

PHP 起步

欢迎来到 PHP 的世界！

PHP 语言是最受欢迎的 Web 开发语言之一。它以学习简单、开发快速、性能稳定而倍受 Web 开发人员的青睐。PHP 不仅使用人员众多、数以万计的 Web 站点用它构建，而且有强大的社区支持，使得无论是用 PHP 开发 Web 应用，还是学习 PHP 语言，都会快速有效、事半功倍。在各种 Web 开发语言、框架、概念纷扰的今天，PHP 仍以其独特魅力吸引更多的 Web 开发人员学习和使用。

选择 PHP，没错的！

1.1 初识 PHP

PHP 是一种服务器端的、嵌入 HTML 的脚本语言。它的语法结构和 C 语言极为相似。为了能够快速编写动态 Web 页面，PHP 还加入了自己的某些语言特征，这些特征都非常容易理解和使用。先来看一个嵌入有 PHP 代码的 HTML 文档。

```
<html>
<head><title>Welcome</title></head>
<body>
<?php
echo "Welcome to PHP's world!";
?>
</body>
</html>
```

1.2 LAMP——锋利四剑客：Linux、Apache、MySQL 和 PHP

要想学习 PHP，就有必要了解和 PHP 关系密切的其他 3 种技术：Linux 操作系统、Apache 网络服务器和 MySQL 数据库。

LAMP 这个名词最早由 Michael Kunze 创造，用来代表 Linux 操作系统、Apache 网络服务器、MySQL 数据库和 PHP（如果可以，Perl 和 Python 也是不错的选择），LAMP 正是这 4 种技术的首字母。

PHP 作为强有力的 Web 开发语言，和 Linux、Apache、MySQL 的支持是密不可分的。它们都是开源软件，并且有强有力的社区支持，它们的完美组合构成了当今 Web 开发世界中不可忽视的一极重要力量。而且，这支力量近年一直在不断地变得更加强大。虽然，这 4 种技术并不是专门被设计成在一起工作的，但多年来，这些软件之间的兼容性不断完善，不仅完改善了个组件之间的协作，扩展出更多的功能，而且在目前几乎所有的 Linux 版本中都默认包含了这些产品，使得这些产品共同组成了一个强大的 Web 应用平台。

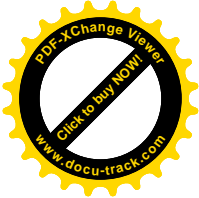
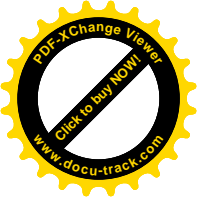
1.3 PHP Web 编程的体系结构和基本内容

概括地说，任何 Web 编程体系，无外乎服务器端和客户端的程序开发。基于 PHP 的 Web 应用开发也不例外，其中客户端的开发包括使用 HTML 设计 Web 页面，使用样式表控制 Web 页面的显示效果，还需要客户端的脚本语言来控制浏览器的特效、验证 HTML 表单数据等，这些脚本语言包括 JavaScript、VBScript 等。服务器端的开发就需要掌握 PHP 语言和 MySQL 数据库的有关内容。

本节的各小节将对这些内容做简要说明，之后的各节将较为详细地介绍 HTML、样式表和 JavaScript，它们都是 Web 开发的基础内容，对这些内容不了解的 PHP 初学者有必要掌握。

1.3.1 了解 PHP、HTML、层叠样式表（CSS）和 JavaScript 及其关系

作为服务器端的脚本语言，PHP 多数情况下都是和 HTML 相互搭配来使用的。PHP 用来



完成和逻辑有关的动态内容。PHP 程序执行的输出结果，通过 HTML 文档表现给用户。一般情况下，在 HTML 页面中需要输出数据的地方内嵌入 PHP 代码，这个页面也就成为了 PHP 程序。

在创建 HTML 页面时，都要考虑页面设计。如页面布局、页面颜色、字体、边距空白等。这些设置可以单独在页面中进行，但那样会使 HTML 页面变得臃肿，也不便于将来的维护。层叠样式表解决了这些问题，它允许页面设计人员在层叠样式表（即 CSS）文件里设定页面表现效果，如字体大小、边距控制等。CSS 使得 HTML 页面的表现更加丰富、美观，也更加容易维护。

1.3.2 HTML 文档

HTML 文档就是使用 HTML 标记语言创建的文档。通过浏览器浏览的网页，几乎都是 HTML 文档，或者是由服务器端程序生成的 HTML 文档。下面就是一个 HTML 文档的代码。

```
<html>
<head><title> HTML 文档示例</title></head>
<body>
<h1>HTML 语言</h1>
<p>First Web Page</p>
</body>
</html>
```

1.3.3 使用样式表实现页面效果

按照 HTML 的设计初衷，文档的内容、结构与格式是分离开来的。如果过多地在 HTML 文档中控制页面的显示效果，会使 HTML 文档的维护越来越难以维护，也违背了 HTML 的设计初衷。为了解决这个问题，引入了样式表的概念。一个简单的样式表如下所示。

说明：样式表负责控制页面的格式和显示效果，HTML 文档负责控制结构和显示内容。

```
p{
font-size:10pt;
}
```

这个样式表定义的含义是，HTML 文档中所有用<p>标签定义的内容，都将使用 10pt 大小的字体显示。关于样式表语法和如何使用样式表，将在后面做讲述。

1.3.4 客户端的响应

Web 页面作为客户端，有时需要直接对用户的请求作出响应。这主要是通过客户端脚本来实现，这些脚本语言包括 JavaScript 等。

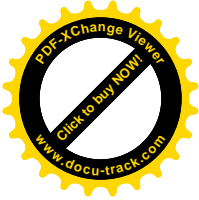
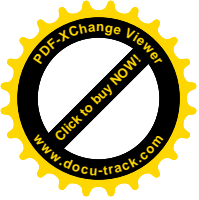
可以在 HTML 页面中加入 JavaScript 脚本，这些脚本可以直接在浏览器里执行。这样，就可以达到在客户端响应用户请求的需求。一般这样的响应主要是验证表单数据、用户操作提示等。

本书将在 1.6 节介绍 JavaScript。

1.3.5 嵌入式脚本页面

PHP 代码一般都是嵌入在 HTML 文档当中，通过服务器解释这些 PHP 代码，并用代码执行产生的结果替换 PHP 代码内容，最后返回给用户的是内嵌代码执行后的 HTML 文档。例如内嵌生成“Hello World”的 HTML 页面代码如下：

```
<html>
<head>
<title>HTML-PHP</title>
</head>
<body>
```



```
<h2><?php ehco "Hello World"; ?></h2>
</body>
</html>
```

1.4 Web 编程的基础知识之一：HTML

HTML 是创建 Web 应用的最基本内容，无论是动态还是静态页面，最终都要产生 HTML 文档。所有的 Web 开发都要涉及到用 HTML 设计 Web 页面。本节将介绍 HTML 语言及如何使用 HTML 创建 Web 页面。

1.4.1 HTTP 协议简介

网络上的计算机之间要进行通信，就必须遵守一定的规则，这种通信规则就是网络协议。协议保证网络上各种不同的计算机之间能够理解彼此传递的消息，好比操不同语言的人们之间，通过翻译来理解对方所说话的含义一样。现在应用最广的 Internet 使用的是 TCP/IP 协议，而浏览 WWW 使用的是 HTTP 协议，即超文本传输协议（HyperText Transfer Protocol），此协议建立在 TCP/IP 协议之上。

浏览网页的过程，其实就是一系列请求/响应的过程。HTTP 协议定义了这个请求/响应过程中请求和响应的格式，及维护 HTTP 链接的内容。

1.4.2 HTML 基本知识：标签

HTML 的全称是 HyperText Markup Language，即超文本标记语言。它是一种简单、通用的标记语言。之所以叫标记语言，是因为，HTML 通过不同的标签，来标记文档的不同部分。读者看到的每个 Web 页面，都是由 HTML 通过一系列定义好的标签生成的。

从简单的文本编辑器，如 Windows 的记事本，到专业化的编辑工具，如 Dreamweaver，都可以用来编辑 HTML 文档，编辑好的 HTML 文档必须按后缀.html 或.htm 来保存，最后，通过浏览器打开 HTML 文档，来查看页面效果。

在 HTML 文档中，标签是包含在“<”和“>”之间的部分，如<p>就是一个标签。标签一般是成对使用的，如和同时使用，其中是开始标签，是结束标签。HTML 的标签不区分大小写，因此和表示的含义相同。

1.4.3 HTML 基本知识：元素

HTML 元素由标签定义，标签所定义的内容就叫“元素”，元素包含在开始标签和结束标签之间。

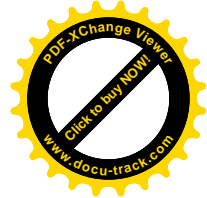
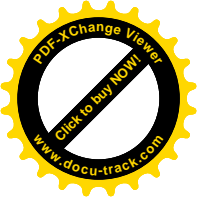
每一种 HTML 元素，一般都会有一个或数个属性，属性用来设置或表示元素的一些特性、名称或显示效果等。属性放在元素标签中，紧跟标签名称之后，它和标签名称之间有一个或数个空格。元素的每个属性都有一个值，属性的值的设定使用“属性=”值”的格式，可以为属性的‘值’加上引号或不加引号。下面的 HTML 代码为标签<form>设置了 name 属性，其值为 login，表示这个表单的名称为 login。

```
<form name= " login " >
```

1.4.4 HTML 基本知识：HTML 的基本元素

元素是组成 HTML 文档的关键，本节介绍 HTML 常用的几种基本元素。

1. 标头元素:HTML 使用标签<head>定义一个标头，结束标签是</head>。
2. 标题元素:这里的标题，是指 HTML 文档中，内容的标题。标题元素由标签<h1>到<h6>定义。
3. 段落元素:HTML 中使用标签<p>和</p>定义一个段落。
4. 字形元素:使用标签和定义一个粗体字形元素
5. 链接:HTML 文档中指向其他 Web 资源，如另一个 HTML 页面、图片等的链接被称为“锚”。在 HTML 中使用标签<a>和定义一个锚元素，即链接元素，也就是说在<a>和之间的内容，会成为一个超链接。



6. 图像元素:使用标签定义一个图片元素,在标签中使用属性 src 来指向一个图片资源,象这样, 其中 url 是指向资源所在位置。这个位置可以是一个 URL,也可以是一个相对地址,

7. 表格元素:使用标签<table>和</table>定义一个表格元素。一个表格由“行”构成,每一行由数据单元构成。表格的“行”用标签<tr>和</tr>定义,数据单元用标签<td>和</td>定义。

8. 列表元素:说明:HTML 的列表分为无序列表和有序列表。

9. 表单元素:HTML 表单是一个包含表单元素的区域,表单元素一般会作为数据,提交给后台服务器做处理。表单域用标签<form>和</form>定义。表单元素是那些定义在表单域里,可以输入信息的元素,如文本框、单选按钮、下拉列表等。

1.4.5 创建 HTML 文档

使用任何一款文本编辑器,都可以编辑 HTML 文档。编辑好的 HTML 文档,按后缀名.html 或 htm 保存,最后通过浏览器访问 HTML 文档。打开一个文本编辑器,键入如下 HTML 代码(代码 1-15),

1.4.6 创建 HTML 列表

HTML 最基本的列表有两种:无序列表和有序列表,列表可以嵌套使用,亦可同时使用。创建无序列表,通过设置标签的 type 属性,可以更改列表项的标志。type 属性的取值可以是 disc, square 或 circle 中的任意一个。

创建有序列表,通过设置标签的 type 属性,可以为有序列表的列表项设置不同的顺序标志。

对于无序列表来说,

1.4.7 创建页面表格

标签<table>和</table>用来创建一个表格。标签<table>主要的常用属性有 width、align 和 border。

width 属性:设置表格的宽度,如下所示。这里定义一个宽度为 600px (600 点像素)的表格。

```
<table width= " 600px " >
```

align 属性,设置表格相对于浏览器区域的对其方式,可以选取的值有:left(居左)、center(居中)或 right(居右)。

border,设置表格的边框宽度,单位为像素。

1.4.8 建立页面表单

使用标签<form>和</form>创建一个表单。<form>的主要属性是 action 和 method。

action 属性:用来指定表单数据被提交后,处理这些数据的程序的地址。如下 HTML 代码,表示当表单提交后,表单的数据将被传到文件 login.php,由 login.php 来处理传入的数据。

```
<form action= " login.php " >
```

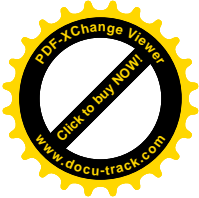
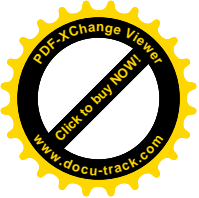
method 属性:指定用何种 HTTP 方式传递数据。

有两种传递数据的方式:POST 方式和 GET 方式。POST 方式将表单数据放在 HTTP 数据的正文部分传递。GET 方式将表单数据加到 action 所指的地址之后传递。

1.5 Web 编程的基础知识之二:层叠样式表(CSS)简介

层叠样式表的英文全称是 Cascading Style Sheet (简称 CSS)。HTML 最初设计时,只是用来定义文档的内容。比如标签<table>就是用来定义一个 HTML 文档的表格。页面的布局由浏览器显示,并不由 HTML 标签控制。

随着一个站点 HTML 文件的越来越多,如果用仍 HTML 标签排版和控制页面显示效果,它的局限性和困难性的问题会日益突出。甚至,在标签无法满足页面显示效果时,有的设计



人员又加入 JavaScript 来控制页面效果。可以想象，这样 HTML 文档会变得越发臃肿，并且越来越难以维护。

层叠样式表 (CSS) 的出现解决了这个问题，即使用 CSS 决定网页内容如何显示，用 CSS 控制页面显示效果。

1.5.1 样式表的基本语法

一个样式 (Style) 的语法由 3 部分构成：Selector (选择器)，属性 (Property)，属性值 (Value)。

格式如下：

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

例如下面的例子，p 就是 selector，color 就是属性，blue 就是属性值。

```
p {color:blue}
```

其中 p 就是指 p 标签 <p>，这个样式 (Style) 的含义表示凡是用 <p> 标签标记的文档内容，其文本颜色显示为蓝色。HTML 中所有的标签都可以作为 selector。

如果想为 Style 添加多个属性，可以在两个属性之间用分号分隔。下面的 Style 就包含两个属性，一个是对齐方式，其值为居中，一个字体颜色，其值为红色，它们之间用分号分隔开。

```
p {text-align:center;color:red}
```

1.5.2 设置页面字体格式

使用 CSS 可以设置字体的名称、大小、显示风格等样式。

1. 字体名称属性：font-family (用这个属性设定字体的名称，如 Arial, Tahoma, Courier 等。)

2. 字体大小属性：font-size (font-size 属性用来设定字体的大小，字体大小的单位有多种，常见的是 pt 和 px。)

3. 字体风格属性：font-style (这个属性有 3 个值可选：normal (正常显示)、italic (斜体显示)、oblique (斜体显示)。normal 是缺省值。)

