YAYASAN PRIMA AGUS TEKNIK

# PENGEMBANGAN
# WEB PHP
## (Hypertext Preprocessor) Language

Dr. Budi Raharjo, S.Kom, M.Kom, MM.

Dr. Budi Raharjo, S.Kom, M.Kom, MM.

# PENGEMBANGAN WEB PHP

## (Hypertext Preprocessor) Language

YAYASAN PRIMA AGUS TEKNIK

PENERBIT :
YAYASAN PRIMA AGUS TEKNIK
Jl. Majapahit No. 605 Semarang
Telp. (024) 6723456. Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

ISBN 978-634-7227-21-8 (PDF)
9 786347 227218

**PENGEMBANGAN WEB PHP**
**(Hypertext Preprocessor) Language**

**Penulis :**
Dr. Budi Raharjo, S.Kom, M.Kom, MM.

**ISBN : 978-634-7227-21-8**

**Editor :**
Dr. Mars Caroline Wibowo. S.T., M.Mm.Tech

**Penyunting :**
Dr. Joseph Teguh Santoso, M.Kom.

**Desain Sampul dan Tata Letak :**
Irdha Yunianto, S.Ds., M.Kom.

**Penebit** :
Yayasan Prima Agus Teknik Bekerja sama dengan
Universitas Sains & Teknologi Komputer (Universitas STEKOM)

**Anggota IKAPI No:** 279 / ALB / JTE / 2023

**Redaksi :**
Jl. Majapahit no 605 Semarang
Telp. (024) 6723456
Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

**Distributor Tunggal :**
**Universitas STEKOM**
Jl. Majapahit no 605 Semarang
Telp. (024) 6723456
Fax. 024-6710144
Email : info@stekom.ac.id

# KATA PENGANTAR

Puji syukur ke hadirat Allah SWT yang telah melimpahkan rahmat, karunia, dan hidayah-Nya, sehingga penulis dapat menyelesaikan penulisan buku yang berjudul ***"Pengembangan Web PHP (Hypertext Preprocessor Language)"*** ini dengan baik dan tepat waktu. Buku ini disusun sebagai kontribusi dalam pengembangan literatur teknologi informasi, khususnya dalam bidang pemrograman web yang menggunakan bahasa pemrograman PHP.

PHP adalah salah satu bahasa pemrograman server-side yang sangat populer dan banyak digunakan dalam dunia pengembangan web karena kemudahan, fleksibilitas, serta kemampuannya untuk membuat konten web dinamis dan interaktif. Buku ini dirancang untuk memberikan pemahaman yang mendalam dan menyeluruh mulai dari konsep dasar hingga teknik-teknik lanjutan dalam PHP, sehingga pembaca dapat menguasai berbagai aspek penting dalam pengembangan web modern.

Materi dalam buku ini disusun secara sistematis dan komprehensif, dimulai dari pengenalan dasar PHP, cara menanamkan kode PHP ke dalam HTML, pengelolaan data dan file, manipulasi string dan array, hingga pemrograman berorientasi objek yang menjadi fondasi penting dalam pengembangan aplikasi skala besar. Selain itu, buku ini juga membahas interaksi dengan sistem file dan server, penggunaan protokol jaringan, pengelolaan tanggal dan waktu, pembuatan gambar dinamis, hingga pengelolaan sesi dan fitur-fitur berguna lainnya yang sering digunakan dalam pengembangan web.

Setiap bab dilengkapi dengan contoh kode praktis dan penjelasan yang mudah dipahami, sehingga pembaca dapat langsung mengaplikasikan ilmu yang diperoleh dalam proyek nyata. Kami berharap buku ini dapat menjadi sumber belajar yang bermanfaat bagi mahasiswa, pengembang web pemula, maupun profesional yang ingin memperdalam kemampuan mereka dalam PHP.

Kami menyadari bahwa dalam penyusunan buku ini masih terdapat kekurangan, oleh karena itu kami sangat mengharapkan kritik dan saran yang membangun demi penyempurnaan edisi berikutnya.

Semarang, Juli 2025
Penulis

Dr. Budi Raharjo, S.Kom, M.Kom, MM

# DAFTAR ISI

# BAB 1
# PENGETAHUAN SINGKAT PHP

Bab ini memberi Anda gambaran singkat tentang sintaksis dan konstruksi bahasa PHP. Jika Anda sudah menjadi programmer PHP, bab ini mungkin akan mengisi beberapa celah dalam pengetahuan Anda. Jika Anda memiliki latar belakang menggunakan C, ASP, atau bahasa pemrograman lain, bab ini akan membantu Anda untuk memahaminya dengan cepat.

Dalam buku ini, Anda akan mempelajari cara menggunakan PHP dengan mengerjakan banyak contoh dunia nyata, yang diambil dari pengalaman kami dalam membangun situs e-commerce. Sering kali buku teks pemrograman mengajarkan sintaksis dasar dengan contoh yang sangat sederhana. Kami memilih untuk tidak melakukannya. Kami menyadari bahwa sering kali yang ingin Anda lakukan adalah menyiapkan sesuatu dan menjalankannya, untuk memahami bagaimana bahasa tersebut digunakan, daripada membaca referensi sintaksis dan fungsi lain yang tidak lebih baik dari manual daring.

Coba contoh-contohnya ketik atau muat dari CD-ROM, ubah, hancurkan, dan pelajari cara memperbaikinya lagi. Dalam bab ini, kita akan mulai dengan contoh formulir pesanan produk daring untuk mempelajari bagaimana variabel, operator, dan ekspresi digunakan dalam PHP. Kami juga akan membahas jenis variabel dan prioritas operator. Anda akan mempelajari cara mengakses variabel formulir dan cara memanipulasinya dengan menghitung total dan pajak pada pesanan pelanggan.

Kemudian kami akan mengembangkan contoh formulir pesanan daring dengan menggunakan skrip PHP kami untuk memvalidasi data masukan. Kami akan meneliti konsep nilai Boolean dan memberikan contoh penggunaan if, else, operator ?:, dan pernyataan switch. Terakhir, kami akan mengeksplorasi perulangan dengan menulis beberapa PHP untuk menghasilkan tabel HTML berulang. Topik utama yang akan Anda pelajari dalam bab ini meliputi

- Menanamkan PHP dalam HTML
- Menambahkan konten dinamis
- Mengakses variabel formulir
- Pengidentifikasi
- Variabel yang dideklarasikan pengguna
- Jenis variabel
- Menetapkan nilai ke variabel
- Konstanta
- Cakupan variabel
- Operator dan prioritas
- Ekspresi
- Fungsi variabel
- Membuat keputusan dengan if, else, dan switch
- Iterasi: while, do, dan for loop

## 1.1 MENGGUNAKAN PHP

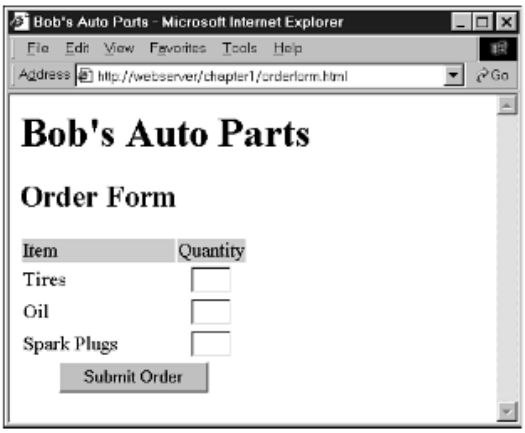
Untuk mempelajari contoh-contoh dalam bab ini dan bagian lain buku ini, Anda memerlukan akses ke server Web dengan PHP yang terinstal. Untuk mendapatkan hasil maksimal dari contoh dan studi kasus, Anda harus menjalankannya dan mencoba mengubahnya. Untuk melakukannya, Anda memerlukan tempat uji coba tempat Anda dapat bereksperimen. Jika PHP tidak terinstal di komputer Anda, Anda harus memulainya dengan menginstalnya, atau meminta administrator sistem untuk menginstalnya untuk Anda.

**Contoh Aplikasi: Bob's Auto Parts**

Salah satu aplikasi paling umum dari bahasa skrip sisi server adalah memproses formulir HTML. Anda akan mulai mempelajari PHP dengan mengimplementasikan formulir pesanan untuk Bob's Auto Parts, sebuah perusahaan suku cadang fiktif

**Formulir Pesanan**

Saat ini, programmer HTML Bob telah menyiapkan formulir pesanan untuk suku cadang yang dijual Bob. Formulir pesanan ditunjukkan pada Gambar 1.1. Ini adalah formulir pesanan yang relatif sederhana, mirip dengan banyak formulir yang mungkin pernah Anda lihat saat berselancar di internet.




**Gambar 1.1 Formulir Pesanan Awal Bob Hanya Mencatat Produk Dan Kuantitas.**

Hal pertama yang ingin Bob lakukan adalah mengetahui apa yang dipesan pelanggannya, menghitung total pesanan pelanggan, dan berapa banyak pajak penjualan yang harus dibayarkan atas pesanan tersebut. Bagian HTML untuk ini ditunjukkan dalam Daftar 1.1. Ada dua hal penting yang perlu diperhatikan dalam kode ini.

**Daftar 1.1 Orderform.Html—Html Untuk Formulir Pesanan Dasar Bob**

```
<form action=”processorder.php” method=post>
<table border=0>
<tr bgcolor=#cccccc>
  <td width=150>Item</td>
  <td width=15>Quantity</td>
</tr>
<tr>
```

```
  <td>Tires</td>
  <td align=center><input type=”text” name=”tireqty” size=3 maxlength=3></td>
</tr>
<tr>
  <td>Oil</td>
  <td align=center><input type=”text” name=”oilqty” size=3 maxlength=3></td>
</tr>
<tr>
  <td>Spark Plugs</td>
  <td align=center><input type=”text” name=”sparkqty” size=3 maxlength=3></td>
</tr>
<tr>
  <td colspan=2 align=center><input type=submit value=”Submit Order”></td>
</tr>
</table>
</form>
```

Hal pertama yang perlu diperhatikan adalah kita telah menetapkan tindakan formulir sebagai nama skrip PHP yang akan memproses pesanan pelanggan. (Kita akan menulis skrip ini selanjutnya.) Secara umum, nilai atribut ACTION adalah URL yang akan dimuat saat pengguna menekan tombol kirim. Data yang diketik pengguna dalam formulir akan dikirim ke URL ini melalui metode yang ditentukan dalam atribut METHOD, baik GET (ditambahkan di akhir URL) atau POST (dikirim sebagai paket terpisah).

Hal kedua yang perlu Anda perhatikan adalah nama bidang formulir tireqty, oilqty, dan sparkqty. Kita akan menggunakan nama-nama ini lagi dalam skrip PHP kita. Karena itu, penting untuk memberi bidang formulir Anda nama yang bermakna yang mudah Anda ingat saat mulai menulis skrip PHP. Beberapa editor HTML akan menghasilkan nama bidang seperti field secara default. Nama-nama ini sulit diingat. Hidup Anda sebagai programmer PHP akan lebih mudah jika nama-nama ini mencerminkan data yang diketik ke dalam bidang tersebut. Anda mungkin ingin mempertimbangkan untuk mengadopsi standar pengkodean untuk nama bidang sehingga semua nama bidang 1 di seluruh situs Anda menggunakan format yang sama. Ini memudahkan untuk mengingat apakah, misalnya, Anda menyingkat kata dalam nama bidang, atau memasukkan garis bawah sebagai spasi.

**Memproses Formulir**

Untuk memproses formulir, kita perlu membuat skrip yang disebutkan dalam atribut ACTION dari tag FORM yang disebut `processorder.php`. Buka editor teks Anda dan buat file ini. Ketik kode berikut:

```
  <html>
  <head>
    <title>Bob’s Auto Parts - Order Results</title>
  </head>
  <body>
  <h1>Bob’s Auto Parts</h1>
```

```
<h2>Order Results</h2>
</body>
</html>
```

Perhatikan, bagaimana semua yang telah kita ketik sejauh ini hanyalah HTML biasa. Sekarang saatnya menambahkan beberapa kode PHP sederhana ke skrip kita.

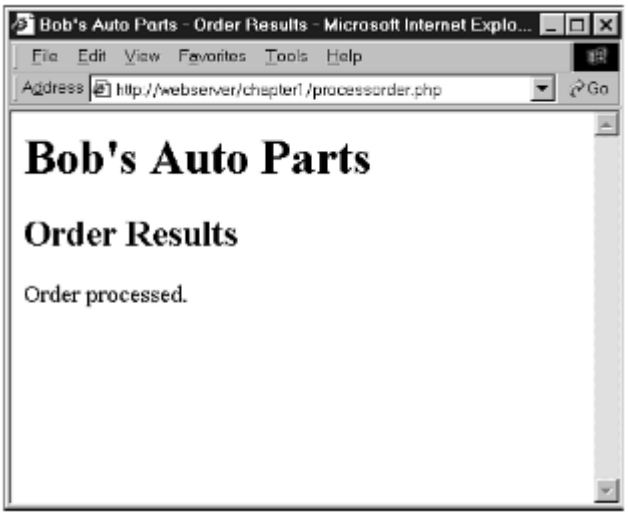
## 1.2 MENANAMKAN PHP DALAM HTML

Di bawah judul <h2> dalam berkas Anda, tambahkan baris berikut:

```
<?
  echo “<p>Order processed.”;
?>
```

Simpan berkas dan muat di peramban Anda dengan mengisi formulir Bob dan mengeklik tombol Kirim. Anda akan melihat sesuatu yang mirip dengan keluaran yang ditunjukkan pada Gambar 1.2. Perhatikan bagaimana kode PHP yang kita tulis disematkan di dalam berkas HTML yang tampak normal. Coba lihat sumbernya dari peramban Anda. Anda akan melihat kode ini:

```
<html>
<head>
  <title>Bob’s Auto Parts - Order Results</title>
</head>
<body>
<h1>Bob’s Auto Parts</h1>
<h2>Order Results</h2>
<p>Order processed.</p></body>
</html>
```




**Gambar 1.2 Teks Yang Diteruskan Ke Konstruksi Echo Php Di-Echo Ke Browser.**

Tidak ada PHP mentah yang terlihat. Ini karena interpreter PHP telah menjalankan skrip dan menggantinya dengan output dari skrip. Ini berarti bahwa dari PHP kita dapat menghasilkan tampilan HTML bersih yang dapat dilihat dengan browser apa pun dengan kata lain, browser pengguna tidak perlu memahami PHP. Ini menggambarkan konsep skrip sisi server secara singkat. PHP telah ditafsirkan dan dijalankan di server Web, berbeda dari JavaScript dan teknologi sisi klien lainnya yang ditafsirkan dan dijalankan dalam browser Web di komputer pengguna.

Kode yang sekarang kita miliki dalam berkas ini terdiri dari empat hal:

- ☑ HTML
- ☑ Tag PHP
- ☑ Pernyataan PHP
- ☑ Whitespace Kita juga dapat menambahkan
- ☑ Komentar

Sebagian besar baris dalam contoh tersebut hanyalah HTML biasa.

**Menggunakan Tag PHP**

Kode PHP dalam contoh sebelumnya dimulai dengan <? dan diakhiri dengan ?>. Hal ini serupa dengan semua tag HTML karena semuanya dimulai dengan simbol kurang dari (<) dan diakhiri dengan simbol lebih dari (>). Simbol-simbol ini disebut tag PHP yang memberi tahu server Web di mana kode PHP dimulai dan diakhiri. Teks apa pun di antara tag akan ditafsirkan sebagai PHP. Teks apa pun di luar tag ini akan diperlakukan sebagai HTML normal. Tag PHP memungkinkan kita untuk keluar dari HTML. Tersedia berbagai gaya tag. Ini adalah gaya pendek. Jika Anda mengalami beberapa masalah saat menjalankan skrip ini, mungkin karena tag pendek tidak diaktifkan dalam instalasi PHP Anda. Mari kita bahas ini secara lebih rinci.

**Gaya Tag PHP**

Sebenarnya ada empat gaya tag PHP yang dapat kita gunakan. Setiap fragmen kode berikut ini setara.

- Gaya pendek

```
<? echo “<p>Order processed.”; ?>
```

Ini adalah gaya tag yang akan digunakan dalam buku ini. Ini adalah tag default yang digunakan pengembang PHP untuk membuat kode PHP.

Gaya tag ini adalah yang paling sederhana dan mengikuti gaya instruksi pemrosesan SGML (*Standard Generalized Markup Language*). Untuk menggunakan jenis tag ini yang merupakan jenis terpendek anda perlu mengaktifkan tag pendek di berkas konfigurasi Anda, atau mengompilasi PHP dengan tag pendek yang diaktifkan. Anda dapat menemukan informasi lebih lanjut tentang cara melakukannya di Lampiran A.

- Gaya XML

```
<? echo “<p>Order processed.”; ?>
```

Gaya tag ini dapat digunakan dengan dokumen XML (*Extensible Markup Language*). Jika Anda berencana untuk menyajikan XML di situs Anda, sebaiknya gunakan gaya tag ini.

- Gaya SCRIPT

```
<SCRIPT LANGUAGE='php'> echo "<p>Order processed."; </SCRIPT>
```

Gaya tag ini adalah yang terpanjang dan akan familier jika Anda pernah menggunakan JavaScript atau VBScript. Gaya ini dapat digunakan jika Anda menggunakan editor HTML yang memberi Anda masalah dengan gaya tag lainnya.

- Gaya ASP

```
<% echo "<p>Order processed."; %>
```

Gaya tag ini sama dengan yang digunakan di *Active Server Pages* (ASP). Gaya ini dapat digunakan jika Anda telah mengaktifkan pengaturan konfigurasi asp_tags. Anda mungkin ingin menggunakan gaya tag ini jika Anda menggunakan editor yang ditujukan untuk ASP atau jika Anda sudah memprogram dalam ASP.

**Pernyataan PHP**

Kita memberi tahu interpreter PHP apa yang harus dilakukan dengan meletakkan pernyataan PHP di antara tag pembuka dan penutup. Dalam contoh ini, kita hanya menggunakan satu jenis pernyataan:

```
echo "<p>Order processed.";
```

Seperti yang mungkin sudah Anda duga, penggunaan konstruksi echo memiliki hasil yang sangat sederhana; ia mencetak (atau menggemakan) string yang diteruskannya ke browser. Pada Gambar 1.2, Anda dapat melihat hasilnya berupa teks "*Pesanan diproses*." yang muncul di jendela browser.

Anda akan melihat bahwa titik koma muncul di akhir pernyataan echo. Ini digunakan untuk memisahkan pernyataan dalam PHP seperti titik yang digunakan untuk memisahkan kalimat dalam bahasa Inggris. Jika Anda pernah memprogram dalam C atau Java sebelumnya, Anda akan terbiasa menggunakan titik koma dengan cara ini. Menghilangkan titik koma adalah kesalahan sintaksis umum yang mudah dibuat. Namun, sama mudahnya untuk menemukan dan memperbaikinya.

**Spasi**

Karakter spasi seperti baris baru (*carriage return*), spasi, dan tab dikenal sebagai spasi. Saya akan menggabungkan paragraf di atasm, dan di bawah dan membentuk satu paragraf kohesif yang menjelaskan bagaimana spasi karakter (spasi) diabaikan dalam PHP dan HTML.

Seperti yang mungkin sudah Anda ketahui, browser mengabaikan spasi dalam HTML. Begitu juga dengan mesin PHP. Perhatikan dua fragmen HTML berikut:

```
<h1>Welcome to Bob’s Auto Parts!</h1><p>What would you like to order today?
```

Dan

```
<h1>Welcome to Bob’s
Auto Parts!</h1>
<p>What would you like to order today?
```

Kedua potongan kode HTML ini menghasilkan keluaran yang identik karena keduanya tampak sama di browser. Namun, Anda dapat dan dianjurkan untuk menggunakan spasi di HTML Anda sebagai bantuan bagi manusia untuk meningkatkan keterbacaan kode HTML Anda. Hal yang sama berlaku untuk PHP. Tidak perlu ada spasi di antara pernyataan PHP, tetapi akan membuat kode lebih mudah dibaca jika kita meletakkan setiap pernyataan pada baris terpisah. Misalnya,

```
echo “hello”;
echo “world”;
```

dan

```
echo “hello”;echo “world”;
```

setara, tetapi versi pertama lebih mudah dibaca.

**Komentar**

Komentar persis seperti itu: Komentar dalam kode berfungsi sebagai catatan bagi orang yang membaca kode tersebut. Komentar dapat digunakan untuk menjelaskan tujuan skrip, siapa yang menulisnya, mengapa mereka menulisnya seperti itu, kapan terakhir kali dimodifikasi, dan seterusnya. Anda biasanya akan menemukan komentar di semua skrip PHP kecuali yang paling sederhana. Interpreter PHP akan mengabaikan teks apa pun dalam komentar. Pada dasarnya parser PHP akan melewati komentar yang setara dengan spasi.

PHP mendukung komentar gaya C, C++, dan skrip shell. Ini adalah komentar multibaris gaya C yang mungkin muncul di awal skrip PHP kita:

```
/* Author: Bob Smith
   Last modified: April 10
   This script processes the customer orders.
*/
```

Komentar multiline harus dimulai dengan /* dan diakhiri dengan */. Seperti dalam C, komentar multiline tidak dapat disarangkan.

Anda juga dapat menggunakan komentar satu baris, baik dalam gaya C++:

```
echo “<p>Order processed.”; // Start printing order
```

atau dalam gaya skrip shell:

```
echo “<p>Order processed.”; # Start printing order
```

Dengan kedua gaya ini, semua yang ada setelah simbol komentar (# atau //) adalah komentar hingga kita mencapai akhir baris atau tag PHP penutup, mana pun yang lebih dulu.

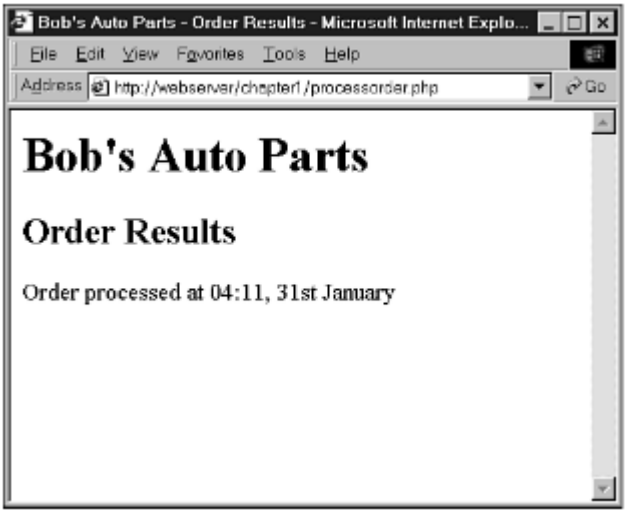
## 1.3 MENAMBAHKAN KONTEN DINAMIS

Sejauh ini, kita belum menggunakan PHP untuk melakukan apa pun yang tidak dapat kita lakukan dengan HTML biasa. Alasan utama menggunakan bahasa skrip sisi server adalah untuk dapat menyediakan konten dinamis bagi pengguna situs. Ini adalah aplikasi penting karena konten yang berubah sesuai dengan kebutuhan pengguna atau seiring waktu akan membuat pengunjung kembali ke situs. PHP memungkinkan kita melakukan ini dengan mudah.

Mari kita mulai dengan contoh sederhana. Ganti PHP di `processorder.php` dengan kode berikut:

```
<?
  echo “<p>Order processed at “;
  echo date(“H:i, jS F”);
  echo “<br>”;
?>
```

`Dalam kode ini, kami menggunakan fungsi `date()` bawaan PHP untuk memberi tahu pelanggan tanggal dan waktu saat pesanannya diproses. Tanggal dan waktu ini akan berbeda setiap kali skrip dijalankan. Output dari skrip yang dijalankan pada satu waktu ditunjukkan pada Gambar 1.3.




**Gambar 1.3 Fungsi `Date()` Php Mengembalikan String Tanggal Yang Diformat.**

**Memanggil Fungsi**

Lihat panggilan ke date(). Ini adalah bentuk umum yang diambil oleh pemanggilan fungsi. PHP memiliki pustaka fungsi yang luas yang dapat Anda gunakan saat mengembangkan aplikasi Web. Sebagian besar fungsi ini perlu memiliki beberapa data yang diteruskan ke mereka dan mengembalikan beberapa data.
Lihat pemanggilan fungsi:

```
date(“H:i, jS F”)
```

Perhatikan bahwa kita meneruskan string (data teks) ke fungsi di dalam sepasang tanda kurung. Ini disebut argumen atau parameter fungsi. Argumen ini adalah input yang digunakan oleh fungsi untuk mengeluarkan beberapa hasil tertentu.

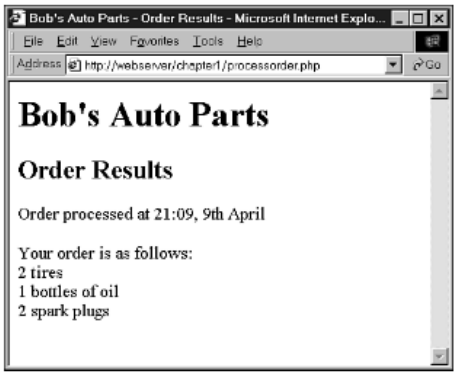
**Fungsi `date()`**

Fungsi `date()` mengharapkan argumen yang Anda berikan berupa string format, yang mewakili gaya keluaran yang Anda inginkan. Setiap huruf dalam string mewakili satu bagian dari tanggal dan waktu. H adalah jam dalam format dua puluh jam, i adalah menit dengan nol di depan jika diperlukan, j adalah hari dalam bulan tanpa nol di depan, S mewakili sufiks ordinal (dalam kasus ini "th"), dan F adalah tahun dalam format empat digit.

## 1.4 MENGAKSES VARIABEL FORMULIR

Tujuan utama penggunaan formulir pesanan adalah untuk mengumpulkan pesanan pelanggan. Mendapatkan detail tentang apa yang diketik pelanggan sangat mudah dalam PHP. Dalam skrip PHP Anda, Anda dapat mengakses setiap bidang formulir sebagai variabel dengan nama yang sama dengan bidang formulir. Mari kita lihat contohnya.
Mulailah dengan menambahkan baris berikut di bagian bawah skrip PHP Anda:

```
echo “<p>Your order is as follows:”;
echo “<br>”;
echo $tireqty.” tires<br>”;
echo $oilqty.” bottles of oil<br>”;
echo $sparkqty.” spark plugs<br>”;
```




**Gambar 1.4 Variabel Formulir Yang Diketik Oleh Pengguna Dapat Diakses Dengan Mudah Di Processorder.Php.**

Jika Anda menyegarkan jendela browser, output skrip akan menyerupai apa yang ditampilkan pada Gambar 1.4. Nilai aktual yang ditampilkan tentu saja akan bergantung pada apa yang Anda ketik pada formulir. Beberapa hal menarik yang perlu diperhatikan dalam contoh ini dibahas di subbagian berikut.

**Variabel Formulir**

Data dari skrip akan berakhir di variabel PHP. Anda dapat mengenali nama variabel di PHP karena semuanya dimulai dengan tanda dolar ($). (Melupakan tanda dolar adalah kesalahan pemrograman yang umum.) Ada dua cara mengakses data formulir melalui variabel.

Dalam contoh ini, dan di seluruh buku ini, kami telah menggunakan gaya singkat untuk merujuk variabel formulir. Dalam kasus ini, Anda akan melihat bahwa nama variabel yang kami gunakan dalam skrip ini sama dengan yang ada di formulir HTML. Hal ini selalu terjadi dengan gaya singkat. Anda tidak perlu mendeklarasikan variabel dalam skrip Anda karena variabel tersebut diteruskan ke skrip Anda, pada dasarnya seperti argumen yang diteruskan ke suatu fungsi. Jika Anda menggunakan gaya ini, Anda dapat, misalnya, mulai menggunakan variabel seperti `$tireqty` seperti yang telah kami lakukan sebelumnya.

Gaya kedua adalah mengambil variabel formulir dari salah satu dari dua larik yang disimpan di `$HTTP_POST_VARS` dan `$HTTP_GET_VARS`. Salah satu larik ini akan menyimpan detail semua variabel formulir. Larik mana yang digunakan bergantung pada apakah metode yang digunakan untuk mengirimkan formulir adalah POST atau GET. Dengan menggunakan gaya ini untuk mengakses data yang diketik ke dalam bidang formulir tireqty pada contoh sebelumnya, Anda akan menggunakan ekspresi

```
$HTTP_POST_VARS[“tireqty”]
```

Anda hanya dapat menggunakan gaya pendek jika Anda telah menyetel direktif register_globals di berkas php.ini Anda ke "Aktif". Ini adalah setelan default di berkas php.ini biasa. Jika Anda ingin register_globals disetel ke "Nonaktif", Anda harus menggunakan gaya kedua. Anda juga perlu menyetel direktif `track_vars` ke "Aktif".

Gaya yang lebih panjang akan berjalan lebih cepat dan menghindari pembuatan variabel secara otomatis yang mungkin tidak diperlukan. Namun, gaya yang lebih pendek lebih mudah dibaca dan digunakan dan sama seperti pada versi PHP sebelumnya. Kedua metode ini mirip dengan yang digunakan dalam bahasa skrip lain seperti Perl, dan mungkin tampak familier. Anda mungkin telah memperhatikan bahwa kami tidak, pada tahap ini, memeriksa konten variabel untuk memastikan bahwa data yang masuk akal telah dimasukkan di setiap bidang formulir. Coba masukkan data yang salah secara sengaja dan amati apa yang terjadi. Setelah Anda membaca sisa bab ini, Anda mungkin ingin mencoba menambahkan beberapa validasi data ke skrip ini.

**Penggabungan String**

Dalam skrip, kami menggunakan echo untuk mencetak nilai yang diketik pengguna di setiap kolom formulir, diikuti oleh beberapa teks penjelasan. Jika Anda perhatikan dengan

saksama pernyataan echo, Anda akan melihat bahwa nama variabel dan teks berikut memiliki titik (.) di antaranya, seperti ini:

```
echo $tireqty.” tires<br>”;
```

Ini adalah operator penggabungan string dan digunakan untuk menambahkan string (potongan teks) bersama-sama. Anda akan sering menggunakannya saat mengirim output ke browser dengan echo. Ini digunakan untuk menghindari keharusan menulis beberapa perintah echo.
Anda juga dapat menulis

```
echo “$tireqty tires<br>”;
```

Ini setara dengan pernyataan pertama. Format mana pun valid, dan yang mana yang Anda gunakan adalah masalah selera pribadi.

**Variabel dan Literal**

Variabel dan string yang kita gabungkan bersama dalam setiap pernyataan gema adalah jenis hal yang berbeda. Variabel adalah simbol untuk data. String adalah data itu sendiri. Ketika kita menggunakan sepotong data mentah dalam program seperti ini, kita menyebutnya literal untuk membedakannya dari variabel. `$tireqty` adalah variabel, simbol yang mewakili data yang diketik pelanggan. Di sisi lain, "tres" adalah literal. Itu dapat diambil pada nilai nominal. Yah, hampir. Ingat contoh kedua sebelumnya? PHP mengganti nama variabel `$tireqty` dalam string dengan nilai yang disimpan dalam variabel. Sebenarnya ada dua jenis string dalam PHP satu dengan tanda kutip ganda dan satu dengan tanda kutip tunggal. PHP akan mencoba dan mengevaluasi string dalam tanda kutip ganda, menghasilkan perilaku yang kita lihat sebelumnya. String dengan tanda kutip tunggal akan diperlakukan sebagai literal sejati.

**Pengidentifikasi**

Pengidentifikasi adalah nama variabel. (Nama fungsi dan kelas juga merupakan pengidentifikasi kita akan membahas fungsi dan kelas di Bab 5 dan 6.) Ada beberapa aturan sederhana tentang pengidentifikasi:

- ✓ Pengidentifikasi dapat memiliki panjang apa pun dan dapat terdiri dari huruf, angka, garis bawah, dan tanda dolar. Namun, Anda harus berhati-hati saat menggunakan tanda dolar dalam pengidentifikasi. Anda akan melihat alasannya di bagian yang disebut, “Variabel Variabel.”
- ✓ Pengidentifikasi tidak dapat dimulai dengan angka.
- ✓ Dalam PHP, pengidentifikasi peka huruf besar/kecil. `$tireqty` tidak sama dengan `$TireQty`. Mencoba menggunakan keduanya secara bergantian merupakan kesalahan pemrograman yang umum. Fungsi bawaan PHP merupakan pengecualian terhadap aturan ini namanya dapat digunakan dalam huruf besar/kecil apa pun.
- ✓ Pengidentifikasi untuk variabel dapat memiliki nama yang sama dengan fungsi bawaan. Namun, hal ini membingungkan dan harus dihindari. Selain itu, Anda tidak dapat membuat fungsi dengan pengenal yang sama dengan fungsi bawaan.

**Variabel yang Dideklarasikan Pengguna**

Anda dapat mendeklarasikan dan menggunakan variabel Anda sendiri selain variabel yang diberikan dari formulir HTML. Salah satu fitur PHP adalah Anda tidak perlu mendeklarasikan variabel sebelum menggunakannya. Variabel akan dibuat saat Anda pertama kali menetapkan nilai padanya lihat bagian berikutnya untuk detailnya.

**Menetapkan Nilai ke Variabel**

Anda menetapkan nilai ke variabel menggunakan operator penugasan, =. Di situs Bob, kita ingin menghitung jumlah total barang yang dipesan dan jumlah total yang harus dibayarkan. Kita dapat membuat dua variabel untuk menyimpan angka-angka ini. Untuk memulainya, kita akan menginisialisasi setiap variabel ini menjadi nol.

Tambahkan baris ini di bagian bawah skrip PHP Anda:

```
$totalqty = 0;
$totalamount = 0.00;
```

Masing-masing dari kedua baris ini membuat sebuah variabel dan menetapkan nilai literal padanya. Anda juga dapat menetapkan nilai variabel ke variabel, misalnya:

```
$totalqty = 0;
$totalamount = $totalqty;
```

**Tipe Variabel**

Tipe variabel mengacu pada jenis data yang disimpan di dalamnya.

**Tipe Data PHP**

PHP mendukung tipe data berikut:

- ☑ Integer: Digunakan untuk bilangan bulat
- ☑ Double: Digunakan untuk bilangan riil
- ☑ String: Digunakan untuk string karakter
- ☑ Array: Digunakan untuk menyimpan beberapa item data dengan tipe yang sama (lihat Bab 3, “Menggunakan Array”)
- ☑ Object: Digunakan untuk menyimpan instance kelas (lihat Bab 6, “PHP Berorientasi Objek”)

PHP juga mendukung tipe pdfdoc dan pdfinfo jika telah diinstal dengan dukungan PDF (Portable Document Format).

**Kekuatan Tipe**

PHP adalah bahasa yang diketik dengan sangat lemah. Dalam kebanyakan bahasa pemrograman, variabel hanya dapat menampung satu tipe data, dan tipe tersebut harus dideklarasikan sebelum variabel dapat digunakan, seperti dalam C. Dalam PHP, tipe variabel ditentukan oleh nilai yang ditetapkan padanya. Misalnya, ketika kita membuat `$totalqty` dan `$totalamount`, tipe awal mereka ditentukan, sebagai berikut:

```
$totalqty=0;
```

```
$totalamount = 0.00;
```

Karena kita menetapkan 0, sebuah integer, ke `$totalqty`, maka ini sekarang menjadi variabel tipe integer. Demikian pula, `$totalamount` sekarang bertipe double.
Anehnya, kita sekarang dapat menambahkan baris ke skrip kita sebagai berikut:

```
$totalamount = “Hello”;
```

Variabel `$totalamount` kemudian akan bertipe string. PHP mengubah tipe variabel sesuai dengan apa yang disimpan di dalamnya pada waktu tertentu. Kemampuan untuk mengubah tipe secara transparan saat itu juga bisa sangat berguna. Ingatlah bahwa PHP "secara otomatis" mengetahui tipe data apa yang Anda masukkan ke dalam variabel Anda. Ia akan mengembalikan data dengan tipe data yang sama setelah Anda mengambilnya dari variabel.

**Pengecoran Tipe**

Anda dapat berpura-pura bahwa suatu variabel atau nilai bertipe berbeda dengan menggunakan pengecoran tipe. Ini bekerja identik dengan cara kerjanya di C. Anda cukup meletakkan tipe sementara dalam tanda kurung di depan variabel yang ingin Anda cor. Misalnya, kita dapat mendeklarasikan dua variabel di atas menggunakan pengecoran.

```
$totalqty = 0;
$totalamount = (double)$totalqty;
```

Baris kedua berarti “Ambil nilai yang disimpan dalam `$totalqty`, tafsirkan sebagai double, dan simpan dalam `$totalamount`.” Variabel `$totalamount` akan bertipe double. Variabel cast tidak mengubah tipe, jadi `$totalqty` tetap bertipe integer.

**Variabel Variabel**

PHP menyediakan satu tipe variabel lainnya variabel variabel. Variabel variabel memungkinkan kita mengubah nama variabel secara dinamis. (Seperti yang Anda lihat, PHP memberikan banyak kebebasan di area ini semua bahasa akan memungkinkan Anda mengubah nilai variabel, tetapi tidak banyak yang akan memungkinkan Anda mengubah tipe variabel, dan bahkan lebih sedikit yang akan memungkinkan Anda mengubah nama variabel.) Cara kerjanya adalah dengan menggunakan nilai satu variabel sebagai nama variabel lain. Misalnya, kita dapat mengatur

```
$varname = “tireqty”;
```

Kita kemudian dapat menggunakan `$$varname` sebagai ganti `$tireqty`. Misalnya, kita dapat mengatur nilai

```
$tireqty:
$$varname = 5;
```

Ini sama persis dengan

```
$tireqty = 5;
```

Ini mungkin tampak agak tidak jelas, tetapi kita akan membahasnya lagi nanti. Daripada harus mencantumkan dan menggunakan setiap variabel formulir secara terpisah, kita dapat menggunakan loop dan variabel untuk memproses semuanya secara otomatis. Ada contoh yang mengilustrasikan hal ini di bagian tentang loop for.

**Konstanta**

Seperti yang Anda lihat sebelumnya, kita dapat mengubah nilai yang disimpan dalam variabel. Kita juga dapat mendeklarasikan konstanta. Konstanta menyimpan nilai seperti variabel, tetapi nilainya ditetapkan sekali dan kemudian tidak dapat diubah di tempat lain dalam skrip. Dalam contoh aplikasi kita, kita dapat menyimpan harga untuk setiap item yang dijual sebagai konstanta. Anda dapat menentukan konstanta ini menggunakan fungsi define:

```
define(“TIREPRICE”, 100);
define(“OILPRICE”, 10);
define(“SPARKPRICE”, 4);
```

Tambahkan baris kode ini ke skrip Anda. Anda akan melihat bahwa nama-nama konstanta semuanya ditulis dalam huruf kapital. Ini adalah konvensi yang dipinjam dari C yang memudahkan untuk membedakan antara variabel dan konstanta secara sekilas. Konvensi ini tidak diwajibkan tetapi akan membuat kode Anda lebih mudah dibaca dan dikelola.

Sekarang kita memiliki tiga konstanta yang dapat digunakan untuk menghitung total pesanan pelanggan. Satu perbedaan penting antara konstanta dan variabel adalah bahwa ketika Anda merujuk ke sebuah konstanta, tidak ada tanda dolar di depannya. Jika Anda ingin menggunakan nilai dari sebuah konstanta, gunakan saja namanya. Misalnya, untuk menggunakan salah satu konstanta yang baru saja kita buat, kita dapat mengetik:

```
echo TIREPRICE;
```

Selain konstanta yang Anda definisikan, PHP juga menetapkan sejumlah besar konstanta miliknya sendiri. Cara mudah untuk mendapatkan gambaran umum tentang konstanta ini adalah dengan menjalankan perintah `phpinfo()`:

```
phpinfo();
```

Ini akan menyediakan daftar variabel dan konstanta PHP yang telah ditetapkan sebelumnya, di antara informasi bermanfaat lainnya. Kami akan membahas beberapa di antaranya seiring berjalannya waktu.

**Cakupan Variabel**

Istilah cakupan mengacu pada tempat-tempat di dalam skrip tempat variabel tertentu terlihat. Tiga jenis cakupan dasar dalam PHP adalah sebagai berikut:

- Variabel global yang dideklarasikan dalam skrip terlihat di seluruh skrip tersebut, tetapi tidak di dalam fungsi.
- Variabel yang digunakan di dalam fungsi bersifat lokal pada fungsi tersebut.
- Variabel yang digunakan di dalam fungsi yang dideklarasikan sebagai global merujuk pada variabel global dengan nama yang sama.

Kami akan membahas cakupan secara lebih terperinci saat membahas fungsi. Untuk saat ini, semua variabel yang kami gunakan akan bersifat global secara default.

## 1.5 OPERATOR

Operator adalah simbol yang dapat Anda gunakan untuk memanipulasi nilai dan variabel dengan melakukan operasi pada keduanya. Kami perlu menggunakan beberapa operator ini untuk menghitung total dan pajak atas pesanan pelanggan. Kami telah menyebutkan dua operator: operator penugasan, =, dan ., operator penggabungan string. Sekarang kita akan melihat daftar lengkapnya. Secara umum, operator dapat mengambil satu, dua, atau tiga argumen, dengan mayoritas mengambil dua. Misalnya, operator penugasan mengambil dua lokasi penyimpanan di sisi kiri simbol =, dan ekspresi di sisi kanan. Argumen ini disebut operan, yaitu, hal-hal yang sedang dioperasikan.

**Operator Aritmatika**

Operator aritmatika sangat mudah digunakan operator ini hanyalah operator matematika biasa. Operator aritmatika ditunjukkan pada Tabel 1.1.

**Tabel 1.1 Operator Aritmatika Php**

| *c* | *Nama* | *Contoh* |
|---|---|---|
| + | Tambah | $a + $b |
| − | Pengurangan | $a − $b |
| * | Perkalian | $a * $b |
| / | Pembagian | $a / $b |
| % | Modulus | $a % $b |

Dengan masing-masing operator ini, kita dapat menyimpan hasil operasi. Misalnya

```
$result = $a + $b;
```

Penjumlahan dan pengurangan bekerja seperti yang Anda harapkan. Hasil dari operator ini adalah untuk menambah atau mengurangi, masing-masing, nilai yang disimpan dalam variabel $a dan $b.

Anda juga dapat menggunakan simbol pengurangan, -, sebagai operator unary (yaitu, operator yang mengambil satu argumen atau operan) untuk menunjukkan angka negatif. Misalnya

```
$a = -1;
```

Perkalian dan pembagian juga berfungsi seperti yang Anda harapkan. Perhatikan penggunaan tanda bintang sebagai operator perkalian, bukan simbol perkalian biasa, dan garis miring sebagai operator pembagian, bukan simbol pembagian biasa.

Operator modulus mengembalikan sisa pembagian variabel `$a` dengan variabel `$b`. Perhatikan potongan kode ini:

```
$a = 27;
$b = 10;
$result = $a%$b
```

Nilai yang disimpan dalam variabel `$result` adalah sisa ketika kita membagi 27 dengan 10; yaitu, 7.

Perlu dicatat bahwa operator aritmatika biasanya diterapkan pada bilangan bulat atau ganda. Jika Anda menerapkannya pada string, PHP akan mencoba dan mengubah string menjadi angka. Jika string berisi "e" atau "E", string akan diubah menjadi ganda; jika tidak, string akan diubah menjadi int. PHP akan mencari digit di awal string dan menggunakannya sebagai nilai jika tidak ada, nilai string akan menjadi nol.

**Operator String**

Kita telah melihat dan menggunakan satu-satunya operator string. Anda dapat menggunakan operator penggabungan string untuk menambahkan dua string dan menghasilkan serta menyimpan hasilnya seperti halnya Anda menggunakan operator penjumlahan untuk menambahkan dua angka.

```
$a = “Bob’s “;
$b = “Auto Parts”;
$result = $a.$b;
```

Variabel `$result` sekarang akan berisi string “`Bob’s Auto Parts`”.

**Operator Penugasan**

Kita telah melihat =, operator penugasan dasar. Selalu rujuk ini sebagai operator penugasan, dan baca sebagai “diatur ke.” Misalnya

```
$totalqty = 0;
```

Ini harus dibaca sebagai "*`$totalqty` ditetapkan ke nol*". Kita akan membahas alasannya saat kita membahas operator perbandingan nanti di bab ini.

**Mengembalikan Nilai dari Penugasan**

Menggunakan operator penugasan akan mengembalikan nilai keseluruhan yang mirip dengan operator lainnya. Jika Anda menulis;

```
$a + $b
```

nilai ekspresi ini adalah hasil dari penambahan variabel $a dan $b. Demikian pula, Anda dapat menulis;

```
$a = 0;
```

Nilai seluruh ekspresi ini adalah nol. Ini memungkinkan Anda melakukan hal-hal seperti

```
$b = 6 + ($a = 5);
```

Ini akan menetapkan nilai variabel $b ke 11. Ini umumnya berlaku untuk penugasan: Nilai seluruh pernyataan penugasan adalah nilai yang ditetapkan ke operan sebelah kiri. Saat menghitung nilai ekspresi, tanda kurung dapat digunakan untuk meningkatkan prioritas subekspresi seperti yang telah kita lakukan di sini. Ini bekerja dengan cara yang persis sama seperti dalam matematika.

**Operator Penugasan Kombinasi**

Selain penugasan sederhana, ada satu set operator penugasan gabungan. Masing-masing operator ini adalah cara singkat untuk melakukan operasi lain pada variabel dan menugaskan hasilnya kembali ke variabel tersebut. Misalnya;

```
$a += 5;
```

Ini setara dengan menulis;

```
$a = $a + 5;
```

Operator penugasan gabungan ada untuk masing-masing operator aritmatika dan untuk operator penggabungan string. Ringkasan semua operator penugasan gabungan dan efeknya ditunjukkan pada Tabel 1.2.

**Tabel 1.2 Operator Penugasan Gabungan Php**

| *Operator* | *Penggunaan* | *Sama/setara dengan* |
|---|---|---|
| += | $a += $b | $a = $a + $b |
| -= | $a -= $b | $a = $a - $b |
| *= | $a *= $b | $a = $a * $b |
| /= | $a /= $b | $a = $a / $b |
| %= | $a %= $b | $a = $a % $b |
| .= | $a .= $b | $a = $a . $b |

**Pra- dan Pasca-Increment dan Decrement**

Operator pra- dan pasca-increment (++) dan decrement (--) mirip dengan operator += dan -=, tetapi dengan beberapa perubahan. Semua operator increment memiliki dua efek mereka increment dan menetapkan nilai. Perhatikan yang berikut:

```
$a=4;
echo ++$a;
```

Baris kedua menggunakan operator pra-increment, disebut demikian karena ++ muncul sebelum $a. Ini memiliki efek pertama, increment $a sebesar 1, dan kedua, mengembalikan nilai yang ditingkatkan. Dalam kasus ini, $a ditingkatkan menjadi 5 dan kemudian nilai 5 dikembalikan dan dicetak. Nilai dari seluruh ekspresi ini adalah 5. (Perhatikan bahwa nilai sebenarnya yang disimpan dalam $a berubah: Kita tidak hanya mengembalikan $a + 1.) Namun, jika ++ berada setelah $a, kita menggunakan operator pasca-peningkatan. Ini memiliki efek yang berbeda. Perhatikan hal berikut:

```
$a=4;
echo $a++;
```

Dalam kasus ini, efeknya terbalik. Pertama, nilai $a dikembalikan dan dicetak, dan kedua, nilainya bertambah. Nilai seluruh ekspresi ini adalah 4. Ini adalah nilai yang akan dicetak. Namun, nilai $a setelah pernyataan ini dieksekusi adalah 5. Seperti yang mungkin dapat Anda tebak, perilakunya serupa untuk operator. Namun, nilai $a dikurangi alih-alih bertambah.

**Referensi**

Tambahan baru dalam PHP 4 adalah operator referensi, & (*ampersand*), yang dapat digunakan bersamaan dengan penugasan. Biasanya, ketika satu variabel ditugaskan ke variabel lain, salinan dibuat dari variabel pertama dan disimpan di tempat lain dalam memori. Misalnya

```
$a = 5;
$b = $a;
```

Baris kode ini membuat salinan kedua dari nilai dalam $a dan menyimpannya di $b. Jika kita kemudian mengubah nilai $a, $b tidak akan berubah:

```
$a = 7; // $b akan tetap 5
```

Anda dapat menghindari membuat salinan dengan menggunakan operator referensi, &. Misalnya

```
$a = 5;
$b = &$a;
$a = 7; // $a dan $b sekarang keduanya 7
```

**Operator Perbandingan**

Operator perbandingan digunakan untuk membandingkan dua nilai. Ekspresi yang menggunakan operator ini mengembalikan salah satu nilai logika benar atau salah tergantung pada hasil perbandingan.

**Operator Persamaan**

Operator perbandingan persamaan, == (dua tanda sama dengan) memungkinkan Anda untuk menguji apakah dua nilai sama. Misalnya, kita dapat menggunakan ekspresi

```
$a == $b
```

untuk menguji apakah nilai yang disimpan dalam $a dan $b sama. Hasil yang dikembalikan oleh ekspresi ini akan benar jika keduanya sama, atau salah jika keduanya tidak sama.

