由PHP解释器执行而非程序员显式执行的转换。

如果程序员使用代码显式地对数值类型进行转换，则这种转换称为显式转换，因为PHP解释器被显式地告知如何执行类型转换。

数值

如果PHP解释器执行数学运算时需要两个数值，它会先将如下情况下的非数值变量转换为数值。

可以看看如下情况的结果会是什么：

- 数值与字符串相加
- 数值与布尔值相加

数值 + 字符串	可视为	返回值	描述
`1 + '1'`	`1 + 1`	`2` (int)	字符串包含一个有效的整数，视为整数
`1 + '1.2'`	`1 + 1.2`	`2.2` (float)	字符串保存一个浮点数，视为浮点数
`1 + '1.2e+3'`	`1 + 1200`	`1201` (float)	字符串保存一个带e的幂函数(指数为3)，视为浮点数
`1 + '5star'`	`1 + 5`	`6` (int)	字符串中的整数跟随有其他字符。该数值被视为整数，后面的字符将被忽略
`1 + '3.5star'`	`1 + 3.5`	`4.5` (float)	字符串保存一个跟随其他字符的浮点数。该数值被视为一个浮点数，后面的字符将被忽略
`1 + 'star9'`	`1 + 0`	`1` (int)	字符串不以整数或浮点数开头，被视为数值0

数值 + 布尔值	可视为	返回值	描述
`1 + true`	`1 + 1`	`2` (int)	布尔值true被视为整数1
`1 + false`	`1 + 0`	`1` (int)	布尔值false被视为整数0

字符串

当PHP解释器试图连接两个字符串时，将遵循如下的规则。

看看如下情况的结果会是什么：

- 将字符串与数值连接
- 将字符串与布尔值连接

字符串 . 数值	可视为	返回值	描述
`'Hi ' . 1`	`'Hi ' . '1'`	`Hi 1` (string)	整数被视为字符串
`'Hi ' . 1.23`	`'Hi ' . '1.23'`	`Hi 1.23` (string)	浮点数被视为字符串