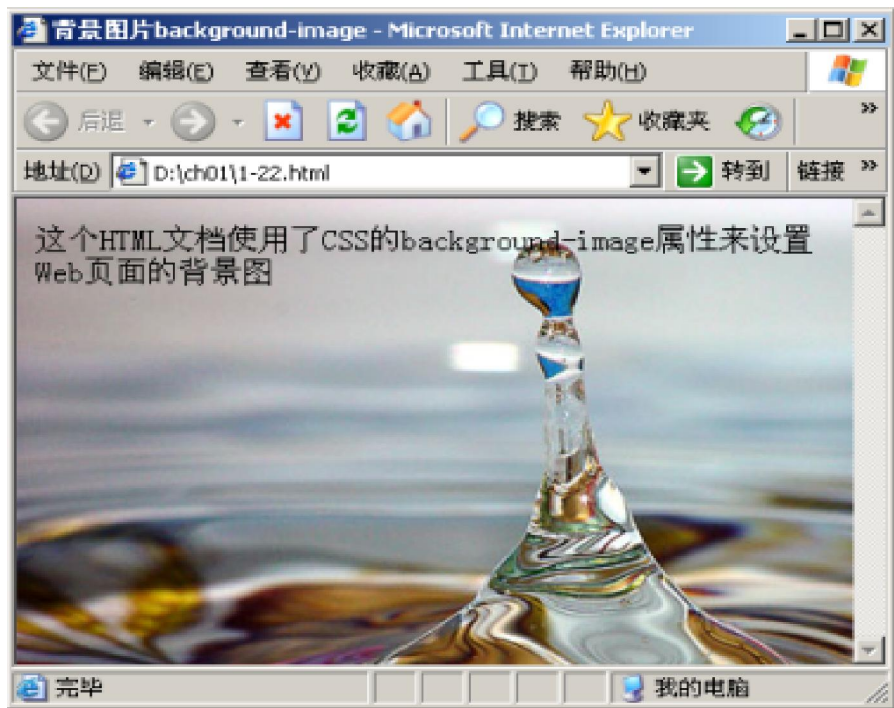
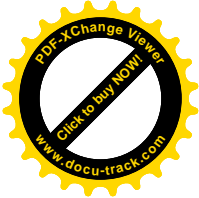
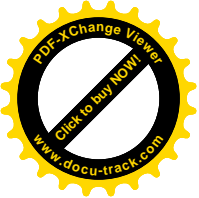
1.5.3 设置页面颜色和背景

使用 CSS 背景颜色属性和背景图片属性，可以很方便地设置 Web 页面的颜色和背景。

1. 背景颜色属性：background-color (此属性为 HTML 元素设定背景颜色，示例代码如下。)

```
body {background-color:#99FF00;} (上面的代码表示 body 这个 HTML 元素的背景颜色值是 #99FF00。)
```

2. 背景图片属性：background-image (此属性为 HTML 元素设定背景图片)



页面背景效果

1.5.4 处理页面的边距和填充

边距属性是用来设置页面中一个元素所占空间的边缘到相邻元素之间的距离。

左边距属性：margin-left：这个属性用来设定左边距的宽度。示例代码如下。

```
.d1{margin-left:1cm}
```

右边距属性：margin-right：这个属性用来设定右边距的宽度。示例代码如下。

```
.d1 {margin-right:1cm}
```

上边距属性：margin-top：这个属性用来设定上边距的宽度。示例代码如下。

```
.d1 {margin-top:1cm}
```

下边距属性：margin-bottom：这个属性用来设定下边距的宽度。示例代码如下。

```
.d1{margin-bottom:1cm}
```

边距属性：margin：这个属性是设定边距宽度的一个快捷的综合写法，用这个属性可以同时设定上下左右边距属性。还可以为上下左右边距设置相同的宽度。示例代码如下。

```
.d1 {margin:1cm}
```

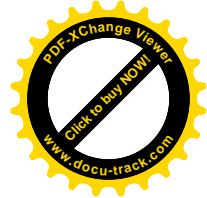
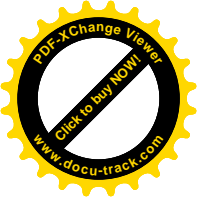
1.5.5 理解 HTML 层的概念

HTML 中，使用标签<div>和</div>来定义一个层，通过 CSS 指定不同的属性值，可以定位层，从而实现页面的布局。

利用层可以非常灵活地放置内容，例如可以将层前后放置、隐藏某些层而显示其他层、在屏幕上移动层等。可以在一个层中放置背景图像，然后在该层的前面放置第二个层，从而包含带有透明背景的文本。通常，可以把层看成一个容器，在层里可以放置其他更多的 HTML 元素。

1.5.6 按 Web 标准建立网页

传统 HTML 布局，一般都是使用表格。表格定位比较简单快捷，但容易出现表格嵌套表格的现象，这样对页面的后期维护很不方便，而且表格越多，越会影响到浏览器解析 HTML 文档的速度，使页面打开的速度变慢。如今，页面设计人员更多地使用 HTML 层和 CSS，用于 Web 页面的布局。



所谓的 Web 标准，其实并没有统一的标准。不过，它似乎朝着使用<div>和 CSS 进行页面布局的方向行进。DIV+CSS 只是具体的实现技术手段，并不能涵盖 web 标准。web 标准不单单是布局的问题，更重要的是信息结构清晰、内容与表现相分离，而 DIV+CSS 技术能较好的实现这种思想。因此，当前看到的多数符合标准的页面都是采用 DIV+CSS 制作。

1.5.7 在网页中引入样式表

最常用的样式表引入方式有：内嵌样式 (Inline Style)、内部样式表 (Internal Style Sheet) 和外部样式表 (External Style Sheet)。

内嵌样式：内嵌样式是写在标签里面的。内嵌样式只对所在的标签有效。如下代码所示。
<p style="font-size:20pt; color:red">用 Style 定义字体</p>

这里在<p>标签内增加样式：style="font-size:20pt; color:red"，这个样式只对当前<p>标签有效。

内部样式：内部样式表是写在 HTML 的<head></head>里面的。内部样式表只对所在的网页有效。

外部样式：外部样式是指，将样式(Styles)写在一个以.css 为后缀的 CSS 文件里，然后在每个需要用到这些样式(Styles)的网页里引用这个 CSS 文件。

1.6 Web 编程的基础知识之三：JavaScript 基础

JavaScript 是一种基于对象和事件驱动并具有安全性能的脚本语言，它是 Web 页面设计的重要组成部分，它被嵌入在 HTML 文档中，由浏览器解释执行。可以使用 JavaScript 设计出更有效果的 Web 页面、验证 Web 页面表单数据、创建 cookie 等。

1.6.1 网页中的 JavaScript

JavaScript 可以出现在 HTML 文档的任何地方，但必须包含在 “ <script language="JavaScript"> ”和“ </script> ”之间。一般情况下，使用“ <script language="JavaScript"> ”和“ </script> ”包含的 JavaScript 代码。

1.6.2 JavaScript 的变量

JavaScript 变量是用来存储数据的地方。这样在需要用这个值的地方就可以用变量来代表，一个变量可以是一个数字、文本等。JavaScript 变量的命名区分大小写，例如，名为 computer 的变量和名为 Computer 的是不一样的。而且，变量名必须以字母或下划线开头。

JavaScript 是一种对变量的数据类型要求不太严格的语言，所以不必声明每一个变量的类型，但在使用变量之前先进行声明是一种好习惯。在 JavaScript 中用 var 语句声明一个变量，象下面这样：

```
var name = value
```

1.6.3 JavaScript 的基本语句

这一小节仅介绍 JavaScript 的几种常用的基本语句，它们是：

if...else 条件语句。

switch 选择语句。

for 循环语句。

while 循环语句。

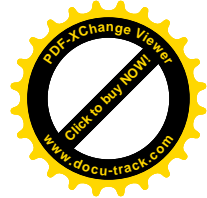
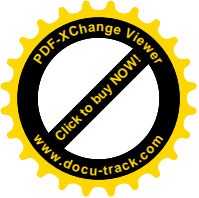
1.6.4 使用简单的对话框

通常会使用 JavaScript 在网页上产生弹出对话框的效果。用 JavaScript 可以创建的对话框主要有以下所示的两种。

警告对话框。

确认对话框。

下面将通过实例代码，来了解如何使用 JavaScript 创建这些对话框。使用函数 alert() 创建警告对话框，它的语法格式如下所示。



alert(" message ")

1.6.5 JavaScript 的基本事件

所谓事件，是指用户与 Web 页面交互时产生的各种操作，例如单击 Web 页面上的超链接或按钮时，就产生一个单击（click）事件（event）。浏览器为了响应某个事件而进行的处理过程，叫做事件处理。

注意：浏览器在程序运行的大部分时间都等待事件的发生，事件能够被 Web 页面中的 JavaScript 捕捉到，并在事件发生时，自动调用 JavaScript 的事件处理函数，完成事件处理。

除此之外，浏览器自己的一些动作也可以产生事件，例如，当浏览器载入一个 Web 页面时，就会发生载入（load）事件，卸载一个页面时，就会发生卸载（unload）事件。本小节将介绍几种常见的 JavaScript 事件，它们是以下所示的 3 种。

鼠标单击事件（onClick）：当用户单击 Web 页面上的超链接或者按钮时，就会触发 onClick 事件。

页面载入和卸载事件（onload 和 onUnload）：onload 事件发生在 Web 页面完全下载完成后。onUnload 事件发生在用户离开（即关闭）当前 Web 页面时。onload 事件通常用来检测浏览器的一些信息，如浏览器类型、版本等。它们也可以用来处理用户 cookie。

提交事件（onSubmit）：该事件发生在页面表单的“提交”按钮按下并放开之后。通常使用该事件的处理过程验证表单数据的有效性。

1.6.6 JavaScript 的函数

函数是一个可以完成特定功能可执行的代码块，它由一条或多条 JavaScript 语句组成。通常将那些会反复使用，或者用来完成某种功能的代码写成函数，以便利于代码的重用。JavaScript 中函数的语法如下所示。

```
function name(var_1,var_2,...,var_n)
{
    statement
}
```

1.6.7 JavaScript 的对象

JavaScript 是一种面向对象的编程语言。一段文字、一个图片、一个表单都可以看作是一个对象，每个对象都有自己的属性、方法和事件。属性表示了该对象的某些特征，如字符串的长度、文本框里的文字等。方法可以理解成该对象可以处理的一些事情。

JavaScript 提供了很多对象，这些对象可以直接在 JavaScript 里使用。本小节介绍 JavaScript 的一些常用的基本对象。

1. String 对象

String 对象代表一个字符串，定义该对象的最简单办法就是直接赋值。如下代码所示。此时，变量 str 就是一个 String 对象，它的值是“Hello JavaScript”。

```
var str = " Hello JavaScript "
```

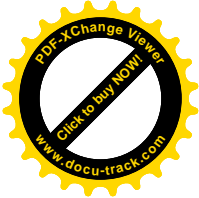
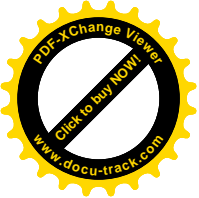
2. 日期对象——Date

顾名思义，日期对象就是用来处理时间的 JavaScript 对象。要定义一个日期对象，需要使用 new 运算符，如下代码所示。

```
var some_date = new Date()
```

3. 数组对象——Array

数组对象用来存放一组数据（或者说是对象），这些数据可以是不同类型的。数组中的每个对象都有一个“下标”，用来表示它在数组中的位置。数组的下标从 0 开始算起，所以，第一个对象的位置是 0，第二个对象的位置是 1，以此类推。JavaScript 中定义一个数组的方法如下代码所示。



```
var my_arr = new Array()
```

4. window 对象

该对象表示的是一个浏览器窗口。引用该对象的方法或属性时，不需要使用“window.xxx”这种形式，而是直接使用该方法或属性。该对象常见的属性如下所示。

name：当前窗口的名称。 states：表示浏览器窗口下方状态栏所显示的内容。通过对改属性发赋值，可以改变浏览器状态栏显示的内容。 self：指浏览器窗口本身。 history：浏览器历史访问对象。

location：浏览器地址对象。 document：文档对象。 window 对象常见的方法如下所示。

open()：打开一个窗口。 close()：关闭一个已打开的窗口。 alert()：弹出一个包含“确定”和“取消”按钮的对话框。该方法已经在前面介绍过。

5. document 对象

document 对象代表浏览器窗口内的文档。该对象包含了整个 HTML 文档，并且可以访问当前页面中的所有元素。该对象最常用的方法就是 write，它向当前 HTML 文档写入数据。前面的内容已多次使用过该方法，这里不再重复举例。document 有一个重要属性：cookie。接下来的一小节，就向读者介绍 JavaScript 中的 cookie。

1.6.8 JavaScript 中的 cookie

cookie 是指当用户访问某一个 Web 站点时，由服务器存储在客户端计算机中的一些变量。它通常用来区别不同的访问用户。当同一台计算机请求访问某个页面时，浏览器也会将 cookie 发送给服务器。

cookie 一般是这样的形式：cookie 名称=cookie 值。cookie 的名字一般使用字母和数字命名，cookie 的值要求是可以 URL 编码的字符。所有的 cookie 都有一个失效期，过了失效期，计算机就会将这个 cookie 删除。JavaScript 中通常使用 document 的 cookie 属性存储 cookie。它的用法如下所示。

```
document.cookie='cookieName='+escape('cookieValue')+';expires=' +  
expirationDateObj.toGMTString();
```

1.7 开始 PHP 之旅：搭建 PHP 开发环境

从这节开始，正式进入和 PHP 有关的内容。本节将介绍 Apache、MySQL 和 PHP 的下载、安装及配置，分别讲述在 WindowsXP 和 Linux/UNIX 下开发环境的搭建。为了便于初学者理解和掌握，本书将按 Windows 操作系统下的开发环境讲述 PHP 语言及其编程。

1.7.1 下载 Apache、MySQL 和 PHP

Apache、MySQL 和 PHP 都是开源产品，均可从其官方网站下载，并且可以免费使用。

1. Apache 的下载

Apache 服务器的官方网站是 <http://httpd.apache.org>

当前，可以通过镜像地址 <http://apache.mirror.phpchina.com/httpd/binaries/win32/>

下载 Windows 的 Apache 安装程序，这个镜像地址列有各个版本的 Apache 安装程序，

2. MySQL 的下载

MySQL 的官方网站是 <http://www.mysql.com>

可以通过 <http://dev.mysql.com/downloads/mysql/5.0.html>

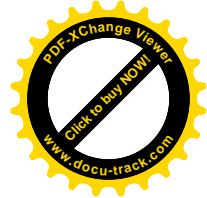
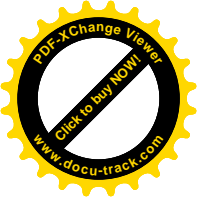
下载最新版本的 MySQL，

3. PHP 的下载

PHP 的官方网站是 <http://www.php.net>

可以通过 <http://www.php.net/downloads.php>

下载最新的 PHP 版本。对于 Windows 系统，可以下载安装版本或者 PHP 压缩包文件，



1.7.2 在 Windows 下配置开发环境

下载完需要的软件后，本节介绍如何实现这些软件的安装和配置。

1. Apache 的安装
2. PHP 的安装
3. 配置开发环境

1.7.3 在 Linux/UNIX 下配置开发环境

因为在 Linux/UNIX 下配置 PHP 的开发环境，对读者的技术要求稍微高一些，而且涉及到一些 Linux/UNIX 方面的基础知识，如命令的使用、程序编译等，考虑到便于初学者学习和掌握，本节讲述比较简略。另外，本书主要以 Windows 为平台讲述 PHP 语言及其开发，因此初学者也可以跳过本节的学习。

1. MySQL 的安装
2. 安装 Apache
3. PHP 的安装

1.7.4 善其事利其器——PHP 编辑器的选择

毋庸置疑，开发人员需要一个功能强大的 IDE (Integrated Development Environment, 集成开发环境)，就好似一个好的猎人，需要一支好的猎枪一样。使用 PHP 做 Web 应用的开发，无论是从开发人员的需求上考虑，还是项目的开发效率上考虑，都需要有一个功能强大而且易用的编辑器做支持。如今有许多编辑器可供 PHP 开发人员选择，它们各有优点，开发人员可以根据自己的需求、使用习惯等方面选择这些编辑器，将它们作为编写代码、开发程序的高效工具。

本节将简单介绍三款常用也是比较受开发人员欢迎的 PHP 编辑器，希望对初学者有所帮助。这三款编辑器是：UltraEdit、Eclipse 和 Zend Studio。

1.8 立竿见影

经过上几节的讲述，配置好了 PHP 的开发环境。从现在开始，通过一些小例子体验一下 PHP，看看 PHP 是怎样一种计算机程序语言，简单了解一下用 PHP 能做哪些事。从向 PHP 问好开始，向它说声“Hello,PHP!”!