Mudah untuk membingungkan ini dengan =, operator penugasan. Ini akan bekerja tanpa memberikan kesalahan, tetapi umumnya tidak akan memberi Anda hasil yang Anda inginkan. Secara umum, nilai bukan nol dievaluasi menjadi benar dan nilai nol menjadi salah. Katakanlah Anda telah menginisialisasi dua variabel sebagai berikut:

```
$a = 5;
$b = 7;
```

Jika Anda kemudian menguji $a = $b, hasilnya akan benar. Mengapa? Nilai $a = $b adalah nilai yang ditetapkan ke sisi kiri, yang dalam kasus ini adalah 7. Ini adalah nilai bukan nol, jadi ekspresi dievaluasi menjadi benar. Jika Anda bermaksud menguji $a == $b, yang dievaluasi menjadi salah, Anda telah memperkenalkan kesalahan logika dalam kode Anda yang bisa sangat sulit ditemukan. Selalu periksa penggunaan kedua operator ini, dan periksa apakah Anda telah menggunakan yang ingin Anda gunakan. Ini adalah kesalahan yang mudah dilakukan, dan Anda mungkin akan melakukannya berkali-kali dalam karier pemrograman Anda.

**Operator Perbandingan Lainnya**

PHP juga mendukung sejumlah operator perbandingan lainnya. Ringkasan semua operator perbandingan ditunjukkan pada Tabel 1.3. Yang perlu diperhatikan adalah operator identik yang baru, ===, yang diperkenalkan pada PHP 4, yang mengembalikan true hanya jika kedua operannya sama dan bertipe sama.

**Tabel 1.3 Operator Perbandingan Php**

| ***Operator*** | ***Nama*** | ***Penggunaan*** |
|---|---|---|
| == | Sama | $a == $b |
| === | Identik | $a ==== $b |
| != | Tidak Sama | $a != $b |
| <> | Tidak Sama | $a <> $b |
| < | Kurang Dari | $a < $b |
| > | Lebih Besar dari | $a > $b |
| <= | kurang dari atau sama dengan | $a <= $b |
| >= | lebih besar dari atau sama dengan | $a >= $b |

**Operator Logika**

Operator logika digunakan untuk menggabungkan hasil kondisi logika. Misalnya, kita mungkin tertarik pada kasus di mana nilai variabel, `$a`, berada di antara 0 dan 100. Kita perlu menguji kondisi `$a >= 0` dan `$a <= 100`, menggunakan operator `AND`, sebagai berikut

```
$a >= 0 && $a <=100
```

PHP mendukung logika `AND`, `OR`, `XOR` (eksklusif atau), dan `NOT`.

Seperangkat operator logika dan penggunaannya dirangkum dalam Tabel 1.4.

**Tabel 1.4 Operator Logika Php**

| Operator | Nama | Penggunaan | Hasil |
|---|---|---|---|
| ! | NOT | !$b | Mengembalikan true jika $b salah, dan sebaliknya |
| && | AND | $a && $b | Mengembalikan true jika $a dan $b keduanya benar; jika tidak, akan mengembalikan false |
| \|\| | OR | $a \|\| $b | Mengembalikan true jika $a atau $b atau keduanya bernilai true; jika tidak, false |
| And | AND | $a and $b | Sama seperti &&, tetapi dengan prioritas lebih rendah |
| or | OR | $a or $b | Sama seperti \|\|, tetapi dengan prioritas lebih rendah |

Operator and dan or memiliki prioritas lebih rendah daripada operator && dan ||. Kami akan membahas prioritas secara lebih rinci nanti dalam bab ini.

**Operator Bitwise**

Operator bitwise memungkinkan Anda untuk memperlakukan integer sebagai serangkaian bit yang digunakan untuk mewakilinya. Anda mungkin tidak akan menemukan banyak kegunaan untuk ini di PHP, tetapi ringkasan operator bitwise ditunjukkan pada Tabel 1.5.

**Tabel 1.5 Operator Bitwise Php**

| Operator | nama | Penggunaan | hasil |
|---|---|---|---|
| & | Bitwise DAN | $a & $b | Bit yang diatur dalam $a dan $b diatur dalam hasil |
| \| | Bitwise OR | $a \| $b | Bit yang diatur dalam $a atau $b diatur dalam hasil |
| ~ | Bitwise NOT | -$a | Bit yang ditetapkan dalam $a tidak ditetapkan dalam hasil, dan sebaliknya |

| | | | |
|---|---|---|---|
| ^ | Bitwise XOR | $a ^ $b | Bit yang diatur dalam $a atau $b tetapi tidak di keduanya diatur dalam hasil |
| << | Geser ke kiri | $a << $b | Menggeser $a ke kiri $b bit |
| >> | Geser ke kanan | $a >> $b | Menggeser $a ke kanan $b bit |

**Operator Lainnya**

Selain operator yang telah kita bahas sejauh ini, masih ada beberapa operator lainnya. Operator koma, **,** ,digunakan untuk memisahkan argumen fungsi dan daftar item lainnya. Biasanya digunakan secara tidak sengaja. Dua operator khusus, *new* dan `->`, masing-masing digunakan untuk membuat instance kelas dan mengakses anggota kelas. Operator ini akan dibahas secara terperinci di Bab 6. Operator array, [ ], memungkinkan kita mengakses elemen array. Operator ini akan dibahas di Bab 3. Ada tiga operator lainnya yang akan kita bahas secara singkat di sini. Operator Terner, operator ini, `?:`, bekerja dengan cara yang sama seperti di C. Operator ini berbentuk kondisi ? nilai jika benar : nilai jika salah Operator `terner` mirip dengan versi ekspresi dari pernyataan `if-else`, yang dibahas nanti di bab ini.
Contoh sederhananya adalah;

```
($grade > 50 ? “Lulus” : “Gagal”);
```

Ekspresi ini mengevaluasi nilai siswa menjadi “*Lulus*” atau “*Gagal*”.

**Operator Penekan Kesalahan**

Operator penekan kesalahan, @, dapat digunakan di depan ekspresi apa pun, yaitu, apa pun yang menghasilkan atau memiliki nilai. Misalnya

```
$a = @(57/0);
```

Tanpa operator @, baris ini akan menghasilkan peringatan bagi nol (coba saja). Dengan operator yang disertakan, kesalahan akan ditekan. Jika Anda menekan peringatan dengan cara ini, Anda harus menulis beberapa kode penanganan kesalahan untuk memeriksa kapan peringatan telah terjadi. Jika Anda telah menyiapkan PHP dengan fitur track_errors yang diaktifkan, pesan kesalahan akan disimpan dalam variabel global `$php_errormsg` .

**Operator Eksekusi**

Operator eksekusi sebenarnya adalah sepasang operator: sepasang tanda petik terbalik (``). Tanda petik terbalik bukanlah tanda petik tunggal biasanya terletak pada tombol yang sama dengan simbol ~ (tilde) pada papan ketik Anda. PHP akan mencoba mengeksekusi apa pun yang ada di antara tanda petik terbalik sebagai perintah pada baris perintah server. Nilai ekspresi adalah keluaran dari perintah tersebut.

Misalnya, di bawah sistem operasi mirip UNIX, Anda dapat menggunakan 1

```
$out = `ls -la`;
```

```
echo “<pre>”.$out.”</pre>”;
```

atau, setara pada server Windows

```
$out = `dir c:`;
echo “<pre>”.$out.”</pre>”;
```

Salah satu versi ini akan memperoleh daftar direktori dan menyimpannya di `$out`. Kemudian dapat digemakan ke browser atau ditangani dengan cara lain. Ada cara lain untuk menjalankan perintah di server. Kami akan membahasnya di Bab 7, “Berinteraksi dengan Sistem Berkas dan Server.”

**Menggunakan Operator: Menghitung Total Formulir**

Sekarang setelah Anda mengetahui cara menggunakan operator PHP, Anda siap menghitung total dan pajak pada formulir pesanan Bob.

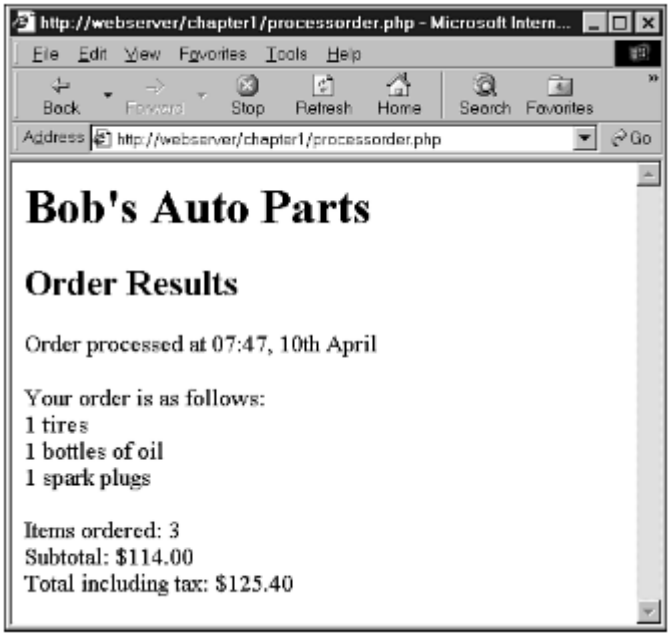
Untuk melakukannya, tambahkan kode berikut di bagian bawah skrip PHP Anda:

```
$totalqty = $tireqty + $oilqty + $sparkqty;
$totalamount = $tireqty * TIREPRICE + $oilqty * OILPRICE + $sparkqty *
  SPARKPRICE;
$totalamount = number_format($totalamount, 2);
echo “<br>\n”;
echo “Items ordered: “.$totalqty.”<br>\n”;
echo “Subtotal: $”.$totalamount.”<br>\n”;
$taxrate = 0.10; // local sales tax is 10%
$totalamount = $totalamount * (1 + $taxrate);
$totalamount = number_format($totalamount, 2);
echo “Total including tax: $”.$totalamount.”<br>\n”;
```

Jika Anda menyegarkan halaman di jendela peramban, Anda akan melihat keluaran yang mirip dengan Gambar 1.5. Seperti yang Anda lihat, kami telah menggunakan beberapa operator dalam bagian kode ini. Kami telah menggunakan operator penjumlahan (+) dan perkalian (*) untuk menghitung jumlah, dan operator penggabungan string (.) untuk menyiapkan keluaran ke peramban.

Kami juga menggunakan fungsi number_format() untuk memformat total sebagai string dengan dua tempat desimal. Ini adalah fungsi dari pustaka Math PHP. Jika Anda mencermati perhitungannya, Anda mungkin bertanya mengapa perhitungan dilakukan dalam urutan tersebut. Misalnya, perhatikan baris ini:

```
$totalamount = $tireqty * TIREPRICE
     + $oilqty * OILPRICE
+ $sparkqty * SPARKPRICE;
```




**Gambar 1.5 Total Pesanan Pelanggan Telah Dihitung, Diformat, Dan Ditampilkan.**

Jumlah total tampaknya benar, tetapi mengapa perkalian dilakukan sebelum penjumlahan? Jawabannya terletak pada urutan prioritas operator, yaitu urutan evaluasinya.

**Urutan Prioritas dan Asosiatif: Mengevaluasi Ekspresi**

Secara umum, operator memiliki urutan prioritas atau urutan evaluasi yang ditetapkan. Operator juga memiliki asosiatif, yaitu urutan evaluasi operator dengan urutan prioritas yang sama. Umumnya, urutannya adalah dari kiri ke kanan (disingkat kiri), dari kanan ke kiri (disingkat kanan), atau tidak relevan. Tabel 1.6 menunjukkan urutan prioritas dan asosiatif operator dalam PHP. Dalam tabel ini, operator dengan urutan prioritas terendah berada di atas, dan urutan prioritas meningkat seiring Anda menelusuri tabel.

**Tabel 1.6 Urutan Prioritas Operator Dalam Php**

| ***Asosiasi*** | ***Operator*** |
|---|---|
| left | , |
| left | or |
| left | xor |
| left | and |
| right | print |
| left | = += -= *= /= .= %= &= \|= ^= ~= <<= >>= |
| Left | ? : |
| Left | \| \| |
| Left | && |
| Left | \| |
| Left | ^ |
| Left | & |

| | |
|---|---|
| n/a | == != === |
| n/a | < <= > >= |
| Left | << >> |
| Left | + - . |
| Left | * / % |
| right | ! ~ ++ -- (int) (double) (string) (array) (object) @ |
| right | [ ] |
| n/a | New |
| n/a | ( ) |

Perhatikan bahwa operator dengan prioritas tertinggi adalah yang belum kita bahas: tanda kurung biasa. Efek dari tanda kurung ini adalah menaikkan prioritas apa pun yang ada di dalamnya. Beginilah cara kita dapat menyiasati aturan prioritas saat diperlukan.
Ingat bagian ini dari contoh terakhir:

```
$totalamount = $totalamount * (1 + $taxrate);
```

Jika kita menulis

```
$totalamount = $totalamount * 1 + $taxrate;
```

operator perkalian, yang memiliki prioritas lebih tinggi daripada operator penjumlahan, akan dilakukan terlebih dahulu, sehingga menghasilkan hasil yang salah. Dengan menggunakan tanda kurung, kita dapat memaksa sub-ekspresi `1 + $taxrate` untuk dievaluasi terlebih dahulu. Anda dapat menggunakan set tanda kurung sebanyak yang Anda suka dalam sebuah ekspresi. Set tanda kurung paling dalam akan dievaluasi terlebih dahulu.

## 1.6 FUNGSI VARIABEL

Sebelum kita meninggalkan dunia variabel dan operator, kita akan melihat fungsi variabel PHP. Ini adalah pustaka fungsi yang memungkinkan kita untuk memanipulasi dan menguji variabel dengan berbagai cara.

**Menguji dan Menetapkan Tipe Variabel**

Sebagian besar fungsi ini berkaitan dengan pengujian tipe suatu fungsi. Dua yang paling umum adalah `gettype()` dan `settype()`. Keduanya memiliki prototipe fungsi berikut; yaitu, inilah yang diharapkan argumen dan apa yang dikembalikannya.

```
string gettype(mixed var);
int settype(string var, string type);
```

Untuk menggunakan `gettype()`, kita memberikannya sebuah variabel. `Gettype` akan menentukan tipe dan mengembalikan string yang berisi nama tipe, atau "tipe tidak dikenal"

jika bukan salah satu tipe standar; yaitu, integer, double, string, array, atau objek. Untuk menggunakan `settype()`, kita berikan variabel yang ingin kita ubah tipenya, dan string yang berisi tipe baru untuk variabel tersebut dari daftar sebelumnya.

Kita dapat menggunakannya sebagai berikut:

```
$a = 56;
echo gettype($a).”<br>”;
settype($a, “double”);
echo gettype($a).”<br>”;
```

Saat `gettype()` dipanggil pertama kali, tipe `$a` adalah integer. Setelah `settype()` dipanggil, tipe akan diubah menjadi double.

PHP juga menyediakan beberapa fungsi pengujian tipe khusus tipe. Masing-masing fungsi ini mengambil variabel sebagai argumen dan mengembalikan true atau false. Fungsi-fungsi tersebut adalah

- `is_array()`
- `is_double(), is_float(), is_real()` (Semua fungsi yang sama)
- `is_long(), is_int(), is_integer()` (Semua fungsi yang sama)
- `is_string()`
- `is_object()`

**Menguji Status Variabel**

PHP memiliki beberapa fungsi untuk menguji status variabel. Yang pertama adalah isset(), yang memiliki prototipe berikut:

```
int isset(mixed var);
```

Fungsi ini mengambil nama variabel sebagai argumen dan mengembalikan true jika ada dan false jika tidak ada.

Anda dapat menghapus variabel dengan menggunakan fungsi pendampingnya, `unset()`. Fungsi ini memiliki prototipe berikut:

```
int unset(mixed var);
```

Fungsi ini membuang variabel yang dilewatkannya dan mengembalikan true. Terakhir ada `empty()`. Fungsi ini memeriksa apakah variabel ada dan memiliki nilai yang tidak kosong, bukan nol, dan mengembalikan true atau false sesuai dengan itu. Fungsi ini memiliki prototipe berikut:

```
int empty(mixed var);
```

Mari kita lihat contoh menggunakan ketiga fungsi ini.

Coba tambahkan kode berikut ke skrip Anda untuk sementara:

```
echo isset($tireqty);
echo isset($nothere);
echo empty($tireqty);
echo empty($nothere);
```

Segarkan halaman untuk melihat hasilnya.

Variabel `$tireqty` harus mengembalikan true dari `isset()` terlepas dari nilai apa yang Anda masukkan atau tidak masukkan di kolom formulir tersebut. Apakah `empty()` atau tidak tergantung pada apa yang Anda masukkan di dalamnya. Variabel `$nothere` tidak ada, jadi akan menghasilkan hasil false dari `isset()` dan hasil true dari `empty()`. Fungsi-fungsi ini dapat berguna untuk memastikan bahwa pengguna mengisi kolom yang sesuai di formulir.

**Menafsirkan Ulang Variabel**

Anda dapat mencapai padanan dari casting variabel dengan memanggil fungsi. Tiga fungsi yang dapat berguna untuk ini adalah

```
int intval(mixed var);
double doubleval(mixed var);
string strval(mixed var);
```

Masing-masing menerima variabel sebagai input dan mengembalikan nilai variabel yang dikonversi ke tipe yang sesuai.

## 1.7 STRUKTUR KONTROL

Struktur kontrol adalah struktur dalam bahasa yang memungkinkan kita untuk mengontrol aliran eksekusi melalui program atau skrip. Anda dapat mengelompokkannya ke dalam struktur kondisional (atau percabangan), dan struktur pengulangan, atau loop. Kami akan mempertimbangkan implementasi spesifik dari masing-masing ini di PHP selanjutnya.

**Membuat Keputusan dengan Kondisional**

Jika kita ingin menanggapi input pengguna kita dengan bijaksana, kode kita harus dapat membuat keputusan. Konstruksi yang memberi tahu program kita untuk membuat keputusan disebut kondisional.

**Pernyataan if**

Kita dapat menggunakan pernyataan if untuk membuat keputusan. Anda harus memberikan pernyataan if kondisi untuk digunakan. Jika kondisinya benar, blok kode berikut akan dieksekusi. Kondisi dalam pernyataan if harus diapit oleh tanda kurung `()`. Misalnya, jika kita memesan tidak ada ban, tidak ada botol oli, dan tidak ada busi dari Bob, itu mungkin karena kita tidak sengaja menekan tombol Kirim. Daripada memberi tahu kita "Pesanan diproses," halaman tersebut dapat memberi kita pesan yang lebih bermanfaat.

Ketika pengunjung tidak memesan barang, kita mungkin ingin mengatakan, "Anda tidak memesan apa pun di halaman sebelumnya!" Kita dapat melakukannya dengan mudah dengan pernyataan if berikut:

```
if( $totalqty == 0 )
echo "Anda tidak memesan apa pun di halaman sebelumnya!<br>";
```

Kondisi yang kita gunakan adalah `$totalqty == 0`. Ingat bahwa operator sama dengan (`==`) berperilaku berbeda dari operator penugasan (`=`). Kondisi `$totalqty == 0` akan benar jika `$totalqty` sama dengan nol. Jika `$totalqty` tidak sama dengan nol, kondisinya akan salah. Ketika kondisinya benar, pernyataan echo akan dieksekusi.

**Blok Kode**

Sering kali kita memiliki lebih dari satu pernyataan yang ingin dieksekusi di dalam pernyataan bersyarat seperti if. Tidak perlu menempatkan pernyataan if baru sebelum setiap pernyataan. Sebaliknya, kita dapat mengelompokkan sejumlah pernyataan bersama sebagai blok. Untuk mendeklarasikan sebuah blok, masukkan dalam kurung kurawal:

```
if( $totalqty == 0 )
{
echo “<font color=red>”;
echo “Anda tidak memesan apa pun di halaman sebelumnya!<br>”; echo “</font>”;
}
```

Tiga baris kode yang diapit kurung kurawal sekarang menjadi blok kode. Jika kondisinya benar, ketiga baris akan dieksekusi. Jika kondisinya salah, ketiga baris akan diabaikan.

**Catatan Tambahan: Membuat Kode Anda Menjorok**

Seperti yang telah disebutkan, PHP tidak peduli bagaimana Anda menata kode Anda. Anda harus membuat kode Anda menjorok agar mudah dibaca. Membuat kode Anda menjorok umumnya digunakan agar kita dapat melihat sekilas baris mana yang hanya akan dieksekusi jika kondisi terpenuhi, pernyataan mana yang dikelompokkan ke dalam blok, dan pernyataan mana yang merupakan bagian dari loop atau fungsi. Anda dapat melihat pada contoh sebelumnya bahwa pernyataan yang bergantung pada pernyataan if dan pernyataan yang membentuk blok tersebut dijorokkan.

**Pernyataan else**

Anda akan sering ingin memutuskan tidak hanya apakah Anda ingin suatu tindakan dilakukan, tetapi juga tindakan mana dari serangkaian tindakan yang mungkin ingin Anda lakukan. Pernyataan else memungkinkan Anda menentukan tindakan alternatif yang akan diambil saat kondisi dalam pernyataan if salah. Kami ingin memperingatkan pelanggan Bob saat mereka tidak memesan apa pun. Di sisi lain, jika mereka memesan, alih-alih memberikan peringatan, kami ingin menunjukkan kepada mereka apa yang mereka pesan.

Jika kami mengatur ulang kode kami dan menambahkan pernyataan else, kami dapat menampilkan peringatan atau ringkasan.

```
if( $totalqty == 0 )
{
  echo “You did not order anything on the previous page!<br>”;
}
  else
       {
         echo $tireqty.” tires<br>”;
         echo $oilqty.” bottles of oil<br>”;
         echo $sparkqty.” spark plugs<br>”;
       }
```

Kita dapat membangun proses logika yang lebih rumit dengan menumpuk pernyataan if di dalam satu sama lain. Dalam kode berikut, ringkasan tidak hanya akan ditampilkan jika kondisi `$totalqty == 0` benar, tetapi juga setiap baris dalam ringkasan hanya akan ditampilkan jika kondisinya sendiri terpenuhi.

```
if( $totalqty == 0)
{
  echo “You did not order anything on the previous page!<br>”;
}
else
{
     if ( $tireqty>0 )
       echo $tireqty.” tires<br>”;
     if ( $oilqty>0 )
       echo $oilqty.” bottles of oil<br>”;
     if ( $sparkqty>0 )
  echo $sparkqty.” spark plugs<br>”;
}
```

**Pernyataan elseif**

Untuk banyak keputusan yang kita buat, ada lebih dari dua pilihan. Kita dapat membuat urutan banyak pilihan menggunakan pernyataan `elseif`. Pernyataan `elseif` adalah kombinasi dari pernyataan `else` dan `if`. Dengan menyediakan urutan kondisi, program dapat memeriksa masing-masing hingga menemukan satu yang benar.

Bob menyediakan diskon untuk pesanan ban dalam jumlah besar. Skema diskon bekerja seperti ini:

- Kurang dari 10 ban dibeli—tidak ada diskon
- 10-49 ban dibeli—diskon 5%
- 50-99 ban dibeli—diskon 10%
- 100 ban atau lebih dibeli—diskon 15%

Kita dapat membuat kode untuk menghitung diskon menggunakan kondisi dan pernyataan `if` dan `elseif`. Kita perlu menggunakan operator `AND (&&)` untuk menggabungkan dua kondisi menjadi satu.

```
if( $tireqty < 10 )
  $discount = 0;
elseif( $tireqty >= 10 && $tireqty <= 49 )
  $discount = 5;
elseif( $tireqty >= 50 && $tireqty <= 99 )
  $discount = 10;
elseif( $tireqty > 100 )
  $discount = 15;
```

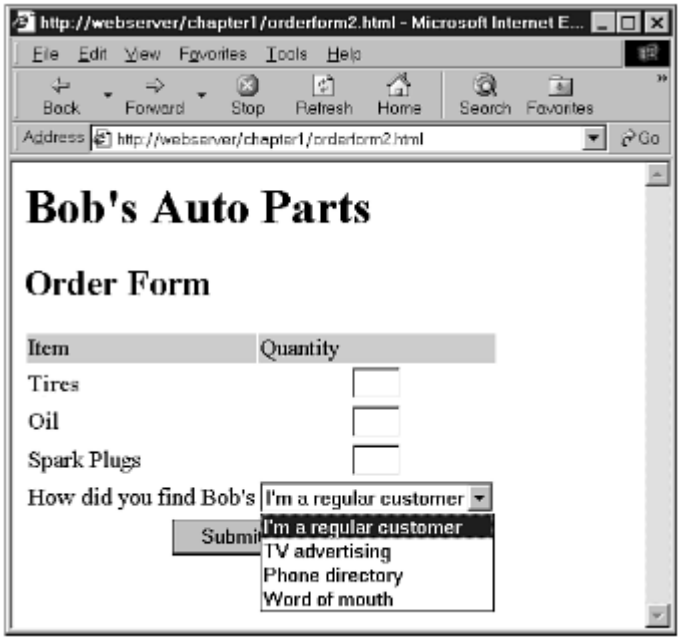
Perhatikan bahwa Anda bebas mengetik `elseif` atau `else if` dengan atau tanpa spasi keduanya benar. Jika Anda akan menulis serangkaian pernyataan `elseif` yang berjenjang, Anda harus menyadari bahwa hanya satu blok atau pernyataan yang akan dieksekusi. Hal itu tidak menjadi masalah dalam contoh ini karena semua kondisi saling eksklusif hanya satu yang dapat bernilai benar pada satu waktu. Jika kita menulis kondisi kita dengan cara yang memungkinkan lebih dari satu bernilai benar pada waktu yang sama, hanya blok atau pernyataan setelah kondisi benar pertama yang akan dieksekusi.

**Pernyataan switch**

Pernyataan switch bekerja dengan cara yang sama dengan pernyataan if, tetapi memungkinkan kondisi untuk mengambil lebih dari dua nilai. Dalam pernyataan if, kondisi dapat berupa benar atau salah. Dalam pernyataan switch, kondisi dapat mengambil sejumlah nilai yang berbeda, selama dievaluasi ke tipe sederhana (integer, string, atau double). Anda perlu memberikan pernyataan case untuk menangani setiap nilai yang ingin Anda tanggapi dan, secara opsional, case default untuk menangani apa pun yang tidak Anda berikan pernyataan case tertentu. Bob ingin tahu bentuk periklanan apa yang berhasil untuknya. Kita dapat menambahkan pertanyaan ke formulir pesanan kita.

Masukkan HTML ini ke dalam formulir pesanan, dan formulir akan menyerupai Gambar 1.6:

```
<tr>
<td>How did you find Bob’s</td>
  <td><select name=”find”>
       <option value = “a”>I’m a regular customer
       <option value = “b”>TV advertising
       <option value = “c”>Phone directory
       <option value = “d”>Word of mouth
  </select>
  </td>
</tr>
```




**Gambar 1.6 Formulir Pesanan Sekarang Menanyakan Kepada Pengunjung Bagaimana Mereka Menemukan Bob's Auto Parts.**

Kode HTML ini telah menambahkan variabel formulir baru yang nilainya akan menjadi "a", "b", "c", atau "d". Kita dapat menangani variabel baru ini dengan serangkaian pernyataan if dan elseif seperti ini:

```
if($find == "a")
  echo "<P>Regular customer.";
elseif($find == "b")
  echo "<P>Customer referred by TV advert.";
elseif($find == "c")
  echo "<P>Customer referred by phone directory.";
elseif($find == "d")
  echo "<P>Customer referred by word of mouth.";
```

Alternatifnya, kita bisa menulis pernyataan switch:

```
switch($find)
{
  case "a" :
       echo "<P>Regular customer.";
       break;
  case "b" :
       echo "<P>Customer referred by TV advert.";
       break;
  case "c" :
       echo "<P>Customer referred by phone directory.";
       break;
```

```
  case “c” :
       echo “<P>Customer referred by word of mouth.”;
       break;
  default :
       echo “<P>We do not know how this customer found us.”;
       break;
}
```

Pernyataan switch berperilaku sedikit berbeda dari pernyataan if atau elseif. Pernyataan if hanya memengaruhi satu pernyataan kecuali Anda sengaja menggunakan kurung kurawal untuk membuat blok pernyataan. `Switch` berperilaku sebaliknya. Ketika `case` dalam `switch` diaktifkan, PHP akan mengeksekusi pernyataan hingga mencapai pernyataan `break.` Tanpa pernyataan `break, switch` akan mengeksekusi semua kode setelah `case` yang bernilai true. Ketika pernyataan `break` tercapai, baris kode berikutnya setelah pernyataan `switch` akan dieksekusi.

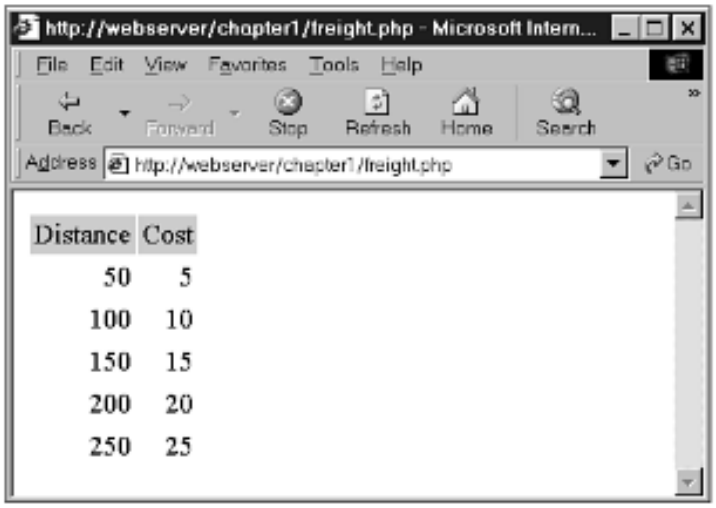
**Membandingkan Kondisi yang Berbeda**

Jika Anda tidak familier dengan pernyataan ini, Anda mungkin bertanya, "Mana yang terbaik?" Itu bukanlah pertanyaan yang dapat kami jawab. Tidak ada yang dapat Anda lakukan dengan satu atau lebih pernyataan `else, elseif,` atau `switch` yang tidak dapat Anda lakukan dengan serangkaian pernyataan `if.` Anda harus mencoba menggunakan kondisi mana pun yang paling mudah dibaca dalam situasi Anda. Anda akan memperoleh gambaran tentang ini dengan pengalaman.

**Iterasi: Mengulangi Tindakan**

Satu hal yang selalu dikuasai komputer dengan sangat baik adalah mengotomatiskan tugas-tugas yang berulang. Jika ada sesuatu yang perlu Anda lakukan dengan cara yang sama beberapa kali, Anda dapat menggunakan loop untuk mengulang beberapa bagian program Anda. Bob menginginkan tabel yang menampilkan biaya pengiriman yang akan ditambahkan ke pesanan pelanggan. Dengan kurir yang digunakan Bob, biaya pengiriman bergantung pada jarak pengiriman paket. Biaya tersebut dapat dihitung dengan rumus sederhana.

Kita ingin tabel pengiriman kita menyerupai tabel pada Gambar 1.7.




**Gambar 1.7 Tabel Ini Menunjukkan Biaya Pengiriman Seiring Bertambahnya Jarak.**

Daftar 1.3 menunjukkan HTML yang menampilkan tabel ini. Anda dapat melihat bahwa tabel ini panjang dan berulang.

**Daftar 1.3 Freight.Html—Html Untuk Tabel Pengiriman Barang Bob**

```
<html>
<body>
<table border = 0 cellpadding = 3>
<tr>
  <td bgcolor = “#CCCCCC” align = center>Distance</td>
  <td bgcolor = “#CCCCCC” align = center>Cost</td>
</tr>
<tr>
  <td align = right>50</td>
  <td align = right>5</td>
</tr>
<tr>
  <td align = right>100</td>
  <td align = right>10</td>
</tr>
<tr>
  <td align = right>150</td>
  <td align = right>15</td>
</tr>
<tr>
  <td align = right>200</td>
  <td align = right>20</td>
</tr>
<tr>
  <td align = right>250</td>
  <td align = right>25</td>
</tr>
```

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

Akan sangat membantu jika, daripada mengharuskan manusia yang mudah bosan yang harus dibayar untuk waktunya mengetik HTML, komputer yang murah dan tak kenal lelah dapat melakukannya. Pernyataan loop memberi tahu PHP untuk mengeksekusi pernyataan atau blok berulang kali.

## 1.8 PERULANGAN WHILE

Jenis loop paling sederhana dalam PHP adalah `loop while`. Seperti pernyataan `if`, `loop` ini bergantung pada suatu kondisi. Perbedaan antara `loop while` dan pernyataan `if` adalah bahwa pernyataan if mengeksekusi blok kode berikut sekali jika kondisinya benar. Loop `while` mengeksekusi blok berulang kali selama kondisinya benar. Anda biasanya menggunakan `loop while` ketika Anda tidak tahu berapa banyak iterasi yang diperlukan untuk membuat kondisinya benar. Jika Anda memerlukan sejumlah iterasi yang tetap, pertimbangkan untuk menggunakan `loop for`.
Struktur dasar `loop while` adalah

```
while( condition ) expression;
```

Perulangan while berikut akan menampilkan angka dari 1 hingga 5.

```
$num = 1; while ($num <= 5 )
{
echo $num.”<BR>”;
$num++;
}
```

Pada awal setiap iterasi, kondisi diuji. Jika kondisi salah, blok tidak akan dieksekusi dan `loop` akan berakhir. Pernyataan berikutnya setelah `loop` akan dieksekusi. Kita dapat menggunakan `loop while` untuk melakukan sesuatu yang lebih berguna, seperti menampilkan tabel pengiriman barang berulang pada Gambar 1.7. Daftar 1.4 menggunakan `loop while` untuk menghasilkan tabel pengiriman barang.

**Daftar 1.4 Freight.Php Menghasilkan Tabel Pengiriman Barang Bob Dengan Php**

```
<body>
<table border = 0 cellpadding = 3>
<tr>
  <td bgcolor = “#CCCCCC” align = center>Distance</td>
  <td bgcolor = “#CCCCCC” align = center>Cost</td>
</tr>
<?
```

```
$distance = 50;
while ($distance <= 250 )
{
  echo “<tr>\n <td align = right>$distance</td>\n”;
  echo “ <td align = right>”. $distance / 10 .”</td>\n</tr>\n”;
  $distance += 50;
}
?>
</table>
</body>
</html>
```

**for Loop**

Cara kita menggunakan `while loop` sebelumnya sangat umum. Kita menetapkan penghitung untuk memulai. Sebelum setiap iterasi, kita menguji penghitung dalam suatu kondisi. Di akhir setiap iterasi, kita memodifikasi penghitung. Kita dapat menulis gaya `loop` ini dalam bentuk yang lebih ringkas menggunakan `for loop.`
Struktur dasar `for loop` adalah:

```
for(ekspresi1; kondisi; ekspresi2) ekspresi3;
```

- *ekspresi1* dieksekusi sekali di awal. Di sini Anda biasanya akan menetapkan nilai awal penghitung.
- *Ekspresi kondisi* diuji sebelum setiap iterasi. Jika ekspresi mengembalikan false, iterasi berhenti. Di sini Anda biasanya akan menguji penghitung terhadap suatu batas.
- *ekspresi2* dieksekusi di akhir setiap iterasi. Di sini Anda biasanya akan menyesuaikan nilai penghitung.
- *ekspresi3* dieksekusi sekali per iterasi. Ekspresi ini biasanya berupa blok kode dan akan berisi sebagian besar kode loop.

Kita dapat menulis ulang contoh while loop pada Daftar 1.4 sebagai for loop. Kode PHP akan menjadi

```
<?
for($distance = 50; $distance <= 250; $distance += 50)
{
  echo “<tr>\n <td align = right>$distance</td>\n”;
  echo “ <td align = right>”. $distance / 10 .”</td>\n</tr>\n”;
}
?>
```

Baik versi while maupun versi for secara fungsional identik. Perulangan for agak lebih ringkas, menghemat dua baris. Kedua jenis perulangan ini setara tidak ada yang lebih baik atau lebih buruk dari yang lain. Dalam situasi tertentu, Anda dapat menggunakan yang mana pun yang menurut Anda lebih intuitif.

Sebagai catatan tambahan, Anda dapat menggabungkan variabel variabel dengan perulangan for untuk mengulang serangkaian bidang formulir yang berulang. Misalnya, jika Anda memiliki bidang formulir dengan nama seperti nama1, nama2, nama3, dan seterusnya, Anda dapat memprosesnya seperti ini:

```
for ($i=1; $i <= $numnames; $i++)
{
  $temp= “name$i”;
  echo $$temp.”<br>”; // or whatever processing you want to do
}
```

Dengan membuat nama-nama variabel secara dinamis, kita dapat mengakses masing-masing bidang secara bergantian.

**Perulangan Do..While**

Jenis perulangan terakhir yang akan kita sebutkan berperilaku sedikit berbeda. Struktur umum pernyataan do..while adalah

```
do
  expression;
while( condition );
```

Perulangan do..while berbeda dari perulangan while karena kondisinya diuji di akhir. Ini berarti bahwa dalam perulangan do..while, pernyataan atau blok dalam perulangan selalu dieksekusi setidaknya satu kali. Bahkan jika kita mengambil contoh ini di mana kondisinya akan salah di awal dan tidak akan pernah menjadi benar, perulangan akan dieksekusi sekali sebelum memeriksa kondisi dan mengakhirinya.

```
$num = 100;
do
{
  echo $num.”<BR>”;
}
while ($num < 1 );
```

**Keluar dari Struktur Kontrol atau Skrip**

Jika Anda ingin berhenti mengeksekusi sepotong kode, ada tiga pendekatan, tergantung pada efek yang ingin Anda capai. Jika Anda ingin berhenti mengeksekusi loop, Anda dapat menggunakan pernyataan break seperti yang dibahas sebelumnya di bagian switch. Jika Anda menggunakan pernyataan break dalam loop, eksekusi skrip akan berlanjut di baris skrip berikutnya setelah loop. Jika Anda ingin melompat ke iterasi loop berikutnya, Anda dapat menggunakan pernyataan continue. Jika Anda ingin menyelesaikan eksekusi seluruh skrip PHP, Anda dapat menggunakan exit. Ini biasanya berguna saat

melakukan pemeriksaan kesalahan. Misalnya, kita dapat memodifikasi contoh sebelumnya sebagai berikut:

```
if( $totalqty == 0)
{
  echo “You did not order anything on the previous page!<br>”;
exit;
}
```

Panggilan untuk keluar menghentikan PHP dari mengeksekusi sisa skrip.

**Berikutnya: Menyimpan Pesanan Pelanggan**

Sekarang Anda telah mengetahui cara menerima dan memanipulasi pesanan pelanggan. Di bab berikutnya, kita akan melihat cara menyimpan pesanan tersebut sehingga dapat diambil dan dipenuhi nanti.

# BAB 2
# MENYIMPAN DAN MENGAMBIL DATA

Sekarang setelah kita mengetahui cara mengakses dan memanipulasi data yang dimasukkan dalam formulir HTML, kita dapat melihat cara menyimpan informasi tersebut untuk digunakan nanti. Dalam kebanyakan kasus, termasuk contoh yang kita lihat di bab sebelumnya, Anda ingin menyimpan data ini dan memuatnya nanti. Dalam kasus kita, kita perlu menulis pesanan pelanggan ke penyimpanan sehingga pesanan tersebut dapat diisi nanti.

Dalam bab ini, kita akan melihat cara menulis pesanan pelanggan dari contoh sebelumnya ke file dan membacanya kembali. Kita juga akan membahas mengapa ini tidak selalu menjadi solusi yang baik. Ketika kita memiliki banyak pesanan, kita harus menggunakan sistem manajemen basis data seperti MySQL. Topik utama yang akan Anda pelajari dalam bab ini meliputi:

- ✪ Menyimpan data untuk nanti
- ✪ Membuka file
- ✪ Membuat dan menulis ke file
- ✪ Menutup file
- ✪ Membaca dari file
- ✪ Mengunci file
- ✪ Menghapus file
- ✪ Fungsi file bermanfaat lainnya
- ✪ Melakukannya dengan cara yang lebih baik: sistem manajemen basis data
- ✪ Bacaan lebih lanjut

## 2.1 MENYIMPAN DATA

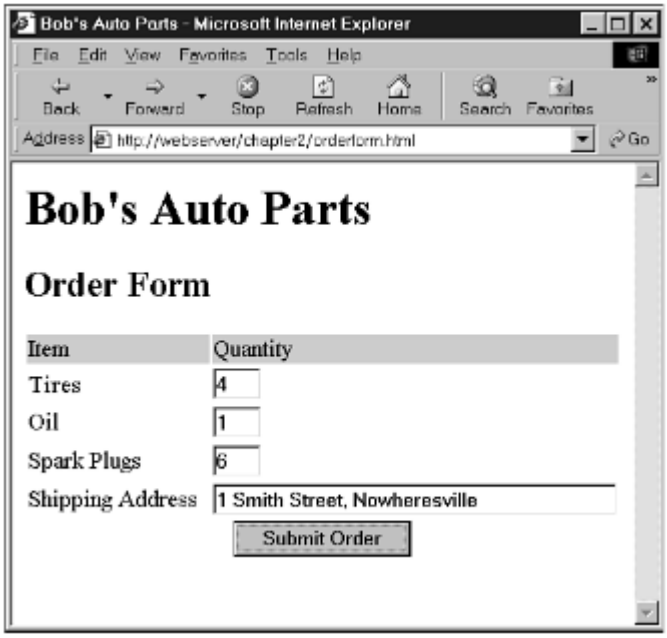
Pada dasarnya ada dua cara untuk menyimpan data: dalam file datar atau dalam basis data. File datar dapat memiliki banyak format, tetapi secara umum, ketika kita merujuk ke file datar, yang kita maksud adalah file teks sederhana. Dalam contoh ini, kita akan menulis pesanan pelanggan ke file teks, satu pesanan per baris. Ini sangat mudah dilakukan, tetapi juga cukup terbatas, seperti yang akan kita lihat nanti dalam bab ini.

Jika Anda berurusan dengan informasi dengan volume yang wajar, Anda mungkin ingin menggunakan basis data sebagai gantinya. Namun, file datar memiliki kegunaannya sendiri dan ada beberapa situasi ketika Anda perlu tahu cara menggunakannya. Menulis dan membaca dari file dalam PHP hampir sama dengan cara yang dilakukan dalam C. Jika Anda pernah melakukan pemrograman C atau skrip shell UNIX, ini semua akan tampak cukup familier bagi Anda.

**Menyimpan dan Mengambil Pesanan Bob**

Dalam bab ini, kita akan menggunakan versi formulir pesanan yang sedikit dimodifikasi yang kita lihat di bab sebelumnya. Kita akan mulai dengan formulir ini dan kode PHP yang kita

tulis untuk memproses data pesanan. Kami telah mengubah formulir tersebut untuk menyertakan cara cepat untuk mendapatkan alamat pengiriman pelanggan. Anda dapat melihat formulir ini pada Gambar 2.1.




**Gambar 2.1 Versi Formulir Pesanan Ini Mendapatkan Alamat Pengiriman Pelanggan.**

Kolom formulir untuk alamat pengiriman disebut alamat. Ini memberi kita variabel yang dapat kita akses sebagai `$address` saat kita memproses formulir dalam PHP, dengan asumsi bahwa kita menggunakan gaya pendek untuk variabel formulir. Ingat bahwa alternatifnya adalah `$HTTP_GET_VARS["address"]` atau `$HTTP_POST_VARS["address"]` jika Anda memilih untuk menggunakan formulir panjang (lihat Bab 1, "Pengetahuan Singkat PHP," untuk detailnya). Kita akan menulis setiap pesanan yang masuk ke file yang sama. Kemudian kita akan membuat antarmuka Web bagi staf Bob untuk melihat pesanan yang telah diterima.

**Tinjauan Umum Pemrosesan Berkas**

Ada tiga langkah untuk menulis data ke berkas:

(1) Buka berkas. Jika berkas belum ada, berkas tersebut perlu dibuat.
(2) Tulis data ke berkas.
(3) Tutup berkas.

Demikian pula, ada tiga langkah untuk membaca data dari berkas:

1. Buka berkas. Jika berkas tidak dapat dibuka (misalnya, jika berkas tidak ada), kita perlu mengenalinya dan keluar dengan baik.
2. Baca data dari berkas.
3. Tutup berkas.

Saat Anda ingin membaca data dari berkas, Anda memiliki pilihan tentang seberapa banyak berkas yang akan dibaca pada satu waktu. Kita akan melihat masing-masing pilihan tersebut secara terperinci. Untuk saat ini, kita akan mulai dari awal dengan membuka berkas.

**Membuka Berkas**

Untuk membuka berkas di PHP, kita menggunakan fungsi `fopen()`. Saat kita membuka berkas, kita perlu menentukan bagaimana kita ingin menggunakannya. Ini dikenal sebagai *mode berkas*.

## 2.2 MODE BERKAS

Sistem operasi pada server perlu mengetahui apa yang ingin Anda lakukan dengan berkas yang sedang Anda buka. Sistem perlu mengetahui apakah berkas dapat dibuka oleh skrip lain saat Anda membukanya, dan untuk mengetahui apakah Anda (pemilik skrip) memiliki izin untuk menggunakannya dengan cara tersebut. Pada dasarnya, mode berkas memberi sistem operasi mekanisme untuk menentukan cara menangani permintaan akses dari orang lain atau skrip dan metode untuk memeriksa apakah Anda memiliki akses dan izin ke berkas tertentu ini.

Ada tiga pilihan yang perlu Anda buat saat membuka berkas:

(1) Anda mungkin ingin membuka berkas untuk dibaca saja, untuk ditulis saja, atau untuk dibaca dan ditulis.
(2) Jika menulis ke berkas, Anda mungkin ingin menimpa konten berkas yang ada atau menambahkan data baru di akhir berkas.
(3) Jika Anda mencoba menulis ke berkas pada sistem yang membedakan antara berkas biner dan teks, Anda mungkin ingin menentukannya.

Fungsi fopen() mendukung kombinasi ketiga opsi ini.

**Menggunakan `fopen()` untuk Membuka File**

Misalnya kita ingin menulis pesanan pelanggan ke file pesanan Bob. Anda dapat membuka file ini untuk menulis dengan perintah berikut:

```
$fp = fopen(“$DOCUMENT_ROOT/../orders/orders.txt”, “w”);
```

Saat fopen dipanggil, ia mengharapkan dua atau tiga parameter. Biasanya Anda akan menggunakan dua, seperti yang ditunjukkan pada baris kode ini. Parameter pertama harus berupa file yang ingin Anda buka. Anda dapat menentukan jalur ke file ini seperti yang telah kita lakukan pada kode sebelumnya file orders.txt kita ada di direktori orders. Kita telah menggunakan variabel bawaan PHP `$DOCUMENT_ROOT`. Variabel ini menunjuk ke dasar pohon dokumen di server Web Anda.

Kami menggunakan `".."` untuk berarti "direktori induk dari direktori `$DOCUMENT_ROOT".` Direktori ini berada di luar pohon dokumen, demi alasan keamanan. Kami tidak ingin berkas ini dapat diakses melalui Web kecuali melalui antarmuka yang kami sediakan. Jalur ini disebut jalur relatif karena menggambarkan posisi dalam sistem berkas yang relatif terhadap `$DOCUMENT_ROOT`. Anda juga dapat menentukan jalur absolut ke berkas tersebut. Ini adalah jalur dari direktori akar (/ pada sistem UNIX dan biasanya C:\ pada sistem Windows).

Di server UNIX kami, ini akan menjadi `/home/book/orders`. Masalah dengan melakukan ini adalah, khususnya jika Anda menghosting situs Anda di server orang lain, jalur absolut mungkin berubah. Kami mempelajarinya dengan cara yang sulit setelah harus mengubah jalur absolut dalam sejumlah besar skrip ketika administrator sistem memutuskan untuk mengubah struktur direktori tanpa pemberitahuan. Jika tidak ada jalur yang ditentukan, berkas akan dibuat atau dicari di direktori yang sama dengan skrip itu sendiri. Ini akan berbeda jika Anda menjalankan PHP melalui semacam Pembungkus CGI dan akan bergantung pada konfigurasi server Anda.

Dalam lingkungan UNIX, garis miring dalam direktori akan berupa garis miring maju (`/`). Jika Anda menggunakan platform Windows, Anda dapat menggunakan garis miring maju atau mundur. Jika Anda menggunakan garis miring mundur, garis miring tersebut harus di-escape (ditandai sebagai karakter khusus) agar fopen dapat memahaminya dengan benar. Untuk meng-escape karakter, Anda cukup menambahkan garis miring mundur tambahan di depannya, seperti yang ditunjukkan berikut ini:

```
$fp = fopen("..\\..\\orders\\orders.txt", "w");
```

Parameter kedua `fopen()` adalah mode file, yang harus berupa string. Ini menentukan apa yang ingin Anda lakukan dengan file tersebut. Dalam kasus ini, kita meneruskan `"w"` ke `fopen()` ini berarti membuka file untuk ditulis. Ringkasan mode file ditunjukkan pada Tabel 2.1.