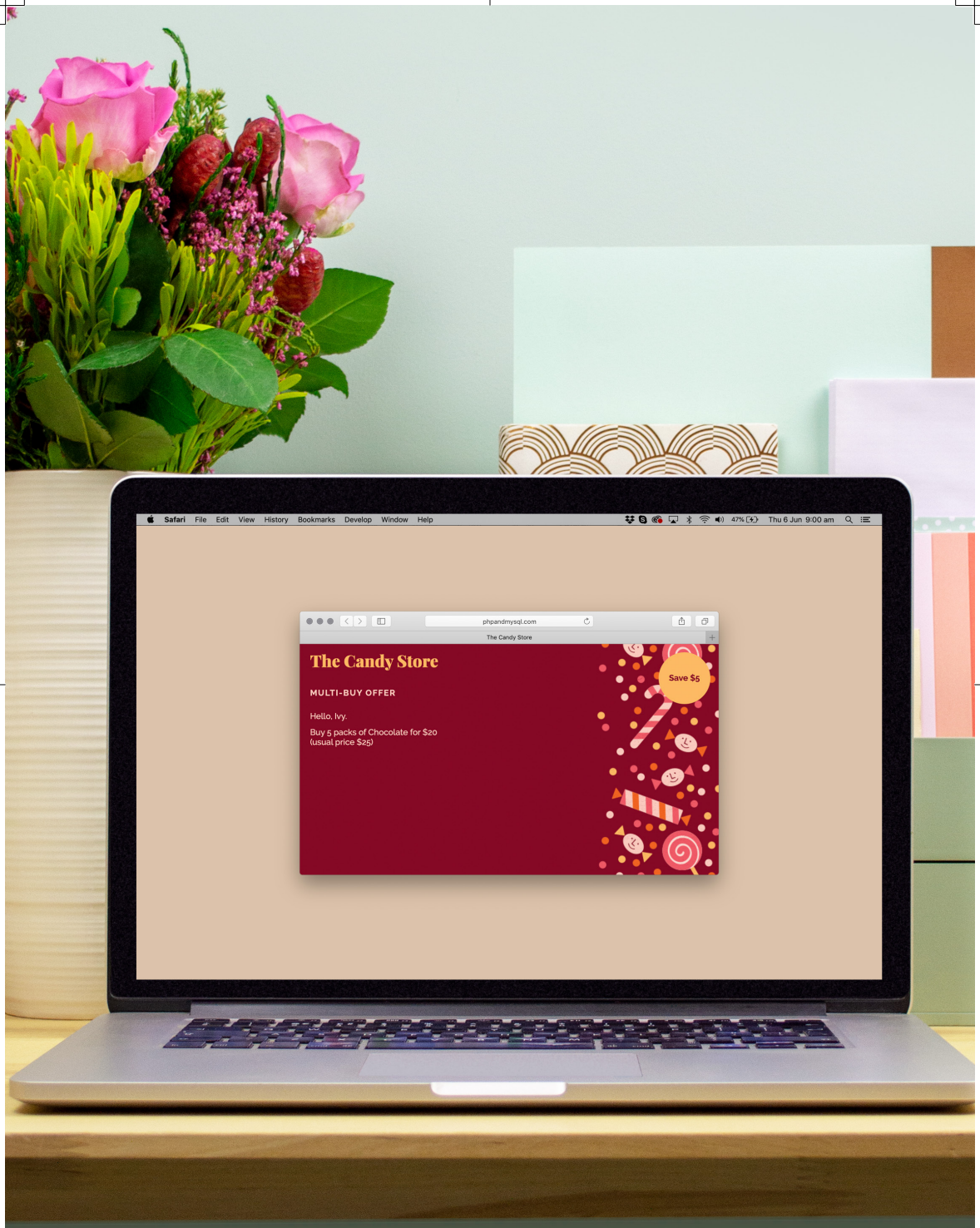
字符串 . 布尔值	可视为	返回值	描述
`'Hi ' . true`	`'Hi ' . '1'`	`Hi 1` (string)	true被视为字符1
`'Hi ' . false`	`'Hi ' . ''`	`Hi` (string)	false被视为空字符串

布尔值

当PHP解释器需要对某个值进行转换得到布尔值时，右表中显示的所有值都将被视为false。

其他任何值(任何文本、除0以外的数值或布尔值true)都被视为true。

值	数据类型	被视为
`false`	布尔值	`false`
`0`	整数	`false`
`0.0`	浮点数	`false`
`'0'`	以0开始的字符串	`false`
`''`	空字符串	`false`
`array[]`	空数组	`false`
`null`	Null	`false`



基本PHP页面

如下示例结合了本章中先前展示的几种技术。

利用PHP文件来创建一个HTML页面，告诉访问者当他们购买多包糖果时可以享有折扣。

该页面中将展示：

- 在变量和数组中保存信息。
- 使用连接操作符连接变量中的文本，为访问者创建个性化的问候语。
- 使用算术操作符执行计算，以确定页面上显示的价格。
- 将PHP解释器创建的新值写入页面的HTML内容中。

另外，如果更新了保存在变量中的值，页面将自动展示新产品和价格。

处理和展示数据

当开始编写PHP文件时，页面中通常混用HTML和 PHP代码。推荐的做法是尽可能将两部分代码分开。

- 首先使用PHP创建将显示在HTML页面中的值，并将这些值保存在变量中(在下一页中，指的是虚线以上的代码)。
- 然后，页面的下方可专注于显示HTML内容。这一部分中，PHP代码应该仅用于显示保存在变量中的值(在下一页中，指的是虚线以下的代码)。