1.8.1 编写第一个 PHP 程序——“Hello,PHP!”

打开最上手的编辑器，来编写第一个 PHP 程序。这个 PHP 程序非常简单，在编辑器键入以上代码后，按文件名 hello.php 保存在 Apache 安装目录的 htdocs 目录下。然后打开浏览器，键入地址：<http://localhost/hello.php>，如果一切正确无误的话，将会看到浏览器显示出“Hello,PHP!”字样，

1.8.2 使用 PHP 处理 HTML 表单

这一小节介绍如何使用 PHP 处理 HTML 表单数据。在这小节，将编写一个 PHP 程序，试着用它处理 HTML 文档提交的数据。

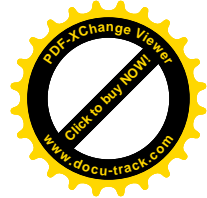
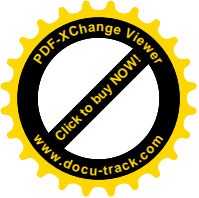
1.9 深入了解 Apache、PHP 和 MySQL

在正式学习 PHP 语言之前，还需要对 Apache、PHP 配置和 MySQL 的有关方面做一些比较深入的说明，这样即有助于读者对前几节知识加深认识，也有助于读者学习和理解后续内容。以下几小节的内容，主要将以 Windows 平台下的配置做讲解，Linux/UNIX 平台下的情形类似。

1.9.1 Apache 服务器目录

Apache 安装完成后，有一些目录需要读者进一步了解，如：conf 目录、htdocs 目录、logs 目录和 modules 目录。

conf 目录：conf 目录下存放着一些 Apache 配置文件，其中最常用到的就是 httpd.conf，这是 Apache 的核心配置文件，Apache 服务器的很多重要配置及功能实现都要在这个文件里



完成。这个文件也是 PHP 开发人员需要经常改动的文件，本书将在下一小结讲述 httpd.conf，设置一些重要的配置项。

htdocs 目录：这个目录被 Apache 默认为服务器的根目录。这就是说，在默认情况下，开发人员编写的 HTML 文档和 PHP 程序，只有放到这个目录下，才可以被访问或被执行。

logs 目录：这个目录下存放着服务器级别的日志文件。如 access.log 记录用户访问的文件及其访问日期时间、方式等。这个目录下的有些文件，有时可以用来做 PHP 程序调试之用，因为服务器记录了些错误在这些日志里，开发人员可以通过这些错误，来调试 PHP 程序。

modules 目录：这个目录下存放有 Apache 执行的核心模块，当 Apache 启动时，它会根据配置从这个目录里载入需要的模块。一般情况下，PHP 开发人员不需要对这个目录了解更多。

1.9.2 进行基本的 Apache 配置

Apache 服务器的很多功能、任务等重要配置，都是通过修改 httpd.conf 来完成的。如设置服务器根目录、服务器超时时间、监听端口、Apache 运行模块的载入、服务器语言字符设置等。下面这段文字就是从 httpd.conf 摘出的一部分。

```
#  
# Timeout: The number of seconds before receives and sends time out.  
#  
Timeout 300
```

1.9.3 Apache 的启动与停止

对于 Windows 用户来说，可以通过 Windows“服务”来启动和关闭 Apache 服务器。进入 Windows 控制面板里的管理工具，打开服务，找到 Apache 一项，通过“操作”菜单项或图标按钮即可完成 Apache 的启动与停止。

1.9.4 PHP 的核心配置——php.ini

与 Apache 相似，PHP 也有自己的核心配置文件 php.ini，PHP 分析器启动时会读取这个文件。与 httpd.conf 不同，php.ini 以英文分号“;”做为注释符。本节主要讲述以下 PHP 基本的配置项，这些配置项均在 php.ini 中设置。

doc_root：该项用来指定 PHP 页面在服务器的根目录，以本书安装情况为例，该项配置如下所示。

```
doc_root = d:\Apache2\htdocs
```

注意：httpd.conf 配置项与其值之间用空格分割，而 php.ini 使用“=”。

extension_dir：该项用来指定 PHP 在哪个目录下查找扩展动态库。以本书安装情况为例，该项配置如下所示。表示从 PHP 安装目录下的 ext 目录里查找扩展动态库。

```
extension_dir = d:\php\ext
```

1.9.5 PHP 文件上传功能的配置

关于 PHP 文件上传功能的配置，主要有以下所示的两项。

upload_tmp_dir：当进行文件上传时，临时文件存放的目录，对于 Linux/UNIX 用户来说，要特别注意，当前 PHP 用户必须拥有这个目录的写权限。

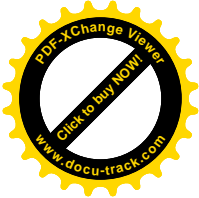
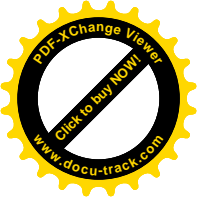
upload_max_filesize：这个配置项指定了允许上传文件大小的最大值，php.ini 默认值是 2M。

1.9.6 PHP 中 session 的配置

在 php.ini 中，有关 session 的基本配置有以下所示的几项。

session.save_handler：这项配置用来设置 session 的存储方式，一般使用默认值 files 即可，代表用文件储存。

session.save_path：这项用来设置 session 的保存路径，以本书为例，将 session 保存在



PHP 安装目录的 session 目录下，如下所示。

```
session.save_path = d:\php\session
```

session.use_cookies：sessionid 的传递方式，默认是 cookie，推荐使用。

1.9.7 PHP 中和电子邮件有关的配置

在这里 PHP 的邮件配置目前只需了解一个 sendmail_path 项即可，其他的配置项将在以后有更多讲述。

sendmail_path 这项仅针对 Linux/UNIX 用户来说，它用来指定 sendmail 程序的目录位置，通常是 /usr/sbin/sendmail 或 /usr/lib/sendmail。

1.9.8 PHP 基本的安全设置

这里的安全设置主要是指 PHP 安全模式方面的内容。基本的配置有以下几项。

safe_mode：是否允许 PHP 的安全模式，默认情况此项配置的值为 Off，即关闭安全模式。

safe_mode_exec_dir：该项表示安全模式下，系统可执行系统程序的目录。这个配置项取决于上一项，如果 PHP 运行于安全模式下，一些系统函数将会拒绝执行不在该目录下的系统程序。

1.9.9 MySQL 数据库系统的启动与关闭

对于 Windows 用户，和 Apache 类似，可以通过 Windows 服务来关闭和启动 MySQL。进入 Windows 控制面板里的管理工具，打开服务，找到 MySQL 一项，通过“操作”菜单项或图标按钮即可完成 MySQL 的启动与停止，

1.9.10 MySQL 对数据的存储

首先了解一下 MySQL 的目录结构。Windows 平台下，

MySQL 会把所有的数据以文件的形式存储在 MySQL 安装目录的 data 目录下。开发人员可以通过 MySQL 创建许多数据库，每创建一个数据库，就会在 data 目录下生成一个子目录，这个子目录的名字就是数据库的名字，在这些子目录下存放的就是真正的数据文件。关于 MySQL 更多的概念与内容，将在第 12 章详细讲述。

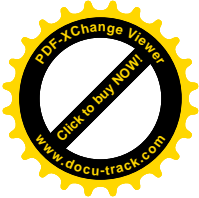
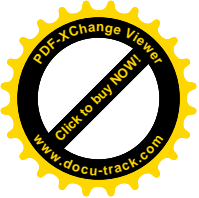
1.10 实例：使用 JavaScript 验证 HTML 表单数据、

本章介绍的重点内容是 HTML、CSS、JavaScript 和 PHP 环境的搭建。其中 HTML、CSS 已经 JavaScript 的相关内容又是 Web 开发的最基础知识，掌握这方面的内容对 Web 开发至关重要。因为在 HTML、CSS 及 JavaScript 三者之间，JavaScript 是比较难掌握的，所以，本节以一个 JavaScript 的应用为实例，作为对 Web 开发基础知识的一个总结。

这个 JavaScript 实例是用来验证 HTML 表单数据的简单程序。任何一个 Web 应用，都离不开数据的提交和处理，这些数据要么被应用程序直接使用，要么被应用程序存入数据库。无论哪种情况，都应该保证浏览器端所提交数据的有效性和正确性。这就要求服务器端程序，在处理数据之前，先对 HTML 表单所提交的数据进行合法性验证，以保证应用程序执行正常，或者保证存入数据库的数据完整有效。

1.11 小结

本章较为详细地介绍了 Web 编程的基础知识和 PHP 开发环境的配置。Web 编程基础部分详细讲述了 Web 基本体系结构、HTML 语言、CSS 和 JavaScript 基础。PHP 开发环境的配置讲述了 Apache、MySQL 和 PHP 的安装及基本配置，其中重点阐述了如何修改 httpd.conf、php.ini 等核心配置文件，及其之间的相互关系。最后通过一个实例验证了一个 PHP 的开发过程，读者也可以通过此例，验证机器的配置是否已经正常使用。



第 2 章 PHP 语言

经过前面预备知识的学习，从本章开始，正式进入 PHP 语言的学习。PHP 作为一种专门用来开发 Web 应用的嵌入式语言，大量借用了 C、C++ 和 Perl 语言的语法，同时加入了一些其它语法特征，使编写 Web 程序更快更有效。之所以说 PHP 是嵌入式语言，是因为用 PHP 开发的 Web 程序，大多都要在 HTML 文档中插入 PHP 代码，或者使用 PHP 代码生成某些 HTML 文档，以满足 Web 应用的需求和特点。

PHP 一般作为 HTTP 服务器（通常是 Apache）的一个模块运行。这意味着，当用户访问到一个含有 PHP 代码的 Web 页面时，HTTP 服务器就会调用这个模块，通过这个模块来分析并执行该页面的 PHP 代码，最终将执行结果返回给用户。PHP 支持多种数据库，如 MySQL、dBase、MS_SQLServer、Oracle 等。这对于基于数据库的 Web 开发来说是大有裨益的。

PHP 从上世纪 90 年代中期问世以来，已经推出了很多版本，到现在已经是 PHP5。本书所讲述的 PHP 语言，将以 PHP5 为准。本书凡是出现“PHP”的地方，除非特别说明，都将指的是 PHP5。

2.1 基本语法

PHP 的语法和 C、C++ 等语言的语法很相似，有 C 语言基础的读者，可以非常轻松地掌握 PHP 的基本语法。即便是没有任何语言基础，也是值得庆幸的，那样不会受其它语言的干扰，可以更快速地接受 PHP 的语法。

事实上，PHP 的语法并不复杂，再加上 PHP 提供了大量的预定义函数，使 PHP 开发事半功倍。只要按本书的讲述、一步步地学习下去，再加上自己的一点信心，相信读者会发现 PHP 很容易学习掌握，并且应用起来也很快速方便。本节，先简单了解一些 PHP 的基本语法。

2.1.1 PHP 分隔符

因为 PHP 是嵌入式脚本语言，需要使用某种分隔符将 PHP 代码和 HTML 的内容区分开来，这里所说的分隔符就是“<?php”和“?>”，它们将 PHP 代码包含在其中，也就是说，所有的 PHP 代码都应该写在“<?php”和“?>”之间。如下代码所示：

```
<p>一个段落</p> //这一行是 HTML，PHP 分析器将会忽略这  
行代码，不做处理  
<?php echo “这段内容由 PHP 代码输出”；?> //这一行是 PHP 代码，PHP 分析器将会执  
行这段代码  
<p>另外一个段落</p>
```

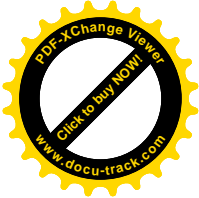
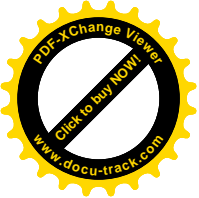
2.1.2 给 PHP 程序添加注释

程序中的注释是指在一个程序文件中，对一个代码块或一条程序语句所作的文字说明，注释是提供给开发人员看的，因此，程序中的注释会被计算机忽略而不会被执行。PHP 中的主要注释风格有：

使用符号“//”添加一个单行的注释。

使用符号“#”添加一个单行的注释。

使用“/*”和“*/”添加一个多行的注释，也可以用来单行注释。



PHP 程序的注释

2.2 变量

变量是任何程序设计语言中一个基本而且重要的概念。本节的内容将讲述 PHP 变量基本概念、变量的类型、PHP 预定义变量以及如何使用 PHP 内置函数对变量进行一些处理。

2.2.1 什么是变量

在程序中可以改变的数据量叫做变量,变量必须有一个名字,用来代表和存放变量的值。PHP 中使用美元符 (\$) 后跟变量名来表示一个变量,如 \$var 就是一个变量。PHP 中的变量名是区分大小写的,因此 \$var 和 \$Var 表示的是不同的两个变量。

PHP 变量的命名需要遵守一定的规则,这个规则是:一个有效的变量名必须由英文字母或下划线开头,后面可以跟任意数量的英文字母、数字、下划线或其组合。如 \$abc、\$_ab_c、\$a1b_c2 都是合法的变量名,而 \$3xyz 就不是合法变量名,因为它以数字开头。

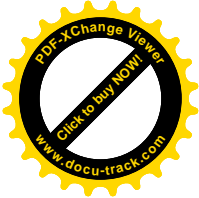
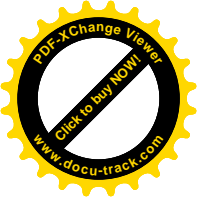
2.2.2 变量的数据类型

PHP 属于弱类型语言。这就是说,变量的数据类型一般不用开发人员指定,PHP 会在程序执行过程中,根据程序上下文环境决定变量的数据类型。如一串数字“789”,在用 echo 语句输出时,它作为字符串处理,但是做数学运算时,它就作为整数处理。PHP 的变量主要有以下类型:

- 整数类型。
- 浮点类型。
- 字符串类型。
- 布尔类型。
- 数组类型。
- 对象。

2.2.3 变量类型的转换

和 C、C++ 等语言不同,PHP 在定义变量时,不需要明确指定变量的类型。也就是说,把一个整数指定给变量 \$v,那么 \$v 就是一个整型变量,如果把一个字符串指定给它,那么