**Tabel 2.1 Ringkasan Mode File Untuk Fopen**

| ***Mode*** | ***Arti*** |
|---|---|
| r | Mode baca-Buka berkas untuk dibaca, mulai dari awal berkas. |
| r+ | Mode baca-Buka berkas untuk membaca dan menulis, dimulai dari awal berkas. |
| w | Mode penulisan-Buka berkas untuk penulisan, mulai dari awal berkas. Jika berkas sudah ada, hapus konten yang ada. Jika tidak ada, coba buat lagi. |
| w+ | Mode penulisan-Buka berkas untuk penulisan dan pembacaan, mulai dari awal berkas. Jika berkas sudah ada, hapus konten yang ada. Jika tidak ada, coba buat lagi. |
| a | Mode penambahan-Buka berkas untuk penambahan (penulisan) saja, mulai dari akhir konten yang sudah ada, jika ada. Jika tidak ada, coba buat. |
| a+ | Mode penambahan-Buka berkas untuk penambahan (penulisan) dan pembacaan, mulai dari akhir konten yang ada, jika ada. Jika tidak ada, coba buat. |
| b | Mode biner-Digunakan bersama dengan salah satu mode lainnya. Anda mungkin ingin menggunakan ini jika sistem berkas Anda membedakan antara berkas biner dan teks. Sistem Windows membedakan; sistem UNIX tidak. |

Modus berkas yang digunakan dalam contoh kita bergantung pada bagaimana sistem akan digunakan. Kita telah menggunakan “w”, yang hanya akan memungkinkan satu pesanan disimpan dalam berkas. Setiap kali pesanan baru diambil, pesanan sebelumnya akan ditimpa. Ini mungkin tidak terlalu masuk akal, jadi sebaiknya kita menentukan modus penambahan:

```
$fp = fopen(“../../orders/orders.txt”, “a”);
```

Parameter ketiga `fopen()` bersifat opsional. Anda dapat menggunakannya jika ingin mencari berkas di include_path (ditetapkan dalam konfigurasi PHP Anda lihat Lampiran A, “Memasang PHP 4 dan MySQL”). Jika ingin melakukannya, tetapkan parameter ini ke 1. Jika Anda memberi tahu PHP untuk mencari `include_path`, Anda tidak perlu memberikan nama direktori atau jalur:

```
$fp = fopen(“orders.txt”, “a”, 1);
```

Jika `fopen()` berhasil membuka berkas, penunjuk ke berkas dikembalikan dan harus disimpan dalam variabel, dalam kasus ini `$fp`. Anda akan menggunakan variabel ini untuk mengakses berkas saat Anda benar-benar ingin membaca atau menulis ke berkas tersebut.

**Membuka Berkas untuk FTP atau HTTP**

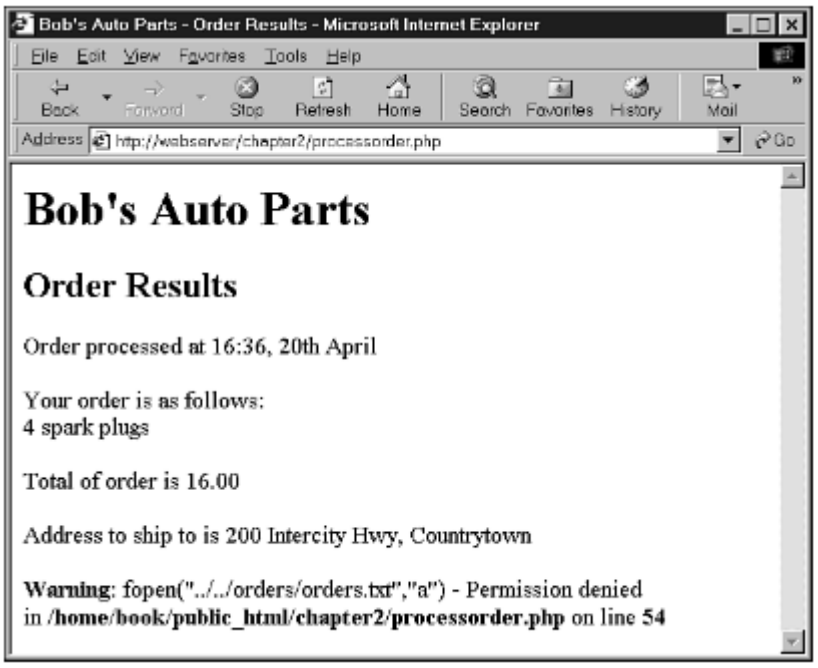
Selain membuka berkas lokal untuk membaca dan menulis, Anda dapat membuka berkas melalui FTP dan HTTP menggunakan `fopen()`. Jika nama berkas yang Anda gunakan dimulai dengan `ftp://`, koneksi FTP mode pasif akan dibuka ke server yang Anda tentukan dan penunjuk ke awal berkas akan dikembalikan. Jika nama berkas yang Anda gunakan dimulai dengan `http://`, koneksi HTTP akan dibuka ke server yang Anda tentukan dan penunjuk ke respons akan dikembalikan. Saat menggunakan mode HTTP, Anda harus menentukan garis miring di akhir nama direktori, seperti yang ditunjukkan berikut ini:

```
http://www.server.com/
```

bukan

**`http://www.server.com`**

Saat Anda menentukan bentuk alamat yang terakhir (tanpa garis miring), server Web biasanya akan menggunakan pengalihan HTTP untuk mengirim Anda ke alamat pertama (dengan garis miring). Cobalah di peramban Anda. Fungsi `fopen()` tidak mendukung pengalihan HTTP, jadi Anda harus menentukan URL yang merujuk ke direktori dengan garis miring di belakangnya. Ingat bahwa nama domain di URL Anda tidak peka huruf besar/kecil, tetapi jalur dan nama file mungkin peka huruf besar/kecil. Masalah saat Membuka File Kesalahan umum yang mungkin Anda buat saat mencoba membuka file adalah mencoba membuka file yang tidak memiliki izin untuk membaca atau menulis. PHP akan memberi Anda peringatan yang mirip dengan yang ditunjukkan pada Gambar 2.2.




**Gambar 2.2 Php Akan Secara Khusus Memperingatkan Anda Saat File Tidak Dapat Dibuka.**

Jika Anda mendapatkan galat ini, Anda perlu memastikan bahwa pengguna yang menjalankan skrip tersebut memiliki izin untuk mengakses file yang Anda coba gunakan. Bergantung pada cara server Anda diatur, skrip tersebut mungkin berjalan sebagai pengguna server Web atau sebagai pemilik direktori tempat skrip tersebut berada.

Pada sebagian besar sistem, skrip akan berjalan sebagai pengguna server Web. Jika skrip Anda berada pada sistem UNIX di direktori `~/public_html/chapter2/`, Anda akan membuat direktori yang dapat ditulisi di seluruh dunia untuk menyimpan pesanan dengan mengetik berikut ini:

```
mkdir ~/orders
chmod 777 ~/orders
```

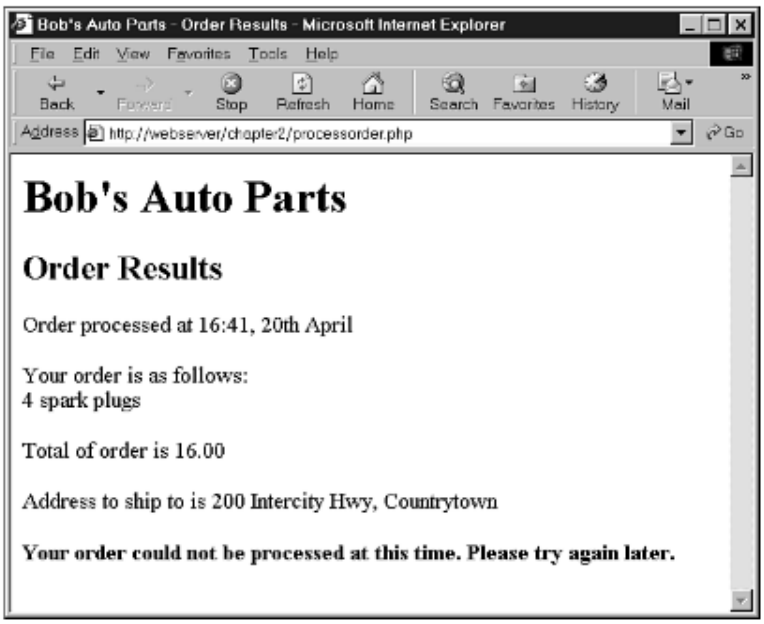
Ingatlah bahwa direktori dan file yang dapat ditulisi oleh siapa saja berbahaya. Anda tidak boleh memiliki direktori yang dapat diakses langsung dari Web sebagai direktori yang dapat ditulisi. Karena alasan ini, direktori pesanan kita berada dua subdirektori di belakang, di atas direktori `public_html`.

Pengaturan izin yang salah mungkin merupakan hal paling umum yang dapat terjadi saat membuka file, tetapi itu bukan satu-satunya hal. Jika file tidak dapat dibuka, Anda benar-benar perlu mengetahui hal ini agar Anda tidak mencoba membaca data dari atau menulis data ke dalamnya. Jika panggilan ke `fopen()` gagal, fungsi akan mengembalikan false. Anda dapat menangani kesalahan dengan cara yang lebih mudah digunakan dengan menekan pesan kesalahan PHP dan memberikan pesan kesalahan Anda sendiri:

```
@ $fp = fopen("$DOCUMENT_ROOT/../orders/orders.txt", "a", 1);
  if (!$fp)
  {

    echo "<p><strong> Your order could not be processed at this time."
      ."Please try again later.</strong></p></body></html>";
  exit;
}
```

Simbol @ di depan panggilan ke `fopen()` memberi tahu PHP untuk menekan kesalahan apa pun yang dihasilkan dari panggilan fungsi. Biasanya merupakan ide yang baik untuk mengetahui kapan sesuatu berjalan salah, tetapi dalam kasus ini kita akan mengatasinya di tempat lain. Perhatikan bahwa simbol @ harus berada di awal baris.

Pernyataan if menguji variabel `$fp` untuk melihat apakah penunjuk file yang valid dikembalikan dari panggilan fopen; jika tidak, pernyataan tersebut akan mencetak pesan kesalahan dan mengakhiri eksekusi skrip. Karena halaman akan berakhir di sini, perhatikan bahwa kita telah menutup tag HTML untuk memberikan HTML yang valid. Output saat menggunakan pendekatan ini ditunjukkan pada Gambar 2.3.




**Gambar 2.3 Menggunakan Pesan Kesalahan Anda Sendiri Sebagai Ganti Pesan Kesalahan Php Dapat Lebih Ramah Pengguna.**

## 2.3 MENULIS KE FILE

Menulis ke file dalam PHP relatif mudah. Anda dapat menggunakan salah satu fungsi `fwrite()` (menulis file) atau `fputs()` (menulis string file); `fputs()` adalah alias untuk `fwrite()`. Kita memanggil `fwrite()` sebagai berikut:

```
fwrite($fp, $outputstring);
```

Ini memberi tahu PHP untuk menulis string yang disimpan dalam `$outputstring` ke file yang ditunjuk oleh `$fp`. Kita akan membahas `fwrite()` secara lebih rinci sebelum kita membahas konten `$outputstring`.

**Parameter untuk fwrite()**

Fungsi `fwrite()` sebenarnya membutuhkan tiga parameter tetapi yang ketiga bersifat opsional. Prototipe untuk `fwrite()` adalah

```
int fputs(int fp, string str, int [length]);
```

Parameter ketiga, length, adalah jumlah maksimum byte yang akan ditulis. Jika parameter ini disediakan, `fwrite()` akan menulis *`string`* ke file yang ditunjuk oleh *`fp`* hingga mencapai akhir *`string`* atau telah menulis *`byte panjang`*, mana saja yang tercapai lebih dulu.

**Format Berkas**

Saat Anda membuat berkas data seperti yang ada dalam contoh kami, format penyimpanan data sepenuhnya terserah Anda. (Namun, jika Anda berencana menggunakan berkas data di aplikasi lain, Anda mungkin harus mengikuti aturan aplikasi tersebut.) Mari buat string yang mewakili satu rekaman di berkas data kita. Kita dapat melakukannya sebagai berikut:

```
$outputstring = $date.”\t”.$tireqty.” tires \t”.$oilqty.” oil\t”
.$sparkqty.” spark plugs\t\$”.$total
.”\t”. $address.”\n”;
```

Dalam contoh sederhana kami, kami menyimpan setiap catatan pesanan pada baris terpisah dalam berkas. Kami memilih untuk menulis satu catatan per baris karena ini memberi kami pemisah catatan sederhana dalam karakter baris baru. Karena baris baru tidak terlihat, kami mewakilinya dengan urutan kontrol `“\n”`. Kami akan menulis bidang data dalam urutan yang sama setiap saat dan memisahkan bidang dengan karakter tab. Sekali lagi, karena karakter tab tidak terlihat, karakter tersebut diwakili oleh urutan kontrol `“\t”`. Anda dapat memilih pembatas yang masuk akal yang mudah dibaca kembali.

Karakter pemisah atau pembatas harus berupa sesuatu yang pasti tidak akan muncul dalam masukan, atau kami harus memproses masukan untuk menghapus atau menghindari setiap contoh pembatas. Kami akan melihat pemrosesan masukan di Bab 4, “Manipulasi String dan Ekspresi Reguler.” Untuk saat ini, kami akan berasumsi bahwa tidak seorang pun akan menempatkan tab ke dalam formulir pesanan kami. Sulit, tetapi bukan tidak mungkin, bagi pengguna untuk menempatkan tab atau baris baru ke dalam bidang masukan HTML satu baris.

Menggunakan pemisah bidang khusus akan memudahkan kita untuk membagi data kembali ke dalam variabel terpisah saat kita membaca kembali data tersebut. Kita akan membahasnya di Bab 3, “Menggunakan Array,” dan Bab 4. Untuk saat ini, kita akan memperlakukan setiap pesanan sebagai string tunggal. Setelah memproses beberapa pesanan, isi berkas akan terlihat seperti contoh yang ditunjukkan pada Daftar 2.1.

**Daftar 2.1 Orders.Txt—Contoh Isi Berkas Pesanan**

```
15:42, 20th April      4 tires      1 oil  6 spark plugs       $434.00
    22 Short St, Smalltown
15:43, 20th April      1 tires      0 oil  0 spark plugs       $100.00
    33 Main Rd, Newtown
15:43, 20th April      0 tires      1 oil  4 spark plugs       $26.00
    127 Acacia St, Springfield
```

**Menutup Berkas**

Setelah selesai menggunakan berkas, Anda perlu menutupnya. Anda harus melakukannya dengan fungsi `fclose()` sebagai berikut:

```
fclose($fp);
```

Fungsi ini akan mengembalikan true jika berkas berhasil ditutup atau false jika tidak. Umumnya, cara ini lebih kecil kemungkinannya untuk salah daripada membuka berkas sejak awal, jadi dalam kasus ini kami memilih untuk tidak mengujinya.

**Membaca dari Berkas**

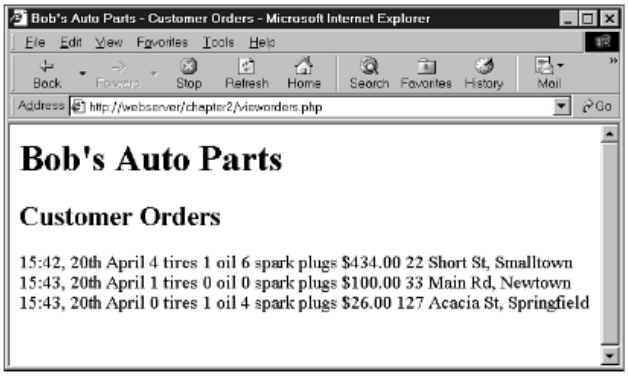
Pelanggan Bob dapat meninggalkan pesanan mereka melalui Web, tetapi jika staf Bob ingin melihat pesanan, mereka harus membuka berkas itu sendiri. Mari kita buat antarmuka Web agar staf Bob dapat membaca berkas dengan mudah. Kode untuk antarmuka ini ditampilkan dalam Daftar 2.2.

**Daftar 2.2 Vieworders.Php — Antarmuka Staf Ke Berkas Pesanan**

```
<html>
<head>
  <title>Bob’s Auto Parts - Customer Orders</title>
</head>
<body>
<h1>Bob’s Auto Parts</h1>
<h2>Customer Orders</h2>
<?

  @ $fp = fopen(“$DOCUMENT_ROOT/../orders/orders.txt”, “r”);

    if (!$fp)
    {
      echo “<p><strong>No orders pending.”
        .”Please try again later.</strong></p></body></html>”;
      exit;
    }

    while (!feof($fp))
    {
      $order= fgets($fp, 100);
```

```
      echo $order.”<br>”;
    }

    fclose($fp);
  ?>
  </body>
  </html>
```

Skrip ini mengikuti urutan yang telah kita bahas sebelumnya: Buka berkas, baca dari berkas, tutup berkas. Output dari skrip ini menggunakan berkas data dari Daftar 2.1 ditunjukkan pada Gambar 2.4.




**Gambar 2.4 Skrip Vieworders.Php Menampilkan Semua Pesanan Yang Saat Ini Ada Di Berkas Orders.Txt Di Jendela Browser.**

Mari kita lihat fungsi-fungsi dalam skrip ini secara terperinci.

**Membuka Berkas untuk Dibaca: `fopen()`**

Sekali lagi, kita membuka berkas menggunakan `fopen()`. Dalam kasus ini, kita membuka berkas hanya untuk dibaca, jadi kita menggunakan mode berkas `"r"`:

```
$fp = fopen(“$DOCUMENT_ROOT/../orders/orders.txt”, “r”);
```

**Mengetahui Kapan Harus Berhenti: feof()**

Dalam contoh ini, kita menggunakan loop while untuk membaca dari berkas hingga akhir berkas tercapai. Loop while menguji akhir berkas menggunakan fungsi `feof()`:

```
while (!feof($fp))
```

Fungsi `feof()` mengambil penunjuk berkas sebagai parameter tunggalnya. Fungsi ini akan mengembalikan true jika penunjuk berkas berada di akhir berkas. Meskipun namanya mungkin tampak aneh, mudah diingat jika Anda tahu bahwa `feof` adalah singkatan dari *File End Of File.* Dalam kasus ini (dan umumnya saat membaca dari sebuah file), kita membaca dari file tersebut hingga EOF tercapai.

**Membaca Baris Sekaligus: `fgets(), fgetss(), dan fgetcsv()`**

Dalam contoh kita, kita menggunakan fungsi `fgets()` untuk membaca dari file:

```
$order= fgets($fp, 100);
```

Fungsi ini digunakan untuk membaca satu baris sekaligus dari sebuah file. Dalam kasus ini, fungsi ini akan membaca hingga menemukan karakter baris baru (`\n`), menemukan EOF, atau telah membaca 99 byte dari file tersebut. Panjang maksimum yang dibaca adalah panjang yang ditentukan dikurangi satu byte.

Ada banyak fungsi berbeda yang dapat digunakan untuk membaca dari berkas. Fungsi `fgets()` berguna saat menangani berkas yang berisi teks biasa yang ingin kita tangani dalam potongan-potongan. Variasi menarik dari `fgets()` adalah `fgetss()`, yang memiliki prototipe berikut:

```
string fgetss(int fp, int length, string [allowable_tags]);
```

Ini sangat mirip dengan `fgets()` kecuali bahwa ia akan menghapus semua tag PHP dan HTML yang ditemukan dalam string. Jika Anda ingin membiarkan tag tertentu, Anda dapat memasukkannya dalam string `allowable_tags`. Anda akan menggunakan `fgetss()` demi keamanan saat membaca berkas yang ditulis oleh orang lain atau berisi masukan pengguna. Mengizinkan kode HTML tanpa batas dalam berkas dapat mengacaukan format yang Anda rencanakan dengan saksama. Mengizinkan PHP tanpa batas dapat memberi pengguna yang berniat jahat kebebasan penuh di server Anda. Fungsi fgetcsv() adalah variasi lain dari fgets(). Prototipe berikut ini:

```
array fgetcsv(int fp, int length, string [delimiter]);
```

Digunakan untuk memecah baris file saat Anda menggunakan karakter pembatas, seperti karakter tab seperti yang kami sarankan sebelumnya atau koma seperti yang umum digunakan oleh spreadsheet dan aplikasi lainnya. Jika kita ingin merekonstruksi variabel dari urutan secara terpisah alih-alih sebagai baris teks, `fgetcsv()` memungkinkan kita melakukannya dengan mudah. Anda memanggilnya dengan cara yang hampir sama seperti Anda memanggil `fgets()`, tetapi Anda meneruskannya dengan pembatas yang Anda gunakan untuk memisahkan bidang. Misalnya

```
$order = fgetcsv($fp, 100, “\t”);
```

akan mengambil baris dari file dan memecahnya di mana pun tab (`\t`) ditemukan. Hasilnya dikembalikan dalam array ( `$order` dalam contoh kode ini). Kami akan membahas array secara lebih rinci di Bab 3. Parameter length harus lebih besar daripada panjang karakter dari baris terpanjang dalam file yang Anda coba baca.

**Membaca Seluruh Berkas: `readfile(), fpassthru(), file()`**

Daripada membaca dari sebuah berkas satu baris dalam satu waktu, kita dapat membaca seluruh berkas sekaligus. Ada tiga cara berbeda untuk melakukannya. Yang pertama menggunakan `readfile()`. Kita dapat mengganti seluruh skrip yang kita tulis sebelumnya dengan satu baris:

```
readfile("$DOCUMENT_ROOT/../orders/orders.txt");
```

Pemanggilan fungsi `readfile()` membuka berkas, menampilkan konten ke keluaran standar (browser), lalu menutup berkas. Prototipe untuk `readfile()` adalah

```
int readfile(string nama_berkas, int [use_include_path]);
```

Parameter opsional kedua menentukan apakah PHP harus mencari file di include_path dan beroperasi dengan cara yang sama seperti di `fopen()`. Fungsi ini mengembalikan jumlah total byte yang dibaca dari file.

Kedua, Anda dapat menggunakan `fpassthru()`. Anda perlu membuka file menggunakan `fopen()` terlebih dahulu. Anda kemudian dapat meneruskan penunjuk file sebagai argumen ke `fpassthru()`, yang akan membuang konten file dari posisi penunjuk ke keluaran standar. Ia menutup file saat selesai. Anda dapat mengganti skrip sebelumnya dengan `fpassthru()` sebagai berikut:

```
$fp=fopen("$DOCUMENT_ROOT/../orders/orders.txt","r");
fpassthru($fp);
```

Fungsi `fpassthru()` mengembalikan true jika pembacaan berhasil dan false jika tidak. Opsi ketiga untuk membaca seluruh file menggunakan fungsi `file()`. Fungsi ini identik dengan `readfile()` kecuali bahwa alih-alih menggemakan file ke keluaran standar, ia mengubahnya menjadi array. Kita akan membahasnya lebih rinci saat kita membahas array di Bab 3. Sekadar referensi, Anda dapat menyebutnya menggunakan

```
$filearray = file($fp);
```

Ini akan membaca seluruh file ke dalam array yang disebut `$filearray`. Setiap baris file disimpan dalam elemen array yang terpisah.

**Membaca Karakter: `fgetc()`**

Pilihan lain untuk pemrosesan file adalah membaca satu karakter pada satu waktu dari sebuah file. Anda dapat melakukannya menggunakan fungsi `fgetc()`. Fungsi ini mengambil penunjuk file sebagai satu-satunya parameter dan mengembalikan karakter berikutnya dalam file. Kita dapat mengganti loop while dalam skrip asli kita dengan loop yang menggunakan

```
fgetc():
while (!feof($fp))
{
  $char = fgetc($fp);
       if (!feof($fp))
    echo ($char==”\n” ? “<br>”: $char);
}
```

Kode ini membaca satu karakter dari file pada satu waktu menggunakan `fgetc()` dan menyimpannya di `$char`, hingga akhir file tercapai. Kemudian, kita melakukan sedikit pemrosesan untuk mengganti karakter akhir baris teks, `\n`, dengan pemisah baris HTML, `<br>`. Ini hanya untuk membersihkan format. Karena browser tidak merender baris baru dalam HTML sebagai baris baru tanpa kode ini, seluruh file akan dicetak pada satu baris. (Coba dan lihat.) Kita menggunakan operator ternary untuk melakukannya dengan rapi.

Efek samping kecil dari penggunaan `fgetc()` alih-alih `fgets()` adalah ia akan mengembalikan karakter EOF sedangkan `fgets()` tidak. Kita perlu menguji `feof()` lagi setelah kita membaca karakter karena kita tidak ingin menggemakan EOF ke browser. Umumnya tidak masuk akal untuk membaca file karakter per karakter kecuali karena suatu alasan kita ingin memprosesnya karakter per karakter. Membaca Panjang Sembarangan: `fread()`

Cara terakhir untuk membaca dari sebuah berkas adalah dengan menggunakan fungsi `fread()` untuk membaca sejumlah byte sembarangan dari berkas. Fungsi ini memiliki prototipe berikut:

```
string fread(int fp, int length);
```

Cara kerjanya adalah dengan membaca hingga panjang byte atau hingga akhir berkas, mana saja yang lebih dulu.

## 2.4 FUNGSI BERKAS

Ada sejumlah fungsi berkas lain yang dapat kita gunakan yang berguna dari waktu ke waktu.

**Memeriksa Apakah Berkas Ada: `file_exists()`**

Jika Anda ingin memeriksa apakah berkas ada tanpa benar-benar membukanya, Anda dapat menggunakan `file_exists()`, sebagai berikut:

```
if (file_exists(“$DOCUMENT_ROOT/../orders/orders.txt”))
  echo “There are orders waiting to be processed.”;
else
  echo “There are currently no orders.”;
```

**Mengetahui Seberapa Besar Sebuah File: `filesize()`**

Anda dapat memeriksa ukuran sebuah file dengan fungsi filesize(). Fungsi ini akan mengembalikan ukuran file dalam byte:

```
echo filesize("$DOCUMENT_ROOT/../orders/orders.txt");
```

Fungsi ini dapat digunakan bersama dengan `fread()` untuk membaca seluruh file (atau sebagian dari file) dalam satu waktu. Kita dapat mengganti seluruh skrip asli kita dengan

```
$fp = fopen("$DOCUMENT_ROOT/../orders/orders.txt", "r");
echo fread( $fp, filesize("$DOCUMENT_ROOT/../orders/orders.txt" ));
fclose( $fp );
```

**Menghapus File: `unlink()`**

Jika Anda ingin menghapus file pesanan setelah pesanan diproses, Anda dapat melakukannya menggunakan `unlink()`. (Tidak ada fungsi yang disebut delete.) Misalnya

```
unlink("$DOCUMENT_ROOT/../orders/orders.txt");
```

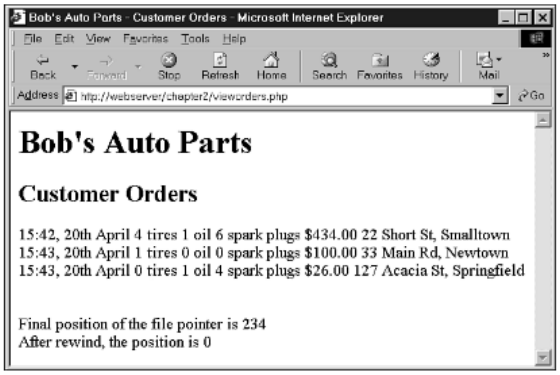
Fungsi ini mengembalikan false jika file tidak dapat dihapus. Ini biasanya terjadi jika izin pada file tidak mencukupi atau jika file tidak ada.

**Menavigasi di Dalam File: `rewind()`, `fseek()`, dan `ftell()`**

Anda dapat memanipulasi dan menemukan posisi penunjuk file di dalam file menggunakan `rewind()`, `fseek()`, dan `ftell()`. Fungsi `rewind()` menyetel ulang penunjuk file ke awal file. Fungsi `ftell()` melaporkan seberapa jauh penunjuk berada dalam file dalam byte. Misalnya, kita dapat menambahkan baris berikut di bagian bawah skrip asli kita (sebelum perintah `fclose()`):

```
echo "Final position of the file pointer is ".(ftell($fp));
echo "<br>";
rewind($fp);
echo "After rewind, the position is ".(ftell($fp));
echo "<br>";
```

Output di browser akan mirip dengan yang ditunjukkan pada Gambar 2.5.




**Gambar 2.5 Setelah Membaca Perintah, Penunjuk Berkas Menunjuk Ke Akhir Berkas, Offset 234 Byte. Panggilan Untuk Memutar Ulang Mengembalikannya Ke Posisi 0, Awal Berkas.**

Fungsi `fseek()` dapat digunakan untuk mengatur penunjuk berkas ke beberapa titik di dalam berkas. Prototipe-nya adalah

```
int fseek(int fp, int offset);
```

Panggilan ke `fseek()` mengatur penunjuk berkas fp pada titik offset byte ke dalam berkas. Fungsi `rewind()` setara dengan memanggil fungsi `fseek()` dengan offset nol. Misalnya, Anda dapat menggunakan `fseek()` untuk menemukan rekaman tengah dalam berkas atau untuk melakukan pencarian biner. Sering kali jika Anda mencapai tingkat kerumitan dalam berkas data di mana Anda perlu melakukan hal-hal semacam ini, hidup Anda akan jauh lebih mudah jika Anda menggunakan basis data.

## 2.5 PENGUNCIAN BERKAS

Bayangkan situasi saat dua pelanggan mencoba memesan produk pada saat yang sama. (Tidak jarang, terutama saat Anda mulai mendapatkan volume lalu lintas apa pun di situs Web.) Bagaimana jika satu pelanggan memanggil `fopen()` dan mulai menulis, lalu pelanggan lain memanggil `fopen()` dan juga mulai menulis? Apa isi akhir berkas tersebut? Apakah akan berupa pesanan pertama diikuti pesanan kedua, atau sebaliknya? Apakah akan berupa satu pesanan atau yang lain? Atau akan berupa sesuatu yang kurang berguna, seperti dua pesanan yang saling terkait? Jawabannya bergantung pada sistem operasi Anda, tetapi sering kali mustahil untuk diketahui.

Untuk menghindari masalah seperti ini, Anda dapat menggunakan penguncian berkas. Ini diimplementasikan dalam PHP menggunakan fungsi `flock()`. Fungsi ini harus dipanggil setelah berkas dibuka, tetapi sebelum data apa pun dibaca dari atau ditulis ke berkas tersebut. Prototipe untuk `flock()` adalah

```
bool flock(int fp, int operation);
```

Anda perlu meneruskannya berupa penunjuk ke berkas yang terbuka dan angka yang mewakili jenis kunci yang Anda perlukan. Ia mengembalikan true jika kunci berhasil diperoleh, dan false jika tidak. Nilai-nilai yang mungkin dari operasi ditunjukkan pada Tabel 2.2.

**Tabel 2.2 Nilai-Nilai Operasi `flock()`**

| *Nilai Operasi* | *Artinya* |
|---|---|
| 1 | Kunci baca. Ini berarti berkas dapat dibagikan dengan pembaca lain. |
| 2 | Kunci penulisan. Ini bersifat eksklusif. File tidak dapat dibagikan. |
| 3 | Lepaskan kunci yang ada. |
| +4 | Menambahkan angka 4 pada operasi akan mencegah pemblokiran saat mencoba mendapatkan kunci. |

Jika Anda akan menggunakan `flock()`, Anda perlu menambahkannya ke semua skrip yang menggunakan file tersebut; jika tidak, itu tidak berguna. Untuk menggunakannya dengan contoh ini, Anda dapat mengubah `processororder.php` sebagai berikut:

```
$fp = fopen(“$DOCUMENT_ROOT/../orders/orders.txt”, “a”, 1);
flock($fp, 2); // lock the file for writing
fwrite($fp, $outputstring);
flock($fp, 3); // release write lock
fclose($fp);
```

Anda juga harus menambahkan kunci ke `vieworders.php:`

```
$fp = fopen(“$DOCUMENT_ROOT /../orders/orders.txt”, “r”);
flock($fp, 1); // lock file for reading
// read from the file
flock($fp, 3); // release read lock
fclose($fp);
```

Kode kita sekarang lebih tangguh, tetapi masih belum sempurna. Bagaimana jika dua skrip mencoba memperoleh kunci pada saat yang sama? Ini akan mengakibatkan kondisi persaingan, di mana proses bersaing untuk mendapatkan kunci tetapi tidak pasti mana yang akan berhasil, yang dapat menyebabkan lebih banyak masalah. Kita dapat melakukannya dengan lebih baik dengan menggunakan DBMS.

**Sistem Manajemen Basis Data**

Sejauh ini semua contoh yang telah kita lihat menggunakan berkas datar. Di bagian berikutnya dari buku ini, kita akan melihat bagaimana Anda dapat menggunakan MySQL, sistem manajemen basis data relasional, sebagai gantinya. Anda mungkin bertanya, "Mengapa saya harus repot-repot?"

**Masalah dengan Penggunaan Berkas Datar**

Ada sejumlah masalah dalam bekerja dengan berkas datar:

- ☑ Ketika berkas menjadi besar, berkas tersebut dapat bekerja sangat lambat.
- ☑ Mencari rekaman atau kelompok rekaman tertentu dalam berkas datar sulit dilakukan. Jika rekamannya berurutan, Anda dapat menggunakan semacam pencarian biner bersama dengan rekaman dengan lebar tetap untuk mencari bidang kunci. Jika Anda ingin menemukan pola informasi (misalnya, Anda ingin menemukan semua pelanggan yang tinggal di Smalltown), Anda harus membaca setiap catatan dan memeriksanya satu per satu.
- ☑ Berurusan dengan akses bersamaan dapat menjadi masalah. Kita telah melihat bagaimana Anda dapat mengunci file, tetapi ini dapat menyebabkan kondisi persaingan yang telah kita bahas sebelumnya. Ini juga dapat menyebabkan kemacetan. Dengan lalu lintas yang cukup banyak di situs, sekelompok besar pengguna mungkin menunggu file dibuka sebelum mereka dapat memesan. Jika menunggu terlalu lama, orang akan pergi ke tempat lain untuk membeli.
- ☑ Semua pemrosesan file yang telah kita lihat sejauh ini berurusan dengan file menggunakan pemrosesan berurutan yaitu, kita mulai dari awal file dan membaca hingga akhir. Jika kita ingin memasukkan catatan ke dalam atau menghapus catatan dari tengah file (akses acak), ini bisa jadi sulit Anda akhirnya membaca seluruh file ke dalam memori, membuat perubahan, dan menulis seluruh file lagi. Dengan file data yang besar, ini menjadi beban yang signifikan.
- ☑ Di luar batasan yang ditawarkan oleh izin file, tidak ada cara mudah untuk menerapkan berbagai tingkat akses ke data.

**Bagaimana RDBMS Memecahkan Masalah Ini**

Sistem manajemen basis data relasional mengatasi semua masalah ini:

- ➲ RDBMS dapat menyediakan akses data yang lebih cepat daripada file datar. Dan MySQL, sistem basis data yang kami gunakan dalam buku ini, memiliki beberapa tolok ukur tercepat dari semua RDBMS.
- ➲ RDBMS dapat dengan mudah dikueri untuk mengekstrak kumpulan data yang sesuai dengan kriteria tertentu.
- ➲ RDBMS memiliki mekanisme bawaan untuk menangani akses bersamaan sehingga Anda sebagai programmer tidak perlu mengkhawatirkannya.
- ➲ RDBMS menyediakan akses acak ke data Anda.
- ➲ RDBMS memiliki sistem hak istimewa bawaan. MySQL memiliki kekuatan khusus di area ini.

Mungkin alasan utama untuk menggunakan RDBMS adalah bahwa semua (atau setidaknya sebagian besar) fungsionalitas yang Anda inginkan dalam sistem penyimpanan data telah diimplementasikan. Tentu, Anda dapat menulis pustaka fungsi PHP Anda sendiri, tetapi mengapa harus menciptakan kembali roda? Pada Bagian II buku ini, “Menggunakan MySQL,” kita akan membahas cara kerja basis data relasional secara umum, dan khususnya cara Anda dapat menyiapkan dan menggunakan MySQL untuk membuat situs web yang didukung basis data.

# BAB 3
# MENGGUNAKAN ARRAY

Bab ini menunjukkan kepada Anda cara menggunakan konstruksi pemrograman yang penting array. Variabel yang kita bahas di bab sebelumnya adalah variabel skalar, yang menyimpan satu nilai. Array adalah variabel yang menyimpan satu set atau rangkaian nilai. Satu array dapat memiliki banyak elemen. Setiap elemen dapat menampung satu nilai, seperti teks atau angka, atau array lainnya. Array yang berisi array lainnya dikenal sebagai array multidimensi.

PHP mendukung array berindeks numerik dan asosiatif. Anda mungkin sudah familier dengan array berindeks numerik jika Anda pernah menggunakan bahasa pemrograman, tetapi kecuali Anda menggunakan PHP atau Perl, Anda mungkin belum pernah melihat array asosiatif sebelumnya. Array asosiatif memungkinkan Anda menggunakan nilai yang lebih berguna sebagai indeks. Daripada setiap elemen memiliki indeks numerik, array dapat memiliki kata-kata atau informasi bermakna lainnya.

Kami akan terus mengembangkan contoh Bob's Auto parts menggunakan array untuk bekerja lebih mudah dengan informasi berulang seperti pesanan pelanggan. Demikian pula, kita akan menulis kode yang lebih pendek dan lebih rapi untuk melakukan beberapa hal yang kita lakukan dengan file di bab sebelumnya.

Topik utama yang dibahas dalam bab ini meliputi

- ☑ Apa itu array?
- ☑ Array yang diindeks secara numerik
- ☑ Array asosiatif
- ☑ Array multidimensi
- ☑ Mengurutkan array
- ☑ Bacaan lebih lanjut

## 3.1 MENGENAL ARRAY

Kita telah membahas variabel skalar di Bab 1, "Pengetahuan Singkat PHP." Variabel skalar adalah lokasi bernama untuk menyimpan nilai; demikian pula, array adalah tempat bernama untuk menyimpan sekumpulan nilai, sehingga memungkinkan Anda untuk mengelompokkan skalar umum.

Daftar produk Bob akan menjadi array untuk contoh kita. Pada Gambar 3.1, Anda dapat melihat daftar tiga produk yang disimpan dalam format array dan satu variabel, yang disebut $products, yang menyimpan tiga nilai. (Kita akan membahas cara membuat variabel seperti ini sebentar lagi.)


**Gambar 3.1 Produk Bob Dapat Disimpan Dalam Array.**

Setelah kita memiliki informasi sebagai array, kita dapat melakukan sejumlah hal yang berguna dengannya. Dengan menggunakan konstruksi perulangan dari Bab 1, kita dapat menghemat pekerjaan dengan melakukan tindakan yang sama pada setiap nilai dalam array. Seluruh rangkaian informasi dapat dipindahkan sebagai satu unit. Dengan cara ini, dengan satu baris kode, semua nilai dapat diteruskan ke suatu fungsi. Misalnya, kita mungkin ingin mengurutkan produk berdasarkan abjad. Untuk mencapainya, kita dapat meneruskan seluruh array ke fungsi `sort()` PHP.

Nilai yang disimpan dalam array disebut elemen array. Setiap elemen array memiliki indeks terkait (juga disebut kunci) yang digunakan untuk mengakses elemen tersebut. Array dalam sebagian besar bahasa pemrograman memiliki indeks numerik yang biasanya dimulai dari nol atau satu. PHP mendukung jenis array ini. PHP juga mendukung array asosiatif, yang sudah tidak asing lagi bagi programmer Perl. Array asosiatif dapat memiliki hampir apa saja sebagai indeks array, tetapi biasanya menggunakan string. Kita akan mulai dengan melihat array yang diindeks secara numerik.

**Array Berindeks Numerik**

Array ini didukung di sebagian besar bahasa pemrograman. Dalam PHP, indeks dimulai dari nol secara default, meskipun Anda dapat mengubahnya.

**Menginisialisasi Array Berindeks Numerik**

Untuk membuat array yang ditunjukkan pada Gambar 3.1, gunakan baris kode PHP berikut:

```
$products = array( “Tires”, “Oil”, “Spark Plugs” );
```

Ini akan membuat array yang disebut products yang berisi tiga nilai yang diberikan *”Tires”, “Oil”,* dan *“Spark Plugs”*. Perhatikan bahwa, seperti echo, `array()` sebenarnya adalah konstruksi bahasa dan bukan fungsi.

Bergantung pada konten yang Anda perlukan dalam array, Anda mungkin tidak perlu menginisialisasinya secara manual seperti pada contoh sebelumnya. Jika Anda memiliki data yang Anda perlukan di array lain, Anda cukup menyalin satu array ke array lain menggunakan operator =. Jika Anda menginginkan urutan angka menaik yang disimpan dalam array, Anda dapat menggunakan fungsi `range()` untuk membuat array secara otomatis untuk Anda. Baris kode berikut akan membuat array yang disebut angka dengan elemen berkisar dari 1 hingga 10:

```
$numbers = range(1,10);
```

Jika Anda memiliki informasi yang tersimpan dalam file di disk, Anda dapat memuat konten array langsung dari file tersebut. Kita akan membahasnya nanti di bab ini di bawah judul *"Memuat Array dari File."*

**Mengakses Konten Array**

Untuk mengakses konten variabel, gunakan namanya. Jika variabel tersebut adalah array, akses konten menggunakan nama variabel dan kunci atau indeks. Kunci atau indeks menunjukkan nilai tersimpan mana yang kita akses. Indeks ditempatkan dalam tanda kurung siku setelah nama.

Ketik `$products[0], $products[1], dan $products[2]` untuk menggunakan konten array produk. Elemen nol adalah elemen pertama dalam array. Ini adalah skema penomoran yang sama seperti yang digunakan dalam C, C++, Java, dan sejumlah bahasa lainnya, tetapi mungkin perlu waktu untuk membiasakan diri jika Anda tidak terbiasa dengannya. Seperti halnya variabel lain, konten elemen array diubah dengan menggunakan operator =. Baris berikut akan mengganti elemen pertama dalam array “Tires” dengan “Fuses”.

```
$products[0] = “Fuses”;
```

Baris berikut dapat digunakan untuk menambahkan elemen baru ”Fuse” di akhir array, sehingga kita memiliki total empat elemen:

```
$products[3] = “Fuses”;
```

Untuk menampilkan konten, kita dapat mengetik

```
echo “$products[0] $products[1] $products[2] $products[3]”;
```

Seperti variabel PHP lainnya, array tidak perlu diinisialisasi atau dibuat terlebih dahulu. Array dibuat secara otomatis saat pertama kali Anda menggunakannya. Kode berikut akan membuat array `$products` yang sama:

```
$products[0] = “Tires”;
$products[1] = “Oil”;
$products[2] = “Spark Plugs”;
```

Jika `$products` belum ada, baris pertama akan membuat array baru dengan hanya satu elemen. Baris berikutnya menambahkan nilai ke array.

**Menggunakan Loop untuk Mengakses Array**

Karena array diindeks oleh serangkaian angka, kita dapat menggunakan loop for untuk menampilkan konten dengan lebih mudah:

```
for ( $i = 0; $i<3; $i++ ) echo “$products[$i] “;
```

Loop ini akan memberikan output yang mirip dengan kode sebelumnya, tetapi akan memerlukan lebih sedikit pengetikan daripada menulis kode secara manual untuk bekerja dengan setiap elemen dalam array yang besar. Kemampuan untuk menggunakan loop sederhana untuk mengakses setiap elemen adalah fitur yang bagus dari array yang diindeks secara numerik. Array asosiatif tidak begitu mudah untuk dilalui, tetapi memungkinkan indeks menjadi bermakna.

## 3.2 ARRAY ASOSIATIF

Dalam array products, kita mengizinkan PHP untuk memberikan setiap item indeks default. Ini berarti bahwa item pertama yang kita tambahkan menjadi item 0, item kedua 1, dan seterusnya. PHP juga mendukung array asosiatif. Dalam array asosiatif, kita dapat mengaitkan kunci atau indeks apa pun yang kita inginkan dengan setiap nilai.

**Menginisialisasi Array Asosiatif**

Kode berikut membuat array asosiatif dengan nama produk sebagai kunci dan harga sebagai nilai.

```
$prices = array( “Tires”=>100, “Oil”=>10, “Spark Plugs”=>4 );
```

**Mengakses Elemen Array**

Sekali lagi, kita mengakses konten menggunakan nama variabel dan kunci, sehingga kita dapat mengakses informasi yang telah kita simpan dalam array harga sebagai `$prices[ “Tires” ]`, `$prices[ “Oil” ]`, and `$prices[ “Spark Plugs” ]`. Seperti array berindeks numerik, array asosiatif dapat dibuat dan diinisialisasi satu elemen pada satu waktu.

Kode berikut akan membuat array `$prices` yang sama. Daripada membuat array dengan tiga elemen, versi ini membuat array dengan hanya satu elemen, lalu menambahkan dua elemen lagi.

```
$prices = array( “Tires”=>100 );
$prices[“Oil”] = 10;
$prices[“Spark Plugs”] = 4;
```

Berikut ini adalah kode lain yang sedikit berbeda, tetapi setara. Dalam versi ini, kita tidak membuat array secara eksplisit sama sekali. Array dibuat untuk kita saat kita menambahkan elemen pertama ke dalamnya.

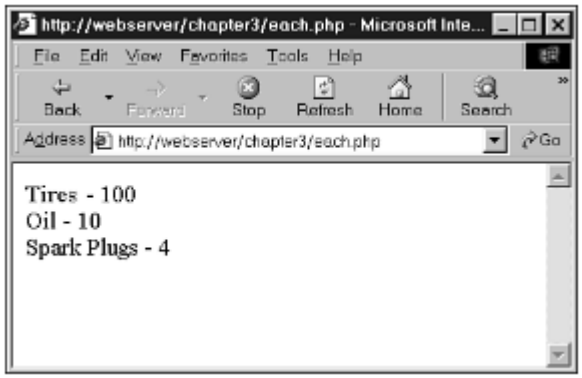
```
$prices = array( “Tires”=>100 );
$prices[“Oil”] = 10;
$prices[“Spark Plugs”] = 4;
```

**Menggunakan Loop dengan `each()` dan `list()`**

Karena indeks dalam array asosiatif ini bukan angka, kita tidak dapat menggunakan penghitung sederhana dalam loop for untuk bekerja dengan array tersebut. Kode berikut mencantumkan konten array `$prices` kita:

```
while( $element = each( $prices ) )
{
  echo $element[ “key” ];
  echo “ - “;
  echo $element[ “value” ];
  echo “<br>”;
}
```

Output dari fragmen skrip ini ditunjukkan pada Gambar 3.2.




**Gambar 3.2 Pernyataan Each Dapat Digunakan Untuk Melakukan Perulangan Melalui Array.**

Pada Bab 1, kita telah membahas perulangan while dan pernyataan echo. Kode sebelumnya menggunakan fungsi `each()`, yang belum pernah kita gunakan sebelumnya. Fungsi ini mengembalikan elemen saat ini dalam array dan menjadikan elemen berikutnya sebagai elemen saat ini. Karena kita memanggil `each()` dalam perulangan while, fungsi ini mengembalikan setiap elemen dalam array secara bergantian dan berhenti saat akhir array tercapai.

Dalam kode ini, variabel `$element` adalah array. Saat kita memanggil `each()`, fungsi ini memberi kita array dengan empat nilai dan empat indeks ke lokasi array. Lokasi key dan 0 berisi kunci elemen saat ini, dan lokasi value dan 1 berisi nilai elemen saat ini. Meskipun tidak ada bedanya yang mana yang Anda pilih, kami telah memilih untuk menggunakan lokasi bernama, daripada yang bernomor. Ada cara yang lebih elegan dan lebih umum untuk melakukan hal yang sama. Fungsi `list()` dapat digunakan untuk membagi array menjadi sejumlah nilai. Kita dapat memisahkan dua nilai yang diberikan fungsi `each()` seperti ini:

```
$list( $product, $price ) = each( $prices );
```

Baris ini menggunakan `each()` untuk mengambil elemen saat ini dari `$prices`, mengembalikannya sebagai array, dan menjadikan elemen berikutnya sebagai elemen saat ini. Baris ini juga menggunakan `list()` untuk mengubah elemen 0 dan 1 dari array yang dikembalikan oleh `each()` menjadi dua variabel baru yang disebut `$product` dan `$price`.

Kita dapat mengulang seluruh array `$prices`, dengan menggemakan konten menggunakan skrip singkat ini.

```
while ( list( $product, $price ) = each( $prices ) ) echo "$product -
  $price<br>";
```

Outputnya sama dengan skrip sebelumnya, tetapi lebih mudah dibaca karena `list()` memungkinkan kita untuk menetapkan nama pada variabel.

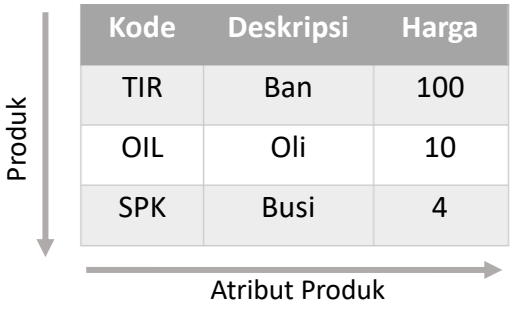
Satu hal yang perlu diperhatikan saat menggunakan `each()` adalah bahwa array melacak elemen saat ini. Jika kita ingin menggunakan array dua kali dalam skrip yang sama, kita perlu menyetel elemen saat ini kembali ke awal array menggunakan fungsi `reset()`. Untuk mengulang array prices lagi, kita ketik yang berikut ini:

```
reset($prices); while ( list( $product, $price ) = each( $prices ) ) echo
  "$product - $price<br>";
```

Ini akan mengembalikan elemen saat ini ke awal array, dan memungkinkan kita untuk melanjutkan lagi.