先关注页面开头处的PHP代码。

① 下一页的示例首先声明一个变量来保存访问者的用户名。之所以命名为$username，是因为变量名应该总是以美元符号开始，后跟一个描述它所保存的数据类型的名称。

② 声明一个名为$greeting的变量，为访问者保存一条问候语。这将使用字符串操作符来连接字符串Hello和访问者的名字。

③ 创建一个名为$offer的变量来保存特价商品的详细信息。它的值是一个有四个元素的数组：

- 购买的商品名
- 购买的数量
- 每袋糖果的原价(无折扣)
- 每袋糖果的折扣价

数组中的第一个元素描述了订单中的商品名，数据类型为字符串。数组中其他元素的值则是整数。

④ 创建一个名为$usual_price的变量。它的值是未打折之前商品的总价。这是通过将数组中保存的两个值(数量和价格)相乘得到的。

⑤ 创建一个名为$offer_price的变量。它的值是打折后商品的总价。这是通过将保存在数组中的数量和折扣价格相乘得到的。

⑥ 创建一个名为$saving的变量来保存客户节省的花销。这是通过从保存在$usual_price(在步骤④中创建)中的值减去保存在$offer_price变量(在步骤⑤中创建)中的值来计算得到的。

页面的后半部分(虚线以下部分)将创建返回给浏览器的HTML。它从HTML DOCTYPE声明开始。PHP只用于写出前面步骤中保存在变量中的值。

⑦ 问候语是单词Hello后面跟着访问者的名字。这里使用echo命令的缩写将问候语写在页面上。

⑧ 保存在$saving变量(在步骤⑥中创建)中的节省的花销显示在黄色圆圈中。CSS用于将这个圆圈放在浏览器窗口的右上角。

⑨ 步骤中的<p>标签用于描述订单的细节，展示了访问者所要购买的糖果数量和名称。

⑩ 随后是保存在$offer_price中的折扣总价和保存在$usual_price中的非折扣总价。

PHP section_a/c01/string-operator.php

```php
<?php
① $username = 'Ivy';                                    // 保存用户名的变量

② $greeting = 'Hello, ' . $username . '.';              // 问候语的值是“Hello,”+用户名

③ $offer = [                                            // 创建数组来保存订单
      'item'     => 'Chocolate',                        // 订单上的商品名
      'qty'      => 5,                                  // 要购买的商品数量
      'price'    => 5,                                  // 每袋糖果的原价
      'discount' => 4,                                  // 每袋糖果的折扣价
  ];

④ $usual_price = $offer['qty'] * $offer['price'];       // 未打折的总价
⑤ $offer_price = $offer['qty'] * $offer['discount'];    // 打折的总价
⑥ $saving      = $usual_price - $offer_price;           // 节省的总花销
?>
```

```html
<!DOCTYPE html>
<html>
  <head>
    <title>The Candy Store</title>
    <link rel="stylesheet" href="css/styles.css">
  </head>
  <body>
    <h1>The Candy Store</h1>

    <h2>Multi-buy Offer</h2>

⑦   <p><?= $greeting ?></p>

⑧   <p class="sticker">Save $<?= $saving ?></p>

⑨   <p>Buy <?= $offer['qty'] ?> packs of <?= $offer['item'] ?>
⑩      for $<?= $offer_price ?><br>(usual price $<?= $usual_price ?>)</p>
  </body>
</html>
```

试一试：在步骤①中将用户名修改为你自己的名字。在步骤②中，更新显示给访问者的问候语，使用Hi(而不是Hello)。

在步骤③中，将$offer数组中键名为qty的值更改为3。

在步骤③中，将糖果的price更新为6。

小结

变量、表达式与操作符

- 变量用于保存每次运行脚本时会发生变化的数据。
- 标量数据类型包含字符串、整数、浮点数和值为true或false的布尔值。
- 数组是一种复合数据类型，用于保存一组相关值。
- 数组中的每一项都称为元素。关联数组中的元素有键和值。索引数组中的元素有索引号和值。
- 字符串操作符用于连接(拼接)字符串中的文本。
- 算术操作符使用数值执行数学运算。
- 比较操作符比较两个值，看一个值是否等于、大于或小于另一个值。
- 可以使用逻辑操作符与(&&)、或(||)、非(!)来组合多个表达式的结果。