它就是一个字符串变量。若要转换类型，在 PHP 程序中也是很自由的，一般不必经过特殊的转换。

当然，PHP 中也可以对变量做强制转换，这点和 C 语言相似，在要转换的变量之前加上目标类型，目标类型用括号括起来。

PHP 中允许的强制类型转换有：

(int)，(integer)——转换成整型。

(bool)，(boolean)——转换成 bool 型。

(float)，(double)，(real)——转换成浮点型。

(string)——转换成字符串。

(array)——转换成数组。

(object)——转换成对象。

2.2.4 可变变量

可变变量是 PHP 中比较特别的一个概念，可变变量是指这样一个变量，它将某个变量的值作为自己的变量名。



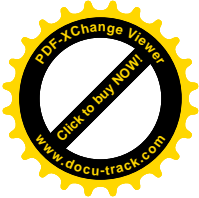
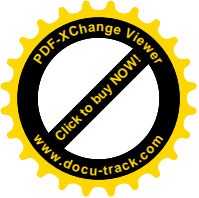
可变变量

2.2.5 PHP 的预定义变量

PHP 提供了大量的预定义变量，可以在程序或文件的任何地方使用它们。这些变量大多数依赖于服务器的版本及其配置。本书主要讲述 PHP5 的预定义变量，这些预定义变量和以前版本的有些不同，有兴趣的读者可自行了解。下面是 PHP 中一些常用的预定义变量。

`$GLOBALS`：包含指向当前程序中全局范围内有效的变量，它是一个数组，该数组的索引（或键名）就是全局变量的名称。

`$SERVER`：该全局变量是一个包含诸如头信息、路径和脚本位置的数组。常见的 `$SERVER` 的元素包括 `PHP_SELF`（当前正在执行的脚本的文件名）、`SERVER_ADD`（当前执行脚本所在服务器的 IP 地址）、`SERVER_NAME`（当前执行脚本所在服务器主机的名称）、`DOCUMENT_ROOT`（当前脚本所在文档的根目录）、`SCRIPT_FILENAME`（当前执行脚本的绝对路径）。



SCRIPT_NAME (当前脚本的路径) HTTP_REFERER (链接到当前页面的前一页面的 URL) REQUEST_URI (访问此页面所需的 URI) 等。

- \$_GET : 通过 HTTP 的 GET 方法提交至脚本的表单变量。
- \$_POST : 通过 HTTP 的 POST 方法提交至脚本的表单变量。
- \$_FILE : 通过 HTTP 的 POST 文件上传提交至脚本的变量。
- \$_COOKIE : 通过 HTTP 的 Cookies 方法提交至脚本的变量。

2.2.6 判断变量的类型

从本小节开始, 将介绍一些处理变量的方法。因为在 PHP 中通常通过一些预定义函数来处理变量, 所以, 需要读者对函数的概念有个大概了解。

简单地说, 函数是指完成某种特定功能的代码块, 可以向函数传入参数, 函数对参数进行处理, 并且将处理结果返回给用户。本书将在后面详细介绍函数的概念。

在 PHP 中, 可以通过以下函数对变量的类型做判断。

- 函数 is_integer 判断变量是否为整数。
- 函数 is_string 判断变量是否为字符串。
- 函数 is_double 判断变量是否为浮点数。
- 函数 is_array 判断一个变量是否为数组。

2.2.7 获取变量的类型

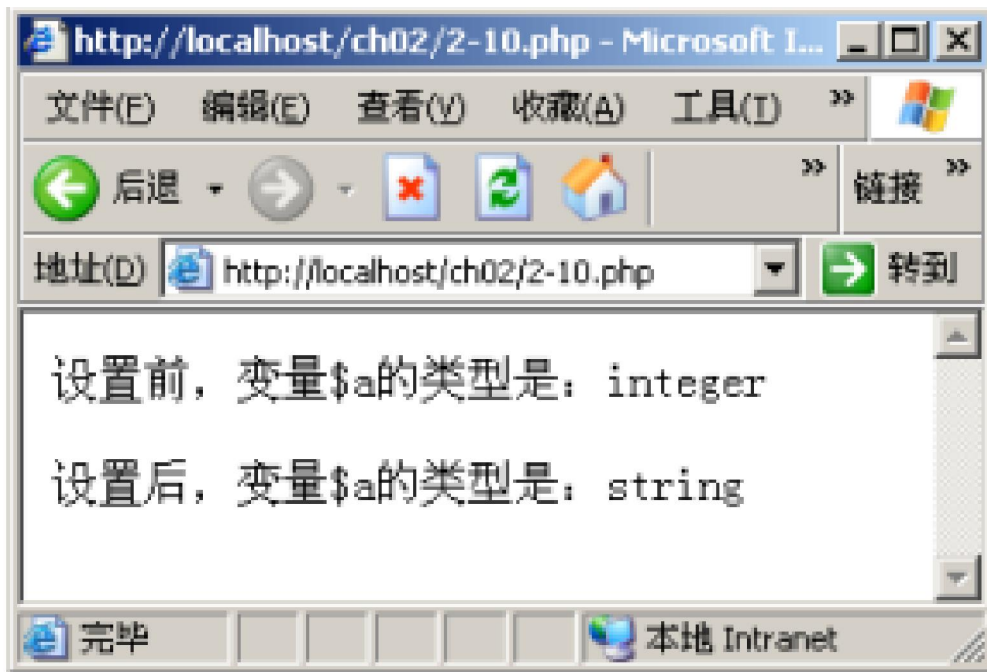
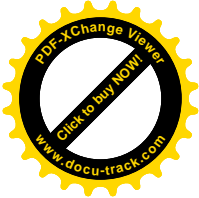
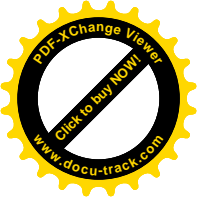
在 PHP 中, 可以使用预定义函数 gettype 取得一个变量的类型, 它接受一个变量作为参数, 返回这个变量的类型。



获取变量的类型

2.2.8 设置变量的类型

使用预定义函数 settype 设置一个变量的类型, 该函数接受两个参数, 第一个参数是变量名, 第二个参数是要设置的变量的数据类型。



设置变量的类型

2.2.9 判断一个变量是否已经定义

使用预定义函数 `isset` 判断一个变量是否已经定义，它接受一个变量作为参数值，返回值如果为 `TRUE`，说明该变量定义过，否则，说明该变量没有被定义。

2.2.10 删除一个变量

使用 `unset` 语句删除一个变量。从 PHP4 开始 `unset` 不再有返回值，因此，严格意义上讲，它并不是一个函数，而是一个 PHP 的语言结构。可以用 `unset` 一次删除多个 PHP 变量。

2.3 常量

和变量相对应的概念是常量。上节介绍了变量、变量的数据类型及对变量的一些操作、处理，这节将介绍常量的概念及使用。

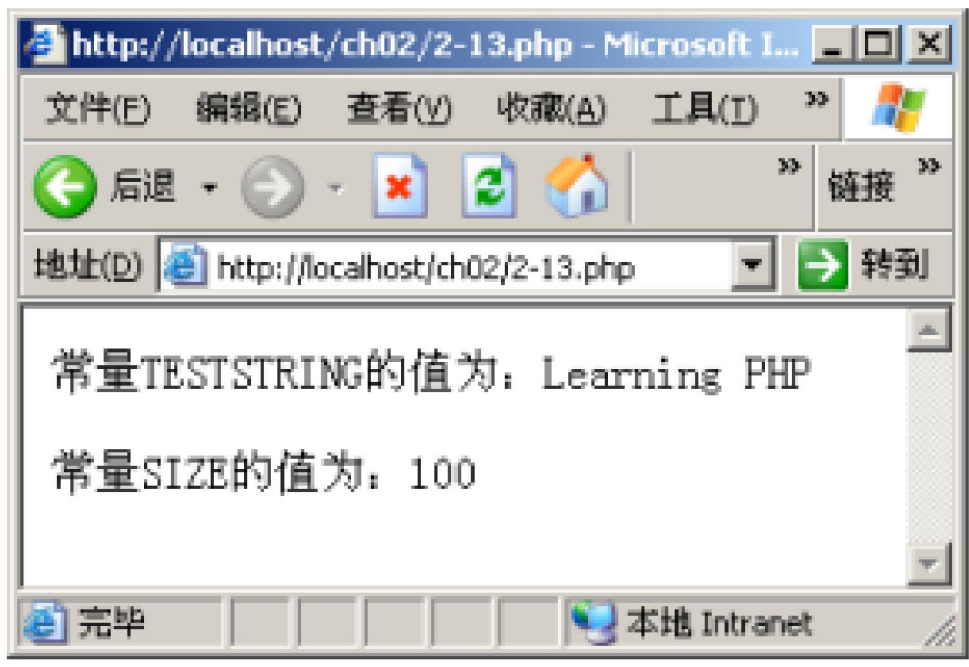
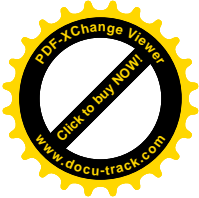
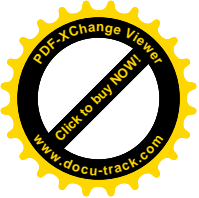
2.3.1 什么是常量

在程序执行过程中，其值不能改变的量叫做常量。这就是说，常量不能再被定义成其它的值。常量也可以分为不同的类型，如 `10`、`0`、`-12` 是整型常量，`1.23`、`-0.45` 是浮点型常量，常量的类型从形式上就可以判别。

PHP 中有一些定义好的常量，在程序中可以直接使用。开发人员也可以根据程序的需要，自己定义新的常量。

2.3.2 定义常量

在 PHP 中通过 `define()` 函数定义一个常量。合法的常量名只能以字母和下划线开始，后面可以跟着任意字母、数字或下划线。常量一旦定义就不能再修改或者取消定义。



常量的定义和结果输出

2.4 表达式

表达式是指程序中任何有值的部分，PHP 中几乎所有内容都是表达式。如 \$a=9 就是一个表达式，这个表达式的含义是：将 9 指定给变量 \$a（即赋值操作，将在下一节介绍）。很明显，“9”的值就是 9，因此“9”本身就是一个表达式。也就是说，“9”是一个值为 9 的表达式，只不过在这里，9 是一个整型常量。同理，变量 \$a 也是一个值为 9 的表达式。从这个例子可以看出两个值：整数常量“9”和被指定值为 9 的变量 \$a，但事实上，还有一个值，就是这个 \$a=9 本身的价值，表达式 \$a=9 的值就是被指定的值——9。

另外一类很常见的表达式就是比较表达式，如 \$a>\$b 等。这些表达式的值要么是 0（表示 FALSE），要么是 1（表示 TRUE），如果表达式成立，则表达式的值为 1，否则，表达式的值为 0。

2.5 运算符

运算符是指，通过一个或多个表达式，来产生另外一个值的某些符号，如“+”、“%”、“.”等都是运算符。在表达式 2+1 中，运算符“+”有两个操作数 1 和 2。具有两个操作数的运算符叫做双目运算符。具有一个操作数的运算符叫做单目运算符，如表达式 -6，运算符“-”只有一个操作数 6，因此，这里的“-”是单目运算符。

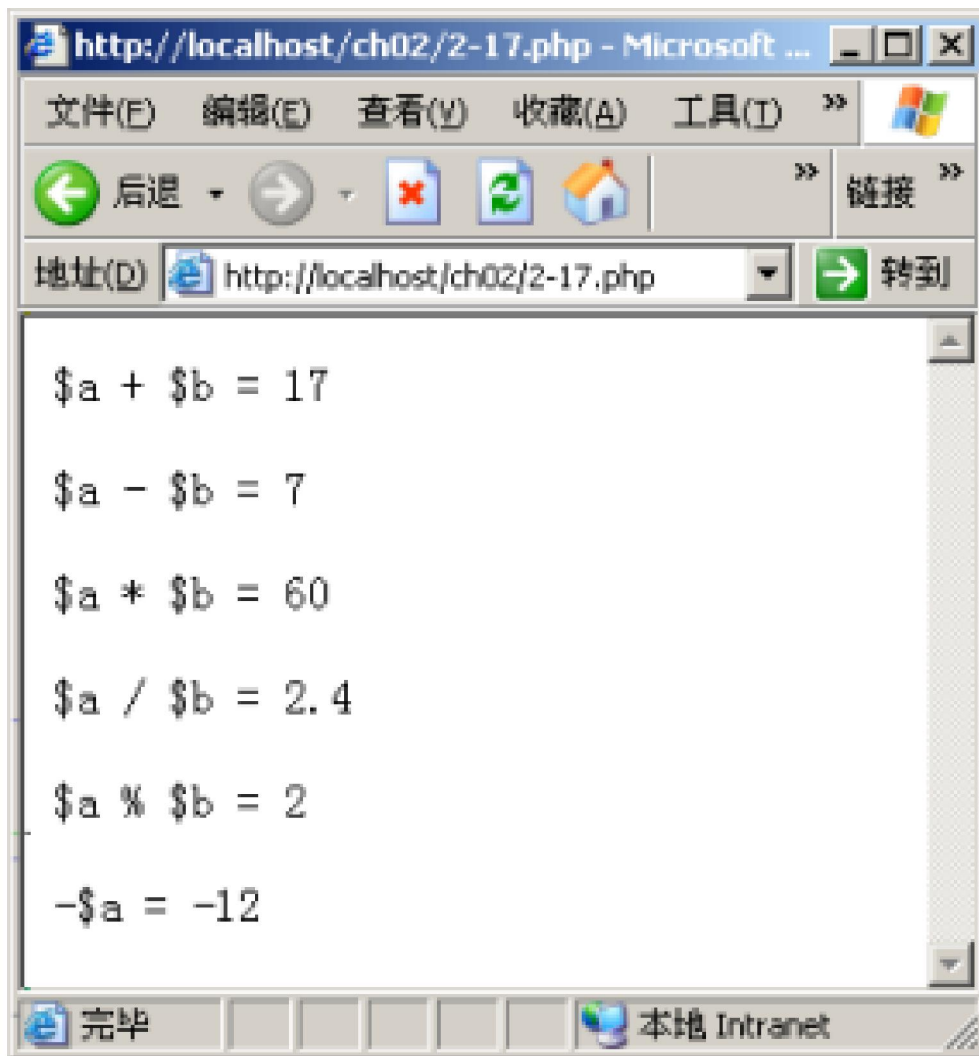
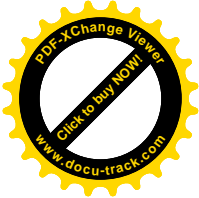
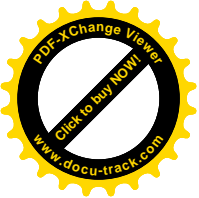
运算符有优先级，即表达式中多种运算符同时出现时，哪种运算符应该首先被应用。本节将介绍基本的运算符，最后介绍基本运算符的优先级。

2.5.1 赋值运算符

在 PHP 中，符号“=”不表示相等，而表示赋值。它的含义是将一个值指定给一个变量，更专业一点地说，它表示把“=”右边的表达式的值赋给左边的变量，如“\$a=5”表示将 5 赋给 \$a。赋值表达式的值也就是所赋的值，这就是说，“\$a=5”的值是 5。除了上述最基本的赋值方式外，还有一种所谓的“组合赋值”，通过例子来说明它的含义和用法。

2.5.2 算术运算符

PHP 的算术运算符有加（+）、减（-）、乘（*）、除（/）和取模（%）、取反（-，即取负值）。这些运算符的用法和学校里学到的数学知识一样，



算术运算符的使用

2.5.3 递增/递减运算符

PHP 有和 C 语言风格相同的递增/递减运算符。递增是指对当前表达式的值增加 1，递减正相反，对表达式的值减 1。本书仅讲述整数表达式的递增/递减运算，下面分别介绍四种风格的递增/递减运算。