**Array Multidimensi**

Array tidak harus berupa daftar kunci dan nilai sederhana setiap lokasi dalam array dapat menampung array lain. Dengan cara ini, kita dapat membuat array dua dimensi. Anda dapat menganggap array dua dimensi sebagai matriks, atau kisi, dengan lebar dan tinggi atau baris dan kolom. Jika kita ingin menyimpan lebih dari satu bagian data tentang setiap produk Bob, kita dapat menggunakan array dua dimensi.




| Kode | Deskripsi | Harga |
|---|---|---|
| TIR | Ban | 100 |
| OIL | Oli | 10 |
| SPK | Busi | 4 |

**Gambar 3.3 Kita Dapat Menyimpan Informasi Lebih Lanjut Tentang Produk Bob Dalam Array Dua Dimensi.**

Gambar 3.3 menunjukkan produk Bob yang direpresentasikan sebagai array dua dimensi dengan setiap baris mewakili produk individual dan setiap kolom mewakili atribut

produk yang disimpan. Dengan menggunakan PHP, kita akan menulis kode berikut untuk mengatur data dalam array yang ditunjukkan pada Gambar 3.3.

```
$products = array( array( “TIR”, “Tires”, 100 ),
  array( “OIL”, “Oil”, 10 ),
  array( “SPK”, “Spark Plugs”, 4 ) );
```

Anda dapat melihat dari definisi ini bahwa array produk kita sekarang berisi tiga array.

Untuk mengakses data dalam array satu dimensi, ingatlah bahwa kita memerlukan nama array dan indeks elemen. Array dua dimensi serupa, kecuali bahwa setiap elemen memiliki dua indeks baris dan kolom. (Baris teratas adalah baris 0 dan kolom paling kiri adalah kolom 0.) Untuk menampilkan konten array ini, kita dapat mengakses setiap elemen secara manual dalam urutan seperti ini:

```
echo “|”.$products[0][0].”|”.$products[0][1].”|”.$products[0][2].”|<BR>”;
echo “|”.$products[1][0].”|”.$products[1][1].”|”.$products[1][2].”|<BR>”;
echo “|”.$products[2][0].”|”.$products[2][1].”|”.$products[2][2].”|<BR>”;
```

Alternatifnya, kita dapat menempatkan perulangan for di dalam perulangan for yang lain untuk memperoleh hasil yang sama.

```
for ( $row = 0; $row < 3; $row++ )
{
  for ( $column = 0; $column < 3; $column++ )
  {
       echo “|”.$products[$row][$column];
  }
  echo “|<BR>”;
}
```

Kedua versi kode ini menghasilkan output yang sama di browser:

```
|TIR|Tires|100|
|OIL|Oil|10|
|SPK|Spark Plugs|4|
```

Satu-satunya perbedaan antara kedua contoh tersebut adalah bahwa kode Anda akan lebih pendek jika Anda menggunakan versi kedua dengan array yang besar. Anda mungkin lebih suka membuat nama kolom daripada angka seperti yang ditunjukkan pada Gambar 3.3. Untuk melakukannya, Anda dapat menggunakan array asosiatif. Untuk menyimpan kumpulan produk yang sama, dengan kolom yang diberi nama seperti pada Gambar 3.3, Anda akan menggunakan kode berikut:

```
$products = array( array( Code =>“TIR”,
Description => “Tires”,
price => 100
),
array( Code => “OIL”,
Description => “Oil”,
price => 10
),
array( Code => “SPK”,
Description => “Spark Plugs”,
price =>4
)
);
```

Array ini lebih mudah digunakan jika Anda ingin mengambil satu nilai. Lebih mudah mengingat bahwa deskripsi disimpan di kolom *Description* daripada mengingat bahwa deskripsi disimpan di kolom 1. Dengan menggunakan array asosiatif, Anda tidak perlu mengingat bahwa item disimpan di $[x][y]$. Anda dapat dengan mudah menemukan data Anda dengan merujuk ke lokasi dengan nama baris dan kolom yang bermakna.

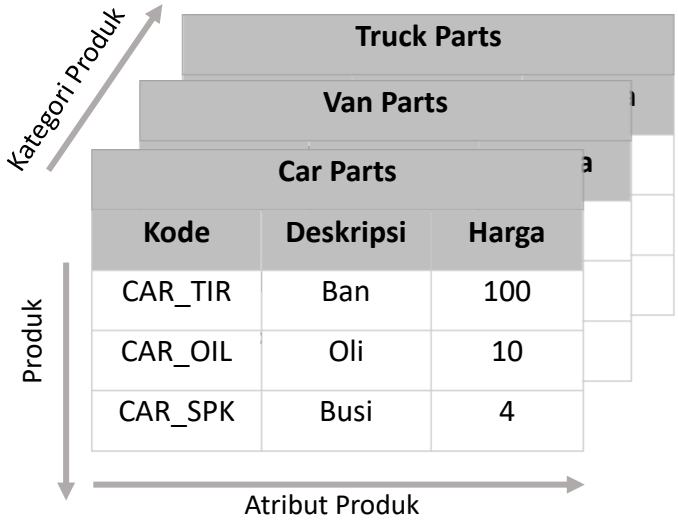
Namun, kita kehilangan kemampuan untuk menggunakan loop for sederhana untuk melangkah melalui setiap kolom secara bergantian. Berikut adalah salah satu cara untuk menulis kode guna menampilkan array ini:

```
for ( $row = 0; $row < 3; $row++ )
{
     echo “|”.$products[$row][“Code”].”|”.$products[$row][“Description”].
          “|”.$products[$row][“Price”].”|<BR>”;
}
```

Dengan menggunakan `for loop`, kita dapat melangkah melalui array `$products` terluar yang diindeks secara numerik. Setiap baris dalam array `$products` kita adalah array asosiatif. Dengan menggunakan fungsi `each()` dan `list()` dalam while loop, kita dapat melangkah melalui array asosiatif. Oleh karena itu, kita memerlukan while loop di dalam for loop.

```
for ( $row = 0; $row < 3; $row++ )
{
  while ( list( $key, $value ) = each( $products[ $row ] ) )
  {
       echo “|$value”;
  }
  echo “|<BR>”;
}
```

Kita tidak perlu berhenti pada dua dimensi dengan cara yang sama seperti elemen array dapat menampung array baru, array baru tersebut pada gilirannya dapat menampung lebih banyak array. Array tiga dimensi memiliki tinggi, lebar, dan kedalaman. Jika Anda merasa nyaman membayangkan array dua dimensi sebagai tabel dengan baris dan kolom, bayangkan tumpukan atau dek tabel tersebut. Setiap elemen akan direferensikan berdasarkan lapisan, baris, dan kolomnya. Jika Bob membagi produknya ke dalam kategori, kita dapat menggunakan array tiga dimensi untuk menyimpannya. Gambar 3.4 menunjukkan produk Bob dalam array tiga dimensi.




**Gambar 3.4 Array Tiga Dimensi Ini Memungkinkan Kita Untuk Membagi Produk Ke Dalam Beberapa Kategori.**

Dari kode yang mendefinisikan array ini, Anda dapat melihat bahwa array tiga dimensi adalah array yang berisi array dari array.

```
$categories = array( array ( array( “TIR”, “Tires”, 100 ),
                             array( “OIL”, “Oil”, 10 ),
                             array( “SPK”, “Spark Plugs”, 4 )
                             ),
                     array ( array( “TIR”, “Tires”, 100 ),
                             array( “OIL”, “Oil”, 10 ),
                             array( “SPK”, “Spark Plugs”, 4 )
                             ),
                     array ( array( “TIR”, “Tires”, 100 ),
                             array( “OIL”, “Oil”, 10 ),
                             array( “SPK”, “Spark Plugs”, 4 )
                             )
                   );
```

Karena array ini hanya memiliki indeks numerik, kita dapat menggunakan perulangan for bersarang untuk menampilkan isinya.

```
for ( $layer = 0; $layer < 3; $layer++ )
{
  echo “Layer $layer<BR>”;
  for ( $row = 0; $row < 3; $row++ )
       {
       for ( $column = 0; $column < 3; $column++ )
         {
         echo “|”.$categories[$layer][$row][$column];
         }
       echo “|<BR>”;
  }
}
```

Karena cara array multidimensi dibuat, kita dapat membuat array empat, lima, atau enam dimensi. Tidak ada batasan bahasa untuk jumlah dimensi, tetapi sulit bagi orang untuk memvisualisasikan konstruksi dengan lebih dari tiga dimensi. Sebagian besar masalah dunia nyata cocok secara logis dengan konstruksi tiga dimensi atau kurang.

**Mengurutkan Array**

Sering kali berguna untuk mengurutkan data terkait yang disimpan dalam array. Mengambil array satu dimensi dan mengurutkannya ke dalam urutan cukup mudah.

**Menggunakan `sort()`**

Kode berikut menghasilkan array yang diurutkan ke dalam urutan abjad menaik:

```
$products = array( “Tires”, “Oil”, “Spark Plugs” ); sort($products);
```

Elemen array kita sekarang akan berada dalam urutan Oil, Spark Plugs, Tires.

Kita juga dapat mengurutkan nilai berdasarkan urutan numerik. Jika kita memiliki array yang berisi harga produk Bob, kita dapat mengurutkannya ke dalam urutan numerik menaik seperti yang ditunjukkan:

```
$prices = array( 100, 10, 4 ); sort($prices);
```

Harga sekarang akan berada dalam urutan 4, 10, 100. Perhatikan bahwa fungsi sort peka huruf besar/kecil. Semua huruf kapital muncul sebelum semua huruf kecil. Jadi, ‘A’ lebih kecil dari ‘Z’, tetapi ‘Z’ lebih kecil dari ‘a’.

**Menggunakan asort() dan ksort() untuk Mengurutkan Array Asosiatif**

Jika kita menggunakan array asosiatif untuk menyimpan item dan harganya, kita perlu menggunakan berbagai jenis fungsi sort untuk menyimpan kunci dan nilai bersama saat diurutkan. Kode berikut membuat array asosiatif yang berisi tiga produk dan harga terkaitnya, lalu mengurutkan array ke dalam urutan harga menaik.

```
$prices = array( “Ban”=>100, “Oli”=>10, “Busi”=>4 ); asort($prices);
```

Fungsi asort() mengurutkan array berdasarkan nilai setiap elemen. Dalam array, nilai adalah harga dan kunci adalah deskripsi tekstual. Jika alih-alih mengurutkan berdasarkan harga, kita ingin mengurutkan berdasarkan deskripsi, kita menggunakan ksort(), yang mengurutkan berdasarkan kunci, bukan nilai. Kode ini akan menghasilkan kunci array yang diurutkan secara alfabetis Oli, Busi, Ban.

```
$prices = array( “Ban”=>100, “Oli”=>10, “Busi”=>4 ); ksort($prices);
```

**Mengurutkan secara Terbalik**

Anda telah melihat sort(), asort(), dan ksort(). Ketiga fungsi pengurutan yang berbeda ini mengurutkan array dalam urutan menaik. Masing-masing fungsi ini memiliki fungsi pengurutan terbalik yang cocok untuk mengurutkan array dalam urutan menurun. Versi terbalik disebut rsort(), arsort(), dan krsort().

Fungsi pengurutan terbalik digunakan dengan cara yang sama seperti fungsi pengurutan. Fungsi rsort() mengurutkan array berindeks numerik satu dimensi ke dalam urutan menurun. Fungsi arsort() mengurutkan array asosiatif satu dimensi ke dalam urutan menurun menggunakan nilai setiap elemen. Fungsi krsort() mengurutkan array asosiatif satu dimensi ke dalam urutan menurun menggunakan kunci setiap elemen.

**Mengurutkan Array Multidimensi**

Mengurutkan array dengan lebih dari satu dimensi, atau dengan sesuatu selain urutan alfabet atau numerik, lebih rumit. PHP tahu cara membandingkan dua angka atau dua string teks, tetapi dalam array multidimensi, setiap elemen adalah array. PHP tidak tahu cara membandingkan dua array, jadi Anda perlu membuat metode untuk membandingkannya. Sebagian besar waktu, urutan kata atau angka cukup jelas tetapi untuk objek yang rumit, hal itu menjadi lebih bermasalah.

**Pengurutan yang Ditentukan Pengguna**

Berikut adalah definisi array dua dimensi yang kami gunakan sebelumnya. Array ini menyimpan tiga produk Bob dengan kode, deskripsi, dan harga untuk masing-masing produk.

```
$products = array( array( “TIR”, “Tires”, 100 ),
                   array( “OIL”, “Oil”, 10 ),
                   array( “SPK”, “Spark Plugs”, 4 ) );
```

Jika kita mengurutkan array ini, dalam urutan apa nilai-nilai akan berakhir? Karena kita tahu apa yang diwakili oleh konten, setidaknya ada dua urutan yang berguna. Kita mungkin ingin produk diurutkan berdasarkan abjad menggunakan deskripsi atau berdasarkan urutan numerik berdasarkan harga. Kedua hasil tersebut mungkin saja, tetapi kita perlu menggunakan fungsi usort() dan memberi tahu PHP cara membandingkan item. Untuk melakukan ini, kita perlu menulis fungsi perbandingan kita sendiri.

Kode berikut mengurutkan array ini berdasarkan abjad menggunakan kolom kedua dalam array deskripsi.

```
function compare($x, $y)
{
  if ( $x[1] == $y[1] )
       return 0;
  else if ( $x[1] < $y[1] )
       return -1;
  else
       return 1;
}

usort($products, compare);
```

Sejauh ini dalam buku ini, kita telah memanggil sejumlah fungsi bawaan PHP. Untuk mengurutkan array ini, kita telah mendefinisikan fungsi kita sendiri. Kita akan membahas penulisan fungsi secara terperinci dalam Bab 5, "Menggunakan Kembali Kode dan Menulis Fungsi," tetapi berikut adalah pengantar singkatnya.

Kita mendefinisikan fungsi menggunakan kata kunci function. Kita perlu memberi fungsi tersebut sebuah nama. Nama harus bermakna, jadi kita akan menyebutnya compare(). Banyak fungsi yang mengambil parameter atau argumen. Fungsi `compare()` kita mengambil dua, satu disebut x dan satu disebut y. Tujuan dari fungsi ini adalah untuk mengambil dua nilai dan menentukan urutannya. Untuk contoh ini, parameter x dan y akan menjadi dua dari array dalam array utama, masing-masing mewakili satu produk. Untuk mengakses Deskripsi dari array x, kita ketik `$x[1]` karena Deskripsi adalah elemen kedua dalam array ini, dan penomoran dimulai dari nol. Kita menggunakan `$x[1]` dan `$y[1]` untuk membandingkan Deskripsi dari array yang diteruskan ke fungsi. Ketika suatu fungsi berakhir, fungsi tersebut dapat memberikan balasan kepada kode yang memanggilnya. Ini disebut mengembalikan nilai. Untuk mengembalikan nilai, kita menggunakan kata kunci return dalam fungsi kita. Misalnya, baris return 1; mengirimkan nilai 1 kembali ke kode yang memanggil fungsi tersebut.

Agar dapat digunakan oleh `usort()`, fungsi `compare()` harus membandingkan x dan y. Fungsi tersebut harus mengembalikan 0 jika x sama dengan y, angka negatif jika lebih kecil, dan angka positif jika lebih besar. Fungsi kita akan mengembalikan 0, 1, atau –1, tergantung pada nilai x dan y. Baris kode terakhir memanggil fungsi bawaan usort() dengan array yang ingin kita urutkan (`$products`) dan nama fungsi perbandingan kita (`compare()`). Jika kita ingin array diurutkan ke dalam urutan lain, kita cukup menulis fungsi perbandingan yang berbeda. Untuk mengurutkan berdasarkan harga, kita perlu melihat kolom ketiga dalam array, dan membuat fungsi perbandingan ini:

```
function compare($x, $y)
{
  if ($x[2] == $y[2])
```

```
    return 0;
      else if ($x[2] < $y[2])
    return -1;
      else
    return 1;
}
```

Ketika usort($products, compare) dipanggil, array akan ditempatkan dalam urutan menaik berdasarkan harga.

Huruf "u" dalam usort() adalah singkatan dari "user" karena fungsi ini memerlukan fungsi perbandingan yang ditentukan pengguna. Versi uasort() dan uksort() dari asort dan ksort juga memerlukan fungsi perbandingan yang ditentukan pengguna. Mirip dengan asort(), uasort() harus digunakan saat mengurutkan array asosiatif berdasarkan nilai. Gunakan asort jika nilai Anda berupa angka atau teks sederhana. Tentukan fungsi perbandingan dan gunakan uasort() jika nilai Anda berupa objek yang lebih rumit seperti array. Mirip dengan ksort(), uksort() harus digunakan saat mengurutkan array asosiatif berdasarkan kunci. Gunakan ksort jika kunci Anda berupa angka atau teks sederhana. Tentukan fungsi perbandingan dan gunakan uksort() jika kunci Anda berupa objek yang lebih rumit seperti array.

**Pengurutan Terbalik Pengguna**

Fungsi sort(), asort(), dan ksort() semuanya memiliki pengurutan terbalik yang cocok dengan huruf "r" pada nama fungsi. Pengurutan yang ditentukan pengguna tidak memiliki varian terbalik, tetapi Anda dapat mengurutkan array multidimensi ke dalam urutan terbalik. Anda menyediakan fungsi perbandingan, jadi tulislah fungsi perbandingan yang mengembalikan nilai yang berlawanan. Untuk mengurutkan ke dalam urutan terbalik, fungsi tersebut harus mengembalikan 1 jika x kurang dari y dan –1 jika x lebih besar dari y. Misalnya

```
function compare($x, $y)
{
  if ($x[2] == $y[2])
       return 0;
  else if ($x[2] < $y[2])
       return -1;
  else
       return 1;
}
```

Memanggil usort($products, *reverseCompare*) sekarang akan mengakibatkan array ditempatkan dalam urutan menurun berdasarkan harga.

**Menata Ulang Array**

Untuk beberapa aplikasi, Anda mungkin ingin memanipulasi urutan array dengan cara lain. Fungsi shuffle() menata ulang elemen array secara acak. Fungsi array_reverse() memberi Anda salinan array dengan semua elemen dalam urutan terbalik.

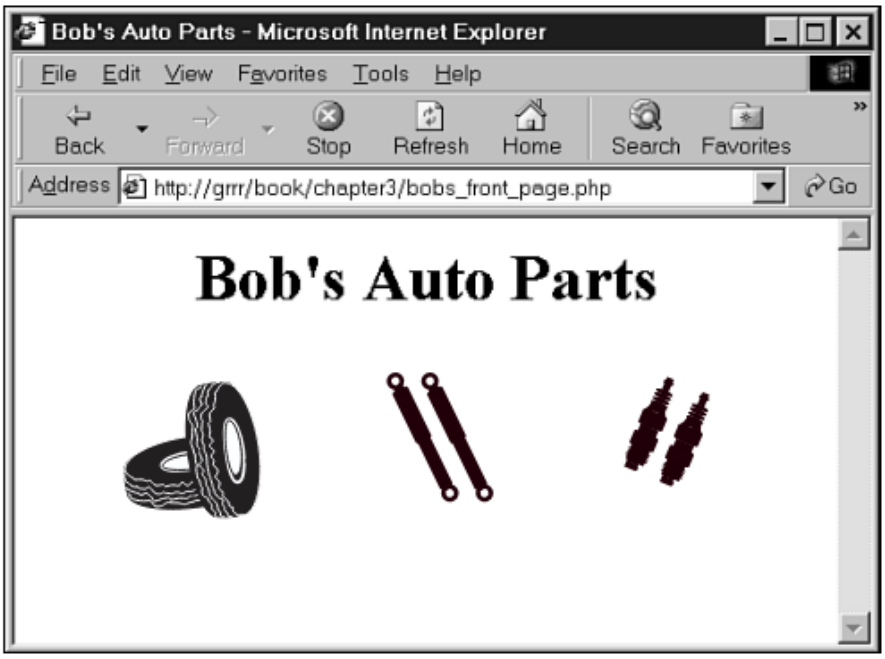
**Menggunakan `shuffle()`**

Bob ingin menampilkan sejumlah kecil produknya di halaman depan situsnya. Ia memiliki sejumlah besar produk, tetapi ingin tiga item yang dipilih secara acak ditampilkan di halaman depan. Agar pengunjung yang berulang tidak bosan, ia ingin tiga produk yang dipilih berbeda untuk setiap kunjungan. Ia dapat dengan mudah mencapai tujuannya jika semua produknya berada dalam array. Daftar 3.1 menampilkan tiga gambar yang dipilih secara acak dengan mengacak array ke dalam urutan acak lalu menampilkan tiga gambar pertama.

**Daftar 3.1 Bobs_Front_Page.Php—Menggunakan Php Untuk Menghasilkan Halaman Depan Yang Dinamis Untuk Suku Cadang Mobil Bob**

```
<?
  $pictures = array( “tire.jpg”, “oil.jpg”, “spark_plug.jpg”,
                  “door.jpg”, “steering_wheel.jpg”,
                  “thermostat.jpg”, “wiper_blade.jpg”,
                  “gasket.jpg”, “brake_pad.jpg”);
  shuffle($pictures);
?>
<html>
<head>
  <title>Bob’s Auto Parts</title>
</head>
<body>
  <center>
    <h1>Bob’s Auto Parts</H1>
    <table width = 100%>
      <tr>
<?
  for ( $i = 0; $i < 3; $i++ )
  {
    echo “<td align = center><img src=\””;
    echo $pictures[$i];
    echo “\” width = 100 height = 100></td>”;
  }
?>
      </tr>
    </table>
  </center>
</body>
</html>
```

Karena kode tersebut memilih gambar acak, kode tersebut menghasilkan halaman yang berbeda hampir setiap kali Anda memuatnya, seperti yang ditunjukkan pada Gambar 3.5.




**Gambar 3.5 Fungsi `Shuffle()` Memungkinkan Kita Untuk Menampilkan Tiga Produk Yang Dipilih Secara Acak.**

**Menggunakan `array_reverse()`**

Fungsi `array_reverse()` mengambil sebuah array dan membuat array baru dengan konten yang sama dalam urutan terbalik. Misalnya, ada sejumlah cara untuk membuat array yang berisi hitungan mundur dari sepuluh hingga satu.

Karena penggunaan `range()` saja akan membuat urutan menaik, maka kita harus menggunakan `rsort()` untuk mengurutkan angka-angka ke dalam urutan menurun. Atau, kita dapat membuat array satu elemen pada satu waktu dengan menulis `for loop`:

```
$numbers = array();
for($i=10; $i>0; $i--)
array_push( $numbers, $i );
```

For() loop dapat berjalan dalam urutan menurun seperti ini. Kita menetapkan nilai awal yang tinggi, dan di akhir setiap loop menggunakan operator -- untuk mengurangi penghitung sebanyak satu. Kami membuat array kosong, lalu menggunakan `array_push()` untuk setiap elemen guna menambahkan satu elemen baru di akhir array. Sebagai catatan tambahan, kebalikan dari `array_push()` adalah `array_pop()`. Fungsi ini menghapus dan mengembalikan satu elemen dari akhir array.

Alternatifnya, kita dapat menggunakan fungsi `array_reverse()` untuk membalikkan array yang dibuat oleh

```
range().
$numbers = range(1,10);
```

```
$numbers = array_reverse($numbers);
```

Perhatikan bahwa `array_reverse()` mengembalikan salinan array yang dimodifikasi. Karena kita tidak menginginkan array asli, kita cukup menyimpan salinan baru di atas yang asli.

## 3.3 MEMUAT ARRAY DARI FILE

Pada Bab 2, "Menyimpan dan Mengambil Data," kita menyimpan pesanan pelanggan dalam sebuah file. Setiap baris dalam file terlihat seperti ini

```
15:42, 20th April 4 tires 1 oil 6 spark plugs $434.00 22 Short St, Smalltown
```

Untuk memproses atau memenuhi pesanan ini, kita dapat memuatnya kembali ke dalam array. Daftar 3.2 menampilkan berkas pesanan saat ini.

**DAFTAR 3.2 Vieworders.Php—Menggunakan PHP Untuk Menampilkan Pesanan Bagi Bob**

```
$orders= file("../../orders/orders.txt");
$number_of_orders = count($orders);
if ($number_of_orders == 0)
{
  echo "<p><strong>No orders pending.
    Please try again later.</strong></p>";
}
for ($i=0; $i<$number_of_orders; $i++)
{
  echo $orders[$i]."<br>";
}
```

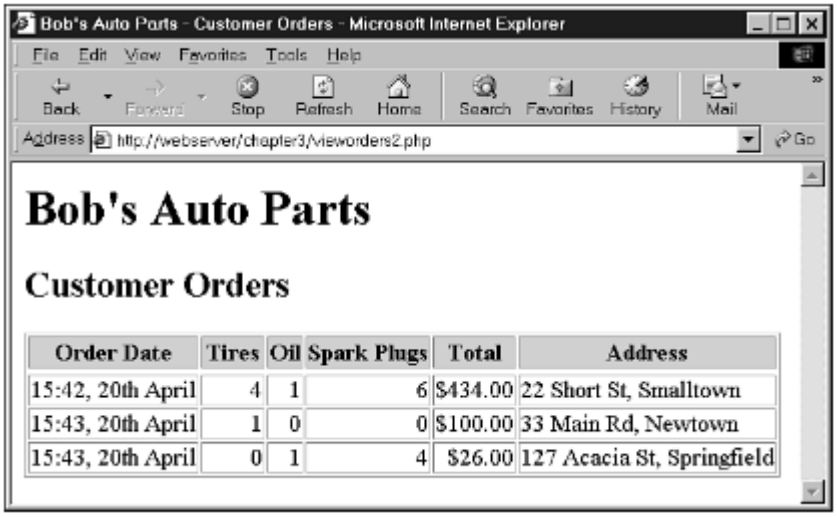
Skrip ini menghasilkan keluaran yang hampir sama persis dengan Listing 2.2 pada bab sebelumnya, yang ditunjukkan pada Gambar 2.4. Kali ini, kita menggunakan fungsi `file()`, yang memuat seluruh berkas ke dalam array. Setiap baris dalam berkas menjadi satu elemen array. Kode ini juga menggunakan fungsi `count()` untuk melihat berapa banyak elemen yang ada dalam array.

**Daftar 3.3 Vieworders2.Php—Menggunakan Php Untuk Memisahkan, Memformat, Dan Menampilkan Pesanan Untuk Bob**

```
<html>
<head>
  <title>Bob's Auto Parts – Customer Orders</title>
</head>
<body>
<h1>Bob's Auto Parts</h1>
<h2>Customer Orders</h2>
<?
  //Read in the entire file.
```

```
    //Each order becomes an element in the array
    $orders= file("../../orders/orders.txt");
    // count the number of orders in the array
    $number_of_orders = count($orders);
    if ($number_of_orders == 0)
    {
      echo "<p><strong>No orders pending.
        Please try again later.</strong></p>";
    }
      echo "<table border=1>\n";
      echo "<tr><th bgcolor = \"#CCCCFF\">Order Date</td>
                <th bgcolor = \"#CCCCFF\">Tires</td>
                <th bgcolor = \"#CCCCFF\">Oil</td>
                <th bgcolor = \"#CCCCFF\">Spark Plugs</td>
                <th bgcolor = \"#CCCCFF\">Total</td>
                <th bgcolor = \"#CCCCFF\">Address</td>
              <tr>";
      for ($i=0; $i<$number_of_orders; $i++)
      {
        //split up each line
        $line = explode( "\t", $orders[$i] );
        //keep only the number of items ordered
        $line[1] = intval( $line[1] );
        $line[2] = intval( $line[2] );
        $line[3] = intval( $line[3] );
        //output each order
        echo "<tr><td>$line[0]</td>
                  <td align = right>$line[1]</td>
                  <td align = right>$line[2]</td>
                  <td align = right>$line[3]</td>
                  <td align = right>$line[4]</td>
                  <td>$line[5]</td>
                </tr>";
      }
      echo "</table>";
?>
</body>
</html>
```

Lebih jauh, kita dapat memuat setiap bagian dari baris pesanan ke dalam elemen array terpisah untuk memproses bagian-bagian tersebut secara terpisah atau memformatnya agar lebih menarik. Listing 3.3 melakukan hal yang sama. Kode dalam Daftar 3.3 memuat seluruh berkas ke dalam array, tetapi tidak seperti contoh dalam Daftar 3.2, di sini kita menggunakan fungsi `explode()` untuk membagi setiap baris, sehingga kita dapat menerapkan beberapa pemrosesan dan pemformatan sebelum mencetak. Output dari skrip ini ditunjukkan pada Gambar 3.6.




**Gambar 3.6 Setelah Membagi Catatan Pesanan Dengan Explode, Kita Dapat Meletakkan Setiap Bagian Pesanan Di Sel Tabel Yang Berbeda Untuk Mendapatkan Hasil Yang Lebih Baik.**

Fungsi explode memiliki prototipe berikut:

```
array explode(string separator, string string)
```

Pada bab sebelumnya, kita menggunakan karakter tab sebagai pemisah saat menyimpan data ini, jadi di sini kita memanggil

```
explode( "\t", $orders[$i] )
```

Ini "meledakkan" string yang dimasukkan menjadi beberapa bagian. Setiap karakter tab menjadi pemisah antara dua elemen. Misalnya, string

```
"15:42, 20 April\t4 ban\t1 oli\t
➥6 busi\t$434.00\t22 Short St, Smalltown"
```

diledakkan menjadi beberapa bagian "`15:42, 20 April`", "`4 ban`", "`1 oli`", "`6 busi`", "`$434.00`", dan "`22 Short St, Smalltown`".

Kita belum melakukan banyak pemrosesan di sini. Daripada menampilkan ban, oli, dan busi pada setiap baris, kita hanya menampilkan jumlah masing-masing dan memberikan baris judul pada tabel untuk menunjukkan apa yang diwakili oleh angka-angka tersebut.

Ada sejumlah cara yang dapat kita gunakan untuk mengekstrak angka dari string ini. Di sini kita menggunakan fungsi `intval()`. Seperti yang disebutkan dalam Bab 1, `intval()` mengubah string menjadi integer. Konversi ini cukup cerdik dan akan mengabaikan bagian-bagian, seperti label dalam contoh ini, yang tidak dapat diubah menjadi integer. Kita akan membahas berbagai cara untuk memproses string di bab berikutnya.

**Manipulasi Array Lainnya**

Sejauh ini, kita baru membahas sekitar setengah dari fungsi pemrosesan array. Banyak fungsi lain yang akan berguna dari waktu ke waktu.

Menavigasi Dalam Array: each, current(), reset(), end(), next(), pos(), dan prev()

Kita sebutkan sebelumnya bahwa setiap array memiliki pointer internal yang menunjuk ke elemen saat ini dalam array. Kita secara tidak langsung menggunakan pointer ini sebelumnya saat menggunakan fungsi each(), tetapi kita dapat langsung menggunakan dan memanipulasi pointer ini.

Jika kita membuat array baru, pointer saat ini diinisialisasi untuk menunjuk ke elemen pertama dalam array. Memanggil current( $array_name ) akan mengembalikan elemen pertama. Memanggil next() atau each() memajukan pointer satu elemen ke depan. Memanggil each( $array_name ) mengembalikan elemen saat ini sebelum memajukan pointer. Fungsi next() berperilaku sedikit berbeda memanggil next( $array_name ) memajukan pointer lalu mengembalikan elemen saat ini yang baru. Kita telah melihat bahwa reset() mengembalikan pointer ke elemen pertama dalam array. Demikian pula, memanggil end( $array_name ) mengirim pointer ke akhir array.

Elemen pertama dan terakhir dalam array dikembalikan oleh reset() dan end(), masing-masing. Untuk bergerak melalui array dalam urutan terbalik, kita dapat menggunakan end() dan prev(). Fungsi prev() adalah kebalikan dari next(). Fungsi ini memindahkan pointer saat ini ke belakang satu elemen lalu mengembalikan elemen saat ini yang baru. Misalnya, kode berikut menampilkan array dalam urutan terbalik:

```
$value = end ($array);
while ($value)
{
  echo “$value<br>”;
  $value = prev($array);
}
```

Jika $array dideklarasikan seperti ini:

```
$array = array(1, 2, 3);
```

output akan muncul di browser sebagai

```
3
2
1
```

Dengan menggunakan each(), current(), reset(), end(), next(), pos(), dan prev(), Anda dapat menulis kode Anda sendiri untuk menavigasi array dalam urutan apa pun.

Menerapkan Fungsi Apa Pun ke Setiap Elemen dalam Array: array_walk()

Terkadang Anda mungkin ingin bekerja dengan atau memodifikasi setiap elemen dalam array dengan cara yang sama. Fungsi `array_walk()` memungkinkan Anda melakukan ini. Prototipe `array_walk()` adalah sebagai berikut:

```
int array_walk(array arr, string func, [mixed userdata])
```

Mirip dengan cara kita memanggil `usort()` sebelumnya, `array_walk()` mengharapkan Anda untuk mendeklarasikan fungsi Anda sendiri.

Seperti yang Anda lihat, `array_walk()` mengambil tiga parameter. Yang pertama, arr, adalah array yang akan diproses. Yang kedua, func, adalah nama fungsi yang ditentukan pengguna yang akan diterapkan pada setiap elemen dalam array. Parameter ketiga, userdata, bersifat opsional. Jika Anda menggunakannya, parameter tersebut akan diteruskan ke fungsi Anda. Anda akan melihat cara kerjanya sebentar lagi.

Fungsi yang ditentukan pengguna yang praktis mungkin adalah fungsi yang menampilkan setiap elemen dengan beberapa format yang ditentukan. Kode berikut menampilkan setiap elemen pada baris baru dengan memanggil fungsi yang ditentukan pengguna `myPrint()` dengan setiap elemen `$array`:

```
function myPrint($value)
{
  echo “$value<BR>”;
}
array_walk($array, myPrint);
```

Fungsi yang Anda tulis harus memiliki tanda tangan tertentu. Untuk setiap elemen dalam array, `array_walk` mengambil kunci dan nilai yang disimpan dalam array, dan apa pun yang Anda berikan sebagai userdata, dan memanggil fungsi Anda seperti ini:

```
Yourfunction(value, key, userdata)
```

Untuk sebagian besar penggunaan, fungsi Anda hanya akan menggunakan nilai dalam array. Untuk beberapa penggunaan, Anda mungkin juga perlu memberikan parameter ke fungsi Anda menggunakan parameter userdata.

Kadang-kadang, Anda mungkin tertarik pada kunci setiap elemen serta nilainya. Fungsi Anda dapat, seperti halnya `MyPrint()`, memilih untuk mengabaikan kunci dan parameter userdata. Untuk contoh yang sedikit lebih rumit, kita akan menulis fungsi yang mengubah nilai dalam array dan memerlukan parameter. Perhatikan bahwa meskipun kita tidak tertarik pada kunci, kita perlu menerimanya untuk menerima parameter ketiga.

```
function myMultiply(&$value, $key, $factor)
{
  $value *= $factor;
}
```

```
array_walk(&$array, “myMultiply”, 3);
```

Di sini kita mendefinisikan sebuah fungsi, `myMultiply()`, yang akan mengalikan setiap elemen dalam array dengan faktor yang diberikan. Kita perlu menggunakan parameter ketiga opsional untuk `array_walk()` guna mengambil parameter untuk diteruskan ke fungsi kita dan menggunakannya sebagai faktor untuk mengalikan. Karena kita memerlukan parameter ini, kita harus mendefinisikan fungsi kita, `myMultiply()`, untuk mengambil tiga parameter—nilai elemen array (`$value`), kunci elemen array (`$key`), dan parameter kita (`$factor`). Kita memilih untuk mengabaikan kunci.

Hal penting yang perlu diperhatikan adalah cara kita meneruskan `$value`. Tanda ampersand (&) sebelum nama variabel dalam definisi `myMultiply()` berarti bahwa `$value` akan diteruskan dengan referensi. Melewati dengan referensi memungkinkan fungsi untuk mengubah isi array.

Kita akan membahas penerusan dengan referensi secara lebih rinci di Bab 5. Jika Anda tidak familier dengan istilah tersebut, untuk saat ini perhatikan saja bahwa untuk meneruskan dengan referensi, kita menempatkan tanda ampersand sebelum nama variabel. Menghitung Elemen dalam Array: `count()`, `sizeof()`, dan `array_count_values()`

Kami menggunakan fungsi `count()` dalam contoh sebelumnya untuk menghitung jumlah elemen dalam array dengan urutan tertentu. Fungsi `sizeof()` memiliki tujuan yang sama persis. Kedua fungsi ini mengembalikan jumlah elemen dalam array yang diberikan kepada mereka. Anda akan mendapatkan hitungan satu untuk jumlah elemen dalam variabel skalar normal dan 0 jika Anda memberikan array kosong atau variabel yang belum ditetapkan. **Fungsi `array_count_values()` lebih kompleks.**

Jika Anda memanggil `array_count_values($array)`, fungsi ini menghitung berapa kali setiap nilai unik muncul dalam array `$array`. (Ini adalah kardinalitas array yang ditetapkan.) Fungsi ini mengembalikan array asosiatif yang berisi tabel frekuensi. Array ini berisi semua nilai unik dari `$array` sebagai kunci. Setiap kunci memiliki nilai numerik yang memberi tahu Anda berapa kali kunci terkait muncul dalam `$array`.
Misalnya, kode berikut

```
$array = array(4, 5, 1, 2, 3, 1, 2, 1);
$ac = array_count_values($array);
```

membuat array bernama `$ac` yang berisi

| Kunci | Nilai |
|---|---|
| 4 | 1 |
| 5 | 1 |
| 1 | 3 |
| 2 | 2 |
| 3 | 1 |

Ini menunjukkan bahwa 4, 5, dan 3 muncul sekali dalam $array, 1 muncul tiga kali, dan 2 muncul dua kali.

**Mengonversi Array ke Variabel Skalar: extract()**

Jika kita memiliki array asosiatif dengan sejumlah pasangan kunci nilai, kita dapat mengubahnya menjadi sekumpulan variabel skalar menggunakan fungsi extract(). Prototipe untuk extract() adalah sebagai berikut:

```
extract(array var_array [, int extract_type] [, string prefix] );
```

Tujuan dari extract() adalah untuk mengambil array dan membuat variabel skalar dengan nama kunci dalam array. Nilai variabel ini ditetapkan ke nilai dalam array.

Berikut adalah contoh sederhananya.

```
$array = array( “key1” => “value1”, “key2” => “value2”, “key3” => “value3”);
extract($array);
echo “$key1 $key2 $key3”;
```

Array memiliki tiga elemen dengan kunci: *key1, key2,* dan *key3*. Dengan menggunakan extract(), kami membuat tiga variabel skalar, $key1, $key2, dan $key3. Anda dapat melihat dari output bahwa nilai dari $key1, $key2, dan $key3 masing-masing adalah *“value1”, “value2”,* dan *“value3”.* Nilai-nilai ini berasal dari array asli.

Ada dua parameter opsional untuk extract(): extract_type dan prefix. Variabel extract_type memberi tahu extract() cara menangani tabrakan. Ini adalah kasus di mana variabel sudah ada dengan nama yang sama dengan kunci. Respons default adalah menimpa variabel yang ada. Empat nilai yang diizinkan untuk extract_type ditunjukkan pada Tabel 3.1.

**Tabel 3.1 Jenis Ekstrak Yang Diizinkan Untuk Extract()**

| ***Type*** | ***Artinya*** |
|---|---|
| EXTR_OVERWRITE | Menimpa variabel yang ada saat terjadi tabrakan. |
| EXTR_SKIP | Melewati elemen saat terjadi tabrakan. |
| EXTR_PREFIX_SAME | Membuat variabel bernama $prefix_key saat terjadi tabrakan. Anda harus memberikan awalan. |
| EXTR_PREFIX_ALL | Memberikan awalan pada semua nama variabel. Anda harus memberikan awalan. |

Dua opsi yang paling berguna adalah default (EXTR_OVERWRITE) dan EXTR_PREFIX_ALL. Dua opsi lainnya mungkin berguna sesekali ketika Anda tahu bahwa tabrakan tertentu akan terjadi dan ingin tombol tersebut dilewati atau diawali. Contoh sederhana menggunakan

EXTR_PREFIX_ALL adalah sebagai berikut. Anda dapat melihat bahwa variabel yang dibuat disebut prefix-underscore-keyname.

```
$array = array( “key1” => “value1”, “key2” => “value2”, “key3” => “value3”);
extract($array, EXTR_PREFIX_ALL, “myPrefix”);
echo “$myPrefix_key1 $myPrefix_key2 $myPrefix_key3”;
```

Perhatikan bahwa agar extract() dapat mengekstrak elemen, kunci elemen tersebut harus berupa nama variabel yang valid, yang berarti kunci yang dimulai dengan angka atau yang menyertakan spasi akan dilewati.

# BAB 4
# MANIPULASI STRING DAN EKSPRESI REGULER

Dalam bab ini, kita akan membahas cara menggunakan fungsi string PHP untuk memformat dan memanipulasi teks. Kita juga akan membahas penggunaan fungsi string atau fungsi ekspresi reguler untuk mencari (dan mengganti) kata, frasa, atau pola lain dalam string. Fungsi-fungsi ini berguna dalam banyak konteks. Anda sering kali ingin membersihkan atau memformat ulang masukan pengguna yang akan disimpan dalam basis data. Fungsi pencarian sangat bagus saat membangun aplikasi mesin pencari (di antara hal-hal lainnya).
Dalam bab ini, kita akan membahas

- ☑ Memformat string
- ☑ Menggabungkan dan memisahkan string
- ☑ Membandingkan string
- ☑ Mencocokkan dan mengganti substring dengan fungsi string
- ☑ Menggunakan ekspresi reguler

## 4.1 SMART FORM MAIL

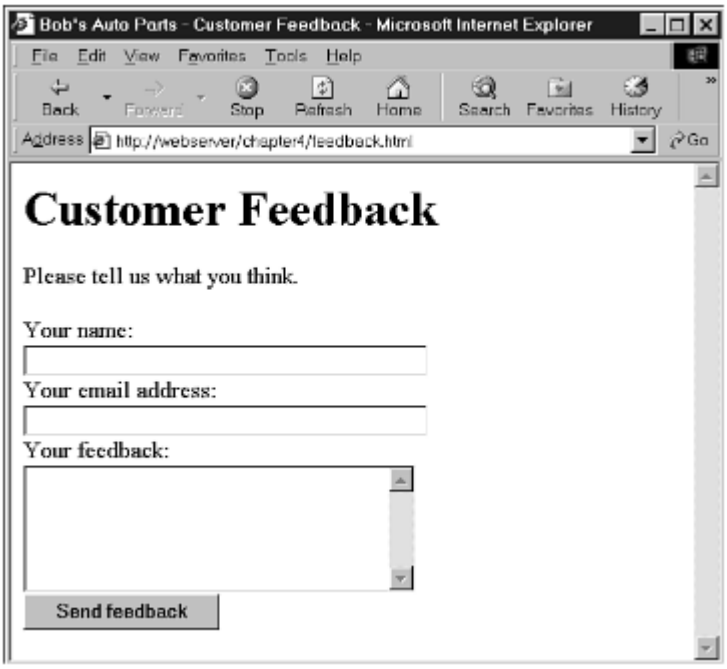
Dalam bab ini, kita akan melihat fungsi string dan ekspresi reguler dalam konteks aplikasi Smart Form Mail. Kita akan menambahkan skrip ini ke situs Bob's Auto Parts yang telah kita lihat dalam beberapa bab terakhir. Kali ini, kita akan membuat formulir umpan balik pelanggan yang mudah digunakan dan umum digunakan bagi pelanggan Bob untuk menyampaikan keluhan dan pujian mereka, seperti yang ditunjukkan pada Gambar 4.1.

**Daftar 4.1 Processfeedback.Php—Skrip Dasar Untuk Isi Formulir Email**

```
<?
  $toaddress = "feedback@bobsdomain.com";
  $subject = "Feedback from web site";
  $mailcontent = "Customer name: ".$name."\n"
               ."Customer email: ".$email."\n"
               ."Customer comments: \n".$feedback."\n";
$fromaddress = "webserver@bobsdomain.com";
  mail($toaddress, $subject, $mailcontent, $fromaddress);
?>
<html>
<head>
  <title>Bob's Auto Parts - Feedback Submitted</title>
</head>
<body>
<h1>Feedback submitted</h1>
<p>Your feedback has been sent.</p>
</body>
</html>
```

Namun, aplikasi kita akan memiliki satu peningkatan dibandingkan dengan banyak yang akan Anda temukan di Web. Alih-alih mengirim formulir melalui email ke alamat email umum seperti feedback@bobsdomain.com, kita akan mencoba memasukkan sedikit kecerdasan ke dalam proses tersebut dengan mencari kata kunci dan frasa pada input, lalu mengirim email tersebut ke karyawan yang tepat di perusahaan Bob. Misalnya, jika email tersebut berisi kata "iklan", kita dapat mengirim umpan balik tersebut ke departemen Pemasaran. Jika email tersebut berasal dari klien terbesar Bob, umpan balik tersebut dapat langsung dikirim ke Bob.

Kita akan mulai dengan skrip sederhana yang ditunjukkan pada Daftar 4.1 dan menambahkannya seiring berjalannya waktu.




**Gambar 4.1 Formulir Umpan Balik Bob Menanyakan Nama, Alamat Email, Dan Komentar Pelanggan.**

Perlu dicatat bahwa secara umum Anda harus memeriksa apakah pengguna telah mengisi semua kolom formulir yang diperlukan menggunakan, misalnya, `isempty()`. Kami telah menghilangkan ini dari skrip dan contoh lainnya demi singkatnya.

Dalam skrip ini, Anda akan melihat bahwa kami telah menggabungkan kolom formulir dan menggunakan fungsi `mail()` PHP untuk mengirimkannya melalui email ke feedback@bobsdomain.com. Kami belum menggunakan `mail()`, jadi kami akan membahas cara kerjanya.

Tidak mengherankan, fungsi ini mengirimkan email. Prototipe untuk `mail()` tampak seperti ini:

```
bool mail(string to, string subject, string message,
string [additional_headers]);
```

Tiga parameter pertama wajib diisi dan masing-masing mewakili alamat tujuan pengiriman email, baris subjek, dan isi pesan. Parameter keempat dapat digunakan untuk mengirim header email tambahan yang valid. Header email yang valid dijelaskan dalam dokumen RFC822, yang tersedia daring jika Anda menginginkan informasi lebih rinci. (RFC atau Requests For Comment merupakan sumber dari banyak standar Internet kita akan membahasnya di Bab 8, "Menggunakan Fungsi Jaringan dan Protokol.") Di sini, kita menggunakan parameter keempat untuk menambahkan alamat "From:" untuk email. Anda juga dapat menggunakannya untuk menambahkan kolom "Reply-To:" dan "Cc:", di antara kolom lainnya. Jika Anda menginginkan lebih dari satu header tambahan, pisahkan saja dengan baris baru `(\n)` di dalam string, seperti berikut:

```
$additional_headers="From: webserver@bobsdomain.com\n"
."Reply-To: bob@bobsdomain.com";
```

Untuk menggunakan fungsi email(), atur instalasi PHP Anda agar mengarah ke program pengiriman email Anda. Jika skrip tersebut tidak berfungsi untuk Anda dalam bentuknya saat ini. Dalam bab ini, kita akan menyempurnakan skrip dasar ini dengan memanfaatkan penanganan string dan fungsi ekspresi reguler PHP.

## 4.2 MEMFORMAT STRING

Anda sering kali perlu merapikan string pengguna (biasanya dari antarmuka formulir HTML) sebelum dapat menggunakannya. Memangkas String: `chop()`, `ltrim()`, dan `trim()` Langkah pertama dalam merapikan adalah memangkas spasi kosong yang berlebih dari string. Meskipun ini tidak pernah wajib, ini dapat berguna jika Anda akan menyimpan string dalam file atau basis data, atau jika Anda akan membandingkannya dengan string lain. PHP menyediakan tiga fungsi yang berguna untuk tujuan ini. Kita akan menggunakan fungsi `trim()` untuk merapikan data masukan kita sebagai berikut:

```
$name=trim($name);
$email=trim($email);
$feedback=trim($feedback);
```

Fungsi `trim()` menghilangkan spasi dari awal dan akhir string, dan mengembalikan string yang dihasilkan. Karakter yang dihilangkannya adalah baris baru dan carriage return (\n dan \r), tab horizontal dan vertikal `(\t dan \v)`, karakter akhir string `(\0)`, dan spasi.

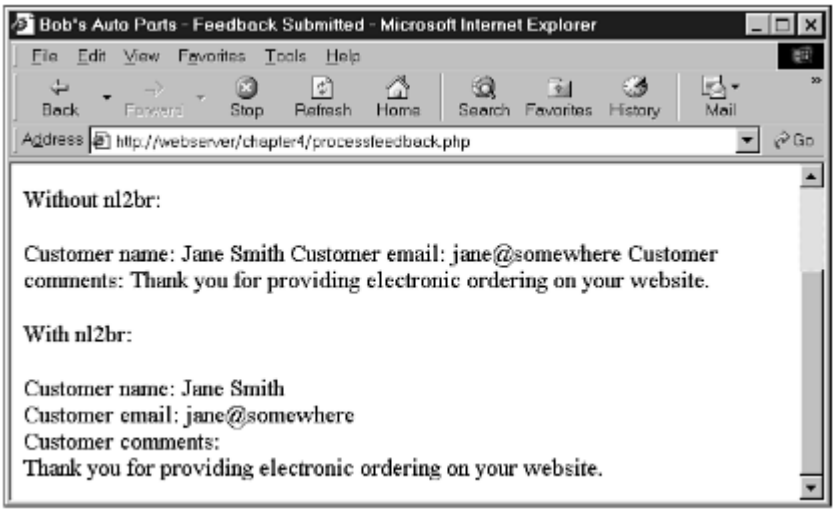
Bergantung pada tujuan khusus Anda, Anda mungkin ingin menggunakan fungsi `ltrim()` atau `chop()` sebagai gantinya. Keduanya mirip dengan `trim()`, mengambil string yang dimaksud sebagai parameter dan mengembalikan string yang diformat. Perbedaan antara ketiganya adalah `trim()` menghilangkan spasi dari awal dan akhir string, `ltrim()` menghilangkan spasi dari awal (atau kiri) saja, dan `chop()` menghilangkan spasi dari akhir (atau kanan) saja.