`$a++`：先返回 `$a` 的值，然后将 `$a` 的值加 1。

`++$a`：先将 `$a` 的值加 1，然后将 `$a` 返回。

`$a--`：先返回 `$a` 的值，然后将 `$a` 的值减 1。

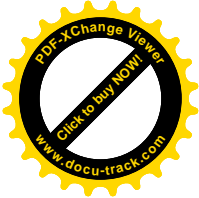
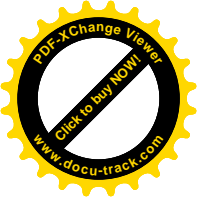
`--$a`：先将 `$a` 的值减 1，然后返回 `$a` 的值。

2.5.4 字符串运算符

字符串运算符只有一个，即字符串的连接运算符“`.`”。这个运算符将两个字符串连接成一个新的字符串。在 2.2.2 小节介绍字符串变量时，简单提到过这个运算符。

其实在此之前，已经在很多示例程序中使用过这个运算符。比如程序要显示一个执行结果，会用“`.`”将一些内容连接起来，然后输出。

2.5.5 逻辑运算符



运算符	名称	举例
!	逻辑非	!\$a
&& (或and)	逻辑与	\$a && \$b
(或or)	逻辑或	\$a \$b
xor	逻辑异或	\$a xor \$b
结果		
如果\$a为FALSE, 则!\$a为TRUE。反之, !\$a为FALSE。		
如果\$a和\$b都为TRUE, 则\$a && \$b为TRUE。		
如果\$a或\$b任意一个为FALSE, 则\$a && \$b为FALSE。		
如果\$a或\$b任意一个为TRUE, 则\$a \$b为TRUE。		
如果\$a和\$b都为FALSE, 则\$a \$b为FALSE		
如果\$a或\$b任意一个为TRUE, 但不同时为TRUE, 则\$a xor \$b为TRUE。		

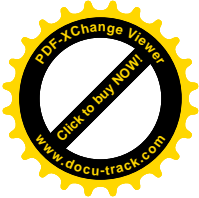
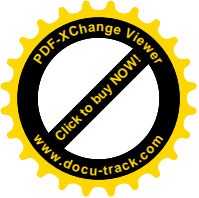
2.5.6 比较运算符

比较运算符用来对两个值进行比较。列举了主要的比较运算符及其可能的运算结果。

运算符及名称	举例
= (等于)	\$a == \$b
!= (不等于)	\$a != \$b
=== (全等于)	\$a === \$b
> (大于)	\$a > \$b
>= (大于等于)	\$a >= \$b
< (小于)	\$a < \$b
<= (小于等于)	\$a <= \$b
结果	
如果 \$a 等于 \$b, 则\$a == \$b的值为TRUE。	
如果 \$a 不等于 \$b, 则\$a != \$b的值为TRUE。	
如果 \$a 等于 \$b, 并且它们的类型也相同, 则\$a === \$b的值为TRUE。	
如果 \$a 大于 \$b, 则\$a > \$b的值为TRUE。	
如果 \$a 大于或者等于 \$b, 则\$a >= \$b的值为TRUE。	
如果 \$a 小于 \$b, 则\$a < \$b的值为TRUE。	
如果 \$a 小于或者等于 \$b, 则\$a <= \$b的值为TRUE。	

2.5.7 运算符的优先级

事实上, 在小学的数学知识中, 就已经学习过运算符的优先级。比如 $1+2 \times 3$ 的结果是 7, 不是 9。因为 \times 号的优先级高于 $+$ 号的优先级。只不过在 PHP 中, 运算符不仅限于加减乘



除。下面列举一些常见的 PHP 运算符的优先级，最上面的优先级最高。

new (new 运算符，将在后面讲述)

++、-- (递增、递减运算符)

*、/、%

+、-、.

&&

||

?: (条件运算符，将在后面讲述)

= (赋值运算符，包含+=、*=、.=等)

and

xor

or

2.6 PHP 程序的流程控制

所有的 PHP 程序都由语句构成，程序就是一系列语句的序列。计算机通过执行这些语句可以完成特定的功能。一般情况下，程序都是从第一条语句开始执行，按顺序执行到最后一句。但有时因为某种情况，需要改变程序的执行顺序，这就需要对程序的流程进行控制。本节将讲述 PHP 程序的各种流程控制结构。

2.6.1 程序流程控制概述

计算机程序的执行方式有 3 种：顺序执行、选择执行、循环执行，通过使用这 3 种控制结构，可以改变程序的执行顺序，以满足开发人员解决问题的需求。

顺序结构使程序从第一条语句开始，按顺序执行到最后一句。在选择结构中，程序可以根据某个条件是否成立，选择执行不同的语句。在循环结构中，可以使程序根据某种条件和指定的次数，使某些语句执行多次。

PHP 程序都是由一系列语句组成，语句通常以分号结尾。此外，可以使用一对花括号“{”和“}”将一组语句组成一个语句组。例如：

```
{  
$i = 123;  
$s = " This is a string ";  
}
```

2.6.2 条件控制语句：if 和 if...else 语句

PHP 通过一系列条件控制语句完成程序的选择执行流程。PHP 中使用 if、if...else(elseif) 语句构建选择程序结构。

1 . if 语句：f 条件语句的结构如下所示。

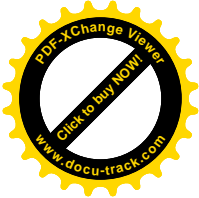
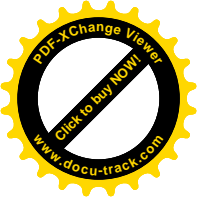
```
if(expr)  
    statement
```

2 . if...else 语句：f...else 语句的结构如下所示。

```
if(expr)  
    statement1  
else  
    statement2
```

3 . if...elseif 语句：f...elseif 语句的结构如下所示。

```
if(expr1)  
    statement1  
elseif(expr2)
```



```
statement2  
...  
else  
statement
```

2.6.3 switch 结构

上一小节讲述了使用 if 或者 if...else 语句的选择控制结构，if 语句通常需要计算逻辑表达式的值。这小节将介绍另一种选择控制结构——switch 结构，它不需要计算逻辑表达式的值。

注意：在 PHP 中，switch、case、break 和 default 都是保留关键字。

switch 结构首先计算表达式 expr 的值，如果 expr 的值与某个 case 的值匹配，则从 case 后面的语句开始执行，直到遇到 break 语句（该语句将在后面详细介绍）或整个 switch 结构结束。比如，如果 expr 的值是 value2，那么语句 statement2 将会被执行。

2.6.4 循环控制语句：for 循环语句

前两小节讲述的是程序的选择控制流程，这节开始讲述程序的循环控制流程。循环控制流程可以控制程序，在满足某些条件的时候，某些语句被循环执行多次。

PHP 的循环控制主要有：for 语句、while 语句和 do...while 语句。本节先介绍 for 循环语句。

for 循环语句的结构如下：

```
for(初始化语句; 循环条件表达式; 更新语句)  
statement
```

2.6.5 while 循环语句

除了 for 循环语句之外，还可以使用 while 语句控制程序循环执行。while 循环语句的结构如下：

```
while(expr)  
statement
```

这里的语句 statement 可以是单条语句，也可以是语句组。该结构的执行流程是：当表达式 expr 的值为真时，就执行循环体——语句 statement，然后再次计算表达式 expr 的值，直到 expr 的值为假，程序中断循环，跳出 while 循环结构。

2.6.6 do...while 循环语句

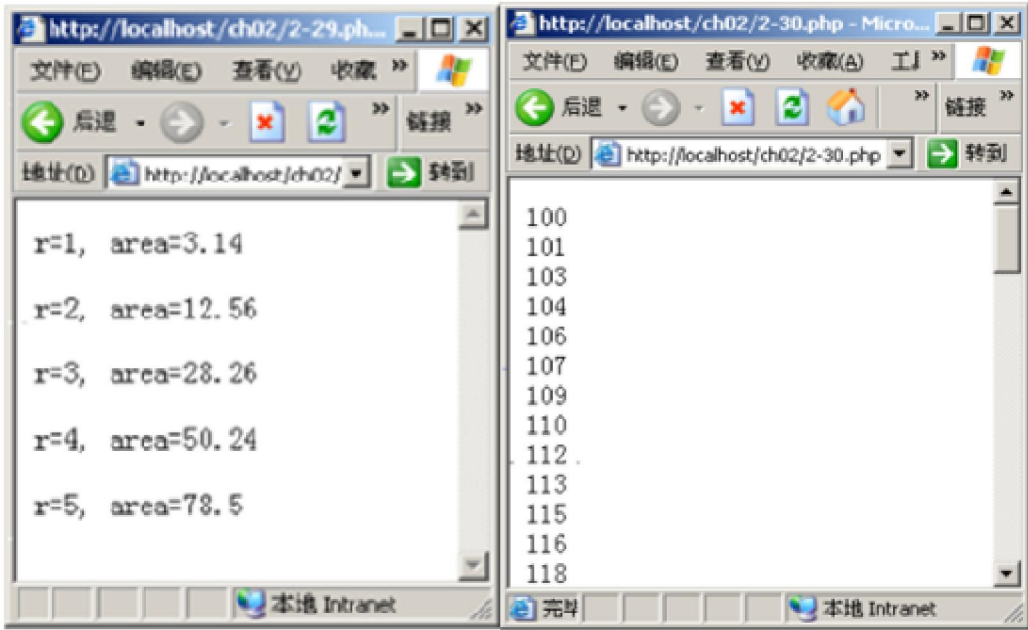
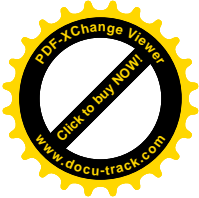
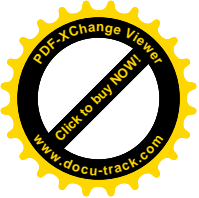
这小节讲述第三种循环控制结构：do...while 语句，它也可以控制程序循环执行。do...while 循环语句的结构如下：

```
do  
statement  
while(expr);
```

这里的语句 statement 可以是单条语句，也可以是语句组。该结构的执行流程是：程序首先执行语句 statement，然后再计算表达式 expr 的值。如果表达式 expr 的值为 TRUE，就再次执行语句 statement。

2.6.7 break 和 continue 语句

在 2.6.3 小节中介绍 switch 语句时，提及过 break 语句，它可以使程序流程跳出 switch 结构。除此之外，break 语句还可以在 for、while 和 do...while 语句中使用，这样可以使程序立即跳出该循环结构。



break 语句的使用

continue 语句的使用

2.6.8 条件运算符：?:

条件运算符 (?:) 的用法是：

```
expr1 ? expr2 : expr2
```

可以看出，条件运算符有 3 个操作数，所以它是三目运算符。它的计算规则是：如果表达式 expr1 的值为 TRUE，那么整个表达式的值就取 expr2 的值，否则，就取 expr3 的值。下面的代码使用条件运算符来获取两个数中最大的那个数。

```
$max = ($a>=$b) ? $a : $b
```

2.7 函数

在很多编程语言中都有函数这个概念。函数将为解决某一问题而编写的代码组织在一起，使得在解决同一个问题时，可以重复这些代码。本节将介绍 PHP 中函数的概念、构建和调用函数等内容。

2.7.1 PHP 中函数的概念

在数学知识里，函数是由参数的定义域和在这个参数定义域上的某种规则组成的。当选定某一参数时，函数的值也是惟一确定的。例如，有这样一个数学函数： $f(x)=2x+3$ ，那么就有 $f(1)=5$ ， $f(3)=9$ 。这里的 1、3 都是函数 f 的参数，而 5、9 都是这些参数对应的函数 f 的值。

PHP 语言中的函数和数学中函数的概念很相似，只不过 PHP 中的函数不仅仅是做一些数学运算，而是要完成更多、更复杂的功能。

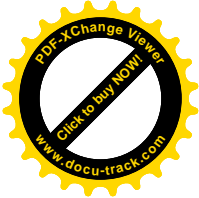
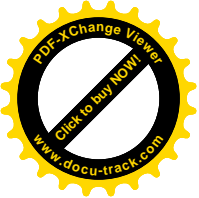
在程序设计中，经常将一些常用的功能模块编写成函数，放在公用函数库中，供程序或其它文件使用。函数就像一些小程序，用它们可以组成更大的程序。函数之间也可以相互调用，完成更复杂的功能，但它们之间是相互独立的，互不隶属。

2.7.2 定义函数和调用函数

PHP 使用下面的语法定义一个函数：

```
function func_name(param_list)
{statement}
```

其中 function 是 PHP 的保留关键字，表示开始定义一个函数。func_name 是函数名，由开发人员自行指定，函数名以字母或下划线开始，后跟任意字母、数字或下划线。函数名后



的一对括号，用来存放函数的参数 param_list，如果所定义的函数不需要传入参数，括号内留空，但不能没有括号。最后花括号括住的语句 statement 叫做函数体，它可以是单条语句，也可以是多条语句，这些语句完成函数所要实现的功能。下面的代码演示了如何定义了一个函数。

2.7.3 函数的参数和函数的返回值

上小节的函数 say_hello() 只能向“Jack”显示问候语，如果想向更多的人显示问候语，该怎么做呢？这就是函数参数的问题。在大多数情况下，调用函数都会有数据的传递，这就是前面讲到的函数参数。将参数传递给函数，函数根据不同的参数会完成不同的功能，或有不同的输出。

在函数 say_hello() 的例子中，如果想要向更多的人显示问候语，可以传递一个参数给函数 say_hello()，这个参数就是不同的人名，在函数体内输出这个变量即可。

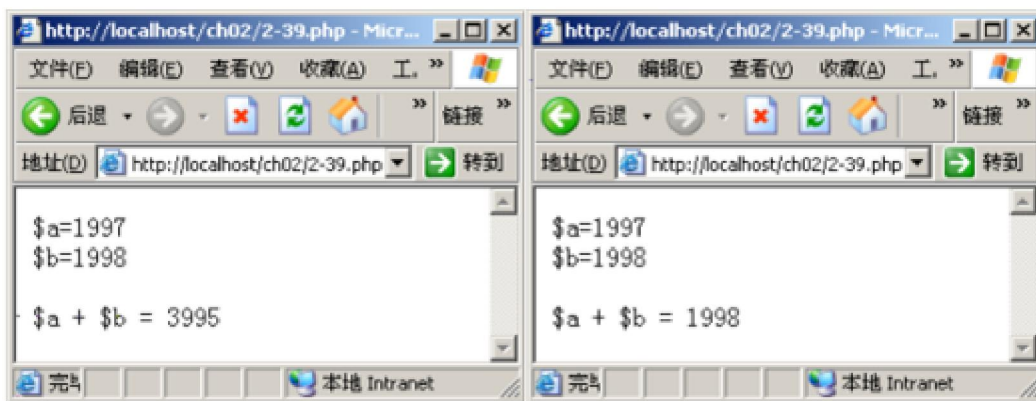
2.7.4 PHP 函数的传值与传址

上小节讲述的向函数传入参数，是按传值方式传入的。传值的含义是指，在函数体内，会生成一个传入值的拷贝，在函数内部对参数的修改，不会影响到传入的值。有时，因为开发的需要，希望在函数内部能够修改传入的值，这就需要函数参数使用传址方式。传址的含义是指，在函数体内，真实引用传入的值，这意味着，在函数体内使用的函数参数，和传入的值完全是同一个，而不单单是传入值的一个拷贝。这时，在函数内部修改了参数的值，同时也就修改了传入的值。

在 PHP 中，要想在函数参数中传址，需要在定义函数时，在参数前加上符号：&。

2.7.5 函数和变量作用域

变量的作用域就是变量的有效范围。对于大多数 PHP 变量，作用域只能有一个。但是，在用户自定义函数中，存在一个单独的局部函数范围。在一个函数内部定义的变量是局部变量，它只在本函数内有效，它的作用域就是当前的函数之内。这就是说，一个在函数外部定义的变量，不会在函数内部起作用，反之亦然。



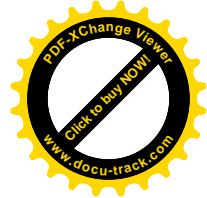
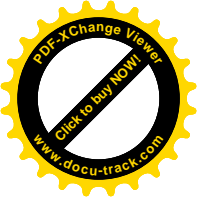
在程序中使用 global 关键字

在程序中未使用 global 关键字

2.8 小结

本章主要介绍了 PHP 的基本语法，包括变量类型、常量、表达式、运算符、程序控制流程和函数的基本概念。其中变量、表达式、程序控制流程和函数是本章的重点知识，读者应该熟练掌握这些内容。

流程控制是程序设计中非常关键的地方，虽然这些控制语句看起来很简单，但使用过程中，如果因为逻辑错误，将可能导致死循环发生。



第 3 章 用 PHP 进行 Web 编程

PHP 是开发 Web 应用的首选语言之一，也是最佳选择。PHP 本身就是为 Web 而生的。它提供了一系列可以使 Web 开发更加方便、更加容易的功能和特性。

本章先介绍用 PHP 进行 Web 编程的一些基本用法，了解用 PHP 进行 Web 编程的一些特性，接着通过一个简单的完整实例实践这些用法和特性，加深对使用 PHP 进行 Web 编程的理解和掌握。

3.1 PHP 的 Web 编程基础

本节将讲述最基本的 PHP Web 编程知识，诸如获取表单数据、处理表单数据、PHP 中的 Session 和上传文件等。

3.1.1 访问和获取 HTML 表单数据

在 PHP 中，可以通过两个预定义变量，很方便地获取 HTML 表单数据。这两个预定义变量在前面提及过：`$_GET` 和 `$_POST`。它们都是 PHP 的自动全局变量，可以直接在 PHP 程序中使用。

变量 `$_GET` 是表单数据组成的数组，它由 HTTP 的 GET 方法传递的表单数据组成。表单元素的名称就是数组的“索引”。这就是说，通过表单元素的名称（即 `name` 属性的值），就可以获得该表单元素的值。例如某表单中，有一个文本输入框，名称为“`user_name`”，那么在 PHP 程序中，就可以通过 `$_GET['user_name']` 获取文本框中用户输入的值。

变量 `$_POST` 的用法和 `$_GET` 类似。通过 HTTP 的 POST 方法获取的表单数据，都将存放在该变量中，该变量也是一个数组。

3.1.2 用 PHP 处理表单数据

在上小节的文档 3-1.html 中，对于表单中的“爱好”多选框，只选择了“阅读”一项。如果做了多个选择，再提交表单，.php 输出的结果就有所不同。

所选择的 3 个 checkbox，只有最后 1 个的值被输出了，其他两个选项的值没有被输出，这并不是所期望的结果。之所以出现这种情况，是因为多选按钮元素 checkbox 的名称都为“`hobby`”，而 PHP 要求，如果表单元素同名，就必须以数组方式命名，并为其 `value` 属性赋值，这样 PHP 才能正确取值。

因此，首先修改 3-1.html 的中表单元素 checkbox 的名称，以数组方式命名 checkbox 元素，即在原来的名称“`hobby`”后加上“`[]`”，

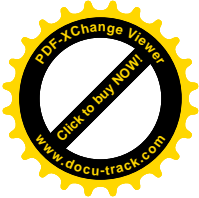
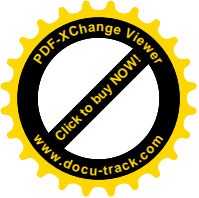
3.1.3 用 PHP 验证表单数据有效性

在上小节的文档 3-1.html 中，对于表单中的“爱好”多选框，只选择了“阅读”一项。如果做了多个选择，再提交表单，3-2.php 输出的结果就有所不同。提交表单后，所选择的 3 个 checkbox，只有最后 1 个的值被输出了，其他两个选项的值没有被输出，这并不是所期望的结果。之所以出现这种情况，是因为多选按钮元素 checkbox 的名称都为“`hobby`”，而 PHP 要求，如果表单元素同名，就必须以数组方式命名，并为其 `value` 属性赋值，这样 PHP 才能正确取值。

3.1.4 PHP 中的 session

session 是 Web 开发中最常见的概念，也是最常用的功能之一。简单地说，是 session 是指用户进入网站到浏览器关闭的这段时间（或过程）。

HTTP 是面向无连接（或无状态）的协议。这意味着，在 HTTP 中，一个完整的请求/响应过程结束之后，客户端（即浏览器）和服务端端的链接就已中断。此时，如果用户再从当前页面访问其他页面，即向服务器发出请求，服务器端并不知道此请求是哪个用户发起的，因此也就无法得知用户的浏览状态。这样就遇到一个问题：当前页面中的某个数据（或变量），无法在接下来访问的页面中使用。而在实际的 Web 开发中，经常要在页面之间传递数据，



而且不同的访问用户，传递的数据是不同的。虽然解决这个问题的办法有很多，但通过 session 解决这个问题，会更加方便、快速、有效。通过 session 记录用户的有关信息，以供用户以此身份向服务器发起请求时，服务器能够根据 session 做出正确的判断，区分不同用户的请求。

3.1.5 PHP 中的文件上传处理

在 Web 开发中，经常会遇到从客户端上传文件到服务器端的问题。通常，文件上传使用的是 HTTP 的 POST 方式，使用 POST 方式传递文件到服务器端。要完成文件上传处理，首先要定义 HTML 表单的 enctype 属性为 "multipart/form-data"，如下代码所示。

```
<form enctype=" multipart/form-data " action=" somefile.php " method=" POST " >
```

在 PHP 程序中，使用全局变量 \$_FILES 处理文件上传。\$_FILES 是一个数组，包含了要上传的文件的信息。下面，以上述 HTML 表单为例，介绍 \$_FILES 数组的内容。

\$_FILES[' myfile '][' name ']表示客户端文件的原始名称，即要上传的文件的文件名。其中 myfile 就是在代码 3-6 中定义的 input 元素的 name 属性的值：<input name="myfile" type="file" />。

\$_FILES[' myfile '][' type ']表示上传文件的类型，例如 " image/gif "。

\$_FILES[' myfile '][' size ']表示已上传文件的大小，单位为字节。

\$_FILES[' myfile '][' tmp_name ']表示文件上传后，在服务器端存储的临时文件名。

\$_FILES[' myfile '][' error ']表示和文件上传的相关错误信息。

3.2 实例：用 PHP 开发一个简单的网站

通过上一小节对 PHP 开发 Web 应用基础知识的学习，读者掌握了使用 PHP 获取和处理表单数据、用 PHP 验证数据的有效完整、PHP 中 session 的使用和处理文件上传等基本技能。本节，将通过一个具体的实例来进一步加强对这些技能的应用。本节还将进一步学习界面设计和布局方面的应用。

3.2.1 网站功能设计

本节制作一个简单数据录入系统。首先用户提供登录名和密码登入系统，之后使用 session 维护用户状态。进入系统后，用户可以录入一些个人信息。该系统还应该实现对于不同用户录入的数据，有不同的显示结果。

该系统客户端使用 CSS 完成页面设计和布局，并且使用 JavaScript 验证数据是否有效。客户端使用 PHP 处理这些数据，并将数据显示至浏览器。

3.2.2 页面设计

初步考虑设计两个页面。一个页面是用户登录界面，另一个是用户信息的录入界面。此外还需要 PHP 程序完成用户验证，和负责获取表单提交的数据，并将表单数据显示出来。

3.2.3 用 JavaScript 实现客户端响应

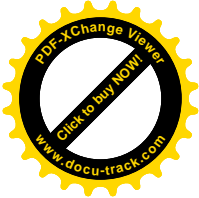
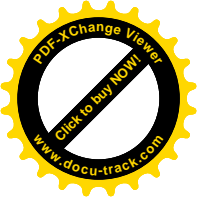
用 JavaScript 可以在客户端验证数据的有效性。在上述登录页面中加入一些 JavaScript 脚本，可以验证用户是否输入了用户名。

其中在<head>标签内加入了 JavaScript 脚本，用以检验用户输入的用户名是否为空，如果为空，则向用户弹出一个提示对话框。此外，该 HTML 文档中还加入了 CSS 代码，定义了页面字体的显示大小、表格单元格背景颜色等页面效果。

3.2.4 服务器端用 PHP 处理请求

当用户输入了用户名后，该数据将提交至一个 PHP 程序做处理。服务器端的 PHP 处理程序需要显示用户刚刚输入的用户名，以及一个录入用户信息的界面。考虑到不同的用户录入的信息都是不同的，因此对于不同的用户，只能看到自己的录入信息，所以要在程序中使用 session 维护不同用户的状态和数据。

3.3 小结



本章讲述了使用 PHP 进行 Web 编程的基础知识和基本技能。包含：用 PHP 获取 HTML 表单数据、用 PHP 处理表单数据、用 PHP 验证数据、PHP 中 session 的使用及使用 PHP 处理文件的上传。最后通过一个简单实例，加深对这些基本技能的实践和掌握。

第 4 章 PHP 对数组的处理

PHP 的一个优势就是提供了丰富的函数，用来处理各种类型的数据、完成一些相对复杂、经常性、重复性多或者和底层有关的操作。这些函数都可以在程序中直接使用。从本章开始，本书就为读者介绍这些 PHP 内置函数的用法。这一章主要讲述一些和数组处理有关的常用 PHP 函数。

4.1 建立数组的方法

在第 2 章介绍数据类型时，已经介绍过建立数组的方法。本节对建立数组的方法做进一步的说明。使用 PHP 的 `array()` 语言结构可以创建一个数组，如下代码所示。

```
$ms_office = array( " word ", " excel ", " outlook ", " access " );
```

上述代码创建了一个名叫 `$ms_office` 的数组，它包含 4 个单元，每个单元存储 1 个值，这些值是：`word`、`excel`、`outlook` 和 `access`，这些值也被称为数组元素。注意，这里的 `array()` 不是函数，它只是 PHP 的语言结构。

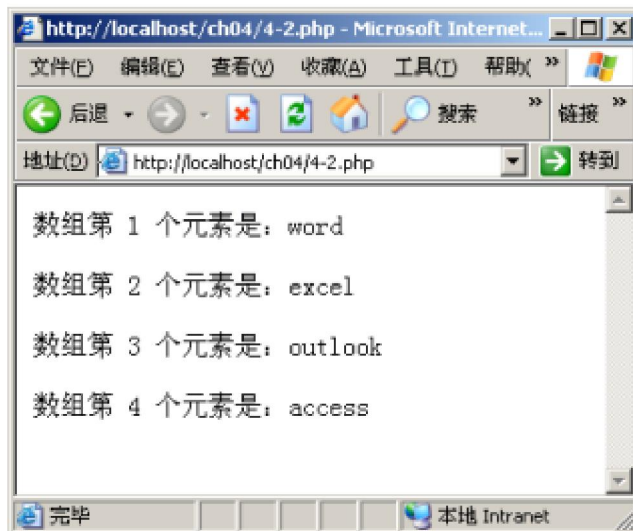
数组的索引又叫“键值”或者“下标”。使用“`=>`”运算符，可以为数组指定索引和值。它的语法格式是：“索引`=>`值”，每对“索引`=>`值”之间使用逗号“`,`”分开。如下代码使用“`=>`”为数组指定索引和值。

4.2 输出数组元素的方法

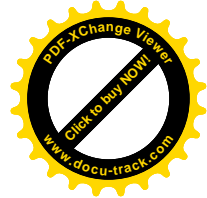
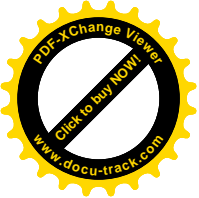
在实际开发中，经常要访问数组的值。如果每次都按数组索引来取数值元素，会非常麻烦。在 PHP 中可以通过各种循环来快速访问数组元素。本节就介绍常用的 `for` 和 `foreach`。

4.2.1 使用 for 循环语句输出数组元素

对于一个按整数顺序索引的数组，可以通过 `for` 循环语句，来依次访问数组元素，



用 for 循环访问数组元素



4.2.2 使用 foreach 循环语句输出数组元素

PHP 还专门为数组提供了一个循环语句：foreach 循环语句，它的语法结构如下所示。

```
foreach($array as $some_var)
{
    statement
}
```

其中 foreach 和 as 都是 PHP 关键字。该语法结构的含义是：按顺序，每次将数组 \$array 中的一个元素存入变量 \$some_var 后执行 statement 语句，然后取下一个元素执行 statement 语句，依次下去，直到所有元素都取到。和 for 循环语句一样，statement 可以是单条语句，也可以是一个语句块。

4.2.3 使用 print_r 函数显示数组元素

有时，为了在程序中查看一个数组的结构（即索引和值的对应关系），需要将数组结构显示到页面上。这可以通过 PHP 的 print_r 函数实现。

HTML 中的 “<pre>” 和 “</pre>” 标签，可以把它们之间文本中的空格、回车、换行、tab 键表现出来，即按照文本原先的布局显示。在代码 4-6 中，调用函数 print_r 之后，也可以加上输出结束标签 “</pre>” 的代码。

4.3 计算数组元素个数

在 PHP 开发中，经常遇到对数组元素个数的计算，通常也称之为计算数组的大小。使用函数 count 计算数组元素的个数，该函数语法如下所示。该函数的返回值为一整数，即数组元素的个数。

```
int count(mixed $var);
```

上述语法格式中的 “mixed” 表示参数 var 可以是多种不同的数据类型，这意味着，该函数不仅仅可以用来计算数组元素个数。“mixed” 这种表示方法会在本书后续内容中继续使用。

4.4 对数组进行分割、合并

数组的分割是指将一个数组拆分成一个或多个数组，而数组的合并，可以看成是分割的逆运算，即将多个数组组合成一个新的数组。下面分别介绍和分割、合并数组有关的 PHP 内置函数。

4.4.1 实现数组分割

使用函数 array_chunk 可以对一个数组进行分割，该函数语法如下所示。

```
array array_chunk(array $input_array, int $size, bool $preserve_key);
```

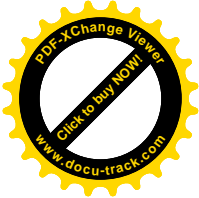
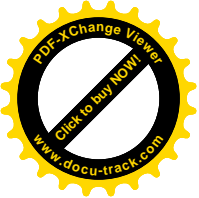
该函数接受 3 个参数，\$input_array 是将要分割的数组（即原数组），\$size 表示原数组被分割后，每个数组中元素的个数，\$preserve_key 是可选参数，它为 TRUE 时，分割后的每个数组的索引使用原数组的索引名；它为 FALSE 时，分割后的每个数组的索引都将使用从 0 开始的整数索引。该函数的返回值是一个数组，这个数组的索引从 0 开始，该数组的每个元素仍然是一个数组，即原数组被分割后的一个数组。