**Memformat String untuk Presentasi**

PHP memiliki serangkaian fungsi yang dapat Anda gunakan untuk memformat ulang string dengan berbagai cara. Menggunakan Pemformatan HTML: Fungsi `nl2br()` Fungsi `nl2br()` mengambil string sebagai parameter dan mengganti semua baris baru di dalamnya dengan tag HTML <BR>. Ini berguna untuk menampilkan string yang panjang ke browser. Misalnya, kami menggunakan fungsi ini untuk memformat umpan balik pelanggan agar dapat ditampilkan kembali:

```
<p>Your feedback (shown below) has been sent.</p>
<p><? echo nl2br($mailcontent); ?> </p>
```

Ingat bahwa HTML mengabaikan spasi kosong, jadi jika Anda tidak memfilter output ini melalui `nl2br()`, output ini akan muncul pada satu baris (kecuali untuk baris baru yang dipaksakan oleh jendela browser). Hal ini diilustrasikan pada Gambar 4.2.




**Gambar 4.2 Penggunaan Fungsi Nl2br() Php Meningkatkan Tampilan String Panjang Dalam Html.**

**Memformat String untuk Dicetak**

Sejauh ini, kita telah menggunakan konstruksi bahasa echo untuk mencetak string ke browser. PHP juga mendukung fungsi `print()`, yang melakukan hal yang sama seperti echo, tetapi karena merupakan fungsi, fungsi ini mengembalikan nilai (0 atau 1, yang menunjukkan

keberhasilan). Kedua teknik ini mencetak string "apa adanya". Anda dapat menerapkan beberapa pemformatan yang lebih canggih menggunakan fungsi `printf()` dan `sprintf()`.

Keduanya bekerja dengan cara yang sama, kecuali bahwa `printf()` mencetak string yang diformat ke browser dan `sprintf()` mengembalikan string yang diformat. Jika Anda sebelumnya telah memprogram dalam C, Anda akan menemukan bahwa fungsi-fungsi ini sama dengan versi C. Jika belum, fungsi-fungsi ini perlu dibiasakan tetapi berguna dan canggih. Prototipe untuk fungsi-fungsi ini adalah

```
string sprintf (string format [, mixed args...])
int printf (string format [, mixed args...])
```

Parameter pertama yang diberikan ke kedua fungsi ini adalah string format yang menggambarkan bentuk dasar keluaran dengan kode format, bukan variabel. Parameter lainnya adalah variabel yang akan disubstitusikan ke dalam string format.
Misalnya, menggunakan echo, kami menggunakan variabel yang ingin kami cetak sebaris, seperti ini:

```
echo “Total amount of order is $total.”;
```

Untuk mendapatkan efek yang sama dengan printf(), Anda akan menggunakan

```
printf (“Total amount of order is %s.”, $total);
```

Jika nilai yang disimpan dalam $total adalah 12,4, kedua pendekatan ini akan mencetaknya sebagai 12,4. Keuntungan dari `printf()` adalah kita dapat menggunakan spesifikasi konversi yang lebih berguna untuk menentukan bahwa `$total` sebenarnya adalah angka floating point, dan bahwa ia harus memiliki dua tempat desimal setelah titik desimal, sebagai berikut:

```
printf (“Total amount of order is %.2f”, $total);
```

Anda dapat memiliki beberapa spesifikasi konversi dalam string format. Jika Anda memiliki n spesifikasi konversi, Anda harus memiliki n argumen setelah string format. Setiap spesifikasi konversi akan diganti dengan argumen yang diformat ulang dalam urutan yang tercantum. Misalnya

```
printf (“Total jumlah pesanan adalah %.2f (dengan pengiriman %.2f) “,
$total, $total_shipping);
```

Di sini, spesifikasi konversi pertama akan menggunakan variabel $total, dan yang kedua akan menggunakan variabel `$total_shipping`.

Setiap spesifikasi konversi mengikuti format yang sama, yaitu

```
%[‘padding_character][-][width][.precision]type
```

Semua spesifikasi konversi dimulai dengan simbol %. Jika Anda benar-benar ingin mencetak simbol %, Anda harus menggunakan %%.

Padding_character bersifat opsional. Karakter ini akan digunakan untuk memberi dimensi pada variabel Anda agar sesuai dengan lebar yang telah Anda tentukan. Contohnya adalah menambahkan angka nol di depan angka seperti penghitung. Simbol - bersifat opsional. Simbol ini menentukan bahwa data di bidang akan diratakan ke kiri, bukan diratakan ke kanan, sebagai default.

Penentu lebar memberi tahu `printf()` seberapa banyak ruang (dalam karakter) yang harus disediakan agar variabel dapat diganti di sini. Penentu presisi harus dimulai dengan titik desimal. Penentu ini harus berisi jumlah tempat setelah titik desimal yang ingin Anda tampilkan. Bagian akhir dari spesifikasi adalah kode tipe. Ringkasannya ditunjukkan pada Tabel 4.1.

**Tabel 4.1 Kode Tipe Spesifikasi Konversi**

| ***Tipe*** | ***Arti*** |
|---|---|
| b | Tafsirkan sebagai bilangan bulat dan cetak sebagai bilangan biner. |
| c | Tafsirkan sebagai bilangan bulat dan cetak sebagai karakter. |
| d | Tafsirkan sebagai bilangan bulat dan cetak sebagai bilangan desimal. |
| f | Tafsirkan sebagai bilangan ganda dan cetak sebagai bilangan floating point. |
| o | Tafsirkan sebagai bilangan bulat dan cetak sebagai bilangan oktal. |
| s | Tafsirkan sebagai string dan cetak sebagai string. |
| x | Tafsirkan sebagai bilangan bulat dan cetak sebagai bilangan heksadesimal dengan huruf kecil untuk digit a–f. |
| X | Tafsirkan sebagai bilangan bulat dan cetak sebagai bilangan heksadesimal dengan huruf besar untuk digit A–F |

Catatan lain, saat kita membahas topik ini, adalah saat mencetak atau menampilkan sesuatu di browser, Anda mungkin telah memperhatikan bahwa kita menggunakan beberapa karakter khusus seperti `\n`. Ini adalah cara menulis karakter khusus ke output. Karakter `\n` adalah baris baru. Karakter utama lainnya yang akan Anda lihat adalah `\t`, atau tab, dan `\s`, atau spasi.

**Mengubah Huruf Besar Kecil pada String**

Anda juga dapat mengubah huruf besar kecil pada string. Ini tidak terlalu berguna untuk aplikasi kita, tetapi kita akan melihat beberapa contoh singkat. Jika kita mulai dengan string subjek, `$subject,` yang kita gunakan untuk email kita, kita dapat mengubah huruf besar kecilnya dengan beberapa fungsi. Efek dari fungsi-fungsi ini dirangkum dalam Tabel 4.2. Kolom pertama menunjukkan nama fungsi, kolom kedua menjelaskan efeknya, kolom ketiga

menunjukkan bagaimana fungsi tersebut akan diterapkan pada string `$subject`, dan kolom terakhir menunjukkan nilai apa yang akan dikembalikan dari fungsi tersebut.

**Tabel 4.2 Fungsi String Case Dan Efeknya**

| *Fungsi* | *Keterangan* | *Menggunakan* | *Nilai* |
|---|---|---|---|
| | | `$subject` | Umpan balik dari situs web |
| `strtoupper()` | Mengubah string menjadi huruf besar | `strtoupper($subject)` | MASUKAN DARI SITUS WEB |
| `strtolower()` | Mengubah String menjadi huruf Kecil | `strtolower($subject)` | Umpan balik dari situs web |
| `ucfirst()` | Kapitalisasi Karakter pertama dari string jika itu alfabet | `ucfirst($subject)` | Umpan balik dari situs web |
| `ucwords()` | Menjadikan huruf kapital pada karakter pertama setiap kata dalam string yang dimulai dengan karakter alfabet | `ucwords($subject)` | Umpan balik dari situs web |

**Memformat String untuk Penyimpanan: `AddSlashes()` dan `StripSlashes()`**

Selain menggunakan fungsi string untuk memformat ulang string secara visual, kita dapat menggunakan beberapa fungsi ini untuk memformat ulang string untuk penyimpanan dalam basis data.

Karakter tertentu benar-benar valid sebagai bagian dari string tetapi dapat menyebabkan masalah, terutama saat memasukkan data ke dalam basis data karena basis data dapat mengartikan karakter ini sebagai karakter kontrol. Yang bermasalah adalah tanda kutip (tunggal dan ganda), garis miring terbalik (\), dan karakter NUL.

Kita perlu menemukan cara untuk menandai, atau melepaskan, karakter ini sehingga basis data seperti MySQL dapat memahami bahwa yang kita maksud adalah karakter khusus literal daripada urutan kontrol. Untuk melepaskan karakter ini, tambahkan garis miring terbalik di depannya. Misalnya, “(tanda kutip ganda) menjadi \” (garis miring terbalik tanda kutip ganda), dan \ (garis miring terbalik) menjadi \\ (garis miring terbalik garis miring terbalik). (Aturan ini berlaku secara universal untuk karakter khusus, jadi jika Anda memiliki \\ dalam string Anda, Anda perlu menggantinya dengan \\\\.)

PHP menyediakan dua fungsi yang dirancang khusus untuk karakter escape. Sebelum Anda menulis string apa pun ke dalam basis data, Anda harus memformat ulang string tersebut dengan `AddSlashes()`, misalnya:

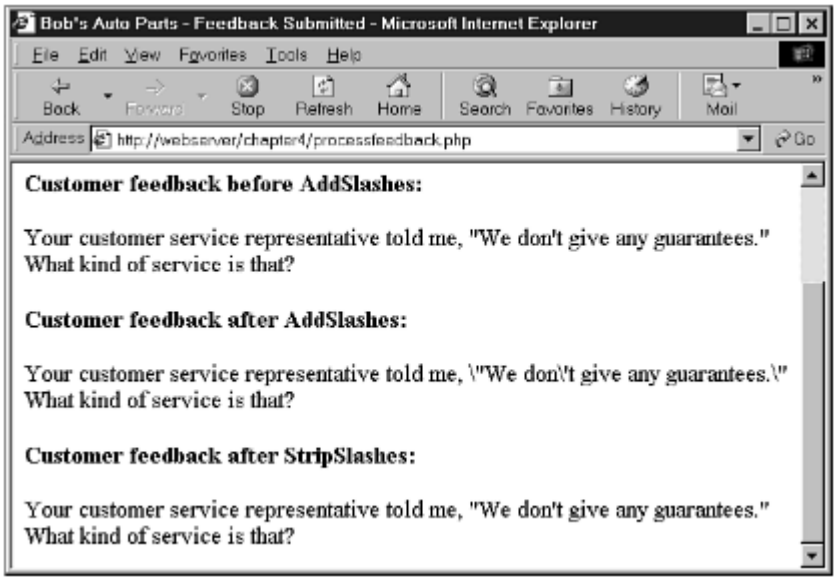
```
$feedback = AddSlashes($feedback);
```

Seperti banyak fungsi string lainnya, `AddSlashes()` mengambil string sebagai parameter dan mengembalikan string yang diformat ulang.

Saat Anda menggunakan `AddSlashes()`, string akan disimpan dalam basis data dengan garis miring di dalamnya. Saat Anda mengambil string, Anda harus ingat untuk menghapus garis miring. Anda dapat melakukannya menggunakan fungsi `StripSlashes()`:

```
$feedback = StripSlashes($feedback);
```

Gambar 4.3 menunjukkan efek sebenarnya dari penggunaan fungsi-fungsi ini pada string.




**Gambar 4.3 Setelah Memanggil Fungsi `Addslashes()`, Semua Tanda Kutip Telah Dihilangkan. `Stripslashes()` Menghapus Tanda Kutip.**

Anda juga dapat mengatur PHP untuk menambahkan dan menghapus tanda kutip secara otomatis. Ini disebut menggunakan tanda kutip ajaib.

## 4.3 MENGGABUNGKAN DAN MEMISAHKAN STRING DENGAN FUNGSI STRING

Sering kali, kita ingin melihat bagian-bagian string secara individual. Misalnya, kita mungkin ingin melihat kata-kata dalam kalimat (misalnya untuk pemeriksaan ejaan), atau membagi nama domain atau alamat email menjadi bagian-bagian komponennya. PHP menyediakan beberapa fungsi string (dan satu fungsi ekspresi reguler) yang memungkinkan kita melakukan ini.

Dalam contoh kita, Bob ingin setiap umpan balik pelanggan dari bigcustomer.com dikirim langsung kepadanya, jadi kita akan membagi alamat email yang diketik pelanggan menjadi beberapa bagian untuk mengetahui apakah alamat tersebut berfungsi untuk pelanggan besar Bob.

**Menggunakan explode(), implode(), dan join()**

Fungsi pertama yang dapat kita gunakan untuk tujuan ini, `explode()`, memiliki prototipe berikut:

```
array explode(string separator, string input);
```

Fungsi ini mengambil input string dan membaginya menjadi beberapa bagian pada string pemisah yang ditentukan. Bagian-bagian tersebut dikembalikan dalam sebuah array. Untuk mendapatkan nama domain dari alamat email pelanggan dalam skrip kita, kita dapat menggunakan kode berikut:

```
$email_array = explode("@", $email);
```

Panggilan ke `explode()` ini membagi alamat email pelanggan menjadi dua bagian: nama pengguna, yang disimpan dalam `$email_array[0]`, dan nama domain, yang disimpan dalam `$email_array[1]`. Sekarang kita dapat menguji nama domain untuk menentukan asal pelanggan, lalu mengirimkan umpan balik mereka ke orang yang tepat:

```
if ($email_array[1]=="bigcustomer.com")
  $toaddress = "bob@bobsdomain.com";
else
  $toaddress = "feedback@bobsdomain.com";
```

Perhatikan jika domain ditulis dengan huruf kapital, ini tidak akan berfungsi. Kita dapat menghindari masalah ini dengan mengonversi domain menjadi semua huruf kapital atau semua huruf kecil, lalu memeriksa:

```
$email_array[1] = strtoupper ($email_array[1]);
```

Anda dapat membalikkan efek `explode()` menggunakan `implode()` atau `join()`, yang identik. Misalnya

```
$new_email = implode("@", $email_array);
```

Ini mengambil elemen array dari `$email_array` dan menggabungkannya dengan string yang dilewatkan pada parameter pertama. Pemanggilan fungsi ini sangat mirip dengan `explode()`, tetapi efeknya berlawanan.

**Menggunakan strtok()**

Tidak seperti `explode()`, yang memecah string menjadi semua bagiannya sekaligus, `strtok()` mengambil bagian (disebut token) dari string satu per satu. `strtok()` adalah alternatif yang berguna untuk menggunakan `explode()` guna memproses kata dari string satu per satu.

Prototipe untuk `strtok()` adalah string strtok(string input, string separator);

Pemisah dapat berupa karakter atau string karakter, tetapi perhatikan bahwa string input akan dibagi pada setiap karakter dalam string pemisah, bukan pada seluruh string pemisah (seperti yang dilakukan explode).

Memanggil `strtok()` tidak semudah yang terlihat dalam prototipe.

Untuk mendapatkan token pertama dari string, Anda memanggil `strtok()` dengan string yang ingin Anda tokenisasi, dan pemisah. Untuk mendapatkan token berikutnya dari string, Anda cukup memberikan satu parameter pemisah. Fungsi tersebut menyimpan pointer internalnya sendiri di tempatnya dalam string. Jika Anda ingin mengatur ulang pointer, Anda dapat memasukkan string ke dalamnya lagi.
`strtok()` biasanya digunakan sebagai berikut:

```
$token = strtok($feedback, “ “);
echo $token.”<br>”;
while ($token!=””)
{
  $token = strtok(“ “);
  echo $token.”<br>”;
};
```

Seperti biasa, sebaiknya periksa apakah pelanggan benar-benar mengetikkan umpan balik dalam formulir, misalnya menggunakan `empty().` Kami telah menghilangkan pemeriksaan ini demi singkatnya. Ini mencetak setiap token dari umpan balik pelanggan pada baris terpisah, dan mengulang hingga tidak ada lagi token. Perhatikan bahwa `strtok()` PHP tidak bekerja persis sama dengan yang ada di C.

Jika ada dua contoh pemisah dalam satu baris di string target Anda (dalam contoh ini, dua spasi dalam satu baris), `strtok()` mengembalikan string kosong. Anda tidak dapat membedakannya dari string kosong yang dikembalikan saat Anda sampai di akhir string target. Selain itu, jika salah satu token adalah 0, string kosong akan dikembalikan. Ini membuat `strtok()` PHP agak kurang berguna daripada yang ada di C. Anda sering kali lebih baik menggunakan fungsi `explode()` saja.

**Menggunakan `substr()`**

Fungsi `substr()` memungkinkan Anda mengakses substring antara titik awal dan akhir string yang diberikan. Ini tidak sesuai untuk contoh kita, tetapi dapat berguna saat Anda perlu mendapatkan bagian dari string format tetap.
Fungsi `substr()` memiliki prototipe berikut:

```
string substr(string string, int start, int [length] );
```

Fungsi ini mengembalikan substring yang disalin dari dalam string.

Kita akan melihat contoh-contoh penggunaan string pengujian ini:

```
$test = “Layanan pelanggan Anda sangat baik”;
```

Jika Anda memanggilnya dengan angka positif untuk *start* (hanya), Anda akan mendapatkan `start` dari posisi awal hingga akhir string. Misalnya,

```
substr($test, 1);
```

mengembalikan “layanan pelanggan kami sangat baik”. Perhatikan bahwa posisi string dimulai dari 0, seperti pada array.
Jika Anda memanggil `substr()` dengan start (hanya) negatif, Anda akan mendapatkan string dari akhir string dikurangi karakter awal hingga akhir string. Misalnya,

```
substr($test, -9);
```

mengembalikan “sangat baik”.

Parameter length dapat digunakan untuk menentukan sejumlah karakter yang akan dikembalikan (jika positif), atau karakter akhir dari urutan pengembalian (jika negatif). Misalnya,

```
substr($test, 0, 4);
```

mengembalikan empat karakter pertama dari string, yaitu, “Anda”. Kode berikut:

```
echo substr($test, 4, -13);
```

mengembalikan karakter antara karakter keempat dan karakter ketiga belas hingga terakhir, yaitu, "layanan pelanggan".

## 4.4 MEMBANDINGKAN STRING

Sejauh ini kita baru saja menggunakan == untuk membandingkan dua string untuk kesetaraan. Kita dapat melakukan beberapa perbandingan yang sedikit lebih canggih menggunakan PHP. Kita telah membaginya menjadi dua kategori: kecocokan parsial dan lainnya. Kita akan membahas yang lainnya terlebih dahulu, lalu masuk ke kecocokan parsial, yang akan kita perlukan untuk mengembangkan contoh Formulir Cerdas lebih lanjut.
Pengurutan String: `strcmp(), strcasecmp(), dan strnatcmp()`

Fungsi-fungsi ini dapat digunakan untuk mengurutkan string. Ini berguna saat mengurutkan data.

Prototipe untuk `strcmp()` adalah `int strcmp(string str1, string str2);`

Fungsi ini mengharapkan untuk menerima dua string, yang akan dibandingkan. Jika keduanya sama, maka akan mengembalikan 0. Jika `str1` muncul setelah (atau lebih besar dari) `str2` dalam urutan leksikografis, `strcmp()` akan mengembalikan angka yang lebih besar dari nol. Jika `str1` lebih kecil dari `str2`, `strcmp()` akan mengembalikan angka yang lebih kecil dari nol. Fungsi ini peka huruf besar/kecil.

**Fungsi `strcasecmp()` identik kecuali tidak peka huruf besar/kecil.**

Fungsi `strnatcmp()` dan kembarannya yang tidak peka huruf besar/kecil, `strnatcasecmp()`, ditambahkan dalam PHP 4. Fungsi-fungsi ini membandingkan string menurut "urutan alami", yang lebih seperti cara manusia melakukannya. Misalnya, `strcmp()` akan mengurutkan string "2" sebagai lebih besar daripada string "12" karena secara leksikografis lebih besar. `strnatcmp()` akan melakukannya sebaliknya.

**Menguji Panjang String dengan `strlen()`**

Kita dapat memeriksa panjang string dengan fungsi `strlen()`. Jika Anda memberikannya string, fungsi ini akan mengembalikan panjangnya. Misalnya, `strlen("hello")` mengembalikan 5.

Ini dapat digunakan untuk memvalidasi data input. Pertimbangkan alamat email pada formulir kita, yang disimpan di $email. Salah satu cara dasar untuk memvalidasi alamat email yang disimpan di $email adalah dengan memeriksa panjangnya. Menurut penalaran saya, panjang minimum alamat email adalah enam karakter misalnya, a@a.to jika Anda memiliki kode negara tanpa domain tingkat kedua, nama server satu huruf, dan alamat email satu huruf. Oleh karena itu, kesalahan dapat terjadi jika alamat tidak sepanjang ini:

```
if (strlen($email) < 6)
{
   echo “That email address is not valid”;
   exit; // finish execution of PHP script
}
```

Jelas, ini adalah cara yang sangat sederhana untuk memvalidasi informasi ini. Kita akan melihat cara yang lebih baik di bagian berikutnya.

**Mencocokkan dan Mengganti Substring dengan Fungsi String**

Biasanya kita ingin memeriksa apakah substring tertentu ada dalam string yang lebih besar. Pencocokan parsial ini biasanya lebih berguna daripada menguji kesetaraan. Dalam contoh Formulir Cerdas kita, kita ingin mencari frasa kunci tertentu dalam umpan balik pelanggan dan mengirim email ke departemen yang sesuai. Jika kita ingin mengirim email yang membahas tentang toko Bob kepada manajer ritel, kita ingin tahu apakah kata "toko" (atau turunannya) muncul dalam pesan tersebut.

Mengingat fungsi-fungsi yang telah kita lihat, kita dapat menggunakan `explode()` atau `strtok()` untuk mengambil kata-kata individual dalam pesan, lalu membandingkannya menggunakan operator == atau `cstrcmp()`. Namun, kita juga dapat melakukan hal yang sama dengan satu panggilan fungsi ke salah satu fungsi pencocokan string atau pencocokan ekspresi reguler. Fungsi-fungsi ini digunakan untuk mencari pola di dalam string. Kita akan melihat setiap rangkaian fungsi satu per satu.

Menemukan String dalam String: `strstr(), strchr(), strrchr(), stristr()` Untuk menemukan string di dalam string lain, Anda dapat menggunakan salah satu fungsi `strstr(), strchr(), strrchr(), atau stristr()`. Fungsi `strstr()` adalah yang paling umum, dan dapat digunakan untuk menemukan string atau karakter yang cocok dalam string yang lebih panjang. Perhatikan bahwa dalam PHP, fungsi `strchr()` sama persis dengan `strstr()`, meskipun namanya menyiratkan bahwa fungsi ini digunakan untuk menemukan karakter dalam string, mirip dengan versi C dari fungsi ini. Dalam PHP, salah satu fungsi ini dapat digunakan untuk menemukan string di dalam string, termasuk menemukan string yang hanya berisi satu karakter.

Prototipe untuk `strstr()` adalah sebagai berikut:

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

Anda meneruskan fungsi tersebut ke haystack yang akan dicari dan needle yang akan ditemukan. Jika kecocokan persis dengan needle ditemukan, fungsi tersebut mengembalikan haystack dari needle dan seterusnya, jika tidak, fungsi tersebut mengembalikan false. Jika needle muncul lebih dari satu kali, string yang dikembalikan akan dimulai dari kemunculan pertama needle.

Misalnya, dalam aplikasi Smart Form, kita dapat memutuskan ke mana akan mengirim email sebagai berikut:

```
$toaddress = “feedback@bobsdomain.com”; // the default value

// Change the $toaddress if the criteria are met
if (strstr($feedback, “shop”))
  $toaddress = “retail@bobsdomain.com”;
else if (strstr($feedback, “delivery”))
  $toaddress = “fulfilment@bobsdomain.com”;
else if (strstr($feedback, “bill”))
  $toaddress = “accounts@bobsdomain.com”;
```

Kode ini memeriksa kata kunci tertentu dalam umpan balik dan mengirimkan email ke orang yang tepat. Misalnya, jika umpan balik pelanggan berbunyi "Saya masih belum menerima pengiriman pesanan terakhir saya," string "pengiriman" akan terdeteksi dan umpan balik akan dikirim ke fulfilment@bobsdomain.com.

Ada dua varian pada strstr(). Varian pertama adalah stristr(), yang hampir identik tetapi tidak peka huruf besar/kecil. Ini akan berguna untuk aplikasi ini karena pelanggan mungkin mengetik "delivery", "Delivery", atau "DELIVERY". Varian kedua adalah strrchr(), yang juga hampir identik, tetapi akan mengembalikan tumpukan jerami dari kemunculan terakhir jarum dan seterusnya.

**Menemukan Posisi Substring: strpos(), strrpos()**

Fungsi strpos() dan strrpos() beroperasi dengan cara yang sama dengan strstr(), kecuali, alih-alih mengembalikan substring, mereka mengembalikan posisi numerik jarum di dalam tumpukan jerami.

Fungsi strpos() memiliki prototipe berikut:

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

Integer yang dikembalikan mewakili posisi kemunculan pertama jarum di dalam tumpukan jerami. Karakter pertama berada di posisi 0 seperti biasa. Misalnya, kode berikut akan menggemakan nilai 4 ke browser:

```
$test = “Hello world”; echo strpos($test, “o”);
```

Dalam kasus ini, kita hanya memasukkan satu karakter sebagai jarum, tetapi bisa berupa string dengan panjang berapa pun.

Parameter offset opsional digunakan untuk menentukan titik di dalam tumpukan jerami untuk memulai pencarian. Misalnya echo strpos($test, “o”, 5); Kode ini akan menggemakan nilai 7 ke browser karena PHP telah mulai mencari karakter o pada posisi 5, dan karena itu tidak melihat karakter pada posisi 4. Fungsi strrpos() hampir identik, tetapi akan mengembalikan posisi kemunculan terakhir jarum di tumpukan jerami. Tidak seperti strpos(), fungsi ini hanya bekerja dengan satu karakter jarum. Oleh karena itu, jika Anda memasukkannya sebagai string sebagai jarum, fungsi ini hanya akan menggunakan karakter pertama dari string tersebut untuk mencocokkan.

Dalam salah satu kasus ini, jika jarum tidak ada di dalam string, strpos() atau strrpos() akan mengembalikan false. Hal ini dapat menjadi masalah karena false dalam bahasa yang diketik lemah seperti PHP setara dengan 0, yaitu karakter pertama dalam suatu string. Anda dapat menghindari masalah ini dengan menggunakan operator === untuk menguji nilai pengembalian:

```
$result = strpos($test, “H”);
if ($result === false)
    echo “Not found”
else
    echo “Found at position 0”;
```

Perhatikan bahwa ini hanya akan berfungsi di PHP 4—di versi sebelumnya Anda dapat menguji false dengan menguji nilai yang dikembalikan untuk melihat apakah itu string (yaitu, false).

**Mengganti Substring: `str_replace()`, `substr_replace()`**

Fungsi find-and-replace dapat sangat berguna dengan string. Kami telah menggunakan find and replace di masa lalu untuk mempersonalisasi dokumen yang dihasilkan oleh PHP—misalnya dengan mengganti `<<name>>` dengan nama seseorang dan `<<address>>` dengan alamatnya. Anda juga dapat menggunakannya untuk menyensor istilah tertentu, seperti dalam aplikasi forum diskusi, atau bahkan dalam aplikasi Smart Form. Sekali lagi, Anda dapat menggunakan fungsi string atau fungsi ekspresi reguler untuk tujuan ini.

Fungsi string yang paling umum digunakan untuk penggantian adalah `str_replace()`. Fungsi ini memiliki prototipe berikut:

```
string str_replace(string needle, string new_needle, string haystack);
```

Fungsi ini akan mengganti semua contoh needle di haystack dengan new_needle. Misalnya, karena orang dapat menggunakan Formulir Cerdas untuk mengeluh, mereka mungkin menggunakan beberapa kata yang berwarna-warni. Sebagai programmer, kita dapat mencegah berbagai departemen Bob disalahgunakan dengan cara itu:

```
$feedback = str_replace($offcolor, "%!@*", $feedback);
```

Fungsi `substr_replace()` digunakan untuk menemukan dan mengganti substring tertentu dari sebuah string. Fungsi ini memiliki prototipe berikut:

```
string substr_replace(string string, string replacement, int start, int [length]
  );
```

Fungsi ini akan mengganti bagian dari `string string` dengan `string replacement`. Bagian mana yang diganti bergantung pada nilai parameter start dan length opsional.

Nilai start mewakili offset ke dalam string tempat penggantian harus dimulai. Jika nilainya 0 atau positif, maka itu adalah offset dari awal string; jika nilainya negatif, maka itu adalah offset dari akhir string. Misalnya, baris kode ini akan mengganti karakter terakhir `dalam $test` dengan "X":

```
$test = substr_replace($test, "X", -1);
```

Nilai panjang bersifat opsional dan mewakili titik di mana PHP akan berhenti mengganti. Jika Anda tidak memberikan nilai ini, string akan diganti dari awal hingga akhir string.

Jika panjangnya nol, string pengganti akan benar-benar disisipkan ke dalam string tanpa menimpa string yang ada. Panjang positif mewakili jumlah karakter yang ingin Anda ganti dengan string baru. Panjang negatif mewakili titik di mana Anda ingin berhenti mengganti karakter, dihitung dari akhir string.

## 4.5 EKSPRESI REGULER

PHP mendukung dua gaya sintaks ekspresi reguler: POSIX dan Perl. Gaya POSIX dari ekspresi reguler dikompilasi ke dalam PHP secara default, tetapi Anda dapat menggunakan gaya Perl dengan mengompilasinya di pustaka PCRE (Perl-compatible regular expression). Kami akan membahas gaya POSIX yang lebih sederhana, tetapi jika Anda sudah menjadi programmer Perl, atau ingin mempelajari lebih lanjut tentang PCRE, baca manual daring di *http://php.net*.

Sejauh ini, semua pencocokan pola yang telah kami lakukan telah menggunakan fungsi string. Kami telah dibatasi pada pencocokan persis, atau pencocokan substring persis. Jika Anda ingin melakukan pencocokan pola yang lebih rumit, Anda harus menggunakan ekspresi reguler. Ekspresi reguler sulit dipahami pada awalnya tetapi bisa sangat berguna.

**Dasar-dasar**

Ekspresi reguler adalah cara untuk menggambarkan pola dalam sepotong teks. Pencocokan persis (atau literal) yang telah kita lakukan sejauh ini adalah bentuk ekspresi reguler. Misalnya, sebelumnya kita mencari istilah ekspresi reguler seperti "toko" dan "pengiriman".

Pencocokan ekspresi reguler dalam PHP lebih seperti pencocokan `strstr()` daripada perbandingan yang sama karena Anda mencocokkan string di suatu tempat dalam string lain. (Itu bisa di mana saja dalam string itu kecuali Anda menentukan sebaliknya.) Misalnya, string "toko" cocok dengan ekspresi reguler "toko". Ia juga cocok dengan ekspresi reguler “h”, “ho”, dan seterusnya. Kita dapat menggunakan karakter khusus untuk menunjukkan meta-makna selain mencocokkan karakter secara tepat.

Misalnya, dengan karakter khusus Anda dapat menunjukkan bahwa suatu pola harus terjadi di awal atau akhir string, bahwa bagian dari suatu pola dapat diulang, atau bahwa karakter dalam suatu pola harus bertipe tertentu. Anda juga dapat mencocokkan berdasarkan kemunculan literal karakter khusus. Kita akan melihat masing-masing karakter ini.

**Set Karakter dan Kelas**

Penggunaan set karakter secara langsung memberikan ekspresi reguler kekuatan yang lebih besar daripada ekspresi pencocokan persis. Set karakter dapat digunakan untuk mencocokkan karakter apa pun dari jenis tertentu set karakter tersebut sebenarnya adalah semacam karakter pengganti. Pertama-tama, Anda dapat menggunakan karakter . sebagai karakter pengganti untuk karakter tunggal lainnya kecuali baris baru `(\n)`. Misalnya, ekspresi reguler ".at", cocok dengan string “cat”, “sat”, dan “mat”, di antara yang lainnya.

Pencocokan karakter pengganti semacam ini sering digunakan untuk pencocokan nama berkas dalam sistem operasi. Namun, dengan ekspresi reguler, Anda dapat lebih spesifik tentang jenis karakter yang ingin Anda cocokkan, dan Anda benar-benar dapat menentukan set tempat karakter tersebut harus berada. Dalam contoh sebelumnya, ekspresi reguler cocok dengan “cat” dan “mat”, tetapi juga cocok dengan “#at”. Jika Anda ingin membatasi ini pada karakter antara a dan z, Anda dapat menentukannya sebagai berikut: `[a-z]`

Apa pun yang diapit oleh karakter kurung siku khusus `[and]` adalah kelas karakter seperangkat karakter yang harus dimiliki oleh karakter yang cocok. Perhatikan bahwa ekspresi

dalam tanda kurung siku hanya cocok dengan satu karakter. Anda dapat mencantumkan satu set, misalnya

```
[aeiou]
```

berarti vokal apa pun.
Anda juga dapat menjelaskan rentang, seperti yang baru saja kita lakukan menggunakan karakter tanda hubung khusus, atau satu set rentang:

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

Set rentang ini mewakili karakter alfabet apa pun dalam huruf besar atau kecil.
Anda juga dapat menggunakan set untuk menentukan bahwa karakter tidak dapat menjadi anggota suatu set. Misalnya,

```
[^a-z]
```

cocok dengan karakter apa pun yang tidak berada di antara a dan z. Simbol sisipan berarti tidak saat ditempatkan di dalam tanda kurung siku. Simbol ini memiliki arti lain saat digunakan di luar tanda kurung siku, yang akan kita bahas sebentar lagi. Selain mencantumkan set dan rentang, sejumlah kelas karakter yang telah ditetapkan sebelumnya dapat digunakan dalam ekspresi reguler. Kelas-kelas ini ditunjukkan pada Tabel 4.3.

**Tabel 4.3 Kelas Karakter Untuk Digunakan Dalam Ekspresi Reguler Gaya Posix**

| *Kelas* | *Keterangan* |
|---|---|
| `[[:alnum:]]` | Karakter alfanumerik |
| `[[:alpha:]]` | Karakter alfabet |
| `[[:lower:]]` | Huruf kecil |
| `[[:upper:]]` | Huruf besar |
| `[[:digit:]]` | Digit desimal |
| `[[:xdigit:]]` | Digit heksadesimal |
| `[[:punct:]]` | Tanda baca |
| `[[:blank:]]` | Tab dan spasi |
| `[[:space:]]` | Karakter spasi |
| `[[:cntrl:]]` | Karakter kontrol |
| `[[:print:]]` | Semua karakter yang dapat dicetak |
| `[[:graph:]]` | Semua karakter yang dapat dicetak kecuali spasi |

**Pengulangan**

Seringkali Anda ingin menentukan bahwa mungkin ada beberapa kemunculan string atau kelas karakter tertentu. Anda dapat merepresentasikannya menggunakan dua karakter

khusus dalam ekspresi reguler Anda. Simbol * berarti pola dapat diulang nol kali atau lebih, dan simbol + berarti pola dapat diulang satu kali atau lebih. Simbol harus muncul langsung setelah bagian ekspresi yang diterapkan. Misalnya

```
[[:alnum:]]+ 4
```

berarti "setidaknya satu karakter alfanumerik."

**Subekspresi**

Sering kali berguna untuk dapat membagi ekspresi menjadi subekspresi sehingga Anda dapat, misalnya, merepresentasikan "setidaknya satu dari string ini diikuti oleh tepat satu dari string tersebut." Anda dapat melakukannya menggunakan tanda kurung, persis seperti yang Anda lakukan dalam ekspresi aritmatika. Misalnya, *(sangat )*besar,* cocok dengan *"besar", "sangat besar", "sangat sangat besar"*, dan seterusnya.

**Subekspresi yang Dihitung**

Kita dapat menentukan berapa kali sesuatu dapat diulang dengan menggunakan ekspresi numerik dalam kurung kurawal `( {} )`. Anda dapat menunjukkan jumlah pengulangan yang tepat `({3}` berarti tepat 3 pengulangan), rentang pengulangan `({2, 4}` berarti dari 2 hingga 4 pengulangan), atau rentang pengulangan yang tidak terbatas `({2,}` berarti setidaknya dua pengulangan).

Misalnya,

```
(very ){1, 3}
```

cocok dengan "sangat", "sangat sangat" dan "sangat sangat sangat".

**Menjangkar ke Awal atau Akhir String**

Anda dapat menentukan apakah subekspresi tertentu harus muncul di awal, akhir, atau keduanya. Ini cukup berguna ketika Anda ingin memastikan bahwa hanya istilah pencarian Anda dan tidak ada yang lain yang muncul dalam string. Simbol sisipan (^) digunakan di awal ekspresi reguler untuk menunjukkan bahwa itu harus muncul di awal string yang dicari, dan `$` digunakan di akhir ekspresi reguler untuk menunjukkan bahwa itu harus muncul di akhir.

Misalnya, ini mencocokkan bob di awal string:

```
^bob
```

Ini mencocokkan com di akhir string:

```
com$
```

Terakhir, ini mencocokkan karakter tunggal apa pun dari a hingga z, dalam string itu sendiri:

```
^[a-z]$
```

## 4.6 PERCABANGAN

Anda dapat merepresentasikan pilihan dalam ekspresi reguler dengan pipa vertikal. Misalnya, jika kita ingin mencocokkan com, edu, atau net, kita dapat menggunakan ekspresi:

```
(com)|(edu)|(net)
```

**Mencocokkan Karakter Spesial Literal**

Jika Anda ingin mencocokkan salah satu karakter spesial yang disebutkan di bagian ini, seperti `.`, `{`, atau `$`, Anda harus meletakkan garis miring `(\)` di depannya. Jika Anda ingin merepresentasikan garis miring, Anda harus menggantinya dengan dua garis miring, `\\`.

**Ringkasan Karakter Khusus**

Ringkasan semua karakter khusus ditampilkan dalam Tabel 4.4 dan 4.5. Tabel 4.4 menunjukkan arti karakter khusus di luar tanda kurung siku, dan Tabel 4.5 menunjukkan artinya saat digunakan di dalam tanda kurung siku.

**Tabel 4.4 Ringkasan Karakter Khusus Yang Digunakan Dalam Ekspresi Reguler Posix Di Luar Tanda Kurung Siku**

| *Karakter* | *Keterangan* |
|---|---|
| \ | Karakter Escape |
| ^ | Cocok di awal string |
| $ | Cocok di akhir string |
| . | . Cocok dengan karakter apa pun kecuali baris baru (\n) |
| \| | Awal cabang alternatif (dibaca sebagai OR) |
| ( | Awal subpola |
| ) | Akhir subpola |
| * | Ulangi 0 kali atau lebih |
| + | Ulangi 1 kali atau lebih |
| { | Awal kuantifier min/maks |
| } | Akhir kuantifier min/maks |

**Tabel 4.5 Ringkasan Karakter Khusus Yang Digunakan Dalam Ekspresi Reguler Posix Di Dalam Tanda Kurung Siku**

| *Karakter* | *Keterangan* |
|---|---|
| \ | Karakter Escape |
| ^ | NOT, hanya jika digunakan pada posisi awal |
| - | Digunakan untuk menentukan rentang karajter |

**Menyatukan Semuanya untuk Formulir Cerdas**

Setidaknya ada dua kemungkinan penggunaan ekspresi reguler dalam aplikasi Formulir Cerdas. Penggunaan pertama adalah untuk mendeteksi istilah tertentu dalam umpan balik pelanggan. Kita bisa sedikit lebih pintar tentang hal ini menggunakan ekspresi reguler. Dengan menggunakan fungsi string, kita harus melakukan tiga pencarian berbeda jika kita ingin

mencocokkan pada "toko", "layanan pelanggan", atau "ritel". Dengan ekspresi reguler, kita dapat mencocokkan ketiganya:

```
shop|customer service|retail
```

Penggunaan kedua adalah untuk memvalidasi alamat email pelanggan dalam aplikasi kami dengan mengodekan format standar alamat email dalam ekspresi reguler. Format tersebut mencakup beberapa karakter alfanumerik atau tanda baca, diikuti oleh simbol @, diikuti oleh serangkaian karakter alfanumerik dan tanda hubung, diikuti oleh titik, diikuti oleh lebih banyak karakter alfanumerik dan tanda hubung dan mungkin lebih banyak titik, hingga akhir rangkaian, yang dikodekan sebagai berikut:

```
^[a-zA-Z0-9_]+@[a-zA-Z0-9\-]+\.[a-zA-Z0-9\-\.]+$
```

Subekspresi `^[a-zA-Z0-9_]+` berarti "memulai rangkaian dengan setidaknya satu huruf, angka, atau garis bawah, atau beberapa kombinasi dari itu."

Simbol @ cocok dengan @ literal. Subekspresi `[a-zA-Z0-9\-]+` cocok dengan bagian pertama nama host termasuk karakter alfanumerik dan tanda hubung. Perhatikan bahwa kami telah menghilangkan tanda hubung karena itu adalah karakter khusus di dalam tanda kurung siku. Kombinasi \. cocok dengan literal; Subekspresi `[a-zA-Z0-9\-\.]+$` cocok dengan sisa nama domain, termasuk huruf, angka, tanda hubung, dan lebih banyak titik jika diperlukan, hingga akhir string.

Sedikit analisis menunjukkan bahwa Anda dapat menghasilkan alamat email yang tidak valid yang masih akan cocok dengan ekspresi reguler ini. Hampir mustahil untuk menemukan semuanya, tetapi ini akan sedikit memperbaiki situasi. Sekarang setelah Anda membaca tentang ekspresi reguler, kita akan melihat fungsi PHP yang menggunakannya.

**Menemukan Substring dengan Ekspresi Reguler**

Menemukan substring adalah aplikasi utama dari ekspresi reguler yang baru saja kita kembangkan. Dua fungsi yang tersedia dalam PHP untuk mencocokkan ekspresi reguler adalah `ereg()` dan `eregi()`. Fungsi `ereg()` memiliki prototipe berikut:

```
int ereg(string pattern, string search, array [matches]);
```

Fungsi ini mencari string pencarian, mencari kecocokan dengan ekspresi reguler dalam pola. Jika kecocokan ditemukan untuk subekspresi pola, kecocokan tersebut akan disimpan dalam array matches, satu subekspresi per elemen array.

Fungsi `eregi()` identik kecuali tidak peka huruf besar/kecil.

Kita dapat mengadaptasi contoh Formulir Cerdas untuk menggunakan ekspresi reguler sebagai berikut:

```
if (!eregi("^[a-zA-Z0-9_]+@[a-zA-Z0-9\-]+\.[a-zA-Z0-9\-\.]+$", $email))
{
  echo "That is not a valid email address. Please return to the"
    ." previous page and try again.";
  exit;
}
$toaddress = "feedback@bobsdomain.com"; // the default value
if (eregi("shop|customer service|retail", $feedback))
  $toaddress = "retail@bobsdomain.com";
else if (eregi("deliver.*|fulfil.*", $feedback))
  $toaddress = "fulfilment@bobsdomain.com";
else if (eregi("bill|account", $feedback))
  $toaddress = "accounts@bobsdomain.com";

if (eregi("bigcustomer\.com", $email))
  $toaddress = "bob@bobsdomain.com";
```

**Mengganti Substring dengan Ekspresi Reguler**

Anda juga dapat menggunakan ekspresi reguler untuk menemukan dan mengganti substring dengan cara yang sama seperti kita menggunakan `str_replace()`. Dua fungsi yang tersedia untuk ini adalah `ereg_replace()` dan `eregi_replace()`. Fungsi `ereg_replace()` memiliki prototipe berikut:

```
string ereg_replace(string pattern, string replacement, string search);
```

Fungsi ini mencari pola ekspresi reguler dalam string pencarian dan menggantinya dengan string replacement. Fungsi `eregi_replace()` identik, tetapi sekali lagi, tidak peka huruf besar/kecil.

**Memisahkan String dengan Ekspresi Reguler**

Fungsi ekspresi reguler lain yang berguna adalah `split()`, yang memiliki prototipe berikut:

```
array split(string pattern, string search, int [max]);
```

Fungsi ini membagi pencarian string menjadi substring pada pola ekspresi reguler dan mengembalikan substring dalam array. Bilangan bulat maksimum membatasi jumlah item yang dapat masuk ke dalam array.

Ini dapat berguna untuk memisahkan nama domain atau tanggal. Misalnya

```
$domain = "yallara.cs.rmit.edu.au";
$arr = split ("\.", $domain);
while (list($key, $value) = each ($arr)) echo "<br>".$value;
```

Ini membagi nama host menjadi lima komponennya dan mencetak masing-masing pada baris terpisah.

***Perbandingan Fungsi String dan Fungsi Ekspresi Reguler***

Secara umum, fungsi ekspresi reguler berjalan kurang efisien daripada fungsi string dengan fungsionalitas serupa. Jika aplikasi Anda cukup sederhana untuk menggunakan ekspresi string, lakukanlah.

# BAB 5
# MENGGUNAKAN KEMBALI KODE DAN MENULIS FUNGSI

Bab ini menjelaskan bagaimana penggunaan kembali kode menghasilkan kode yang lebih konsisten, andal, dan mudah dipelihara, dengan sedikit usaha. Kami akan menunjukkan teknik untuk memodulasi dan menggunakan kembali kode, dimulai dengan penggunaan sederhana require() dan include() untuk menggunakan kode yang sama di lebih dari satu halaman. Kami akan menjelaskan mengapa ini lebih unggul daripada penyertaan sisi server. Contoh yang diberikan akan mencakup penggunaan file penyertaan untuk mendapatkan tampilan dan nuansa yang konsisten di seluruh situs Anda. Kami akan menjelaskan cara menulis dan memanggil fungsi Anda sendiri menggunakan fungsi pembuatan halaman dan formulir sebagai contoh.

Dalam bab ini, kami akan membahas

- Mengapa menggunakan kembali kode?
- Menggunakan `require()` dan `include()`
- Pengantar fungsi
- Mengapa Anda harus mendefinisikan fungsi Anda sendiri?
- Struktur fungsi dasar
- Parameter
- Mengembalikan nilai
- Melewati referensi versus melewati nilai
- Cakupan
- Rekursi

## 5.1 ALASAN MENGGUNAKAN KEMBALI KODE

Salah satu tujuan insinyur perangkat lunak adalah menggunakan kembali kode sebagai pengganti menulis kode baru. Hal ini bukan karena para insinyur perangkat lunak adalah kelompok yang sangat malas. Penggunaan kembali kode yang ada mengurangi biaya, meningkatkan keandalan, dan meningkatkan konsistensi. Idealnya, proyek baru dibuat dengan menggabungkan komponen yang dapat digunakan kembali yang sudah ada, dengan pengembangan minimal dari awal.

**Biaya**

Selama masa pakai perangkat lunak, waktu yang dihabiskan untuk memelihara, memodifikasi, menguji, dan mendokumentasikannya akan jauh lebih banyak daripada waktu yang dihabiskan untuk menulisnya. Jika Anda menulis kode komersial, Anda harus berusaha membatasi jumlah baris yang digunakan dalam organisasi.

Salah satu cara paling praktis untuk mencapainya adalah dengan menggunakan kembali kode yang sudah digunakan daripada menulis versi yang sedikit berbeda dari kode yang sama untuk tugas baru. Lebih sedikit kode berarti biaya yang lebih rendah. Jika ada perangkat lunak yang memenuhi persyaratan proyek baru, belilah. Biaya untuk membeli

perangkat lunak yang sudah ada hampir selalu lebih rendah daripada biaya untuk mengembangkan produk yang setara. Namun, berhati-hatilah jika ada perangkat lunak yang sudah ada yang hampir memenuhi persyaratan Anda. Memodifikasi kode yang sudah ada bisa lebih sulit daripada menulis kode baru.

**Keandalan**

Jika suatu modul kode digunakan di suatu tempat di organisasi Anda, kode tersebut mungkin telah diuji secara menyeluruh. Meskipun hanya beberapa baris, ada kemungkinan bahwa jika Anda menulis ulang kode tersebut, Anda akan mengabaikan sesuatu yang disertakan oleh penulis asli atau sesuatu yang ditambahkan ke kode asli setelah ditemukan cacat selama pengujian. Kode yang sudah ada dan matang biasanya lebih andal daripada kode baru yang "hijau".

**Konsistensi**

Antarmuka eksternal ke sistem Anda, termasuk antarmuka pengguna dan antarmuka ke sistem luar, harus konsisten. Diperlukan kemauan dan upaya yang sungguh-sungguh untuk menulis kode baru yang konsisten dengan cara kerja bagian lain dari sistem. Jika Anda menggunakan kembali kode yang menjalankan bagian lain dari sistem, fungsionalitas Anda secara otomatis akan konsisten. Selain keuntungan ini, menggunakan kembali kode akan mengurangi pekerjaan Anda, selama kode asli bersifat modular dan ditulis dengan baik. Saat Anda bekerja, cobalah untuk mengenali bagian kode yang mungkin dapat Anda gunakan lagi di masa mendatang.