4.4.2 实现数组合并

使用函数 array_merge 可以合并一个或多个数组，该函数的语法如下所示。

```
array array_merge($array...);
```

该函数接受一个或多个数组作为参数，PHP5 中，它只接受 array 类型的参数。该函数返回一个合并后的数组。



使用 array_merge 合并数组

4.5 处理数组元素和键值位置

PHP 的每个数组内部均会有一个指针，指向当前元素。这里所谓的指针，可以把它理解成程序即将要处理的元素所在的位置。处理数组元素和键值（索引）的位置，就是获取数组某个位置上的元素，或获取该元素后，将该指针移动到其他位置。PHP 中，处理数组元素和键值位置的函数主要有以下几个。

current：该函数返回数组的当前元素，不移动当前指针位置。

next：该函数返回当前元素的下一个元素的值，并将指针向后移动一位，下一个元素不存在时，返回 FALSE。

prev：该函数返回当前元素的上一个元素的值，并将指针向前移动一位，上一个元素不存在时，返回 FALSE。

end：该函数返回数组的最后一个元素的值。

4.6 对数组进行排序

对数组排序实际指的是对数组元素的排序。使用 PHP 的有关内置函数，可以对数组元素按多种方式进行排序，下面分别介绍这些函数的用法。

4.6.1 使用 sort 对数组进行排序

使用函数 sort 对数组元素按顺向排序，即按字母由前向后或按数字由小到大排序。该函数的语法如下所示。

```
bool sort(array &$arr [,int $sort_flag]);
```

该函数接受一个数组作为输入参数，如果处理成功，该函数返回 TRUE，否则，该函数返回 FALSE。该函数将会删除所要排序的数组原有的索引值，从而为该数组使用新的索引值。代码 4-11 演示了如何使用该函数为数组元素排序。

4.6.2 使用 rsort 对数组进行逆向排序

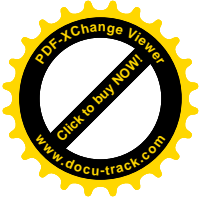
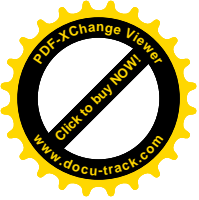
使用函数 rsort 对数组元素做逆向排序，即按字母从后到前或按数字由大到小排序。该函数的语法如下所示。

```
bool rsort(array &$arr [,int $sort_flag]);
```

该函数接受一个数组作为输入参数，如果处理成功，函数返回 TRUE，否则返回 FALSE。该函数将会删除所要排序的数组原有的索引值，从而为该数组使用新的索引值。

4.6.3 排序时保持原数组的索引

前面介绍的两个函数 sort 和 rsort，都会为要排序的数组重新建立索引。有时，对数组排序后，仍需要保持原数组的索引，对于这种情况，在 PHP 中可以使用如下两个函数。



asort : 该函数对数组元素进行排序, 并且保持数组的原索引关系不变。该函数仍然对数组元素按顺向排序。如果处理成功, 该函数返回 TRUE, 否则返回 FALSE。

ksort : 该函数对数组元素按索引名顺序排序, 并且保持数组的原索引关系不变, 如果处理成功该函数返回 TRUE, 否则返回 FALSE。该函数主要用于关联数组。

4.6.4 数组的随机排序

PHP 提供一个随机排序数组元素的函数 shuffle(), 该函数的语法如下所示。

```
bool shuffle (array $array)
```

该函数为参数数组 \$array 中的元素赋予新的键名, 这将删除原有的键名而不仅仅是重新排序。如果执行成功该函数返回 TRUE。

4.6.5 数组的反向排序

本节最后, 介绍一个可以将数组元素按原顺序反向排序的函数, 该函数是 array_reverse()。其语法如下所示。

```
array array_reverse (array $array [, bool $preserve_keys])
```

该函数返回一个与原数组单元顺序相反的数组, 原数组以参数 \$array 传入该函数, 该函数的第 2 个参数是可选的, 如果第 2 个参数 \$preserve_keys 设置为 TRUE, 则原数组的键名将会被保留。

4.7 重置一个数组

PHP 中, 重置一个数组指的是, 将数组中的内部指针移动到该数组的第一个元素所在的单元。使用函数 reset 可以完成对数组的重置。该函数的语法如下所示。

```
mixed reset(array &$arr);
```

该函数接受一个数组类型的变量, 将函数内部指针指向数组第一个单元, 并将第一个单元的元素返回。如果数组为空, 该函数返回 FALSE。

4.8 用数组对变量赋值

PHP 中, 可以将数组的元素作为值赋给变量, 使用语言结构 list() 可以完成这个功能。严格意义上讲, list() 和 array() 一样, 不是函数。

在上述程序中, 通过 list, 将数组 \$planet 的第一个元素 'Earth' 赋给了变量 \$no1, 将第二个元素赋给了变量 \$no2, 将第 3 个元素赋给了变量 \$no3。

4.9 快速创建数组

函数 range() 可以快速创建制定元素范围的数组。其语法如下所示。

```
array range(mixed $low, mixed $high [, number $step])
```

函数 range() 返回一个数组, 其中元素为参数从 \$low 到 \$high 的序列, 包括它们本身。例如 range(1,9) 相当于 array(1,2,3,4,5,6,7,8,9), 如果 \$low > \$high, 则序列将从 \$high 到 \$low。该函数的第 3 个参数 \$step 是可选的, 如果给出了 step 的值, 它将被作为元素之间的跨度值。step 应该为正值。如果未指定, step 则默认为 1。

4.10 压入和弹出数组元素

这里所说的压入是指, 向数组的末尾增加一个元素。而弹出是指, 将数组中的最后一个元素取出。PHP 提供两个函数分别完成这两个功能, 这两个函数是 array_push() 和 array_pop()。

4.10.1 压入数组元素

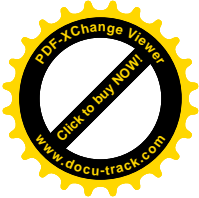
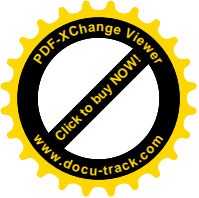
函数 array_push() 将一个或多个元素压入数组的末尾, 其语法如下所示。

```
int array_push(array $array, mixed $var)
```

该函数将参数 \$array 当成一个数据栈, 并将传入的参数 \$var 压入 \$array 的末尾。array 的长度将根据入栈变量的数目增加, 执行正常时, 该函数返回数组 \$array 新的单元总数。

4.10.2 弹出数组元素

函数 array_pop() 将数组最后一个元素弹出, 其语法如下所示。



mixed array_pop(array \$array)

该函数弹出并返回参数\$array 数组的最后一个单元，并将数组\$array 的长度减一。如果参数\$array 为空（或者不是数组），该函数将返回 NULL。需要注意的是，使用本函数后会重置数组指针，相当于使用了函数 reset()。

4.11 改变数组字符索引名的大小写

PHP 提供了一个函数，可以将以字符串作为数组索引（键名）的大小写全部更换，这个函数是 array_change_key_case()，该函数的语法如下所示。

array array_change_key_case(array \$input [, int \$case])

函数 array_change_key_case()将参数\$input 数组中的所有索引改为全小写或大写。可选参数\$case 用来指定大小写转换方式，它为两个常量值：CASE_UPPER 和 CASE_LOWER。没有指定该参数时，该函数使用的默认值是 CASE_LOWER。

4.12 对数组的集合处理

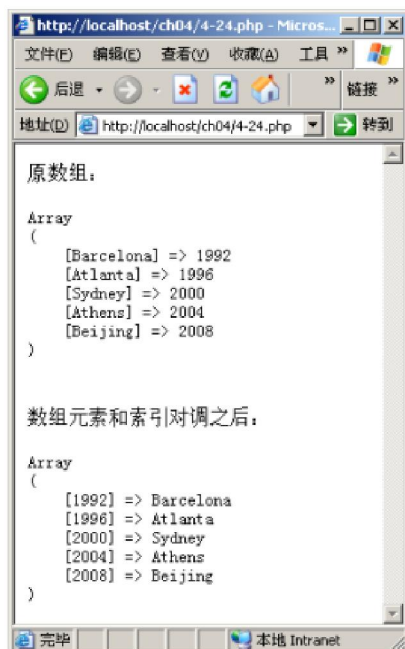
函数 array_intersect()可以用来计算数组的交集，其语法如下所示。

array array_intersect(array \$arr1, array \$arr2 [, \$arr ...])

函数 array_intersect()返回一个数组，该数组包含了在参数数组\$arr1 中出现，同时也出现在所有其它参数数组中的值，注意键名保留不变。

4.13 交换数组索引和元素

PHP 中的数组可以通过函数实现索引和元素的交换，这对程序实现某些操作带来极大的便利，这个函数就是 array_flip()，该函数的语法如下所示。



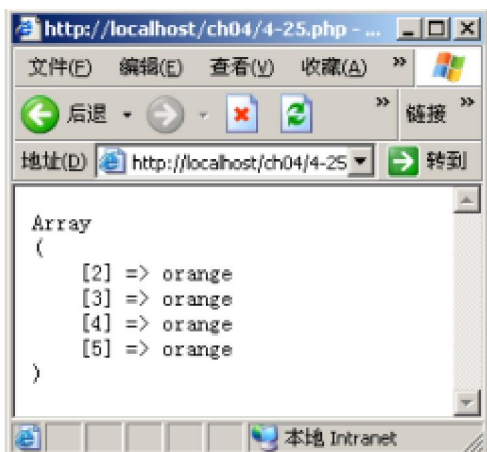
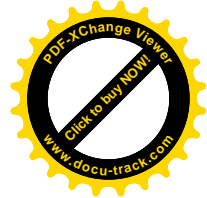
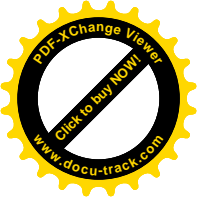
数组元素和索引的对换

4.14 快速填充数组元素

PHP 提供函数快速填充数组，将数组中的某些或所有元素都置成通一个值，这个函数是 array_fill()，其语法如下所示。

array array_fill(int \$start, int \$num, mixed \$value)

函数 array_fill()有 3 个参数：其中参数\$start 指定要填充元素的开始索引，参数\$num 表示将\$num 个数组元素以参数\$value 填充。



快速填充数组元素

4.15 统计数组元素出现次数

函数 `count()` 用来计算整个数组元素的个数，而函数 `array_count_values()` 用来统计数组中各元素出现的次数，即数组中各元素的个数。函数 `array_count_values()` 的语法如下所示。

```
array array_count_values(array $arr)
```

该函数返回一个数组，这个数组把参数数组 `$arr` 中的元素作为索引，元素出现的次数作为对应索引的值。

函数 `array_count_values()` 统计出数组 `$arr` 中，各个元素出现的次数，而且从这个执行结果可以看出，函数 `array_count_values()` 统计元素时，是元素大小写的，比如 `Say` 和 `say` 会分别被统计。

4.16 检查数组索引是否存在

PHP 提供函数 `array_key_exists()` 检查给定的索引是否存在于数组中，该函数语法如下所示。

```
bool array_key_exists(mixed $key, array $search)
```

该函数查找由参数 `$key` 指定的索引，是否存在于由参数 `$sarch` 指定的数组中，如果存在，该函数返回 `TRUE`，否则或者出错时，该函数返回 `FALSE`。参数 `$key` 可以是任何能作为数组索引的值。

4.17 取得数组中的所有索引

PHP 提供函数 `array_keys()` 用来获取数组中所有的索引名，该函数语法如下所示。

```
array array_keys(array $arr [, mixed $search_value [, bool $strict]])
```

该函数会将参数数组 `$arr` 中，所有的索引名返回到一个数组中。参数 `search_value` 是可选的，如果指定该参数，则函数 `array_keys` 只返回该元素值对应的索引名，可选参数 `$strict` 用来进行全等比较。

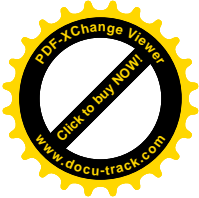
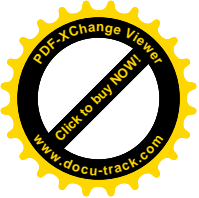
4.18 用回调函数处理数组

PHP 支持回调函数操作给定数组的元素上，这也是 PHP 数组函数的强大功能之一。函数 `array_map()` 就可以完成这样的工作，它将回调函数作用到给定数组的元素上。其语法如下所示。

```
array array_map(callback_func, array $arr1 [, array ...])
```

函数 `array_map()` 返回一个数组，该数组包含了参数 `$arr1` 中的所有元素经过函数 `callback_func` 处理过之后的元素。callback 接受参数的数目应该和传递给 `array_map()` 函数的数组数目一致。

4.19 其他数组处理函数



本节介绍两个在实际应用中经常会用到的两个 PHP 内置函数：in_array()和 key()。下面分别加以介绍。

in_array：该函数可以判断某个值是否存在于数组中。如果在数组中找到该值，函数返回 TRUE，否则返回 FALSE。

key：该函数可以获取数组中当前单元的索引值。

4.19.1 判断某个值是否存在于数组

代码 4-30 演示了函数 in_array 的用法。

注意：从该执行结果可以看出，in_array 是区分大小写的。“Mars”和“mars”被认为是两个不同的值。



使用 in_array 检查数组中是否存在某值

4.19.2 获取数组中当前单元的索引值

代码(详细内容请参照本书)演示了函数 key 的用法。这段程序首先将当前指针指向数组的最后一个单元，然后通过函数 key 获取当前元素（即最后一个元素）的索引值。



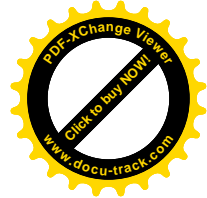
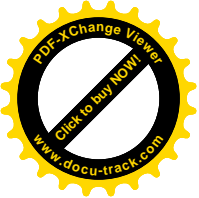
使用 key 获取数组当前元素索引值

4.20 小结

本章介绍了对数组进行处理的一些主要的 PHP 内置函数和语言结构。他们包括：

使用 array 创建数组。使用 for、foreach 等循环访问数组元素。使用函数 count()计算数组元素个数。使用函数 array_chunk()/array_merge()分割/合并数组。处理数组元素和键值位置的函数 current()、prev()、next()、end()等。对数组元素排序的函数 sort()、rsort()、asort()、ksort()等。使用 reset()重置数组。使用 list()将数组元素赋值给变量。使用函数 shuffle()对数组元素进行随机排序。快速创建数组的函数 range()。使用函数 array_push()、array_pop()压入和弹出数组元素。

使用函数 array_change_key_case()改变字符索引名的大小写。对数组做集合处理的函数



array_intersect()、array_intersect_assoc()、array_diff()和 array_diff_assoc()。交换数组索引和元素的函数 array_flip()。快速填充数组元素的函数 array_fill()。使用函数 array_count_values() 统计数组元素出现的次数。使用函数 array_key_exists() 检查索引是否存在。使用函数 array_keys() 取得数组中的所有索引。使用回调函数处理数组的函数 array_map()。使用 in_array() 检查数组中是否存在某值。使用 key() 获取数组中当前单元的索引值。本章介绍了较多关于处理数组的 PHP 函数，读者需要再实践中慢慢掌握这些函数及其用法。

第 5 章 PHP 对字符串的处理

PHP 对字符串的处理的完美支持，是 PHP 的强大功能之一。PHP 提供了数十个用来处理字符串的内置函数，使用这些函数，可以在 PHP 程序中很方便地完成对字符串的各种操作。本章为读者介绍 PHP 开发中一些比较常用的字符串处理函数。

5.1 对字符串做分割和合并

在 PHP 程序中，可以使用函数 explode 用一个字符串分割另一个字符串，该函数的语法如下所示。

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

函数 explode 使用字符串 \$separator (该函数的第 1 个参数) 作为边界点，来分割字符串 \$str (该函数的第 2 个参数)。该函数的返回值是一个数组，字符串被分割后，分割的部分会存入该数组。该函数最后一个参数 \$limit 是可选参数。如果设置了该参数，则返回的数组中最多有 limit 个元素，最后一个元素包含字符串 \$str 的所有剩余部分。

5.2 对字符串做比较和替换

字符串比较通常是指，判断两个字符串是否一样，用更专业的说法，就是比较两个字符串的大小。PHP 通过字符的 ASCII 码来决定字符的大小，从而决定字符串的大小。例如，字母 a 的 ASCII 码是 097，字母 b 的 ASCII 码是 098，因此 'b' > 'a'。字母表中，靠后字母的 ASCII 码大，所有的小写字母比大写字母的 ASCII 码大。

5.2.1 比较两个字符串大小

PHP 提供函数 strcmp 对两个字符串比较大小。该函数语法如下所示。

```
int strcmp(string $str1, string $str2);
```

函数 strcmp 接受两个参数，它们是待比较的字符串。该函数返回值是整数，如果字符串 \$str1 和字符串 \$str2 相等，则函数返回 0，如果字符串 \$str1 小于字符串 \$str2，则函数返回 -1，如果字符串 \$str1 大于字符串 \$str2，则函数返回值大于 0。

5.2.2 字符串的替换

在实际开发中，有时需要将字符串中的某些字符串替换成其他字符串，这可以通过 PHP 的内置函数 str_replace 来完成。该函数的语法如下所示。

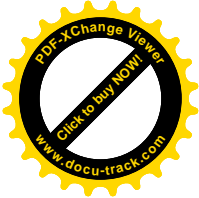
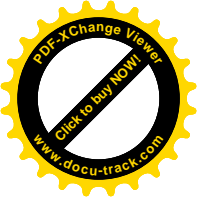
```
string str_replace(string $search, string $replace, string $str);
```

该函数将字符串 \$str 中的 \$search 部分全部替换成字符串 \$replace，并且返回替换后的字符串。

5.3 输出打印字符串

向 Web 页面输出字符串是 PHP 程序最常用的操作之一。事实上，在本节之前的几乎示例程序中都有输出字符串的代码。在 PHP 输出字符串的方式有如下两种。

(1) 使用 print。它的用法如下所示。



```
int print(string $str);
```

print 将字符串\$str 输出，并且总是返回 1。严格意义上讲，print 不是函数，它和 echo 一样，只是 PHP 的语言结构，因此，可以不使用括号将其参数括住。

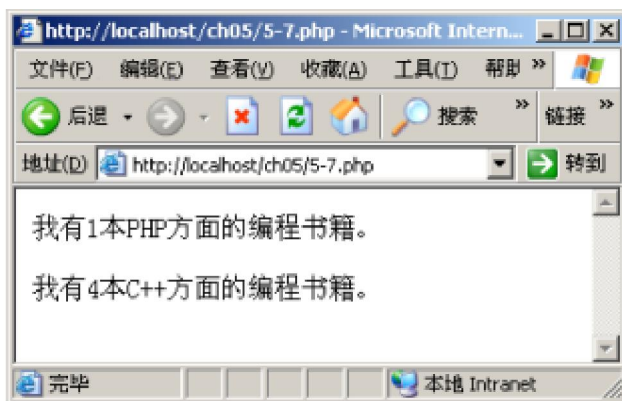
(2) 使用 echo。使用 echo 输出字符串，在此之前已经大量使用，这里不再举例说明。

5.4 格式化字符串

将字符串转换成某种特定的格式，叫做字符串的格式化。比如一本书的价格是 39 元，一般都会格式化成“39.00”的形式。对字符串进行格式化，是程序开发中经常遇到处理操作之一。PHP 语言提供函数 sprintf，将字符串格式化，该函数的语法如下所示。

```
string sprintf(string $format, mixed $args);
```

参数 \$format 是要转换的格式，该参数通常以百分号%开始，到转换字符为止。例如，%1.2f 就是一个转换格式，其中 f 就是转换字符，它表示将字符串格式化成浮点数。这种用法，对于有 C 语言基础的读者来说，一定不会陌生。该函数返回格式化后的字符串。



sprintf 的简单应用

5.5 获取字符串中的一个子串

获取字符串中的一个子串是指，获取这个字符串的某一部分。在 PHP 中，可以使用函数 substr 来获取字符串的某一部分。该函数的语法如下所示。

```
string substr(string $str, int $start, int $length);
```

函数 substr 将字符串 \$str 中，从位置 \$start 的字符开始，截取长度为 \$length 的字符串，然后返回该字符串。需要注意的是，字符串的第 1 个字符的位置是 0，第 2 个字符的位置是 1，以此类推。

5.6 删除字符串末尾部分的空白字符

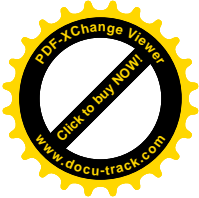
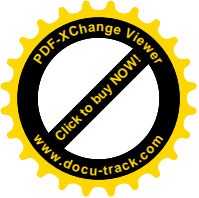
PHP 程序有时需要去掉字符串中的空白字符，比如用户从网页提交数据时，在文本框输入内容后，多敲入了一些空格，而 PHP 处理程序很可能要将这些数据存入数据库，这时，不希望将这些没有实际意义的空白字符存入数据库，因此要将这些空白字符处理掉。

PHP 提供了一些函数，分别可以处理字符串末尾、开头的空白字符，也提供了同时处理字符串两端空白字符的函数。本节先介绍处理字符串末尾部分空白字符的内置函数。在 PHP 中，可以使用函数 rtrim 删除字符串末尾部分的空白字符。该函数的语法如下所示。

```
string rtrim(string $str);
```

5.7 删除字符串开始部分的空白字符

和函数 rtrim 对应的是函数 ltrim，该函数将字符串开始部分的空白字符删除掉。从这个函数的命名可以看出，ltrim 中的字母 l 代表的是 left，一个字符串的 left 就是它的开始部分，所以，ltrim 表示去掉字符串开始部分的空白字符。同理，rtrim 中的首字母 r 表示 right。这样，初学者就容易记住这两个函数及其用法了。函数 ltrim 的语法如下所示。



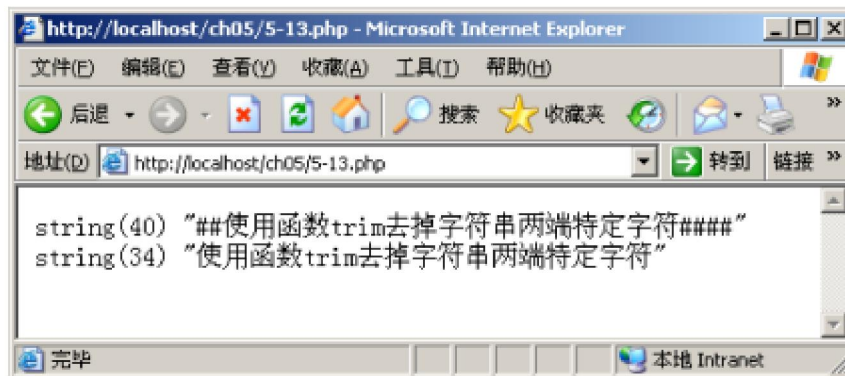
string ltrim(string \$str);

该函数将字符串\$str 开始部分的空白字符删除，返回开始部分没有空白的字符串。

5.8 删除字符串两头的空白字符

如果希望删除字符串两头的空白字符，可以使用函数 trim，该函数的语法如下所示。

string trim(string \$str);



是用 trim 删除字符串两端特定字符

5.9 获取字符串的长度

字符串的长度一般是指，组成字符串的字符的数量。PHP 提供函数 strlen 来获取一个字符串的长度。该函数的语法如下所示。

int strlen(string \$str);

该函数返回字符串\$str 的长度。



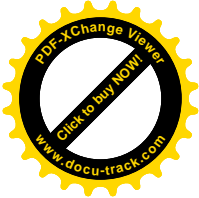
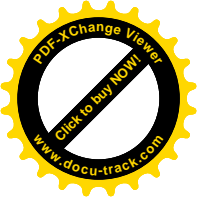
使用 strlen 获取字符串长度

5.10 对字符串做大小写转换

PHP 提供两个函数分别完成对字符串中字母的大小写转换，它们是：

strtoupper(\$str)，将字符串\$str 中的字母全部转换为大写字母。

strtolower(\$str)，将字符串\$str 中的字母全部转换为小写字母。



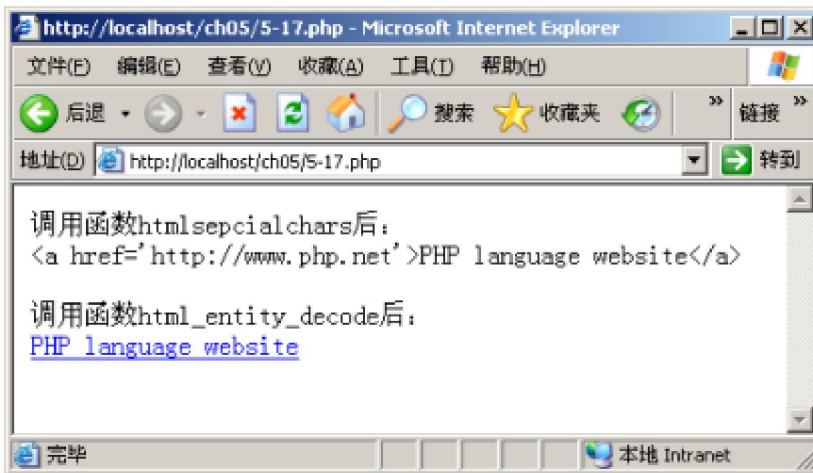
对字符串做大小写转换

5.11 处理含有 HTML 标记的字符串

使用 PHP 进行 Web 页面开发，经常会遇到处理 HTML 标记的问题。PHP 提供的函数 `htmlspecialchars` 完成对字符串中 HTML 标记的处理，该函数可将字符串中有关字符转换成 HTML 实体，即 HTML 字符编码。该函数的语法如下所示。

```
string htmlspecialchars(string $str);
```

函数 `htmlspecialchars` 将字符串 `$str` 中的 HTML 标记（如 `<`、`>` 等）转换成相关编码的形式。



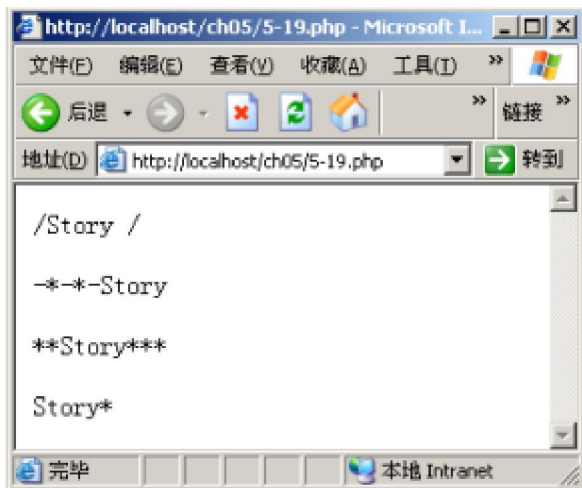
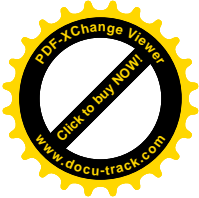
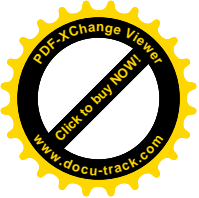
使用 `html_entity_decode` 转换 HTML 实体

5.12 多次生成一个字符串

函数 `str_repeat()` 可以用来重复生成一个字符串，该函数的语法如下所示。

```
string str_repeat (string $input, int $num)
```

函数 `str_repeat()` 有两个参数，第 1 个参数 `$input` 是要重复生成的字符串，第 2 个参数 `$num` 是要重复的次数。参数 `$num` 的取值必须大于等于 0，如果 `$num` 设置为 0，该函数会返回一个空字符串。



使用字符串对另一字符串进行补全

5.13 将字符串分析到变量

这小节所说的分析字符串，是指分析 URL 中的查询字符串。例如一个 URL 是：`http://www.somesite.com/index.php?i=000&key=1a2b3c&cnt=10`，那么该 URL 的查询字符串就是 `i=000&key=1a2b3c&cnt=10`。PHP 提供函数 `parse_str()` 可以分析类似的字符，并将分析结果赋值给变量。函数 `parse_str()` 的语法如下所示。

```
void parse_str(string $str [, array $arr])
```

本函数可将 URL 的查询字符串参数 `$str` 解析，返回的变量名及值就和查询字符串中的名称及值相对应。该函数第 2 个参数一个指定的数组，是可选的。如果指定这个数组参数，那么函数 `parse_str()` 将分析字符串得到变量作为数组索引、值作为数组元素存入该数组。

5.14 转换字符串到数组

PHP 提供的函数 `str_split()`，可将字符串转换成一个数组，即将数组分割成几个部分，每个部分的值作为数组的元素。该函数的语法如下所示。

```
array str_split(string $str [, int $split_length])
```

该函数将字符串 `$str` 做分割，将分割后的字符串各部分存入数组，并将数组返回。可选参数 `$split_length` 用来指定分割字符串长度。如果没有指定该参数，函数将把字符串 `$str` 按一个字符的长度分割。

5.15 计算字符串的 Hash (哈希)

所谓的哈希 (Hash) 也叫散列，它就是指把任意长度的输入 (又叫做预映射)，通过散列算法，变换成固定长度的输出，该输出就是散列值。

Hash 主要用于信息安全领域的加密算法，它把一些不同长度的信息转化成杂乱的 128 位的编码，这个编码，叫做 Hash 值。也可以说，Hash 就是找到一种数据内容和数据存放地址之间的映射关系。常见的 Hash 算法有 MD5，SHA1 等。

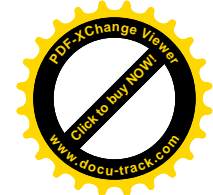
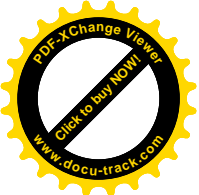
PHP 中提供了对应的函数，完成对字符串做 MD5 哈希计算的函数，也提供函数完成 SHA1 哈希计算的函数，这两个函数分别是 `md5()` 和 `sha1()`，下面分别加以简要说明。函数 `md5()` 的语法如下所示。

```
string md5(string $str)
```

该函数用来计算参数字符串 `$str` 的 MD5 哈希，另外一个计算哈希的函数 `sha1()` 的语法如下所示。

```
string sha1(string $str)
```

5.16 小结



本章通过介绍一些主要的字符串处理函数，来学习在 PHP 程序中如何完成对字符串的操作。这些函数都是比较基本的，读者务必掌握。本章讲到的函数包括：将字符串分割后存入数组的函数 `explode()`。将数组中的元素合并成字符串的函数 `implode()`。比较两个字符串大小的函数 `strcmp()`。字符串替换函数 `str_replace()`。输出字符串的语言结构 `echo` 和 `print`。字符串格式化函数 `sprintf()`。

截取字符串的函数 `substr()`。删除字符串开头和末尾空白字符的函数 `ltrim()`、`rtrim()`、`trim()`。获取字符串长度的函数 `strlen()`。对字符串做大小写转换的函数 `strtolower()` 和 `strtoupper()`。处理含有 HTML 标记的函数 `htmlentities()`、`htmlspecialchars()` 和 `html_entity_decode()`。重复生成字符串的函数 `str_repeat()`。分析字符串到变量的函数 `parse_str()`。转换字符串到数组的函数 `str_split()`。计算字符串 Hash 的函数 `md5()`。和 `sha1()`。