## 5.2 MENGGUNAKAN REQUIRE() DAN INCLUDE()

PHP menyediakan dua pernyataan yang sangat sederhana, namun sangat berguna, untuk memungkinkan Anda menggunakan kembali semua jenis kode. Dengan menggunakan pernyataan require() atau include(), Anda dapat memuat file ke dalam skrip PHP Anda. File tersebut dapat berisi apa pun yang biasanya Anda ketik dalam skrip termasuk pernyataan PHP, teks, tag HTML, fungsi PHP, atau kelas PHP. Pernyataan ini bekerja mirip dengan Server Side Includes yang ditawarkan oleh banyak server Web dan pernyataan #include dalam C atau C++.

**Menggunakan require()**

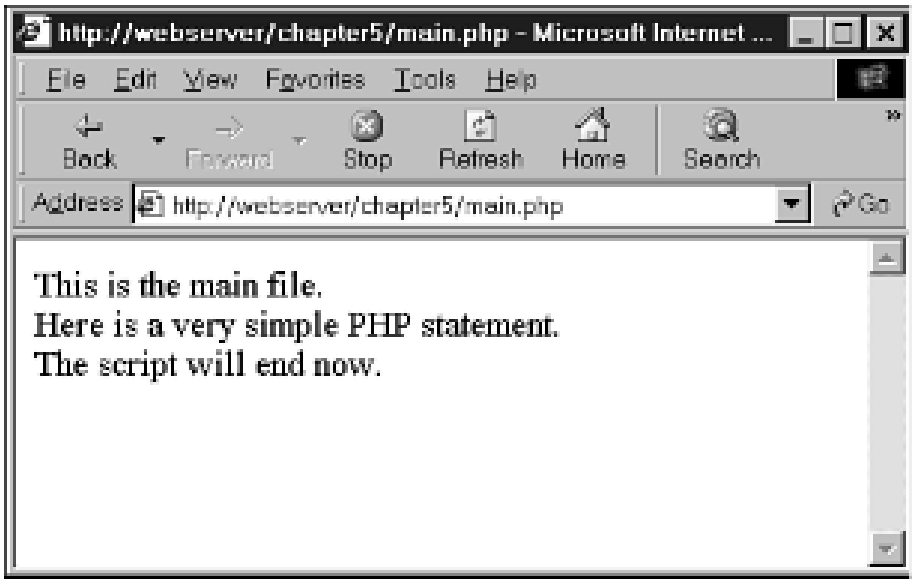
Kode berikut disimpan dalam file bernama reusable.php:

```
<?
  echo "Here is a very simple PHP statement.<BR>";
?>
```

Kode berikut disimpan dalam file bernama main.php:

```
<?
  echo "This is the main file.<BR>";
  require( "reusable.php" );
  echo "The script will end now.<BR>";
?>
```

Jika Anda memuat reusable.php, Anda mungkin tidak akan terkejut saat "Berikut adalah pernyataan PHP yang sangat sederhana." muncul di peramban Anda. Jika Anda memuat main.php, sesuatu yang sedikit lebih menarik terjadi. Output skrip ini ditunjukkan pada Gambar 5.1.




**Gambar 5.1 Output Main.Php Menunjukkan Hasil Pernyataan Require().**

Diperlukan file untuk menggunakan pernyataan require(). Pada contoh sebelumnya, kita menggunakan file bernama reusable.php. Saat kita menjalankan skrip, pernyataan `require()` `require( "reusable.php" );` diganti dengan konten file yang diminta, dan skrip kemudian dieksekusi. Ini berarti bahwa saat kita memuat main.php, skrip berjalan seolah-olah ditulis sebagai berikut:

```
<?
   echo “This is the main file.<BR>”;
   echo “Here is a very simple PHP statement.<BR>”;
   echo “The script will end now.<BR>”;
?>
```

Saat menggunakan require(), Anda perlu memperhatikan berbagai cara penanganan ekstensi nama file dan tag PHP.

**Ekstensi Nama File dan Require()**

PHP tidak melihat ekstensi nama file pada file yang diperlukan. Ini berarti Anda dapat memberi nama file apa pun yang Anda pilih selama Anda tidak akan memanggilnya secara langsung. Saat Anda menggunakan `require()` untuk memuat file, file tersebut secara efektif akan menjadi bagian dari file PHP dan dieksekusi sebagaimana adanya. Biasanya, pernyataan PHP tidak akan diproses jika ada dalam file yang disebut, misalnya, page.html. PHP biasanya

hanya dipanggil untuk mengurai file dengan ekstensi yang ditentukan seperti .php. Namun, jika Anda memuat page.html ini melalui pernyataan `require()`, PHP apa pun di dalamnya akan diproses. Oleh karena itu, Anda dapat menggunakan ekstensi apa pun yang Anda inginkan untuk menyertakan file, tetapi sebaiknya Anda mencoba untuk tetap menggunakan konvensi yang masuk akal, seperti .inc.

Satu hal yang perlu diperhatikan adalah jika file yang diakhiri dengan .inc atau ekstensi nonstandar lainnya disimpan di pohon dokumen Web dan pengguna langsung memuatnya di browser, mereka akan dapat melihat kode dalam teks biasa, termasuk kata sandi apa pun. Oleh karena itu, penting untuk menyimpan file yang disertakan di luar pohon dokumen, atau menggunakan ekstensi standar.

**Tag PHP dan require()**

Dalam contoh kami, file yang dapat digunakan kembali (reusable.php) ditulis sebagai berikut:

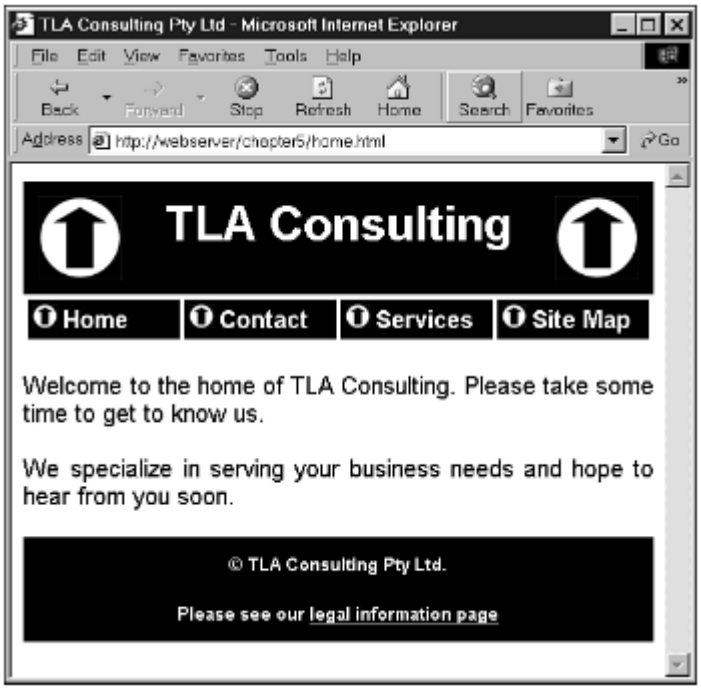
```
<?
  echo “Here is a very simple PHP statement.<BR>”;
?>
```

Kami menempatkan kode PHP di dalam file dalam tag PHP. Anda perlu melakukan ini jika Anda ingin kode PHP dalam file yang diperlukan diperlakukan sebagai kode PHP. Jika Anda tidak membuka tag PHP, kode Anda hanya akan diperlakukan sebagai teks atau HTML dan tidak akan dieksekusi.

**Menggunakan require() untuk Templat Situs Web**

Jika perusahaan Anda memiliki tampilan dan nuansa yang konsisten pada halaman di situs Web, Anda dapat menggunakan PHP untuk menambahkan templat dan elemen standar ke halaman menggunakan `require()`. Misalnya, situs Web perusahaan fiktif TLA Consulting memiliki sejumlah halaman dengan tampilan dan nuansa yang ditunjukkan pada Gambar 5.2. Ketika halaman baru diperlukan, pengembang dapat membuka halaman yang sudah ada, memotong teks yang sudah ada dari tengah file, memasukkan teks baru, dan menyimpan file dengan nama baru.

Pertimbangkan skenario ini: Situs Web telah ada sejak lama, dan sekarang ada puluhan, ratusan, atau bahkan ribuan halaman yang semuanya mengikuti gaya umum. Keputusan dibuat untuk mengubah sebagian tampilan standar bisa berupa perubahan kecil, seperti menambahkan alamat email di bagian bawah setiap halaman atau menambahkan satu entri baru ke menu navigasi. Apakah Anda ingin membuat perubahan kecil itu pada puluhan, ratusan, atau bahkan ribuan halaman?




**Gambar 5.2 Tla Consulting Memiliki Tampilan Dan Nuansa Standar Untuk Semua Halaman Web-Nya.**

Menggunakan kembali bagian-bagian HTML yang umum untuk semua halaman secara langsung merupakan pendekatan yang jauh lebih baik daripada memotong dan menempel pada puluhan, ratusan, atau bahkan ribuan halaman. Kode sumber untuk beranda (home.html) yang ditunjukkan pada Gambar 5.2 diberikan dalam Daftar 5.1.

**Daftar 5.1 Home.Html—Html Yang Menghasilkan Beranda Tla Consulting**

```
<html>
<head>
  <title>TLA Consulting Pty Ltd</title>
  <style>
    h1 {color:white; font-size:24pt; text-align:center;
      font-family:arial,sans-serif}
    .menu {color:white; font-size:12pt; text-align:center;
        font-family:arial,sans-serif; font-weight:bold}
    td {background:black}
    p {color:black; font-size:12pt; text-align:justify;
      font-family:arial,sans-serif}
    p.foot {color:white; font-size:9pt; text-align:center;
        font-family:arial,sans-serif; font-weight:bold}
    a:link,a:visited,a:active {color:white}
  </style>
</head>
<body>

<!-- page header -->
```

```
<table width=”100%” cellpadding = 12 cellspacing =0 border = 0>
<tr bgcolor = black>
  <td align = left><img src = “logo.gif”></td>
  <td>
    <h1>TLA Consulting</h1>
  </td>
  <td align = right><img src = “logo.gif”></td>
</tr>
</table>

<!-- menu -->
<table width = “100%” bgcolor = white cellpadding = 4 cellspacing = 4>
<tr >
  <td width = “25%”>
    <img src = “s-logo.gif”> <span class=menu>Home</span></td>
  <td width = “25%”>
    <img src = “s-logo.gif”> <span class=menu>Contact</span></td>
  <td width = “25%”>
    <img src = “s-logo.gif”> <span class=menu>Services</span></td>
  <td width = “25%”>
    <img src = “s-logo.gif”> <span class=menu>Site Map</span></td>
</tr>
</table>

<!-- page content -->
<p>Welcome to the home of TLA Consulting.
Please take some time to get to know us.</p>
<p>We specialize in serving your business needs
and hope to hear from you soon.</p>

<!-- page footer -->
<table width = “100%” bgcolor = black cellpadding = 12 border = 0>
  <tr>
    <td>
      <p class=foot>&copy; TLA Consulting Pty Ltd.</p>
        <p class=foot>Please see our <a href =””>legal information
        page</a></p>
    </td>
  </tr>
  </table>
</body>
</html>
```

Anda dapat melihat pada Daftar 5.1 bahwa sejumlah bagian kode yang berbeda ada dalam berkas ini. Kepala HTML berisi definisi *Cascading Style Sheet* (CSS) yang digunakan oleh halaman. Bagian yang diberi label "header halaman" menampilkan nama dan logo

perusahaan, "bilah menu" membuat bilah navigasi halaman, dan "konten halaman" adalah teks yang unik untuk halaman ini. Di bawahnya terdapat footer halaman. Kita dapat membagi berkas ini dan memberi nama bagian-bagiannya header.inc, home.php, dan footer.inc. Baik header.inc maupun footer.inc berisi kode yang akan digunakan kembali pada halaman lain. Berkas home.php adalah pengganti home.html, dan berisi konten halaman yang unik dan dua pernyataan require() seperti yang ditunjukkan pada Daftar 5.2.

**Daftar 5.2 Home.Php—Php Yang Menghasilkan Beranda Tla**

```
<?
  require(“header.inc”);
?>
  <!-- page content -->
  <p>Welcome to the home of TLA Consulting.
  Please take some time to get to know us.</p>
  <p>We specialize in serving your business needs
  and hope to hear from you soon.</p>
<?
  require(“footer.inc”);
?>
```

Pernyataan require() di home.php memuat header.inc dan footer.inc.

Seperti yang disebutkan, nama yang diberikan pada berkas-berkas ini tidak memengaruhi cara pemrosesannya saat kita memanggilnya melalui require(). Konvensi yang umum, tetapi sepenuhnya opsional, adalah menyebut berkas-berkas parsial yang akan disertakan dalam berkas-berkas lain dengan something.inc (di sini inc adalah singkatan dari include). Menempatkan berkas-berkas include Anda di direktori yang dapat dilihat oleh skrip Anda juga merupakan hal yang umum, dan merupakan ide yang bagus, tetapi tidak mengizinkan berkas-berkas include Anda dimuat satu per satu melalui server Web.

Ini akan mencegah berkas-berkas ini dimuat satu per satu yang akan a) mungkin menghasilkan beberapa galat jika ekstensi berkasnya adalah .php tetapi hanya berisi sebagian halaman atau skrip, atau b) memungkinkan orang untuk membaca kode sumber Anda jika Anda telah menggunakan ekstensi lain. Berkas header.inc berisi definisi CSS yang digunakan halaman, tabel yang menampilkan nama perusahaan dan menu navigasi seperti yang ditunjukkan pada Daftar 5.3. Berkas footer.inc berisi tabel yang menampilkan footer di bagian bawah setiap halaman. Berkas ini ditunjukkan pada Daftar 5.4.

**Daftar 5.3 Header.Inc—Header Yang Dapat Digunakan Kembali Untuk Semua Halaman Web TLA**

```
<html>
<head>
  <title>TLA Consulting Pty Ltd</title>
  <style>
    h1 {color:white; font-size:24pt; text-align:center;
```

```
        font-family:arial,sans-serif}
      .menu {color:white; font-size:12pt; text-align:center;
          font-family:arial,sans-serif; font-weight:bold}
      td {background:black}
      p {color:black; font-size:12pt; text-align:justify;
        font-family:arial,sans-serif}
      p.foot {color:white; font-size:9pt; text-align:center;
          font-family:arial,sans-serif; font-weight:bold}
      a:link,a:visited,a:active {color:white}
    </style>
  </head>
  <body>

    <!-- page header -->
    <table width=”100%” cellpadding = 12 cellspacing =0 border = 0>
    <tr bgcolor = black>
      <td align = left><img src = “logo.gif”></td>
    <td>
        <h1>TLA Consulting</h1>
    </td>
    <td align = right><img src = “logo.gif”></td>
  </tr>
  </table>

  <!-- menu -->
  <table width = “100%” bgcolor = white cellpadding = 4 cellspacing = 4>
    <tr >
      <td width = “25%”>
        <img src = “s-logo.gif”> <span class=menu>Home</span></td>
      <td width = “25%”>
        <img src = “s-logo.gif”> <span class=menu>Contact</span></td>
      <td width = “25%”>
        <img src = “s-logo.gif”> <span class=menu>Services</span></td>
    <td width = “25%”>
        <img src = “s-logo.gif”> <span class=menu>Site Map</span></td>
  </tr>
  </table>
```

**Daftar 5.4 Footer.Inc—Footer Yang Dapat Digunakan Kembali Untuk Semua Halaman Web TLA**

```
  <!-- page footer -->
    <table width = “100%” bgcolor = black cellpadding = 12 border = 0>
    <tr>
      <td>
        <p class=foot>&copy; TLA Consulting Pty Ltd.</p>
        <p class=foot>Please see our
```

```
                <a href =”legal.php3”>legal information page</a></p>
      </td>
    </tr>
    </table>
  </body>
  </html>
```

Pendekatan ini akan memberi Anda situs Web yang tampak konsisten dengan sangat mudah, dan Anda dapat membuat halaman baru dengan gaya yang sama dengan mengetik sesuatu seperti:

```
<? require(“header.inc”); ?>
Here is the content for this page
<? require(“footer.inc”); ?>
```

Yang terpenting, bahkan setelah kita membuat banyak halaman menggunakan header dan footer ini, mudah untuk mengubah berkas header dan footer. Baik Anda membuat perubahan teks kecil, atau mendesain ulang tampilan situs secara menyeluruh, Anda hanya perlu membuat perubahan satu kali. Kita tidak perlu mengubah setiap halaman di situs secara terpisah karena setiap halaman dimuat dalam berkas header dan footer. Contoh yang ditunjukkan di sini hanya menggunakan HTML biasa di badan, header, dan footer. Ini tidak perlu terjadi. Di dalam berkas-berkas ini, kita dapat menggunakan pernyataan PHP untuk membuat bagian-bagian halaman secara dinamis.

**Menggunakan auto_prepend_file dan auto_append_file**

Jika kita ingin menggunakan `require()` untuk menambahkan header dan footer ke setiap halaman, ada cara lain yang dapat kita lakukan. Dua opsi konfigurasi dalam berkas php.ini adalah auto_prepend_file dan auto_append_file. Dengan menyetelnya ke berkas header dan footer, kita memastikan bahwa keduanya akan dimuat sebelum dan sesudah setiap halaman.

Untuk Windows, setelannya akan menyerupai berikut ini:

```
auto_prepend_file = “c:/inetpub/include/header.inc”
auto_append_file = “c:/inetpub/include/footer.inc”
```

Untuk UNIX, bentuknya akan menyerupai berikut ini:

```
auto_prepend_file = “/home/username/include/header.inc”
auto_append_file = “/home/username/include/footer.inc
```

Jika kita menggunakan perintah ini, kita tidak perlu mengetik pernyataan `require()`, tetapi header dan footer tidak akan lagi menjadi opsional pada halaman. Jika Anda menggunakan server Web Apache, Anda dapat mengubah berbagai opsi konfigurasi seperti ini untuk direktori individual. Untuk melakukannya, server Anda harus diatur agar memungkinkan file konfigurasi

utamanya ditimpa. Untuk mengatur penambahan dan penambahan otomatis untuk direktori, buat file bernama .htaccess di direktori tersebut. File tersebut harus berisi dua baris berikut:

```
php3_auto_prepend_file /home/username/include/header.inc
php3_auto_append_file /home/username/include/footer.inc
```

Perhatikan bahwa sintaksnya sedikit berbeda dari opsi yang sama di php.ini, begitu pula php_value di awal baris: Tidak ada tanda sama dengan. Sejumlah pengaturan konfigurasi php.ini lainnya juga dapat diubah dengan cara ini.

Sintaks ini berubah dari PHP 3. Jika Anda menggunakan versi lama, baris dalam file .htaccess Anda akan menyerupai ini:

```
php3_auto_prepend_file /home/username/include/header.inc
php3_auto_append_file /home/username/include/footer.inc
```

Menetapkan opsi dalam file .htaccess daripada di php.ini atau file konfigurasi server Web Anda memberi Anda banyak fleksibilitas. Anda dapat mengubah pengaturan pada mesin bersama yang hanya memengaruhi direktori Anda. Anda tidak perlu memulai ulang server Web, dan Anda tidak memerlukan akses administrator. Kelemahan metode .htaccess adalah file dibaca dan diurai setiap kali file dalam direktori tersebut diminta daripada hanya sekali saat memulai, jadi ada penalti kinerja.

**Menggunakan `include()`**

Pernyataan `require()` dan `include()` sangat mirip, tetapi ada beberapa perbedaan penting dalam cara kerjanya. Pernyataan `include()` dievaluasi setiap kali pernyataan dieksekusi, dan tidak dievaluasi sama sekali jika pernyataan tidak dieksekusi. Pernyataan `require()` dieksekusi pertama kali pernyataan diurai, terlepas dari apakah blok kode yang memuatnya akan dieksekusi. Kecuali server Anda sangat sibuk, ini tidak akan membuat banyak perbedaan tetapi itu berarti bahwa kode dengan pernyataan `require()` di dalam pernyataan bersyarat tidak efisien.

```
if($variable == true)
{
require(“file1.inc”);
}
else
{
    require(“file2.inc”);
}
```

Kode ini akan memuat kedua berkas secara tidak perlu setiap kali skrip dijalankan, tetapi hanya menggunakan satu berkas tergantung pada nilai `$variabel`. Akan tetapi, jika kode ditulis menggunakan dua pernyataan `include()`, hanya satu berkas yang akan dimuat dan digunakan seperti pada versi berikut:

```
if($variable == true)
{
include(“file1.inc”);
}
else
{
    include(“file2.inc”);
}
```

Tidak seperti file yang dimuat melalui pernyataan `require()`, file yang dimuat melalui `include()` dapat mengembalikan nilai. Oleh karena itu, kita dapat memberi tahu bagian lain dari program tentang keberhasilan atau kegagalan dalam file yang disertakan, atau mengembalikan jawaban atau hasil.

Kita mungkin memutuskan bahwa kita sering membuka file dan daripada mengetik ulang baris kode yang sama setiap saat, kita menginginkan file include untuk membukanya bagi kita. File include kita mungkin disebut "openfile.inc" dan menyerupai yang berikut:

```
<?
@ $fp = fopen($name, $mode);
   if (!$fp)
   {
      echo “<p><strong> Oh No! I could not open the file.</strong></p>”;
      return 0;
   }
   else
   {
      return 1;
   }
?>
```

Berkas ini akan mencoba membuka berkas bernama `$name` menggunakan mode yang diberikan oleh `$mode`. Jika gagal, ia akan memberikan pesan kesalahan dan mengembalikan 0. Jika berhasil, ia akan mengembalikan 1 dan tidak menghasilkan output apa pun. Kita dapat memanggil berkas ini dalam skrip sebagai berikut:

```
$name = “file.txt”;
$mode = “r”;
$result = include(“openfile.php”);
if( $result == 1 )
{
   // do what we wanted to do with the file
   // refer to $fp created in the include file
}
```

Perhatikan bahwa kita dapat membuat variabel dalam berkas utama atau dalam berkas yang disertakan atau diperlukan, dan variabel tersebut akan ada di keduanya. Perilaku ini sama untuk pernyataan `require()` dan `include()`. Anda tidak dapat menggunakan `require()` dengan cara yang persis seperti yang ditunjukkan di sini karena Anda tidak dapat mengembalikan nilai dari pernyataan `require()`. Mengembalikan nilai dapat berguna karena memungkinkan Anda untuk memberi tahu bagian selanjutnya dari program Anda tentang kegagalan, atau untuk melakukan pemrosesan mandiri dan mengembalikan jawaban.

Fungsi adalah sarana yang bahkan lebih baik daripada berkas yang disertakan untuk memecah kode menjadi modul mandiri. Kita akan melihat fungsi selanjutnya. Jika Anda bertanya-tanya mengapa, mengingat keuntungan `include()` dibandingkan `require()`, Anda akan menggunakan `require()`, jawabannya adalah karena fungsi tersebut sedikit lebih cepat.

## 5.3 MENGGUNAKAN FUNGSI DALAM PHP

Fungsi ada di sebagian besar bahasa pemrograman. Fungsi digunakan untuk memisahkan kode yang melakukan satu tugas yang terdefinisi dengan baik. Hal ini membuat kode lebih mudah dibaca dan memungkinkan kita untuk menggunakan kembali kode tersebut setiap kali kita perlu melakukan tugas yang sama. Fungsi adalah modul kode mandiri yang mengatur antarmuka pemanggilan, melakukan beberapa tugas, dan secara opsional mengembalikan hasil. Dalam bab-bab sebelumnya, kami secara rutin memanggil sejumlah fungsi yang sudah ada di dalam PHP. Kami juga telah menulis beberapa fungsi sederhana tetapi mengabaikan detailnya. Di bagian ini, kami akan membahas pemanggilan dan penulisan fungsi secara lebih rinci.

**Memanggil Fungsi**

Baris berikut adalah pemanggilan fungsi yang paling sederhana:

```
function_name();
```

Ini memanggil fungsi bernama `function_name` yang tidak memerlukan parameter. Baris kode ini mengabaikan nilai apa pun yang mungkin dikembalikan oleh fungsi ini. Sejumlah fungsi dipanggil dengan cara yang persis seperti ini. Fungsi `phpinfo()` sering kali berguna dalam pengujian karena menampilkan versi PHP yang terinstal, informasi tentang PHP, pengaturan server Web, dan nilai berbagai variabel PHP dan server. Fungsi ini tidak mengambil parameter apa pun, dan kami biasanya mengabaikan nilai pengembaliannya, jadi pemanggilan ke `phpinfo()` akan menyerupai yang berikut:

```
phpinfo();
```

Sebagian besar fungsi memang memerlukan satu atau beberapa parameter informasi yang diberikan ke suatu fungsi saat dipanggil yang memengaruhi hasil eksekusi fungsi tersebut. Kita meneruskan parameter dengan menempatkan data atau nama variabel yang menyimpan data

di dalam tanda kurung setelah nama fungsi. Panggilan ke suatu fungsi dengan parameter menyerupai yang berikut:

```
function_name(“parameter”);
```

Dalam kasus ini, parameter yang kita gunakan adalah string yang hanya berisi kata parameter, tetapi panggilan berikut juga baik-baik saja tergantung pada fungsinya:

```
function_name(2);
function_name(7.993);
function_name($variable);
```

Pada baris terakhir, `$variabel` dapat berupa jenis variabel PHP apa pun, termasuk array.

Parameter dapat berupa jenis data apa pun, tetapi fungsi tertentu biasanya memerlukan jenis data tertentu. Anda dapat melihat berapa banyak parameter yang dibutuhkan suatu fungsi, apa yang direpresentasikan oleh masing-masing parameter, dan jenis data apa yang dibutuhkan masing-masing parameter dari prototipe fungsi.
Ini adalah prototipe untuk fungsi `fopen()`:

```
int fopen( string nama_file, string mode, [int use_include_path] );
```

Prototipe memberi tahu kita sejumlah hal, dan penting bagi Anda untuk mengetahui cara menafsirkan spesifikasi ini dengan benar. Dalam kasus ini, kata int sebelum nama fungsi memberi tahu kita bahwa fungsi ini akan mengembalikan bilangan bulat. Parameter fungsi berada di dalam tanda kurung. Dalam kasus `fopen()`, tiga parameter ditampilkan dalam prototipe. Parameter `nama_file` dan mode berupa string dan parameter berupa bilangan bulat. Tanda kurung siku di sekitar `use_include_path` menunjukkan bahwa parameter ini bersifat opsional. Kita dapat memberikan nilai untuk parameter opsional atau kita dapat memilih untuk mengabaikannya, dan nilai default akan digunakan.

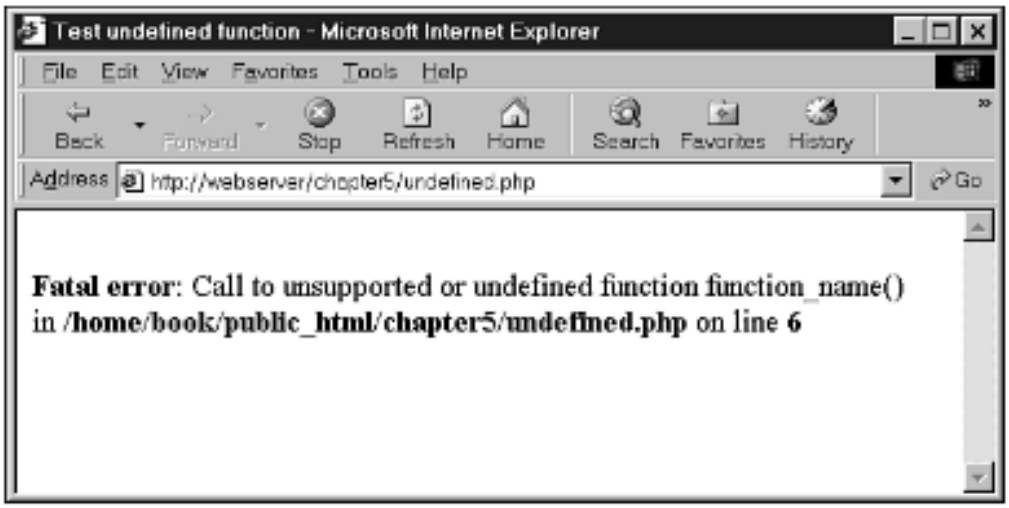
Setelah membaca prototipe untuk fungsi ini, kita mengetahui bahwa fragmen kode berikut akan menjadi panggilan yang valid ke `fopen()`:

```
$name = “myfile.txt”;
$openmode = “r”;
$fp = fopen($name, $openmode)
```

Kode ini memanggil fungsi bernama `fopen()`. Nilai yang dikembalikan oleh fungsi akan disimpan dalam variabel `$fp`. Kami memilih untuk meneruskan ke fungsi tersebut sebuah variabel bernama `$name` yang berisi string yang mewakili berkas yang ingin kami buka, dan sebuah variabel bernama `$openmode` yang berisi string yang mewakili mode di mana kami ingin membuka berkas tersebut. Kami memilih untuk tidak memberikan parameter ketiga yang opsional.

**Panggilan ke Fungsi yang Tidak Didefinisikan**

Jika Anda mencoba memanggil fungsi yang tidak ada, Anda akan mendapatkan pesan kesalahan seperti yang ditunjukkan pada Gambar 5.3. Pesan galat yang diberikan PHP biasanya sangat berguna. Pesan ini memberi tahu kita secara pasti di berkas mana galat itu terjadi, di baris skrip mana galat itu terjadi, dan nama fungsi yang kita coba panggil. Ini akan memudahkan kita untuk menemukan dan memperbaikinya.




**Gambar 5.3 Pesan Galat Ini Merupakan Hasil Pemanggilan Fungsi Yang Tidak Ada.**

Ada dua hal yang perlu diperiksa jika Anda melihat pesan galat ini:

1. Apakah nama fungsi dieja dengan benar?
2. Apakah fungsi itu ada dalam versi PHP yang Anda gunakan?

Tidak selalu mudah untuk mengingat bagaimana nama fungsi dieja. Misalnya, beberapa nama fungsi yang terdiri dari dua kata memiliki garis bawah di antara kata-katanya dan beberapa tidak. Fungsi `stripslashes()` menjalankan dua kata itu bersama-sama, sedangkan fungsi `strip_tags()` memisahkan kata-kata itu dengan garis bawah. Salah mengeja nama fungsi dalam pemanggilan fungsi akan menghasilkan galat seperti yang ditunjukkan pada Gambar 5.3. Banyak fungsi yang digunakan dalam buku ini tidak ada di PHP 3.0 karena buku ini mengasumsikan bahwa Anda menggunakan setidaknya PHP 4.0.

Dalam setiap versi Terbaru, fungsi baru didefinisikan dan jika Anda menggunakan versi lama, fungsionalitas dan kinerja yang ditambahkan membenarkan pemutakhiran. Untuk melihat kapan fungsi tertentu ditambahkan, Anda dapat melihat manual daring di www.php.net. Mencoba memanggil fungsi yang tidak dideklarasikan dalam versi yang Anda jalankan akan mengakibatkan galat seperti yang ditunjukkan pada Gambar 5.3.

**Nama Fungsi dan Huruf Besar/Kecil**

Perlu dicatat bahwa panggilan ke fungsi tidak peka huruf besar/kecil, jadi memanggil `function_name(), Function_Name(),` atau `FUNCTION_NAME()` semuanya valid dan akan memiliki hasil yang sama. Anda bebas menggunakan huruf besar/kecil dengan cara apa pun yang menurut Anda mudah dibaca, tetapi Anda harus berusaha untuk konsisten. Konvensi

yang digunakan dalam buku ini, dan sebagian besar dokumentasi PHP lainnya, adalah menggunakan semua huruf kecil.

Penting untuk dicatat bahwa nama fungsi berperilaku berbeda dengan nama variabel. Nama variabel peka huruf besar/kecil, jadi `$Name` dan `$name` adalah dua variabel terpisah, tetapi `Name()` dan `name()` adalah fungsi yang sama. Pada bab sebelumnya, Anda telah melihat banyak contoh menggunakan beberapa fungsi bawaan PHP. Namun, kekuatan sebenarnya dari bahasa pemrograman berasal dari kemampuan membuat fungsi Anda sendiri.

**Mengapa Anda Harus Menentukan Fungsi Anda Sendiri?**

Fungsi bawaan PHP memungkinkan Anda berinteraksi dengan file, menggunakan basis data, membuat grafik, dan terhubung ke server lain. Namun, dalam karier Anda, akan ada banyak waktu ketika Anda perlu melakukan sesuatu yang tidak diramalkan oleh pembuat bahasa tersebut. Untungnya, Anda tidak terbatas pada penggunaan fungsi bawaan karena Anda dapat menulis fungsi Anda sendiri untuk melakukan tugas apa pun yang Anda suka. Kode Anda mungkin merupakan campuran dari fungsi yang ada yang dikombinasikan dengan logika Anda sendiri untuk melakukan tugas bagi Anda.

Jika Anda menulis blok kode untuk tugas yang mungkin ingin Anda gunakan kembali di sejumlah tempat dalam skrip atau dalam sejumlah skrip, sebaiknya Anda mendeklarasikan blok tersebut sebagai fungsi. Mendeklarasikan fungsi memungkinkan Anda menggunakan kode Anda sendiri dengan cara yang sama seperti fungsi bawaan. Anda cukup memanggil fungsi Anda dan memberinya parameter yang diperlukan. Ini berarti Anda dapat memanggil dan menggunakan kembali fungsi yang sama berkali-kali di seluruh skrip Anda.

## 5.4 STRUKTUR FUNGSI DASAR

Deklarasi fungsi membuat atau mendeklarasikan fungsi baru. Deklarasi dimulai dengan kata kunci fungsi, menyediakan nama fungsi, parameter yang diperlukan, dan berisi kode yang akan dieksekusi setiap kali fungsi ini dipanggil. Berikut ini adalah deklarasi fungsi trivial:

```
function my_function()
{
  echo “My function was called”;
}
```

Deklarasi fungsi ini diawali dengan `function`, sehingga pembaca manusia dan parser PHP mengetahui bahwa yang mengikutinya akan menjadi fungsi yang ditentukan pengguna. Nama fungsi tersebut adalah `my_function`. Kita dapat memanggil fungsi baru kita dengan pernyataan berikut: `my_function()`; Seperti yang mungkin Anda duga, memanggil fungsi ini akan menghasilkan teks “Fungsi saya dipanggil.” yang muncul di browser penampil.

Fungsi bawaan tersedia untuk semua skrip PHP, tetapi jika Anda mendeklarasikan fungsi Anda sendiri, fungsi tersebut hanya tersedia untuk skrip tempat fungsi tersebut dideklarasikan. Sebaiknya Anda memiliki satu file yang berisi fungsi yang umum digunakan. Anda kemudian dapat memiliki pernyataan `require()` di semua skrip Anda untuk membuat

fungsi Anda tersedia. Di dalam suatu fungsi, kurung kurawal mengapit kode yang melakukan tugas yang Anda perlukan. Di antara kurung kurawal ini, Anda dapat memiliki apa pun yang sah di tempat lain dalam skrip PHP termasuk pemanggilan fungsi, deklarasi variabel atau fungsi baru, pernyataan `require()` atau `include()`, dan HTML biasa. Jika kita ingin keluar dari PHP dalam suatu fungsi dan mengetik HTML biasa, kita melakukannya dengan cara yang sama seperti di bagian lain skrip dengan tag penutup PHP diikuti oleh HTML. Berikut ini adalah modifikasi legal dari contoh sebelumnya dan menghasilkan output yang sama:

```
<?
    function my_function()
    {
?>
My function was called
<?
    }
?>
```

Perhatikan bahwa kode PHP disertakan dalam tag pembuka dan penutup PHP yang cocok. Untuk sebagian besar contoh fragmen kode kecil dalam buku ini, kami tidak menampilkan tag ini. Tag-tag ini ditampilkan di sini karena diperlukan dalam contoh serta di atas dan di bawahnya.

## 5.5 MEMBERI NAMA FUNGSI ANDA

Hal terpenting yang perlu dipertimbangkan saat memberi nama fungsi Anda adalah nama tersebut harus singkat tetapi deskriptif. Jika fungsi Anda membuat tajuk halaman, `pageheader()` atau `page_header()` mungkin merupakan nama yang bagus.

Beberapa batasannya adalah sebagai berikut:

- ✓ Fungsi Anda tidak boleh memiliki nama yang sama dengan fungsi yang sudah ada.
- ✓ Nama fungsi Anda hanya boleh berisi huruf, angka, dan garis bawah.
- ✓ Nama fungsi Anda tidak boleh dimulai dengan angka.

Banyak bahasa yang memungkinkan Anda untuk menggunakan kembali nama fungsi. Fitur ini disebut dengan fungsi yang berlebihan. Akan tetapi, PHP tidak mendukung fungsi yang berlebihan, jadi fungsi Anda tidak dapat memiliki nama yang sama dengan fungsi bawaan atau fungsi yang ditentukan pengguna yang sudah ada.

Perhatikan bahwa meskipun setiap skrip PHP mengetahui semua fungsi bawaan, fungsi yang ditentukan pengguna hanya ada dalam skrip tempat fungsi tersebut dideklarasikan. Ini berarti Anda dapat menggunakan kembali nama fungsi dalam berkas yang berbeda, tetapi hal ini akan menyebabkan kebingungan dan harus dihindari.

Nama fungsi berikut ini sah:

```
name()
name2()
name_three()
```

```
_namefour()
```

Berikut ini adalah hal-hal yang ilegal:

```
5name()
name-six()
fopen()
```

(Yang terakhir akan sah jika belum ada.)

**Parameter**

Agar dapat berfungsi, sebagian besar fungsi memerlukan satu atau beberapa parameter. Parameter memungkinkan Anda memasukkan data ke dalam suatu fungsi. Berikut adalah contoh fungsi yang memerlukan parameter. Fungsi ini mengambil array satu dimensi dan menampilkannya sebagai tabel.

```
function create_table($data)
{
  echo “<table border = 1>”;
  reset($data); // Remember this is used to point to the beginning
  $value = current($data);
  while ($value)
  {
    echo “<tr><td>$value</td></tr>\n”;
    $value = next($data);
  }
  echo “</table>”;
}
```

Jika kita memanggil fungsi `create_table()` sebagai berikut:

```
$my_array = array(“Line one.”,”Line two.”,”Line three.”);
create_table($my_array);
```

kita akan melihat output seperti yang ditunjukkan pada Gambar 5.4.


**Gambar 5.4 Tabel Html Ini Adalah Hasil Pemanggilan `Create_Table()`.**

Dengan meneruskan parameter, kita dapat memasukkan data yang dibuat di luar fungsi dalam kasus ini, array `$data` ke dalam fungsi. Seperti halnya fungsi bawaan, fungsi yang ditentukan pengguna dapat memiliki beberapa parameter dan parameter opsional. Kita dapat meningkatkan fungsi `create_table()` dengan berbagai cara, tetapi salah satu caranya adalah dengan mengizinkan pemanggil untuk menentukan batas atau atribut lain dari tabel. Berikut adalah versi fungsi yang telah ditingkatkan. Fungsi ini sangat mirip, tetapi memungkinkan kita untuk secara opsional mengatur lebar batas tabel, spasi sel, dan pengisi sel.

```
function create_table2( $data, $border =1, $cellpadding = 4, $cellspacing = 4 )
{
   echo “<table border = $border cellpadding = $cellpadding”
        .” cellspacing = $cellspacing>”;
   reset($data);
   $value = current($data);
   while ($value)
   {
      echo “<tr><td>$value</td></tr>\n”;
      $value = next($data);
   }
   echo “</table>”;
}
```

Parameter pertama untuk `create_table2()` masih diperlukan. Tiga parameter berikutnya

Parameter pertama untuk `create_table2()` masih diperlukan. Tiga parameter berikutnya bersifat opsional karena kami telah menetapkan nilai default untuk parameter tersebut. Kita dapat membuat output yang sangat mirip dengan yang ditunjukkan pada Gambar 5.4 dengan panggilan ke `create_table2()` ini.

```
create_table2($my_array);
```

Jika kita ingin data yang sama ditampilkan dalam gaya yang lebih menyebar, kita dapat memanggil fungsi baru kita sebagai berikut:

```
create_table2($my_array, 3, 8, 8);
```

Nilai opsional tidak perlu disediakan semuanya kita dapat menyediakan beberapa dan mengabaikan beberapa. Parameter akan ditetapkan dari kiri ke kanan. Ingatlah bahwa Anda tidak dapat mengabaikan satu parameter opsional tetapi menyertakan yang tercantum kemudian. Dalam contoh ini, jika Anda ingin meneruskan nilai untuk cellspacing, Anda juga harus meneruskan satu untuk cellpadding. Ini adalah penyebab umum kesalahan pemrograman. Ini juga merupakan alasan mengapa parameter opsional ditetapkan terakhir dalam daftar parameter apa pun.
Pemanggilan fungsi berikut:

```
create_table2($my_array, 3);
```

sepenuhnya sah, dan akan mengakibatkan $border ditetapkan ke 3 dan `$cellpadding` dan `$cellspacing` ditetapkan ke default-nya.

**Cakupan**

Anda mungkin telah memperhatikan bahwa ketika kita perlu menggunakan variabel di dalam file yang diperlukan atau disertakan, kita cukup mendeklarasikannya dalam skrip sebelum pernyataan `require()` atau `include()`, tetapi ketika menggunakan fungsi, kita secara eksplisit meneruskan variabel tersebut ke dalam fungsi. Hal ini sebagian karena tidak ada mekanisme untuk secara eksplisit meneruskan variabel ke berkas yang diperlukan atau disertakan, dan sebagian karena cakupan variabel berperilaku berbeda untuk fungsi. Cakupan variabel mengontrol di mana variabel tersebut terlihat dan dapat digunakan. Bahasa pemrograman yang berbeda memiliki aturan yang berbeda yang mengatur cakupan variabel. PHP memiliki aturan yang cukup sederhana:

- Variabel yang dideklarasikan di dalam fungsi berada dalam cakupan dari pernyataan di mana variabel tersebut dideklarasikan hingga kurung tutup di akhir fungsi. Ini disebut cakupan fungsi. Variabel-variabel ini disebut variabel lokal.
- Variabel yang dideklarasikan di luar fungsi berada dalam cakupan dari pernyataan di mana variabel tersebut dideklarasikan hingga akhir berkas, tetapi tidak di dalam fungsi. Ini disebut cakupan global. Variabel-variabel ini disebut variabel global.
- Penggunaan pernyataan `require()` dan `include()` tidak memengaruhi cakupan. Jika pernyataan tersebut digunakan di dalam fungsi, cakupan fungsi berlaku. Jika tidak di dalam fungsi, cakupan global berlaku.
- Kata kunci global dapat digunakan untuk secara manual menentukan bahwa variabel yang didefinisikan atau digunakan di dalam fungsi akan memiliki cakupan global.
- Variabel dapat dihapus secara manual dengan memanggil `unset($variable_name)`. Suatu variabel tidak lagi berada dalam cakupan jika telah dihapus.

Contoh-contoh berikut mungkin dapat membantu memperjelas berbagai hal.

Kode berikut tidak menghasilkan output. Di sini kita mendeklarasikan variabel yang disebut $var di dalam fungsi fn(). Karena variabel ini dideklarasikan di dalam suatu fungsi, variabel ini memiliki cakupan fungsi dan hanya ada di tempat variabel ini dideklarasikan, hingga akhir fungsi. Ketika kita merujuk lagi ke $var di luar fungsi, variabel baru yang disebut $var akan dibuat. Variabel baru ini memiliki cakupan global, dan akan terlihat hingga akhir file. Sayangnya, jika satu-satunya pernyataan yang kita gunakan dengan variabel $var baru ini adalah echo, variabel ini tidak akan pernah memiliki nilai.

```
function fn()
{
  $var = “contents”;
}
echo $var;
```

Contoh berikut adalah kebalikannya. Kita mendeklarasikan variabel di luar fungsi, lalu mencoba menggunakannya di dalam fungsi.

```
function fn()
{
  echo “inside the function, \$var = “.$var.”<br>”;
  $var = “contents2”;
  echo “inside the function, \$var = “.$var.”<br>”;
}
$var = “contents 1”;
fn();
echo “outside the function, \$var = “.$var.”<br>”;
```

Output dari kode ini akan menjadi sebagai berikut:

```
inside the function, $var =
inside the function, $var = contents 2
outside the function, $var = contents 1
```

Fungsi tidak akan dieksekusi hingga dipanggil, jadi pernyataan pertama yang dieksekusi adalah $var = “contents 1”;. Ini akan membuat variabel bernama $var, dengan cakupan global dan konten “contents 1”. Pernyataan berikutnya yang dieksekusi adalah panggilan ke fungsi fn(). Baris-baris di dalam pernyataan dieksekusi secara berurutan. Baris pertama dalam fungsi merujuk ke variabel bernama $var. Saat baris ini dieksekusi, fungsi tidak dapat melihat $var sebelumnya yang kita buat, jadi fungsi akan membuat yang baru dengan cakupan fungsi dan menggemakannya. Ini akan membuat baris pertama output.

Baris berikutnya dalam fungsi menetapkan konten $var menjadi “contents 2”. Karena kita berada di dalam fungsi, baris ini mengubah nilai $var lokal, bukan yang global.

Baris kedua output memverifikasi bahwa perubahan ini berhasil. Fungsi sekarang telah selesai, jadi baris terakhir skrip dieksekusi. Pernyataan gema ini menunjukkan bahwa nilai variabel global tidak berubah. Jika kita ingin suatu variabel yang dibuat dalam suatu fungsi menjadi global, kita dapat menggunakan kata kunci global sebagai berikut:

```
function fn()
{
  global $var;
  $var = “contents”;
  echo “inside the function, \$var = “.$var.”<br>”;
}

fn();
echo “outside the function, \$var = “.$var.”<br>”;
```

Dalam contoh ini, variabel `$var` secara eksplisit didefinisikan sebagai global, yang berarti bahwa setelah fungsi dipanggil, variabel tersebut akan berada di luar fungsi tersebut. Output dari skrip ini akan menjadi sebagai berikut:

```
inside the function, $var = contents
outside the function, $var = contents
```

Perhatikan bahwa variabel tersebut berada dalam cakupan dari titik di mana baris global `$var`; dieksekusi. Kita dapat mendeklarasikan fungsi di atas atau di bawah tempat kita memanggilnya. (Perhatikan bahwa cakupan fungsi sangat berbeda dari cakupan variabel!) Lokasi deklarasi fungsi tidak penting, yang penting adalah tempat kita memanggil fungsi dan karenanya mengeksekusi kode di dalamnya.

Gunakan kata kunci global di bagian atas skrip saat variabel pertama kali digunakan untuk mendeklarasikan bahwa variabel tersebut harus berada dalam cakupan di seluruh skrip. Ini mungkin penggunaan kata kunci global yang lebih umum. Lihat dari contoh sebelumnya bahwa sangat sah untuk menggunakan kembali nama variabel untuk variabel di dalam dan di luar fungsi tanpa gangguan di antara keduanya. Namun, ini umumnya merupakan ide yang buruk karena tanpa membaca kode dengan saksama dan memikirkan cakupan, orang mungkin berasumsi bahwa variabel tersebut adalah satu dan sama.

**Pass by Reference Versus Pass by Value**

Jika kita ingin menulis fungsi bernama `increment()` yang memungkinkan kita untuk menambah nilai, kita mungkin tergoda untuk mencoba menulisnya sebagai berikut:

```
function increment($value, $amount = 1)
  {
    $value = $value +$amount;
  }
```

Kode ini tidak akan berguna. Output dari kode pengujian berikut akan menjadi “10”.

```
$value = 10;
increment ($value);
echo $value;
```

Isi `$value` tidak berubah.

Hal ini dikarenakan aturan cakupan. Kode ini membuat variabel bernama `$value` yang berisi 10. Kemudian kode ini memanggil fungsi `increment()`. Variabel `$value` dalam fungsi dibuat saat fungsi dipanggil. Satu ditambahkan ke dalamnya, jadi nilai `$value` adalah 11 di dalam fungsi, hingga fungsi berakhir, dan kita kembali ke kode yang memanggilnya. Dalam kode ini, variabel `$value` adalah variabel yang berbeda, dengan cakupan global, dan karenanya tidak berubah. cara untuk mengatasinya adalah dengan mendeklarasikan `$value` dalam fungsi sebagai global, tetapi ini berarti bahwa untuk menggunakan fungsi ini, variabel yang ingin kita tambahkan harus diberi nama `$value`. Pendekatan yang lebih baik adalah dengan menggunakan *pass by reference.*

Cara normal pemanggilan parameter fungsi disebut *pass by value*. Saat Anda memberikan parameter, variabel baru dibuat yang berisi nilai yang diberikan. Ini adalah salinan dari yang asli. Anda bebas mengubah nilai ini dengan cara apa pun, tetapi nilai variabel asli di luar fungsi tetap tidak berubah. Pendekatan yang lebih baik adalah menggunakan *pass by reference*. Di sini, saat parameter dilewatkan ke fungsi, alih-alih membuat variabel baru, fungsi tersebut menerima referensi ke variabel asli.

Referensi ini memiliki nama variabel, dimulai dengan tanda dolar, dan dapat digunakan dengan cara yang sama persis seperti variabel lain. Perbedaannya adalah alih-alih memiliki nilai sendiri, ia hanya merujuk ke variabel asli. Setiap modifikasi yang dilakukan pada referensi juga memengaruhi variabel asli. Kami menetapkan bahwa parameter harus menggunakan pass by reference dengan menempatkan ampersand (&) sebelum nama parameter dalam definisi fungsi. Tidak ada perubahan yang diperlukan dalam pemanggilan fungsi. Contoh `increment()` sebelumnya dapat dimodifikasi agar satu parameter dilewatkan dengan referensi, dan akan berfungsi dengan benar.

```
function increment(&$value, $amount = 1)
{
   $value = $value +$amount;
}
```

Sekarang kita memiliki fungsi yang berfungsi, dan bebas memberi nama variabel yang ingin kita tambahkan sesuai keinginan. Seperti yang telah disebutkan, membingungkan bagi manusia untuk menggunakan nama yang sama di dalam dan di luar fungsi, jadi kita akan memberi variabel dalam skrip utama nama baru. Kode pengujian berikut sekarang akan menampilkan 10 sebelum panggilan ke `increment()`, dan 11 setelahnya.

```
$a = 10;
echo $a;
increment ($a);
echo $a ;
```

## 5.6 KEMBALI DARI FUNGSI

Kata kunci return menghentikan eksekusi suatu fungsi. Ketika suatu fungsi berakhir karena semua pernyataan telah dieksekusi atau kata kunci return digunakan, eksekusi kembali ke pernyataan setelah fungsi dipanggil. Jika Anda memanggil fungsi berikut, hanya pernyataan echo pertama yang akan dieksekusi.

```
function test_return()
{
  echo “This statement will be executed”;
  return;
  echo “This statement will never be executed”;
}
```

Jelas, ini bukan cara yang sangat berguna untuk menggunakan return. Biasanya, Anda hanya ingin melakukan return dari tengah fungsi sebagai respons terhadap kondisi yang terpenuhi. Kondisi error adalah alasan umum untuk menggunakan pernyataan return guna menghentikan eksekusi fungsi sebelum selesai. Misalnya, jika Anda menulis fungsi untuk mencari tahu mana dari dua angka yang lebih besar, Anda mungkin ingin keluar jika ada angka yang hilang.

```
function larger( $x, $y )
{
  if (!isset($x)||!isset($y))
  {
    echo “this function requires two numbers”;
    return;
  }
  if ($x>=$y)
    echo $x;
  else
    echo $y;
}
```

Fungsi bawaan `isset()` memberi tahu Anda apakah suatu variabel telah dibuat dan diberi nilai. Dalam kode ini, kita akan memberikan pesan kesalahan dan mengembalikan jika salah satu parameter belum ditetapkan dengan nilai. Kita menguji ini dengan menggunakan `!isset()`, yang berarti “BUKAN `isset()`”, sehingga pernyataan if dapat dibaca sebagai “jika x tidak ditetapkan atau jika y tidak ditetapkan”. Fungsi akan mengembalikan jika salah satu dari kondisi ini benar. Jika pernyataan return dijalankan, baris kode berikutnya dalam fungsi akan

diabaikan. Eksekusi program akan kembali ke titik saat fungsi dipanggil. Jika kedua parameter ditetapkan, fungsi akan menggemakan yang lebih besar dari keduanya.
Keluaran dari kode berikut:

```
$a = 1;
$b = 2.5;
$c = 1.9;
larger($a, $b);
larger($c, $a);
larger($d, $a);
```

Keluaran dari kode berikut:

```
2.5
1.9
this function requires two numbers
```

**Mengembalikan Nilai dari Fungsi**

Keluar dari suatu fungsi bukanlah satu-satunya alasan untuk menggunakan return. Banyak fungsi menggunakan pernyataan return untuk berkomunikasi dengan kode yang memanggilnya. Daripada menggemakan hasil perbandingan dalam fungsi `larger()` kita, fungsi kita mungkin akan lebih berguna jika kita mengembalikan jawabannya. Dengan cara ini, kode yang memanggil fungsi dapat memilih apakah dan bagaimana cara menampilkan atau menggunakannya. Fungsi bawaan yang setara `max()` berperilaku seperti ini.
Kita dapat menulis fungsi `larger()` kita sebagai berikut:

```
function larger ($x, $y)
{
  if (!isset($x)||!isset($y))
    return -1.7E+308;
  else if ($x>=$y)
    return $x;
  else
    return $y;
}
```

Di sini kita mengembalikan nilai yang lebih besar dari dua nilai yang dimasukkan. Kita akan mengembalikan angka yang jelas berbeda jika terjadi kesalahan. Jika salah satu angka hilang, kita dapat mengembalikan tidak ada atau mengembalikan –1,7□10308. Ini adalah angka yang sangat kecil dan tidak mungkin tertukar dengan jawaban yang sebenarnya. Fungsi bawaan max() tidak mengembalikan apa pun jika kedua variabel tidak disetel, dan jika hanya satu yang disetel, mengembalikan yang itu.
Kode berikut:

```
$a = 1; $b = 2.5; $c = 1.9;
echo larger($a, $b).”<br>”;
echo larger($c, $a).”<br>”;
echo larger($d, $a).”<br>”;
```

akan menghasilkan keluaran ini:

```
2.5
1.9
-1.7E+308
```

Fungsi yang menjalankan beberapa tugas, tetapi tidak perlu mengembalikan nilai, sering kali mengembalikan true atau false untuk menunjukkan apakah fungsi tersebut berhasil atau gagal. Nilai true dan false dapat direpresentasikan dengan 1 dan 0, berturut-turut.

## 5.7 BLOK KODE

Kita mendeklarasikan bahwa sekelompok pernyataan adalah blok dengan menempatkannya dalam kurung kurawal. Hal ini tidak memengaruhi sebagian besar operasi kode Anda, tetapi memiliki implikasi khusus termasuk cara struktur kontrol seperti loop dan kondisional dijalankan.

Dua contoh berikut bekerja dengan sangat berbeda.

**Contoh Tanpa Blok Kode**

```
for($i = 0; $i < 3; $i++ )
  echo “Line 1<br>”;
  echo “Line 2<br>”;
```

**Contoh dengan Blok Kode**

```
for($i = 0; $i < 3; $i++ )
{
  echo “Line 1<br>”;
  echo “Line 2<br>”;
}
```

Dalam kedua contoh, loop for diulang tiga kali. Pada contoh pertama, hanya satu baris tepat di bawah ini yang dieksekusi oleh for loop. Output dari contoh ini adalah sebagai berikut:

```
Line 1
Line 1
Line 1
Line 2
```

Contoh kedua menggunakan blok kode untuk mengelompokkan dua baris menjadi satu. Ini berarti bahwa kedua baris dieksekusi tiga kali oleh for loop. Output dari contoh ini adalah sebagai berikut:

```
Line 1
```

```
Line 2
Line 1
Line 2
Line 1
Line 2
```

Karena kode dalam contoh-contoh ini diberi indentasi dengan benar, Anda mungkin dapat melihat perbedaan di antara keduanya sekilas. Indentasi kode dimaksudkan untuk memberikan interpretasi visual kepada pembaca tentang baris mana yang terpengaruh oleh for loop. Namun, perlu dicatat bahwa spasi tidak memengaruhi cara PHP memproses kode. Dalam beberapa bahasa, blok kode memengaruhi cakupan variabel. Ini tidak terjadi dalam PHP.

**Rekursi**

Fungsi rekursif didukung dalam PHP. Fungsi rekursif adalah fungsi yang memanggil dirinya sendiri. Fungsi-fungsi ini khususnya berguna untuk menavigasi struktur data dinamis seperti linked list dan pohon. Namun, beberapa aplikasi berbasis Web memerlukan struktur data dengan kompleksitas ini, sehingga kita hanya menggunakan rekursi secara minimal. Rekursi dapat digunakan sebagai pengganti iterasi dalam banyak kasus karena keduanya memungkinkan Anda melakukan sesuatu secara berulang-ulang. Fungsi rekursif lebih lambat dan menggunakan lebih banyak memori daripada iterasi, jadi Anda harus menggunakan iterasi sedapat mungkin. Demi kelengkapan, kita akan melihat contoh singkat yang ditunjukkan dalam Daftar 5.5.

**Daftar 5.5 Recursion.Php—Sangat Mudah Membalikkan String Menggunakan Rekursi—Versi Iteratif Juga Ditampilkan**

```
function reverse_r($str)
{
  if (strlen($str)>0)
    reverse_r(substr($str, 1));
  echo substr($str, 0, 1);
  return;
}

function reverse_i($str)
{
  for ($i=1; $i<=strlen($str); $i++)
  {
    echo substr($str, -$i, 1);
  }
  return;
}
```

Dalam daftar ini, kami telah menerapkan dua fungsi. Keduanya akan mencetak string secara terbalik. Fungsi `reverse_r()` bersifat rekursif, dan fungsi `reverse_i()` bersifat iteratif. Fungsi `reverse_r()` mengambil string sebagai parameter. Saat Anda memanggilnya, fungsi tersebut akan memanggil dirinya sendiri, setiap kali meneruskan karakter kedua hingga terakhir dari string tersebut. Misalnya, jika Anda memanggil:

```
reverse_r(“Hello”);
```

fungsi tersebut akan memanggil dirinya sendiri beberapa kali, dengan parameter berikut:

```
reverse_r(“ello”);
reverse_r(“llo”);
reverse_r(“lo”);
reverse_r(“o”);
reverse_r(“”);
```

Setiap panggilan yang dilakukan fungsi itu sendiri akan membuat salinan baru dari kode fungsi di memori server, tetapi dengan parameter yang berbeda. Ini seperti berpura-pura bahwa kita benar-benar memanggil fungsi yang berbeda setiap kali. Ini menghentikan contoh fungsi agar tidak membingungkan.

Dengan setiap panggilan, panjang string yang dimasukkan diuji. Ketika kita mencapai akhir string (`strlen()==0`), kondisinya gagal. Contoh fungsi terbaru (`reverse_r(“”)`) kemudian akan melanjutkan dan menjalankan baris kode berikutnya, yaitu menggemakan karakter pertama dari string yang dilewatkan dalam kasus ini, tidak ada karakter karena string tersebut kosong. Kemudian, contoh fungsi ini mengembalikan kontrol ke contoh yang memanggilnya, yaitu `reverse_r(“o”)`. Ini mencetak karakter pertama dalam stringnya ”o” dan mengembalikan kontrol ke contoh yang memanggilnya.

Proses berlanjut mencetak karakter dan kemudian kembali ke contoh fungsi di atasnya dalam urutan pemanggilan hingga kontrol dikembalikan ke program utama. Ada sesuatu yang sangat elegan dan matematis tentang solusi rekursif. Namun, dalam kebanyakan kasus, Anda lebih baik menggunakan solusi iteratif. Kode untuk ini juga ada di daftar 5.5. Perhatikan bahwa itu tidak lagi (meskipun ini tidak selalu terjadi dengan fungsi iteratif) dan melakukan hal yang persis sama.

Perbedaan utamanya adalah bahwa fungsi rekursif akan membuat salinan dirinya sendiri dalam memori dan menimbulkan overhead dari beberapa panggilan fungsi. Anda mungkin memilih untuk menggunakan solusi rekursif ketika kodenya jauh lebih pendek dan lebih elegan daripada versi iteratif, tetapi itu tidak akan sering terjadi dalam domain aplikasi ini. Meskipun rekursi tampak lebih elegan, programmer sering lupa untuk menyediakan kondisi penghentian untuk rekursi. Ini berarti bahwa fungsi akan berulang hingga server kehabisan memori, atau hingga waktu eksekusi maksimum terlampaui, mana pun yang lebih dulu.

# BAB 6
# PHP BERORIENTASI OBJEK

## 6.1 KONSEP BERORIENTASI OBJEK

Bahasa pemrograman modern biasanya mendukung atau bahkan memerlukan pendekatan berorientasi objek untuk pengembangan perangkat lunak. Pengembangan Berorientasi Objek (OO) berupaya menggunakan klasifikasi, hubungan, dan properti objek dalam sistem untuk membantu pengembangan program.

**Kelas dan Objek**

Dalam konteks perangkat lunak OO, objek dapat berupa hampir semua item atau konsep objek fisik seperti meja atau pelanggan; atau objek konseptual yang hanya ada dalam perangkat lunak, seperti area input teks atau file. Umumnya, kita paling tertarik pada objek konseptual termasuk objek dunia nyata yang perlu direpresentasikan dalam perangkat lunak. Perangkat lunak berorientasi objek dirancang dan dibangun sebagai sekumpulan objek mandiri dengan atribut dan operasi yang berinteraksi untuk memenuhi kebutuhan kita. Atribut adalah properti atau variabel yang berhubungan dengan objek. Operasi adalah metode, tindakan, atau fungsi yang dapat dilakukan objek untuk memodifikasi dirinya sendiri atau untuk beberapa efek eksternal.

Keunggulan utama perangkat lunak berorientasi objek adalah kemampuannya untuk mendukung dan mendorong enkapsulasi juga dikenal sebagai penyembunyian data. Pada dasarnya, akses ke data dalam suatu objek hanya tersedia melalui operasi objek, yang dikenal sebagai antarmuka objek. Fungsionalitas suatu objek terikat pada data yang digunakannya. Kita dapat dengan mudah mengubah detail tentang bagaimana objek diimplementasikan untuk meningkatkan kinerja, menambahkan fitur baru, atau memperbaiki bug tanpa harus mengubah antarmuka, yang dapat memiliki efek berantai di seluruh proyek. Di bidang pengembangan perangkat lunak lainnya, OO adalah norma dan perangkat lunak berorientasi fungsi dianggap kuno. Karena sejumlah alasan, sebagian besar skrip Web sayangnya masih dirancang dan ditulis menggunakan pendekatan ad hoc yang mengikuti metodologi berorientasi fungsi.

Ada sejumlah alasan untuk ini. Mayoritas proyek Web relatif kecil dan mudah. Anda dapat mengambil gergaji dan membangun rak rempah-rempah kayu tanpa merencanakan pendekatan Anda dan Anda dapat berhasil menyelesaikan sebagian besar proyek perangkat lunak Web dengan cara yang sama karena ukurannya yang kecil. Namun, jika Anda mengambil gergaji dan mencoba membangun rumah tanpa perencanaan formal, Anda tidak akan mendapatkan hasil yang berkualitas, jika Anda mendapatkan hasil sama sekali hal yang sama berlaku untuk proyek perangkat lunak besar.

Banyak proyek Web berkembang dari serangkaian halaman hyperlink menjadi aplikasi yang kompleks. Aplikasi kompleks ini, baik disajikan melalui kotak dialog dan jendela atau melalui halaman HTML yang dibuat secara dinamis, memerlukan metodologi pengembangan yang dipikirkan dengan baik. Orientasi objek dapat membantu Anda mengelola kompleksitas

dalam proyek Anda, meningkatkan penggunaan kembali kode, dan dengan demikian mengurangi biaya pemeliharaan. Dalam perangkat lunak OO, sebuah objek adalah kumpulan data dan operasi tersimpan yang unik dan dapat diidentifikasi yang beroperasi pada data tersebut. Misalnya, kita mungkin memiliki dua objek yang mewakili tombol. Bahkan jika keduanya memiliki label "OK", lebar 60 piksel, tinggi 20 piksel, dan atribut lainnya yang identik, kita tetap harus dapat menangani satu tombol atau yang lainnya. Dalam perangkat lunak, kita memiliki variabel terpisah yang bertindak sebagai pegangan (pengidentifikasi unik) untuk objek.

Objek dapat dikelompokkan ke dalam kelas. Kelas mewakili sekumpulan objek yang mungkin berbeda dari satu individu ke individu lainnya, tetapi harus memiliki sejumlah kesamaan. Kelas berisi objek yang semuanya memiliki operasi yang sama yang berperilaku dengan cara yang sama dan atribut yang sama yang mewakili hal yang sama, meskipun nilai atribut tersebut akan bervariasi dari satu objek ke objek lainnya. Kata benda sepeda dapat dianggap sebagai kelas objek yang menggambarkan banyak sepeda berbeda dengan banyak fitur atau atribut umum seperti dua roda, warna dan ukuran, dan operasi, seperti bergerak. Sepeda saya sendiri dapat dianggap sebagai objek yang sesuai dengan kelas sepeda. Sepeda memiliki semua fitur umum dari semua sepeda termasuk operasi gerakan yang berperilaku sama seperti gerakan sepeda lainnya—meskipun jarang digunakan. Atribut sepeda saya memiliki nilai unik karena sepeda saya berwarna hijau, dan tidak semua sepeda berwarna seperti itu.

**Polimorfisme**

Bahasa pemrograman berorientasi objek harus mendukung polimorfisme, yang berarti bahwa kelas yang berbeda dapat memiliki perilaku yang berbeda untuk operasi yang sama. Misalnya, jika kita memiliki kelas mobil dan kelas sepeda, keduanya dapat memiliki operasi gerakan yang berbeda. Untuk objek di dunia nyata, hal ini jarang menjadi masalah. Sepeda tidak mungkin menjadi bingung dan mulai menggunakan operasi gerakan mobil sebagai gantinya. Namun, bahasa pemrograman tidak memiliki akal sehat dunia nyata, jadi bahasa tersebut harus mendukung polimorfisme untuk mengetahui operasi gerakan mana yang akan digunakan pada objek tertentu.

Polimorfisme lebih merupakan karakteristik perilaku daripada objek. Dalam PHP, hanya fungsi anggota kelas yang dapat bersifat polimorfik. Perbandingan di dunia nyata adalah kata kerja dalam bahasa alami, yang setara dengan fungsi anggota. Pertimbangkan cara sepeda dapat digunakan dalam kehidupan nyata. Anda dapat membersihkannya, memindahkannya, membongkarnya, memperbaikinya, atau mengecatnya, di antara hal-hal lainnya. Kata kerja ini menggambarkan tindakan umum karena Anda tidak tahu jenis objek apa yang sedang ditindaklanjuti. (Jenis abstraksi objek dan tindakan ini adalah salah satu karakteristik pembeda kecerdasan manusia.) Misalnya, menggerakkan sepeda memerlukan tindakan yang sama sekali berbeda dari yang diperlukan untuk menggerakkan mobil, meskipun konsepnya serupa. Kata kerja bergerak dapat dikaitkan dengan serangkaian tindakan tertentu hanya setelah objek yang ditindaklanjuti diketahui.

**Pewarisan**

Pewarisan memungkinkan kita untuk membuat hubungan hierarkis antara kelas menggunakan subkelas. Subkelas mewarisi atribut dan operasi dari superkelasnya. Misalnya, mobil dan sepeda memiliki beberapa kesamaan. Kita dapat menggunakan kelas kendaraan untuk memuat hal-hal seperti atribut warna dan operasi gerakan yang dimiliki semua kendaraan, lalu membiarkan kelas mobil dan sepeda kita mewarisi dari kendaraan. Dengan pewarisan, Anda dapat membangun dan menambahkan kelas yang sudah ada. Dari kelas dasar yang sederhana, Anda dapat memperoleh kelas yang lebih kompleks dan khusus saat dibutuhkan. Ini membuat kode Anda lebih dapat digunakan kembali, yang merupakan salah satu keuntungan penting dari pendekatan berorientasi objek.

Menggunakan pewarisan dapat menghemat pekerjaan kita jika operasi dapat ditulis sekali dalam superkelas daripada berkali-kali dalam subkelas yang terpisah. Ini juga memungkinkan kita untuk memodelkan hubungan dunia nyata dengan lebih akurat. Jika kalimat tentang dua kelas masuk akal dengan "adalah" di antara kelas-kelas tersebut, pewarisan mungkin tepat. Kalimat "mobil adalah kendaraan" masuk akal, tetapi kalimat "kendaraan adalah mobil" tidak masuk akal karena tidak semua kendaraan adalah mobil. Oleh karena itu, mobil dapat mewarisi dari kendaraan.

## 6.2 MEMBUAT KELAS, ATRIBUT, OPERASI DALAM PHP

Sejauh ini, kita telah membahas kelas dengan cara yang cukup abstrak. Saat membuat kelas dalam PHP, Anda harus menggunakan kata kunci class.

**Struktur Kelas**

Definisi kelas minimal terlihat seperti berikut:

```
class classname
{
 }
```

Agar bermanfaat, kelas kita memerlukan atribut dan operasi. Kita membuat atribut dengan mendeklarasikan variabel dalam definisi kelas menggunakan kata kunci var. Kode berikut membuat kelas bernama classname dengan dua atribut, `$attribute1` dan `$attribute2`.

```
class classname
{
   var $attribute1;
   var $attribute2;
}
```

membuat operasi dengan mendeklarasikan fungsi dalam definisi kelas. Kode berikut akan membuat kelas bernama `classname` dengan dua operasi yang tidak melakukan apa pun. Operasi `operation1()` tidak mengambil parameter dan `operation2()` mengambil dua parameter.

```
class classname
{
  function operation1()
  {
  }
  function operation2($param1, $param2)
  {
  }
}
```

**Konstruktor**

Sebagian besar kelas akan memiliki jenis operasi khusus yang disebut konstruktor. Konstruktor dipanggil saat objek dibuat, dan biasanya juga melakukan tugas inisialisasi yang berguna seperti menyetel atribut ke nilai awal yang masuk akal atau membuat objek lain yang dibutuhkan oleh objek ini. Konstruktor dideklarasikan dengan cara yang sama seperti operasi lain, tetapi memiliki nama yang sama dengan kelas. Meskipun kita dapat memanggil konstruktor secara manual, tujuan utamanya adalah dipanggil secara otomatis saat objek dibuat. Kode berikut mendeklarasikan kelas dengan konstruktor:

```
class classname
{
  function classname($param)
  {
    echo “Constructor called with parameter $param <br>”;
  }
}
```

Satu hal yang perlu diingat adalah bahwa PHP tidak mendukung fungsi yang berlebihan, yang berarti Anda hanya dapat menyediakan satu fungsi dengan nama tertentu, termasuk konstruktornya. (Ini adalah fitur yang didukung dalam banyak bahasa OO.)

**Instansiasi**

Setelah kita mendeklarasikan sebuah kelas, kita perlu membuat sebuah objek individu tertentu yang merupakan anggota kelas—untuk digunakan. Ini juga dikenal sebagai membuat instance atau membuat instance kelas. Kita membuat sebuah objek menggunakan kata kunci new. Kita perlu menentukan kelas mana yang akan menjadi instance objek kita, dan menyediakan parameter apa pun yang diperlukan oleh konstruktor kita.

Kode berikut mendeklarasikan sebuah kelas yang disebut classname dengan sebuah konstruktor, lalu membuat tiga objek bertipe classname:

```
class classname
{
  function classname($param)
  {
```

```
      echo “Constructor called with parameter $param <br>”;
    }
  }

  $a = new classname(“First”);
  $b = new classname(“Second”);
  $c = new classname();
```

Karena konstruktor dipanggil setiap kali kita membuat objek, kode ini menghasilkan keluaran berikut:

```
  Constructor called with parameter First
  Constructor called with parameter Second
  Constructor called with parameter
```

**Menggunakan Atribut Kelas**

Di dalam kelas, Anda memiliki akses ke penunjuk khusus yang disebut `$this`. Jika atribut kelas Anda saat ini disebut `$attribute`, Anda menyebutnya sebagai `$this->attribute` saat menyetel atau mengakses variabel dari suatu operasi di dalam kelas. Kode berikut menunjukkan penyetelan dan pengaksesan atribut di dalam kelas:

```
  class classname
  {
    var $attribute;
    function operation($param)
    {
      $this->attribute = $param
      echo $this->attribute;
    }
  }
```

Beberapa bahasa pemrograman memungkinkan Anda membatasi akses ke atribut dengan mendeklarasikan data tersebut sebagai data pribadi atau data yang dilindungi. Fitur ini tidak didukung oleh PHP, jadi semua atribut dan operasi Anda dapat dilihat di luar kelas (dengan kata lain, semuanya bersifat publik).

Kita dapat melakukan tugas yang sama seperti yang ditunjukkan sebelumnya dari luar kelas, dengan menggunakan sintaksis yang sedikit berbeda.

```
  class classname
  {
    var $attribute;
  }
  $a = new classname();
  $a->attribute = “value”;
```

```
echo $a->attribute
```

Mengakses atribut secara langsung dari luar kelas bukanlah ide yang baik. Salah satu keuntungan dari pendekatan berorientasi objek adalah pendekatan ini mendorong enkapsulasi. Meskipun Anda tidak dapat memaksakan penyembunyian data dalam PHP, dengan sedikit kemauan, Anda dapat memperoleh keuntungan yang sama.

Jika Anda menulis fungsi aksesor daripada mengakses atribut kelas secara langsung, Anda dapat membuat semua akses melalui satu bagian kode. Saat Anda pertama kali menulis fungsi aksesor, tampilannya mungkin seperti berikut:

```
class classname
{
  var $attribute;
  function get_attribute()
  {
    return $this->attribute;
  }
  function set_attribute($new_value)
  {
    $this->attribute = $new_value;
  }
}
```

Kode ini hanya menyediakan fungsi untuk mengakses atribut bernama `$attribute`. Kita memiliki fungsi bernama `get_attribute()` yang hanya mengembalikan nilai `$attribute`, dan fungsi bernama `set_attribute()` yang menetapkan nilai baru ke `$attribute`.

Sekilas, kode ini mungkin tampak menambahkan sedikit atau tidak ada nilai. Dalam bentuknya saat ini, hal ini mungkin benar, tetapi alasan untuk menyediakan fungsi aksesor sederhana: Kita kemudian hanya akan memiliki satu bagian kode yang mengakses atribut tertentu. Dengan hanya satu titik akses, kita dapat menerapkan pemeriksaan untuk memastikan bahwa hanya data yang masuk akal yang disimpan. Jika kemudian kita berpikir bahwa nilai `$attribute` seharusnya hanya antara nol dan seratus, kita dapat menambahkan beberapa baris kode sekali dan memeriksa sebelum mengizinkan perubahan. Fungsi `set_attribute()` kita dapat diubah menjadi seperti berikut:

```
function set_attribute($new_value)
{
  if( $new_value >= 0 && $newvalue <= 100 )
    $this->attribute = $new_value;
}
```

Perubahan ini sepele, tetapi jika kita tidak menggunakan fungsi aksesor, kita harus menelusuri setiap baris kode dan mengubah setiap akses ke `$attribute`, suatu tindakan yang

membosankan dan rawan kesalahan. Dengan hanya satu titik akses, kita bebas mengubah implementasi yang mendasarinya. Jika karena suatu alasan, kita memilih untuk mengubah cara penyimpanan `$attribute`, fungsi aksesor memungkinkan kita melakukan ini dan hanya mengubah kode di satu tempat.

Kita mungkin memutuskan bahwa daripada menyimpan `$attribute` sebagai variabel, kita hanya akan mengambilnya dari basis data saat dibutuhkan, menghitung nilai terkini setiap kali diminta, menyimpulkan nilai dari nilai atribut lain, atau mengodekan data kita sebagai tipe data yang lebih kecil. Perubahan apa pun yang kita putuskan untuk dilakukan, kita cukup mengubah fungsi aksesor kita. Bagian kode lainnya tidak akan terpengaruh selama kita membuat fungsi aksesor tetap menerima atau mengembalikan data yang diharapkan oleh bagian lain dari program.

**Memanggil Operasi Kelas**

Kita dapat memanggil operasi kelas dengan cara yang sama seperti kita memanggil atribut kelas. Jika kita memiliki kelas berikut:

```
class classname
{
   function operation1()
   {
   }
   function operation2($param1, $param2)
   {
   }
}
```

dan buat objek bertipe classname yang disebut $a sebagai berikut:

```
$a = new classname();
```

Kita kemudian memanggil operasi dengan cara yang sama seperti kita memanggil fungsi lain: dengan menggunakan nama mereka dan menempatkan parameter yang mereka butuhkan dalam tanda kurung. Karena operasi ini milik sebuah objek dan bukan fungsi normal, kita perlu menentukan objek mana yang menjadi milik mereka. Nama objek digunakan dengan cara yang sama seperti atribut objek sebagai berikut:

```
$a->operation1();
$a->operation2(12, “test”);
```

Jika operasi kami menghasilkan sesuatu, kami dapat menangkap data pengembalian tersebut sebagai berikut:

```
$x = $a->operation1();
$y = $a->operation2(12, “test”);
```

**Menerapkan Pewarisan dalam PHP**

Jika kelas kita akan menjadi subkelas dari kelas lain, Anda dapat menggunakan kata kunci extends untuk menentukannya. Kode berikut membuat kelas bernama B yang mewarisi dari beberapa kelas yang telah didefinisikan sebelumnya bernama A.

```
class B extends A
{
   var $attribute2;
   function operation2()
   {
   }
}
```

Jika kelas A dideklarasikan sebagai berikut:

```
class A
{
   var $attribute1;
   function operation1()
   {
   }
}
```

semua akses berikut terhadap operasi dan atribut objek tipe B akan valid:

```
$b = new B();
$b->operation1();
$b->attribute1 = 10;
$b->operation2();
$b->attribute2 = 10;
```

Perhatikan bahwa karena kelas B memperluas kelas A, kita dapat merujuk ke `operation1()` dan `$attribute1`, meskipun keduanya dideklarasikan di kelas A. Sebagai subkelas A, B memiliki semua fungsi dan data yang sama. Selain itu, B telah mendeklarasikan atribut dan operasinya sendiri.

Penting untuk dicatat bahwa pewarisan hanya bekerja dalam satu arah. Subkelas atau anak mewarisi fitur-fiturnya dari induk atau superkelasnya, tetapi induknya tidak mengambil fitur-fitur dari anak. Ini berarti bahwa dua baris terakhir dalam kode ini salah:

```
$a = new A();
$a->operation1();
$a->attribute1 = 10;
$a->operation2();
$a->attribute2 = 10;
```

Kelas A tidak memiliki operasi2() atau atribut2.

**Overriding**

Kami telah menunjukkan subkelas yang mendeklarasikan atribut dan operasi baru. Mendeklarasikan ulang atribut dan operasi yang sama juga valid dan terkadang berguna. Kita dapat melakukan ini untuk memberikan atribut dalam subkelas nilai default yang berbeda dengan atribut yang sama dalam superkelasnya, atau untuk memberikan operasi dalam subkelas fungsionalitas yang berbeda dengan operasi yang sama dalam superkelasnya. Ini disebut overriding.
Misalnya, jika kita memiliki kelas A:

```
class A
{
   var $attribute = “default value”;
   function operation()
   {
      echo “Something<br>”;
      echo “The value of \$attribute is $this->attribute<br>”;
   }
}
```

dan ingin mengubah nilai default $attribute dan menyediakan fungsionalitas baru untuk operasi(), kita dapat membuat kelas B berikut, yang menggantikan $attribute dan operasi():

```
class B extends A
{
   var $attribute = “different value”;
   function operation()
   {
      echo “Something else<br>”;
      echo “The value of \$attribute is $this->attribute<br>”;
   }
}
```

Mendeklarasikan B tidak akan memengaruhi definisi asli A. Perhatikan dua baris kode berikut:

```
$a = new A();
$a -> operation();
```

Kami telah membuat sebuah objek bertipe A dan memanggil fungsi operasinya(). Ini akan menghasilkan

```
Something
The value of $attribute is default value
```

membuktikan bahwa membuat B tidak mengubah A. Jika kita membuat objek bertipe B, kita akan mendapatkan output yang berbeda.
Kode ini

```
$b = new B();
$b -> operation();
```

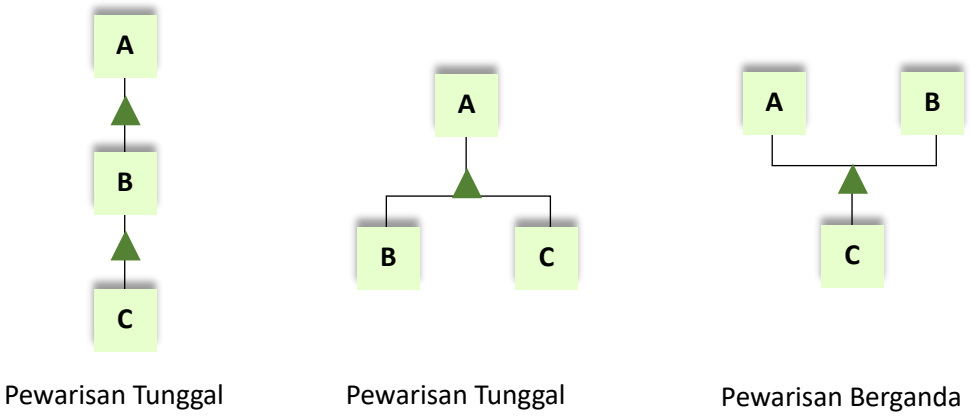
akan menghasilkan

```
Something else
The value of $attribute is different value
```

Dengan cara yang sama seperti menyediakan atribut atau operasi baru dalam subkelas tidak memengaruhi superkelas, mengganti atribut atau operasi dalam subkelas tidak memengaruhi superkelas. Subkelas akan mewarisi semua atribut dan operasi dari superkelasnya, kecuali jika Anda menyediakan pengganti. Jika Anda menyediakan definisi pengganti, ini akan diutamakan dan menggantikan definisi asli. Tidak seperti beberapa bahasa OO lainnya, PHP tidak memungkinkan Anda untuk mengganti fungsi dan tetap dapat memanggil versi yang ditetapkan dalam induknya. Pewarisan dapat berlapis-lapis. Kita dapat mendeklarasikan kelas yang secara imajinatif disebut C, yang memperluas B dan karenanya mewarisi fitur dari B dan dari induk B, A. Kelas C dapat kembali memilih atribut dan operasi mana dari induknya yang akan diganti dan diganti.

## 6.3 PEWARISAN GANDA

Beberapa bahasa OO mendukung pewarisan ganda, tetapi PHP tidak. Ini berarti bahwa setiap kelas hanya dapat mewarisi dari satu induk. Tidak ada batasan untuk berapa banyak anak yang dapat berbagi satu induk. Mungkin tidak langsung jelas apa artinya ini. Gambar 6.1 menunjukkan tiga cara berbeda yang dapat dilakukan oleh tiga kelas bernama A, B, dan C.




Pewarisan Tunggal Pewarisan Tunggal Pewarisan Berganda

**Gambar 6.1 Php Tidak Mendukung Pewarisan Berganda.**

Kombinasi kiri menunjukkan kelas C mewarisi dari kelas B, yang selanjutnya mewarisi dari kelas A. Setiap kelas memiliki paling banyak satu induk, jadi ini adalah pewarisan tunggal yang benar-benar valid dalam PHP. Kombinasi tengah menunjukkan kelas B dan C mewarisi dari kelas A. Setiap kelas memiliki paling banyak satu induk, jadi sekali lagi ini adalah pewarisan tunggal yang valid. Kombinasi kanan menunjukkan kelas C mewarisi dari kelas A dan kelas B. Dalam kasus ini, kelas C memiliki dua induk, jadi ini adalah pewarisan berganda dan tidak valid dalam PHP.

**Mendesain Kelas**

Sekarang setelah Anda mengetahui beberapa konsep di balik objek dan kelas serta sintaksis untuk mengimplementasikannya dalam PHP, sekarang saatnya untuk melihat cara mendesain kelas yang berguna. Banyak kelas dalam kode Anda akan mewakili kelas atau kategori objek dunia nyata. Kelas yang mungkin Anda gunakan dalam pengembangan Web mungkin mencakup halaman, komponen antarmuka pengguna, kereta belanja, penanganan kesalahan, kategori produk, atau pelanggan. Objek dalam kode Anda juga dapat mewakili contoh spesifik dari kelas yang disebutkan sebelumnya, misalnya, beranda, tombol tertentu, atau keranjang belanja yang digunakan oleh Fred Smith pada waktu tertentu. Fred Smith sendiri dapat direpresentasikan oleh objek bertipe pelanggan. Setiap barang yang dibeli Fred dapat direpresentasikan sebagai objek, yang termasuk dalam kategori atau kelas.

Pada bab sebelumnya, kami menggunakan file include sederhana untuk memberikan perusahaan fiktif kami, TLA Consulting, tampilan dan nuansa yang konsisten di seluruh halaman situs Web mereka. Dengan menggunakan kelas dan kekuatan pewarisan yang menghemat waktu, kami dapat membuat versi yang lebih canggih dari situs yang sama. Kami ingin dapat dengan cepat membuat halaman untuk TLA yang terlihat dan berperilaku dengan cara yang sama.

Halaman-halaman tersebut harus dapat dimodifikasi agar sesuai dengan berbagai bagian situs. Kita akan membuat kelas Page. Tujuan utama kelas ini adalah untuk membatasi jumlah HTML yang dibutuhkan untuk membuat halaman baru. Kelas ini akan memungkinkan kita mengubah bagian-bagian yang berubah dari satu halaman ke halaman lainnya, sambil secara otomatis membuat elemen-elemen yang tetap sama. Kelas ini akan menyediakan kerangka kerja yang fleksibel untuk membuat halaman-halaman baru dan tidak akan mengorbankan kebebasan kita. Karena kita membuat halaman dari skrip dan bukan dengan HTML statis, kita dapat menambahkan sejumlah hal cerdas termasuk fungsionalitas untuk mengaktifkan hal-hal berikut:

- ☑ Memungkinkan kita untuk hanya mengubah elemen-elemen halaman di satu tempat. Jika kita mengubah pemberitahuan hak cipta atau menambahkan tombol tambahan, kita hanya perlu membuat perubahan di satu tempat.
- ☑ Memiliki konten default untuk sebagian besar bagian halaman, tetapi dapat mengubah setiap elemen jika diperlukan, dengan menetapkan nilai-nilai khusus untuk elemen-elemen seperti judul dan metatag.

- ☑ Mengenali halaman mana yang sedang dilihat dan mengubah elemen-elemen navigasi agar sesuai tidak ada gunanya memiliki tombol yang membawa Anda ke halaman beranda yang terletak di halaman beranda.
- ☑ Memungkinkan kita mengganti elemen standar untuk halaman tertentu. Misalnya, jika kita menginginkan tombol navigasi yang berbeda di beberapa bagian situs, kita harus dapat mengganti yang standar.

**Menulis Kode untuk Kelas**

Setelah memutuskan seperti apa tampilan keluaran kode kita, dan beberapa fitur yang kita inginkan, bagaimana cara mengimplementasikannya? Kita akan membahasnya nanti di buku ini tentang desain dan manajemen proyek untuk proyek besar. Untuk saat ini, kita akan berkonsentrasi pada bagian-bagian khusus untuk menulis PHP berorientasi objek. Kelas kita akan memerlukan nama logis. Karena mewakili halaman, maka kelas tersebut akan disebut Halaman. Untuk mendeklarasikan kelas yang disebut Halaman, kita mengetik

```
class Page
{
}
```

Kelas kita memerlukan beberapa atribut. Kita akan menetapkan elemen yang mungkin ingin kita ubah dari satu halaman ke halaman lainnya sebagai atribut kelas kita. Konten utama halaman, yang akan berupa kombinasi tag HTML dan teks, akan disebut `$content`. Kita dapat mendeklarasikan konten dengan baris kode berikut dalam definisi kelas:

```
var $content;
```

Kita juga dapat mengatur atribut untuk menyimpan judul halaman. Kita mungkin ingin mengubahnya untuk menunjukkan dengan jelas halaman tertentu yang sedang dilihat pengunjung kita. Daripada memiliki judul kosong, kita akan memberikan judul default dengan deklarasi berikut:

```
var $title = “TLA Consulting Pty Ltd”;
```

Sebagian besar halaman Web komersial menyertakan metatag untuk membantu mesin pencari mengindeksnya. Agar bermanfaat, metatag mungkin harus berubah dari halaman ke halaman. Sekali lagi, kita akan memberikan nilai default:

```
var $keywords = “TLA Consulting, Three Letter Abbreviation,
                some of my best friends are search engines”;
```

Tombol navigasi yang ditampilkan pada halaman asli pada Gambar 5.2 (lihat bab sebelumnya) mungkin harus tetap sama dari halaman ke halaman untuk menghindari kebingungan orang, tetapi untuk mengubahnya dengan mudah, kita akan menjadikannya atribut juga. Karena

mungkin ada sejumlah tombol yang bervariasi, kita akan menggunakan array, dan menyimpan teks untuk tombol dan URL yang seharusnya dituju.

```
var $buttons = array( “Home” => “home.php”,
                “Contact” => “contact.php”,
                “Services” => “services.php”,
                “site Map” => “map.php”
                );
```

Untuk menyediakan beberapa fungsi, kelas kita juga memerlukan operasi. Kita dapat memulai dengan menyediakan fungsi aksesor untuk mengatur dan mendapatkan nilai atribut yang kita definisikan. Semua ini berbentuk seperti ini:

```
function SetContent($newcontent)
{
  $this->content = $newcontent;
}
```

Karena kecil kemungkinan kita akan meminta salah satu nilai ini dari luar kelas, kita memilih untuk tidak menyediakan kumpulan fungsi GET yang cocok.

Tujuan utama kelas ini adalah untuk menampilkan halaman HTML, jadi kita memerlukan fungsi. Kita telah memanggil fungsi Display(), dan fungsinya adalah sebagai berikut:

```
function Display()
{
  echo “<html>\n<head>\n”;
  $this -> DisplayTitle();
  $this -> DisplayKeywords();
  $this -> DisplayStyles();
  echo “</head>\n<body>\n”;
  $this -> DisplayHeader();
  $this -> DisplayMenu($this->buttons);
  echo $this->content;
  $this -> DisplayFooter();
  echo “</body>\n</html>\n”;
}
```

Fungsi ini menyertakan beberapa pernyataan gema sederhana untuk menampilkan HTML, tetapi sebagian besar terdiri dari panggilan ke fungsi lain di kelas. Seperti yang mungkin sudah Anda duga dari namanya, fungsi-fungsi lain ini menampilkan bagian-bagian halaman. Tidak wajib untuk memecah fungsi seperti ini. Semua fungsi terpisah ini mungkin hanya digabungkan menjadi satu fungsi besar. Kami memisahkannya karena sejumlah alasan.

Setiap fungsi harus memiliki tugas yang ditentukan untuk dilakukan. Semakin sederhana tugas ini, semakin mudah penulisan dan pengujian fungsi tersebut. Jangan melangkah terlalu jauh jika Anda memecah program menjadi terlalu banyak unit kecil, program tersebut mungkin sulit dibaca. Dengan menggunakan pewarisan, kita dapat mengganti operasi. Kita dapat mengganti satu fungsi Display() yang besar, tetapi kecil kemungkinan kita ingin mengubah cara seluruh halaman ditampilkan. Akan jauh lebih baik untuk memecah fungsi tampilan menjadi beberapa tugas mandiri dan dapat mengganti hanya bagian-bagian yang ingin kita ubah.

Fungsi Display kita memanggil `DisplayTitle()`, `DisplayKeywords()`, `DisplayStyles()`, `DisplayHeader()`, `DisplayMenu()`, dan `DisplayFooter()`. Ini berarti kita perlu mendefinisikan operasi-operasi ini. Salah satu peningkatan PHP 4 dibandingkan PHP 3 adalah kita dapat menulis operasi atau fungsi dalam urutan logis ini, memanggil operasi atau fungsi sebelum kode aktual untuk fungsi tersebut. Dalam PHP 3 dan banyak bahasa lainnya, kita perlu menulis fungsi atau operasi sebelum dapat dipanggil.

Sebagian besar operasi kita cukup sederhana dan perlu menampilkan beberapa HTML dan mungkin isi atribut kita. Daftar 6.1 menunjukkan kelas lengkap, yang telah kita simpan sebagai page.inc untuk disertakan atau diperlukan ke dalam berkas lain.

**Daftar 6.1** page.inc—Kelas Page Kita Menyediakan Cara Fleksibel yang Mudah untuk Membuat Halaman TLA

```
<?
class Page
{

  // class Page's attributes
  var $content;
  var $title = "TLA Consulting Pty Ltd";
  var $keywords = "TLA Consulting, Three Letter Abbreviation,
                   some of my best friends are search engines";
  var $buttons = array( "Home" => "home.php",
                        "Contact" => "contact.php",
                        "Services" => "services.php",
                        "Site Map" => "map.php"
                      );
  // class Page's operations

  function SetContent($newcontent)
  {
    $this->content = $newcontent;
  }

  function SetTitle($newtitle)
  {
    $this->title = $newtitle;
```

```
    }

    function SetKeywords($newkeywords)
    {
      $this->keywords = $newkeywords;
    }

    function SetButtons($newbuttons)
    {
      $this->buttons = $newbuttons;
    }

    function Display()
    {
      echo “<html>\n<head>\n”;
        $this -> DisplayTitle();
        $this -> DisplayKeywords();
        $this -> DisplayStyles();
      echo “</head>\n<body>\n”;
        $this -> DisplayHeader();
        $this -> DisplayMenu($this->buttons);
      echo $this->content;
        $this -> DisplayFooter();
      echo “</body>\n</html>\n”;
    }

    function DisplayTitle()
    {
      echo “<title> $this->title </title>”;
    }

    function DisplayKeywords()
    {
      echo “<META name=\”keywords\” content=\”$this->keywords\”>”;
    }

    function DisplayStyles()
    {
  ?>
    <style>
      h1 {color:white; font-size:24pt; text-align:center;
        font-family:arial,sans-serif}
      .menu {color:white; font-size:12pt; text-align:center;
        font-family:arial,sans-serif; font-weight:bold}
      td {background:black}
      p {color:black; font-size:12pt; text-align:justify;
        font-family:arial,sans-serif}
```

```
    p.foot {color:white; font-size:9pt; text-align:center;
           font-family:arial,sans-serif; font-weight:bold}
    a:link,a:visited,a:active {color:white}
  </style>
<?
  }

  function DisplayHeader()
  {
?>
  <table width=”100%” cellpadding = 12 cellspacing =0 border = 0>
  <tr bgcolor = black>
    <td align = left><img src = “logo.gif”></td>
    <td>
      <h1>TLA Consulting Pty Ltd</h1>
    </td>
  <td align = right><img src = “logo.gif”></td>
  </tr>
  </table>
<?
  }

  function DisplayMenu($buttons)
  {
    echo “<table width = \”100%\” bgcolor = white”
         .” cellpadding = 4 cellspacing = 4>\n”;
    echo “ <tr>\n”;

    //calculate button size
    $width = 100/count($buttons);

    while (list($name, $url) = each($buttons))
    {
      $this -> DisplayButton($width, $name, $url,
               !$this->IsURLCurrentPage($url));
    }
    echo “ </tr>\n”;
    echo “</table>\n”;
  }

  function IsURLCurrentPage($url)
  {
    if(strpos( $GLOBALS[“SCRIPT_NAME”], $url )==false)
    {
      return false;
    }
    else
```

```
    {
      return true;
    }
  }

  function DisplayButton($width, $name, $url, $active = true)
  {
    if ($active)
    {
      echo "<td width = \"$width%\">
            <a href = \"$url\">
            <img src = \"s-logo.gif\" alt = \"$name\" border = 0></a>
            <a href = \"$url\"><span class=menu>$name</span></a></td>";
    }
    else
    {
      echo "<td width = \"$width%\">
            <img src = \"side-logo.gif\">
            <span class=menu>$name</span></td>";
    }
  }

  function DisplayFooter()
  {
?>

    <table width = "100%" bgcolor = black cellpadding = 12 border = 0>
    <tr>
      <td>
        <p class=foot>&copy; TLA Consulting Pty Ltd.</p>
        <p class=foot>Please see our
                  <a href ="">legal information page</a></p>
      </td>
    </tr>
  </table>
<?
  }
}
?>
```

Saat membacanya, perhatikan bahwa DisplayStyles(), DisplayHeader(), dan DisplayFooter() perlu menampilkan blok HTML statis yang besar, tanpa pemrosesan PHP. Oleh karena itu, kami cukup menggunakan tag PHP akhir (?>), mengetik HTML kami, lalu memasukkan kembali PHP dengan tag PHP terbuka (<?) saat berada di dalam fungsi. Dua operasi lain didefinisikan dalam kelas ini.

Operasi DisplayButton() menghasilkan satu tombol menu. Jika tombol tersebut menunjuk ke halaman yang sedang kita buka, kita akan menampilkan tombol yang tidak aktif, yang terlihat sedikit berbeda, dan tidak terhubung ke mana pun. Ini menjaga tata letak halaman tetap konsisten dan memberi pengunjung lokasi visual. Operasi IsURLCurrentPage() menentukan apakah URL untuk tombol menunjuk ke halaman saat ini. Banyak teknik yang dapat digunakan untuk menemukan ini. Kami telah menggunakan fungsi string strpos() untuk melihat apakah URL yang diberikan terdapat dalam salah satu variabel set server. Pernyataan strpos( $GLOBALS["SCRIPT_NAME"], $url ) akan mengembalikan angka jika string dalam $url berada di dalam variabel global SCRIPT_NAME, atau false jika tidak.

Untuk menggunakan kelas halaman ini, kita perlu menyertakan page.inc dalam skrip dan memanggil Display(). Kode dalam Daftar 6.2 akan membuat beranda TLA Consulting dan memberikan output yang sangat mirip dengan yang kita buat sebelumnya pada Gambar 5.2. Kode dalam Daftar 6.2 melakukan hal berikut:

1. Menggunakan require untuk menyertakan konten page.inc, yang berisi definisi kelas Page.
2. Membuat instance dari kelas Page. Instance tersebut disebut $homepage.
3. Memanggil operasi SetContent() dalam objek $homepage dan memberikan beberapa teks dan tag HTML agar muncul di halaman.
4. Memanggil operasi Display() dalam objek $homepage agar halaman ditampilkan di browser pengunjung.

**Daftar 6.2 Home.Php—Beranda Ini Menggunakan Kelas Page Untuk Melakukan Sebagian Besar Pekerjaan Yang Terlibat Dalam Pembuatan Halaman**

```
<?
  require ("page.inc");

  $homepage = new Page();

  $homepage -> SetContent("<p>Welcome to the home of TLA Consulting.
                  Please take some time to get to know us.</p>
                  <p>We specialize in serving your business needs
                  and hope to hear from you soon.</p>"
            );
  $homepage -> Display();
?>
```

Anda dapat melihat pada Daftar 6.2 bahwa kita perlu melakukan sedikit pekerjaan untuk membuat halaman baru menggunakan kelas Page ini. Menggunakan kelas dengan cara ini berarti bahwa semua halaman kita harus sangat mirip.

Jika kita ingin beberapa bagian situs menggunakan varian dari halaman standar, kita cukup menyalin page.inc ke file baru bernama page2.inc dan membuat beberapa

perubahan. Ini berarti bahwa setiap kali kita memperbarui atau memperbaiki bagian dari page.inc, kita perlu mengingat untuk membuat perubahan yang sama pada page2.inc. Cara yang lebih baik adalah menggunakan pewarisan untuk membuat kelas baru yang mewarisi sebagian besar fungsinya dari Page, tetapi mengganti bagian yang perlu berbeda. Untuk situs TLA, kita ingin mengharuskan halaman layanan menyertakan bilah navigasi kedua. Skrip yang ditunjukkan pada Daftar 6.3 melakukan ini dengan membuat kelas baru bernama ServicesPage yang mewarisi dari Page. Kita menyediakan array baru bernama $row2buttons yang berisi tombol dan tautan yang kita inginkan di baris kedua. Karena kami ingin kelas ini berperilaku dengan cara yang hampir sama, kami hanya mengganti bagian yang ingin kami ubah—operasi Display()

**Daftar 6.3 Services.Php—Halaman Layanan Mewarisi Dari Kelas Halaman Tetapi Mengganti Display() Untuk Mengubah Output**

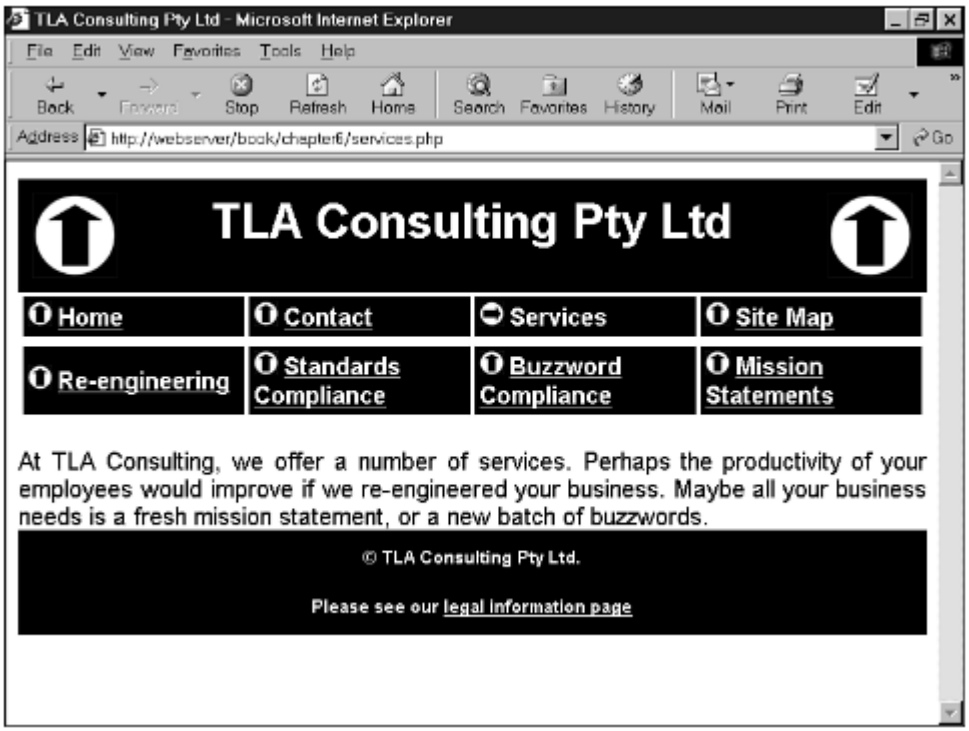
```
<?
  require (“page.inc”);

  class ServicesPage extends Page
  {
    var $row2buttons = array( “Re-engineering” => “reengineering.php”,
                              “Standards Compliance” => “standards.php”,
                              “Buzzword Compliance” => “buzzword.php”,
                              “Mission Statements” => “mission.php”
                        );
    function Display()
  {
    echo “<html>\n<head>\n”;
    $this -> DisplayTitle();
    $this -> DisplayKeywords();
    $this -> DisplayStyles();
    echo “</head>\n<body>\n”;
    $this -> DisplayHeader();
    $this -> DisplayMenu($this->buttons);
    $this -> DisplayMenu($this->row2buttons);
    echo $this->content;
    $this -> DisplayFooter();
    echo “</body>\n</html>\n”;
  }
}

$services = new ServicesPage();
$content =”<p>At TLA Consulting, we offer a number of services. Perhaps
          the productivity of your employees would improve if we re
          engineered your business. Maybe all your business needs is a
          fresh mission statement, or a new batch of buzzwords.”;
$services -> SetContent($content);
```

```
$services -> Display();
?>
```

`Display()` yang kita overriding sangat mirip, tetapi mengandung satu baris tambahan

```
$this -> DisplayMenu($this->row2buttons);
```

untuk memanggil `DisplayMenu()` untuk kedua kalinya dan membuat bilah menu kedua. Di luar definisi kelas, kita membuat contoh kelas ServicesPage, menetapkan nilai yang kita inginkan sebagai nilai non-default, dan memanggil `Display()`. Seperti yang ditunjukkan pada Gambar 6.2, kita memiliki varian baru dari halaman standar kita. Satu-satunya kode baru yang perlu kita tulis adalah untuk bagian-bagian yang berbeda.




**Gambar 6.2 Halaman Layanan Dibuat Menggunakan Pewarisan Untuk Menggunakan Kembali Sebagian Besar Halaman Standar Kita.**

Membuat halaman melalui kelas PHP memiliki keuntungan yang jelas. Dengan kelas yang melakukan sebagian besar pekerjaan untuk kita, kita perlu melakukan lebih sedikit pekerjaan untuk membuat halaman baru. Kita dapat memperbarui semua halaman kita sekaligus hanya dengan memperbarui kelas. Dengan menggunakan pewarisan, kita dapat memperoleh versi kelas yang berbeda dari yang asli tanpa mengorbankan keuntungannya. Seperti kebanyakan hal dalam hidup, keuntungan ini tidak datang tanpa biaya.

Membuat halaman dari skrip memerlukan lebih banyak upaya prosesor komputer daripada sekadar memuat halaman HTML statis dari disk dan mengirimkannya ke browser. Di

situs yang sibuk, ini akan menjadi penting, dan Anda harus berupaya untuk menggunakan halaman HTML statis atau menyimpan hasil skrip Anda jika memungkinkan untuk mengurangi beban pada server.

# BAB 7
# BERINTERAKSI DENGAN SISTEM FILE DAN SERVER

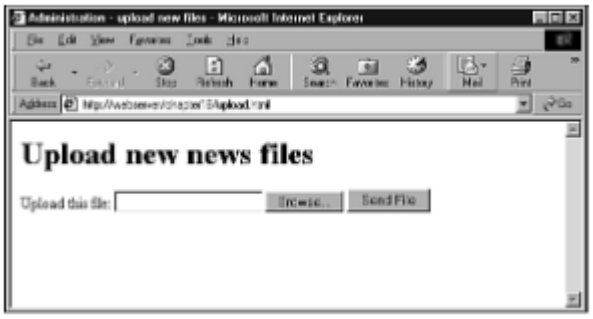
Dalam Bab 2, “Menyimpan dan Mengambil Data,” kita melihat cara membaca data dari dan menulis data ke berkas di server Web. Dalam bab ini, kita akan membahas fungsi PHP lain yang memungkinkan kita berinteraksi dengan sistem berkas di server Web. Kita akan membahas;

- Mengunggah berkas dengan PHP
- Menggunakan fungsi direktori
- Berinteraksi dengan berkas di server
- Menjalankan program di server
- Menggunakan variabel lingkungan server

Untuk membahas penggunaan fungsi-fungsi ini, kita akan melihat sebuah contoh. Pertimbangkan situasi di mana Anda ingin klien Anda dapat memperbarui sebagian konten situs Web—misalnya, berita terkini tentang perusahaan mereka. (Atau mungkin Anda menginginkan antarmuka yang lebih ramah daripada FTP untuk diri Anda sendiri.) Salah satu pendekatan untuk ini adalah dengan membiarkan klien mengunggah berkas konten sebagai teks biasa. Berkas-berkas ini kemudian akan tersedia di situs, melalui templat yang telah Anda rancang dengan PHP, seperti yang kita lakukan di Bab 6, “PHP Berorientasi Objek.” Sebelum kita menyelami fungsi sistem berkas, mari kita lihat sekilas cara kerja pengunggahan berkas.

## 7.1 PENGUNGGAHAN BERKAS

Salah satu bagian fungsionalitas PHP yang sangat berguna adalah dukungan untuk pengunggahan HTTP. Alih-alih berkas datang dari server ke peramban menggunakan HTTP, berkas-berkas tersebut pergi ke arah yang berlawanan, yaitu dari peramban ke server. Biasanya Anda mengimplementasikan ini dengan antarmuka formulir HTML. Yang akan kita gunakan dalam contoh kita ditunjukkan pada Gambar 7.1.




**Gambar 7.1** Formulir HTML yang kita gunakan untuk pengunggahan berkas memiliki kolom dan jenis kolom yang berbeda dari formulir HTML normal.

Seperti yang dapat Anda lihat, formulir tersebut memiliki kotak tempat pengguna dapat memasukkan nama berkas, atau mengeklik tombol Telusuri untuk menelusuri berkas yang tersedia secara lokal. Anda mungkin belum pernah melihat formulir unggah berkas sebelumnya. Kita akan melihat cara menerapkannya sebentar lagi. Setelah nama berkas dimasukkan, pengguna dapat mengeklik Kirim Berkas, dan berkas akan diunggah ke server, tempat skrip PHP menunggunya.

**HTML untuk Unggah Berkas**

Untuk menerapkan unggah berkas, kita perlu menggunakan beberapa sintaks HTML yang khusus dibuat untuk tujuan ini. HTML untuk formulir ini ditunjukkan pada Daftar 7.1.

**Daftar 7.1 Upload.HTML—formulir HTML untuk unggah berkas**

```
<html>
<head>
  <title>Administration - upload new files</title>
</head>
<body>
<h1>Upload new news files</h1>
<form      enctype=”multipart/form-data”      action=”upload.php”
 method=post>
  <input type=”hidden” name=”MAX_FILE_SIZE” value=”1000”>
  Upload this file: <input name=”userfile” type=”file”>
  <input type=”submit” value=”Send File”>
</form>
</body>
</html>
```

Perhatikan bahwa formulir ini menggunakan POST. Pengunggahan berkas juga akan berfungsi dengan metode PUT yang didukung oleh Netscape Composer dan Amaya. Pengunggahan berkas tidak akan berfungsi dengan GET.

Fitur tambahan dalam formulir ini adalah;

- Pada tag <form>, Anda harus menyetel atribut enctype=”multipart/form-data” untuk memberi tahu server bahwa berkas disertakan bersama informasi formulir biasa.
- Anda harus memiliki kolom formulir yang menyetel ukuran berkas maksimum yang dapat diunggah. Ini adalah kolom tersembunyi, dan ditampilkan di sini sebagai
  `<input type=”hidden” name=”MAX_FILE_SIZE” value=”1000”>`
  Nama kolom formulir ini harus MAX_FILE_SIZE. Nilainya adalah ukuran maksimum (dalam byte) berkas yang akan Anda izinkan untuk diunggah orang.
- Anda memerlukan input jenis file, yang ditampilkan di sini sebagai
  `<input name=”userfile” type=”file”>`

Anda dapat memilih nama apa pun yang Anda suka untuk berkas tersebut, tetapi ingatlah nama ini karena Anda akan menggunakan nama ini untuk mengakses berkas Anda dari skrip PHP penerima.

**Menulis PHP untuk Menangani Berkas**

Menulis PHP untuk menangkap berkas cukup mudah.

Saat berkas diunggah, berkas akan masuk ke lokasi sementara di server Web. Ini adalah direktori sementara default server Web. Jika Anda tidak memindahkan atau mengganti nama berkas sebelum skrip Anda selesai dieksekusi, berkas akan dihapus. Mengingat formulir HTML Anda memiliki kolom yang disebut userfile, Anda akan berakhir dengan empat variabel yang diteruskan ke PHP:

- Nilai yang disimpan dalam `$userfile` adalah tempat berkas disimpan sementara di server Web.
- Nilai yang disimpan dalam `$userfile_name` adalah nama berkas di sistem pengguna.
- Nilai yang disimpan dalam `$userfile_size` adalah ukuran berkas dalam byte.
- Nilai yang disimpan dalam `$userfile_type` adalah tipe MIME dari file tersebut, misalnya, text/plain atau image/gif.

Anda juga dapat mengakses variabel-variabel ini melalui array `$HTTP_POST_FILES`, sebagai berikut:

- `$HTTP_POST_FILES['userfile']['tmp_name']`
- `$HTTP_POST_FILES['userfile']['name']`
- `$HTTP_POST_FILES['userfile']['size']`
- `$HTTP_POST_FILES['userfile']['type']`

Mengingat Anda mengetahui di mana file tersebut berada dan apa namanya, Anda sekarang dapat menyalinnya ke suatu tempat yang berguna. Di akhir eksekusi skrip Anda, file sementara akan dihapus. Oleh karena itu, Anda harus memindahkan atau mengganti nama file jika Anda ingin menyimpannya.

Dalam contoh kita, kita akan menggunakan file yang diunggah sebagai artikel berita terkini, jadi kita akan menghapus semua tag yang mungkin ada di dalamnya, dan memindahkannya ke direktori yang lebih berguna. Skrip yang melakukan ini ditunjukkan pada Daftar 7.2.

**Daftar 7.2 Upload.php—php untuk menangkap file dari formulir HTML**

```
<head>
  <title>Uploading...</title>
</head>
<body>
<h1>Uploading file...</h1>
<?
  if ($userfile==”none”)
  {
    echo “Problem: no file uploaded”;
    exit;
  }
  if ($userfile_size==0)
  {
    echo “Problem: uploaded file is zero length”;
```

```
    exit;
  }
      if ($userfile_type != “text/plain”)
  {
    echo “Problem: file is not plain text”;
    exit;
  }
  if (!is_uploaded_file($userfile))
  {
    echo “Problem: possible file upload attack”;
    exit;
  }
  $upfile = “/home/book/uploads/”.$userfile_name;
  if ( !copy($userfile, $upfile))
  {
    echo “Problem: Could not move file into directory”;
    exit;
  }
  echo “File uploaded successfully<br><br>”;
  $fp = fopen($upfile, “r”);
  $contents = fread ($fp, filesize ($upfile));
  fclose ($fp);
  $contents = strip_tags($contents);
  $fp = fopen($upfile, “w”);
  fwrite($fp, $contents);
  fclose($fp);
  echo “Preview of uploaded file contents:<br><hr>”;
  echo $contents;
  echo “<br><hr>”;
  ?>
</body>
</html>
<?
  // This function is from the PHP manual.
  // is_uploaded_file is built into PHP4.0.3.
  // Prior to that, we can use this code.
  function is_uploaded_file($filename) {
  if   (!$tmp_file = get_cfg_var(‘upload_tmp_dir’)) {
       $tmp_file = dirname(tempnam(‘’, ‘’));
  }
  $tmp_file .= ‘/’ . basename($filename);
  /* User might have trailing slash in php.ini... */
  return (ereg_replace(‘/+’, ‘/’, $tmp_file) == $filename);
  }
?>
```

Yang cukup menarik, sebagian besar skrip ini adalah pengecekan kesalahan. Pengunggahan berkas melibatkan potensi risiko keamanan, dan kita perlu mengurangi risiko ini jika memungkinkan. Kita perlu memvalidasi berkas yang diunggah secermat mungkin untuk memastikannya aman untuk ditampilkan kepada pengunjung kita. Mari kita bahas bagian utama skrip ini.

Pertama, kita periksa apakah `$userfile` adalah "none". Ini adalah nilai yang ditetapkan oleh PHP jika tidak ada berkas yang diunggah. Kita juga menguji apakah berkas tersebut memiliki beberapa konten (dengan menguji apakah `$userfile_size` lebih besar dari 0), dan apakah konten tersebut bertipe benar (dengan menguji `$userfile_type`).

Kemudian kita periksa apakah berkas yang kita coba buka benar-benar telah diunggah dan bukan berkas lokal seperti `/etc/passwd`. Kita akan membahasnya nanti. Jika semuanya berjalan lancar, kita salin berkas tersebut ke direktori include kita. Kita gunakan `/home/book/uploads/` dalam contoh ini—berada di luar pohon dokumen Web, dan karenanya merupakan tempat yang baik untuk meletakkan berkas yang akan disertakan di tempat lain.

Kemudian kita buka berkas tersebut, bersihkan tag HTML atau PHP apa saja yang mungkin ada di dalam berkas menggunakan fungsi `strip_tags()`, lalu tulis kembali berkas tersebut. Akhirnya, kami menampilkan isi berkas sehingga pengguna dapat melihat bahwa berkas mereka berhasil diunggah. Hasil dari satu (berhasil) menjalankan skrip ini ditunjukkan pada Gambar 7.2.

Pada bulan September 2000, sebuah eksploitasi diumumkan yang dapat memungkinkan seorang cracker untuk mengelabui skrip pengunggahan file Anda agar memproses file lokal seolah-olah telah diunggah. Eksploitasi ini didokumentasikan di milis BUGTRAQ.


**Gambar 7.2** Setelah file disalin dan diformat ulang, file yang diunggah ditampilkan sebagai konfirmasi kepada pengguna bahwa pengunggahan berhasil.

Kami telah menggunakan fungsi `is_uploaded_file()` untuk memastikan bahwa file yang kami proses benar-benar telah diunggah dan bukan file lokal seperti `/etc/passwd`. Fungsi

ini akan ada di PHP versi 4.0.3. Pada saat penulisan, rilis saat ini adalah 4.0.2, jadi kami telah menggunakan kode contoh untuk fungsi ini dari manual PHP. Kecuali Anda menulis skrip penanganan unggahan dengan hati-hati, pengunjung yang berniat jahat dapat memberikan nama berkas sementara miliknya sendiri dan meyakinkan skrip Anda untuk menangani berkas tersebut seolah-olah berkas tersebut adalah berkas yang diunggah.

Karena banyak skrip unggahan berkas yang menampilkan kembali data yang diunggah kepada pengguna, atau menyimpannya di suatu tempat yang dapat dimuat, hal ini dapat menyebabkan orang dapat mengakses berkas apa pun yang dapat dibaca oleh server Web. Hal ini dapat mencakup berkas sensitif seperti `/etc/passwd` dan kode sumber PHP termasuk kata sandi basis data Anda.

**Masalah Umum**

Ada beberapa hal yang perlu diingat saat melakukan pengunggahan file.

- Contoh sebelumnya mengasumsikan bahwa pengguna telah diautentikasi di tempat lain. Anda tidak boleh mengizinkan sembarang orang mengunggah file ke situs Anda.
- Jika Anda mengizinkan pengguna yang tidak tepercaya atau tidak diautentikasi untuk mengunggah file, sebaiknya Anda bersikap paranoid tentang kontennya. Hal terakhir yang Anda inginkan adalah skrip berbahaya diunggah dan dijalankan. Anda harus berhati-hati, bukan hanya terhadap jenis dan konten file seperti yang kita bahas di sini, tetapi juga terhadap nama file itu sendiri. Sebaiknya ganti nama file yang diunggah menjadi sesuatu yang Anda ketahui "aman".
- Jika Anda menggunakan NT atau komputer berbasis Windows lainnya, pastikan untuk menggunakan \\ alih-alih \di jalur file seperti biasa.
- Jika Anda mengalami masalah agar ini berfungsi, periksa file php.ini Anda. Anda perlu menyetel perintah upload_tmp_dir untuk menunjuk ke beberapa direktori yang dapat Anda akses. Anda mungkin juga perlu menyesuaikan perintah memory_limit jika ingin mengunggah file besar—ini akan menentukan ukuran file maksimum dalam byte yang dapat Anda unggah.
- Jika PHP berjalan dalam mode aman, Anda akan mendapatkan pesan kesalahan tentang tidak dapat mengakses file sementara. Ini hanya dapat diperbaiki dengan tidak menjalankannya dalam mode aman atau dengan menulis skrip non-PHP yang menyalin file ke lokasi yang dapat diakses. Anda kemudian dapat menjalankan skrip ini dari skrip PHP Anda. Kita akan melihat cara menjalankan program di server dari PHP menjelang akhir bab ini.

## 7.2 MENGGUNAKAN FUNGSI DIREKTORI

Setelah pengguna mengunggah beberapa file, akan berguna bagi mereka untuk dapat melihat apa yang telah diunggah dan memanipulasi file konten. PHP memiliki serangkaian fungsi direktori dan sistem file yang berguna untuk tujuan ini.

**Membaca dari Direktori**

Pertama, kita akan menerapkan skrip untuk memungkinkan penelusuran direktori dari konten yang diunggah. Menjelajahi direktori sebenarnya sangat mudah di PHP. Dalam Daftar 7.3, kami menunjukkan skrip sederhana yang dapat digunakan untuk tujuan ini.

**Daftar 7.3 Browsedir.php—daftar direktori file yang diunggah**

```
<html>
<head>
  <title>Browse Directories</title>
</head>
<body>
<h1>Browsing</h1>
<?
  $current_dir = "/home/book/uploads/";
  $dir = opendir($current_dir);
  echo "Upload directory is $current_dir<br>";
  echo "Directory Listing:<br><hr><br>";
  while ($file = readdir($dir))
  {
    echo "$file<br>";
  }
  echo "<hr><br>";
  closedir($dir);
?>
</body>
</html>
```

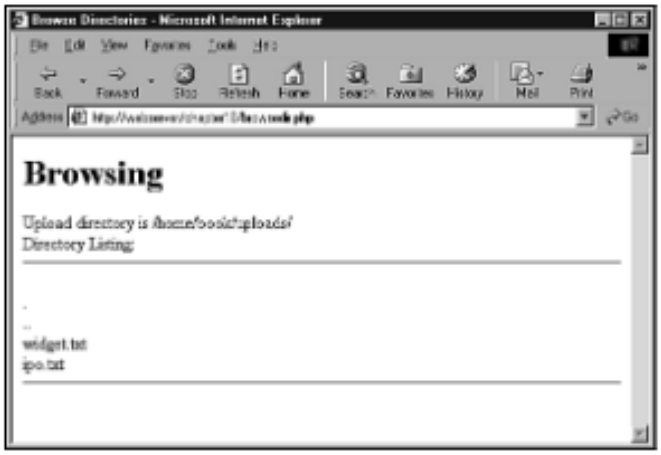
Skrip ini menggunakan fungsi `opendir()`, `closedir()`, dan `readdir()`. Fungsi `opendir()` digunakan untuk membuka direktori untuk dibaca. Penggunaannya sangat mirip dengan penggunaan `fopen()` untuk membaca dari berkas. Alih-alih memberikan nama berkas, Anda harus memberikannya nama direktori:

```
$dir = opendir($current_dir);
```

Fungsi ini mengembalikan handle direktori, lagi-lagi dengan cara yang hampir sama seperti `fopen()` mengembalikan handle berkas.

Saat direktori terbuka, Anda dapat membaca nama berkas dari direktori tersebut dengan memanggil `readdir($dir)`, seperti yang ditunjukkan dalam contoh. Fungsi ini mengembalikan false saat tidak ada lagi berkas yang harus dibaca. (Perhatikan bahwa fungsi ini juga akan mengembalikan false jika membaca berkas bernama "0"—tentu saja, Anda dapat mengujinya jika kemungkinan besar terjadi.) Berkas tidak diurutkan dalam urutan tertentu, jadi jika Anda memerlukan daftar yang diurutkan, Anda harus membacanya ke dalam array dan mengurutkannya. Saat Anda selesai membaca dari direktori, Anda memanggil

`closedir($dir)` untuk menyelesaikannya. Ini sama saja dengan memanggil `fclose()` untuk sebuah file. Contoh keluaran skrip penelusuran direktori ditunjukkan pada Gambar 7.3.




**Gambar 7.3** Daftar direktori menunjukkan semua berkas dalam direktori yang dipilih, termasuk direktori . (direktori saat ini) dan (satu tingkat di atas). Anda dapat memilih untuk memfilternya.

Jika Anda menyediakan penelusuran direktori melalui mekanisme ini, sebaiknya batasi direktori yang dapat ditelusuri sehingga pengguna tidak dapat menelusuri daftar direktori di area yang biasanya tidak tersedia untuknya. Fungsi terkait dan terkadang berguna adalah `rewinddir($dir)`, yang menyetel ulang pembacaan nama berkas ke awal direktori. Sebagai alternatif untuk fungsi ini, Anda dapat menggunakan kelas dir yang disediakan oleh PHP. Kelas ini memiliki properti handle dan path, serta metode `read()`, `close()`, dan `rewind()`, yang bekerja secara identik dengan alternatif non-kelas.

**Mendapatkan Info Tentang Direktori Saat Ini**

Kita dapat memperoleh beberapa informasi tambahan yang diberikan path ke berkas. Fungsi `dirname($path)` dan `basename($path)` masing-masing mengembalikan bagian direktori dari path dan bagian nama berkas dari path. Ini dapat berguna untuk peramban direktori kita, khususnya jika kita mulai membangun struktur direktori kompleks berisi konten berdasarkan nama direktori dan nama berkas yang bermakna.

Kita juga dapat menambahkan indikasi seberapa banyak ruang yang tersisa untuk unggahan ke daftar direktori kita dengan menggunakan fungsi `diskfreespace($path)`. Jika Anda meneruskan jalur ke direktori ke fungsi ini, fungsi ini akan mengembalikan jumlah byte yang kosong pada disk (Windows) atau sistem berkas (UNIX) tempat direktori berada.

**Membuat dan Menghapus Direktori**

Selain membaca informasi tentang direktori secara pasif, Anda dapat menggunakan fungsi PHP `mkdir()` dan `rmdir()` untuk membuat dan menghapus direktori. Anda hanya dapat membuat atau menghapus direktori di jalur yang dapat diakses oleh pengguna yang menjalankan skrip tersebut. Menggunakan `mkdir()` lebih rumit dari yang Anda kira.

Diperlukan dua parameter, jalur ke direktori yang diinginkan (termasuk nama direktori baru), dan izin yang Anda inginkan agar dimiliki direktori tersebut, misalnya,

```
mkdir("/tmp/testing", 0777);
```

Namun, izin yang Anda daftarkan belum tentu merupakan izin yang akan Anda dapatkan. Umask saat ini akan di-AND (seperti pengurangan) dengan nilai ini untuk mendapatkan izin yang sebenarnya. Misalnya, jika umask adalah 022, Anda akan mendapatkan izin 0755.

Anda mungkin ingin mengatur ulang umask sebelum membuat direktori untuk mengatasi efek ini, dengan memasukkan

```
$oldumask = umask(0);
mkdir("/tmp/testing", 0777);
umask($oldumask);
```

Kode ini menggunakan fungsi `umask()`, yang dapat digunakan untuk memeriksa dan mengubah umask saat ini. Fungsi ini akan mengubah umask saat ini menjadi apa pun yang diteruskan dan mengembalikan umask lama, atau jika dipanggil tanpa parameter, fungsi ini hanya akan mengembalikan umask saat ini. Fungsi `rmdir()` menghapus direktori, sebagai berikut:

```
rmdir("/tmp/testing");
 atau
rmdir("c:\\tmp\\testing");
```

## 7.3 BERINTERAKSI DENGAN SISTEM BERKAS

Selain melihat dan mendapatkan informasi tentang direktori, kita dapat berinteraksi dengan dan mendapatkan informasi tentang berkas di server Web. Sebelumnya kita telah melihat penulisan ke dan pembacaan dari berkas. Sejumlah besar fungsi berkas lainnya tersedia.

**Dapatkan Info Berkas**

Kita dapat mengubah bagian skrip penelusuran direktori yang membaca berkas sebagai berikut:

```
while ($file = $dir->read())
{
  echo "<a href=\"filedetails.php?file=".$file."\">".$file."</a><br>";
}
```

Kita kemudian dapat membuat skrip `filedetails.php` untuk memberikan informasi lebih lanjut tentang sebuah berkas. Isi berkas ini ditampilkan dalam Daftar 7.4. Satu peringatan tentang skrip ini: Beberapa fungsi yang digunakan di sini tidak didukung di Windows, termasuk `fileowner()` dan `filegroup()`, atau tidak didukung dengan baik.

**Daftar 7.4 Filedetails.php—fungsi status berkas dan hasilnya**

```
<html>
<head>
  <title>File Details</title>
</head>
<body>
<?
  $current_dir = "/home/book/uploads/";
  $file = basename($file); // strip off directory information for security
  echo "<h1>Details of file: ".$file."</h1>";
  $file = $current_dir.$file;

  echo "<h2>File data</h2>";
  echo "File last accessed: ".date("j F Y H:i", fileatime($file))."<br>";
  echo "File last modified: ".date("j F Y H:i", filemtime($file))."<br>";

  $user = posix_getpwuid(fileowner($file));
  echo "File owner: ".$user["name"]."<br>";

  $group = posix_getgrgid(filegroup($file));
  echo "File group: ".$group["name"]."<br>";

  echo "File permissions: ".decoct(fileperms($file))."<br>";

  echo "File type: ".filetype($file)."<br>";

  echo "File size: ".filesize($file)." bytes<br>";

  echo "<h2>File tests</h2>";

  echo "is_dir: ".(is_dir($file)? "true" : "false")."<br>";
  echo "is_executable: ".(is_executable($file)? "true" : "false")."<br>";
  echo "is_file: ".(is_file($file)? "true" : "false")."<br>";
  echo "is_link: ".(is_link($file)? "true" : "false")."<br>";
  echo "is_readable: ".(is_readable($file)? "true" : "false")."<br>";
  echo "is_writable: ".(is_writable($file)? "true" : "false")."<br>";
?>
</body>
</html>
```

Hasil dari satu contoh yang dijalankan pada Daftar 7.4 ditunjukkan pada Gambar 7.4.


**Gambar 7.4** Tampilan Detail File memperlihatkan informasi sistem file tentang suatu file. Perhatikan bahwa izin diperlihatkan dalam format oktal.

Mari kita bahas tentang fungsi-fungsi yang digunakan dalam Daftar 7.4. Seperti yang disebutkan sebelumnya, fungsi `basename()` mendapatkan nama file tanpa direktori. (Anda juga dapat menggunakan fungsi `dirname()` untuk mendapatkan nama direktori tanpa nama file.) Fungsi `fileatime()` dan `filemtime()` mengembalikan cap waktu saat file terakhir diakses dan terakhir dimodifikasi. Kami telah memformat ulang cap waktu menggunakan fungsi `date()` agar lebih mudah dibaca manusia. Fungsi-fungsi ini akan mengembalikan nilai yang sama pada beberapa sistem operasi (seperti dalam contoh) tergantung pada informasi apa yang disimpan sistem.

Fungsi `fileowner()` dan `filegroup()` mengembalikan ID pengguna (uid) dan ID grup (gid) file. Ini dapat diubah menjadi nama menggunakan fungsi `posix_getpwuid()` dan `posix_getgrgid()`, yang membuatnya sedikit lebih mudah dibaca. Fungsi-fungsi ini mengambil uid atau gid sebagai parameter dan mengembalikan array asosiatif berisi informasi tentang pengguna atau grup, termasuk nama pengguna atau grup, seperti yang telah kami gunakan dalam skrip ini.

Fungsi `fileperms()` mengembalikan izin pada file. Kami telah memformat ulang izin tersebut sebagai angka oktal menggunakan fungsi `decoct()` untuk memasukkannya ke dalam format yang lebih familiar bagi pengguna UNIX. Fungsi filetype() mengembalikan beberapa informasi tentang jenis file yang sedang diperiksa. Hasil yang mungkin adalah fifo, char, dir, block, link, file, dan unknown.

Fungsi `filesize()` mengembalikan ukuran file dalam byte. Set fungsi kedua—`is_dir()`, `is_executable()`, `is_file()`, `is_link()`, `is_readable()`, dan `is_writable()`—semuanya menguji atribut bernama dari sebuah file dan mengembalikan true atau false. Sebagai alternatif, kami dapat menggunakan fungsi `stat()` untuk mengumpulkan banyak informasi yang sama. Ketika diberikan sebuah file, fungsi ini

mengembalikan array yang berisi data yang mirip dengan fungsi-fungsi ini. Fungsi `lstat()` serupa, tetapi untuk digunakan dengan tautan simbolik.

Semua fungsi status file cukup mahal untuk dijalankan dalam hal waktu. Oleh karena itu, hasilnya di-cache. Jika Anda ingin memeriksa beberapa informasi berkas sebelum dan sesudah perubahan, Anda perlu memanggil untuk menghapus hasil sebelumnya.

```
clearstatcache();
```

Jika Anda ingin menggunakan skrip sebelumnya sebelum dan sesudah mengubah beberapa data berkas, Anda harus mulai dengan memanggil fungsi ini untuk memastikan data yang dihasilkan mutakhir.

**Mengubah Properti Berkas**

Selain melihat properti berkas, kita dapat mengubahnya. Masing-masing fungsi `chgrp`(berkas, grup), `chmod`(berkas, izin), dan `chown`(berkas, pengguna) berperilaku serupa dengan padanan UNIX-nya. Tak satu pun dari fungsi ini akan berfungsi di sistem berbasis Windows, meskipun `chown()` akan dijalankan dan selalu mengembalikan true.

Fungsi `chgrp()` digunakan untuk mengubah grup berkas. Fungsi ini hanya dapat digunakan untuk mengubah grup ke grup tempat pengguna menjadi anggotanya kecuali pengguna adalah root. Fungsi `chmod()` digunakan untuk mengubah izin pada berkas. Izin yang Anda berikan kepadanya berada dalam format chmod UNIX yang biasa—Anda harus mengawalinya dengan “0” untuk menunjukkan bahwa izin tersebut dalam format oktal, misalnya,

```
chmod(“somefile.txt”, 0777);
```

Fungsi `chown()` digunakan untuk mengubah pemilik file. Fungsi ini hanya dapat digunakan jika skrip berjalan sebagai root, yang seharusnya tidak pernah terjadi.

## 7.4 MEMBUAT, MENGHAPUS, DAN MEMINDAHKAN FILE

Anda dapat menggunakan fungsi sistem file untuk membuat, memindahkan, dan menghapus file. Pertama, dan paling mudah, Anda dapat membuat file, atau mengubah waktu terakhir file dimodifikasi, menggunakan fungsi `touch()`. Fungsi ini bekerja mirip dengan perintah UNIX touch. Fungsi ini memiliki prototipe berikut:

```
int touch (string file, [int time])
```

Jika file sudah ada, waktu modifikasinya akan diubah ke waktu saat ini, atau waktu yang diberikan pada parameter kedua jika ditentukan. Jika Anda ingin menentukannya, waktu tersebut harus diberikan dalam format cap waktu. Jika file tidak ada, file akan dibuat. Anda juga dapat menghapus file menggunakan fungsi `unlink()`. (Perhatikan bahwa fungsi ini tidak disebut;

```
delete—tidak ada penghapusan.)
```

Anda menggunakannya seperti ini:

```
unlink($filename);
```

Ini adalah salah satu fungsi yang tidak berfungsi dengan build Win32. Namun, Anda dapat menghapus file di Windows dengan `system("del filename.ext")`; Anda dapat menyalin dan memindahkan file dengan fungsi `copy()` dan `rename()`, sebagai berikut:

```
copy($source_path, $destination_path);
rename($oldfile, $newfile);
```

Anda mungkin telah memperhatikan bahwa kami menggunakan `copy()` dalam Listing 16.2.

Fungsi `rename()` berfungsi ganda sebagai fungsi untuk memindahkan file dari satu tempat ke tempat lain karena PHP tidak memiliki fungsi pemindahan. Apakah Anda dapat memindahkan file dari satu sistem file ke sistem file lain, dan apakah file ditimpa saat `rename()` digunakan bergantung pada sistem operasi, jadi periksa efeknya pada server Anda. Selain itu, berhati-hatilah dengan jalur yang Anda gunakan ke nama file. Jika relatif, ini akan relatif terhadap lokasi skrip, bukan file asli.

**Menggunakan Fungsi Eksekusi Program**

Kita akan beralih dari fungsi sistem file sekarang, dan melihat fungsi yang tersedia untuk menjalankan perintah di server. Ini berguna saat Anda ingin menyediakan front end berbasis Web ke sistem berbasis baris perintah yang ada. Misalnya, kami telah menggunakan perintah ini untuk menyiapkan front end bagi pengelola milis ezmlm. Kami akan menggunakannya lagi saat membahas studi kasus di buku ini nanti.

Ada empat teknik yang dapat Anda gunakan untuk menjalankan perintah di server Web. Semuanya cukup mirip, tetapi ada beberapa perbedaan kecil.

**1. `exec()`**

Fungsi `exec()` memiliki prototipe berikut:

string exec (string command [, array result [, int return_value]]) Anda memasukkan perintah yang ingin Anda jalankan, misalnya, exec("ls -la");

Fungsi `exec()` tidak memiliki output langsung.

Fungsi ini mengembalikan baris terakhir dari hasil perintah.

Jika Anda memasukkan variabel sebagai result, Anda akan mendapatkan array string yang mewakili setiap baris output. Jika Anda memasukkan variabel sebagai return_value, Anda akan mendapatkan kode pengembalian.

**2. `passthru()`**

Fungsi `passthru()` memiliki prototipe berikut:

void passthru (string command [, int return_value])

Fungsi `passthru()` secara langsung menampilkan output-nya ke browser. (Ini berguna jika outputnya biner, misalnya, beberapa jenis data gambar.)

Tidak mengembalikan apa pun.

Parameter bekerja dengan cara yang sama seperti parameter `exec()`.

**3. `system()`**

Fungsi `system()` memiliki prototipe berikut:

string system (string command [, int return_value])

Fungsi menggemakan output perintah ke browser. Ia mencoba untuk membersihkan output setelah setiap baris (dengan asumsi Anda menjalankan PHP sebagai modul server), yang membedakannya dari `passthru()`.

Ia mengembalikan baris terakhir output (jika berhasil) atau false (jika gagal). Parameter bekerja dengan cara yang sama seperti pada fungsi lainnya.

**4. Backticks**

Kami telah menyebutkan ini secara singkat di Bab 1, "Pengetahuan Singkat PHP." Ini sebenarnya adalah operator eksekusi.

Ia tidak memiliki output langsung. Hasil dari mengeksekusi perintah dikembalikan sebagai string, yang kemudian dapat digemakan atau apa pun yang Anda suka. Skrip yang ditunjukkan pada Daftar 7.5 mengilustrasikan cara menggunakan masing-masingnya dengan cara yang setara.

**Daftar 7.5 Progex.php—fungsi status file dan hasilnya**

```
<?
  echo "<pre>";

  // exec version
  exec("ls -la", $result);
  foreach ($result as $line)
    echo "$line\n";

  echo "<br><hr><br>";

  // passthru version
  passthru("ls -la");

  echo "<br><hr><br>";

  // system version
  $result = system("ls -la");

  echo "<br><hr><br>";

  //backticks version
  $result = `ls -al`;
  echo $result;

  echo "</pre>";
```

```
?>
```

Kita dapat menggunakan salah satu pendekatan ini sebagai alternatif skrip penelusuran direktori yang kita tulis sebelumnya.

Jika Anda berencana untuk menyertakan data yang dikirimkan pengguna sebagai bagian dari perintah yang akan Anda jalankan, Anda harus selalu menjalankannya melalui fungsi `escapeshellcmd()` terlebih dahulu. Ini menghentikan pengguna dari menjalankan perintah dengan niat jahat (atau sebaliknya) di sistem Anda. Anda dapat menyebutnya seperti ini, misalnya,

```
System(escapeshellcmd($command_with_user_data));
```

**Berinteraksi dengan Lingkungan: `Getenv()` dan `Putenv()`**

Sebelum kita meninggalkan bagian ini, kita akan melihat bagaimana Anda dapat menggunakan variabel lingkungan dari dalam PHP. Ada dua fungsi untuk tujuan ini: `getenv()`, yang memungkinkan Anda untuk mengambil variabel lingkungan, dan `putenv()`, yang memungkinkan Anda untuk mengatur variabel lingkungan. Perhatikan bahwa lingkungan yang kita bicarakan di sini adalah lingkungan tempat PHP berjalan di server.

Anda bisa mendapatkan daftar semua variabel lingkungan PHP dengan menjalankan `phpinfo()`. Beberapa lebih berguna daripada yang lain; misalnya,

```
getenv(“HTTP_REFERER”);
```

akan mengembalikan URL halaman tempat pengguna membuka halaman saat ini. Anda juga dapat mengatur variabel lingkungan sebagaimana diperlukan dengan `putenv()`, misalnya,

```
$home = “/home/nobody”;
putenv (“HOME=$home “);
```

# BAB 8
# MENGGUNAKAN FUNGSI JARINGAN DAN PROTOKOL

Dalam bab ini, kita akan melihat fungsi berorientasi jaringan dalam PHP yang memungkinkan skrip Anda berinteraksi dengan seluruh Internet. Ada banyak sekali sumber daya di luar sana, dan beragam protokol tersedia untuk menggunakannya. Di bagian ini, kita akan mempertimbangkan:

- Tinjauan umum protokol yang tersedia
- Mengirim dan membaca email
- Menggunakan layanan Web lain melalui HTTP
- Menggunakan fungsi pencarian jaringan
- Menggunakan FTP
- Menggunakan komunikasi jaringan generik dengan cURL

## 8.1 PENGERTIAN PROTOKOL

Protokol adalah aturan komunikasi untuk situasi tertentu. Misalnya, Anda mengetahui protokol saat bertemu orang lain: Anda menyapa, berjabat tangan, berkomunikasi sebentar, lalu mengucapkan selamat tinggal. Protokol jaringan komputer serupa. Seperti protokol manusia, protokol komputer yang berbeda digunakan untuk situasi dan aplikasi yang berbeda. Kita menggunakan HTTP, Hypertext Transfer Protocol, untuk mengirim dan menerima halaman Web. Anda mungkin juga pernah menggunakan FTP, file transfer protocol, untuk mentransfer file antarmesin di jaringan. Masih banyak lagi yang lain.

Protokol dan Standar Internet lainnya dijelaskan dalam dokumen yang disebut RFC, atau Permintaan Komentar. Protokol ini ditetapkan oleh *Internet Engineering Task Force* (IETF). RFC tersedia secara luas di Internet. Sumber dasarnya adalah Editor RFC di *http://www.rfc-editor.org/*.

Jika Anda mengalami masalah saat bekerja dengan protokol tertentu, RFC adalah sumber yang berwenang dan sering kali berguna untuk memecahkan masalah kode Anda. Namun, RFC sangat terperinci dan sering kali mencapai ratusan halaman. Beberapa contoh RFC yang terkenal adalah RFC2616, yang menjelaskan protokol HTTP/1.1, dan RFC822, yang menjelaskan format pesan email Internet.

Dalam bab ini, kita akan melihat aspek-aspek PHP yang menggunakan beberapa protokol ini. Secara khusus, kita akan membahas tentang pengiriman email dengan SMTP, membaca email dengan POP dan IMAP, menghubungkan ke server Web lain melalui HTTP dan HTTPS, dan mentransfer file dengan FTP.

**Mengirim dan Membaca Email**

Cara utama untuk mengirim email dalam PHP adalah dengan menggunakan fungsi `mail()` sederhana. Kami membahas penggunaan fungsi ini di Bab 4, “Manipulasi String dan Ekspresi Reguler,” jadi kami tidak akan membahasnya lagi di sini. Fungsi ini menggunakan SMTP *(Simple Mail Transfer Protocol)* untuk mengirim email.

Anda dapat menggunakan berbagai kelas yang tersedia secara gratis untuk menambah fungsionalitas `mail()`. SMTP hanya untuk mengirim email. Protokol IMAP (*Internet Message Access Protocol*, dijelaskan dalam RFC2060) dan POP (*Post Office Protocol*, dijelaskan dalam RFC1939 atau STD0053) digunakan untuk membaca email dari server email. Protokol ini tidak dapat mengirim email. IMAP digunakan untuk membaca dan memanipulasi pesan email yang disimpan di server, dan lebih canggih daripada POP yang umumnya digunakan hanya untuk mengunduh pesan email ke klien dan menghapusnya dari server. PHP dilengkapi dengan pustaka IMAP. Ini juga dapat digunakan untuk membuat POP dan NNTP *(Network News Transfer Protocol)* serta koneksi IMAP.

**Menggunakan Layanan Web Lainnya**

Salah satu hal hebat yang dapat Anda lakukan dengan Web adalah menggunakan, memodifikasi, dan menanamkan layanan dan informasi yang ada ke dalam halaman Anda sendiri. PHP mempermudah hal ini. Mari kita lihat contoh untuk mengilustrasikannya. Bayangkan bahwa perusahaan tempat Anda bekerja ingin agar kutipan saham perusahaan Anda ditampilkan di berandanya. Informasi ini tersedia di beberapa situs bursa saham di suatu tempat—tetapi bagaimana kita mendapatkannya?

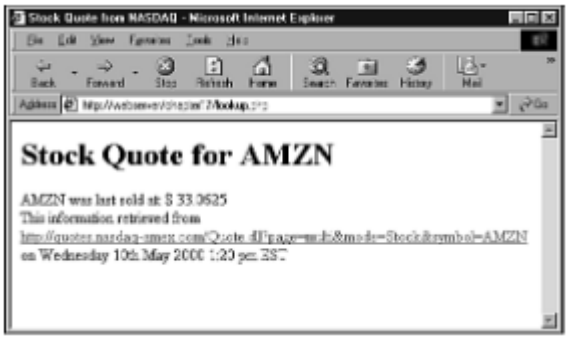
Awali dengan menemukan URL sumber asli untuk informasi tersebut. Bila Anda mengetahui hal ini, setiap kali seseorang membuka beranda Anda, Anda dapat membuka koneksi ke URL tersebut, mengambil halaman tersebut, dan menarik informasi yang Anda perlukan. Sebagai contoh, kami telah menyusun skrip yang mengambil dan memformat ulang kutipan saham dari NASDAQ. Untuk tujuan contoh ini, kami telah mengambil harga saham *Amazon.com* saat ini. (Informasi yang ingin Anda sertakan di halaman Anda mungkin berbeda, tetapi prinsipnya sama.) Skrip ini ditampilkan dalam Daftar 8.1.

**Daftar 8.1 Lookup.php—skrip mengambil kutipan saham dari nasdaq untuk saham dengan simbol ticker tercantum dalam simbol**

```
<html>
<head>
  <title>Stock Quote from NASDAQ</title>
</head>
<body>
<?
  // choose stock to look at
  $symbol=”AMZN”;
  echo “<h1>Stock Quote for $symbol</h1>”;
  // connect to URL and read information
  $theurl = “http://quotes.nasdaq-amex.com/Quote.dll?”
           .”page=multi&mode=Stock&symbol=”.$symbol;
  if (!($fp = fopen($theurl, “r”)))
  {
    echo “Could not open URL”;
    exit;
  }
```

```
    $contents = fread($fp, 1000000);
    fclose($fp);
    // find the part of the page we want and output it
    $pattern = "(\\$[0-9 ]+\\.[0-9]+)";
    if (eregi($pattern, $contents, $quote))
    {
      echo "$symbol was last sold at: ";
      echo $quote[1];
    } else
      {
        echo "No quote available";
      };
    // acknowledge source
    echo "<br>"
  ."This information retrieved from <br>"
  ."<a href=\"$theurl\">$theurl</a><br>"
  ."on ".(date("l jS F Y g:i a T"));
  ?>
  </body>
  </html>
```

Output dari satu contoh yang dijalankan pada Daftar 8.1 ditunjukkan pada Gambar 8.1.




**Gambar 17.1** Skrip tersebut menggunakan ekspresi reguler untuk menarik kutipan saham dari informasi yang diambil dari NASDAQ.

Skrip itu sendiri cukup mudah—bahkan, skrip tersebut tidak menggunakan fungsi apa pun yang belum pernah kita lihat sebelumnya, hanya aplikasi baru dari fungsi tersebut. Anda mungkin ingat bahwa ketika kita membahas pembacaan dari file di Bab 2, "Menyimpan dan Mengambil Data," kita menyebutkan bahwa Anda dapat menggunakan fungsi file untuk membaca dari URL. Itulah yang telah kita lakukan dalam kasus ini. Panggilan ke `fopen()`

```
$fp = fopen($theurl, "r")
```

mengembalikan pointer ke awal halaman di URL yang kita berikan. Kemudian tinggal membaca dari halaman di URL tersebut dan menutupnya lagi:

```
$contents = fread($fp, 1000000);
fclose($fp);
```

Anda akan melihat bahwa kami menggunakan angka yang sangat besar untuk memberi tahu PHP seberapa banyak yang harus dibaca dari berkas tersebut. Dengan berkas di server, Anda biasanya menggunakan `filesize($file)`, tetapi ini tidak berfungsi dengan URL.

Setelah kami melakukan ini, kami memiliki seluruh teks halaman Web di URL tersebut yang tersimpan di `$contents`. Kami kemudian dapat menggunakan ekspresi reguler dan fungsi `eregi()` untuk menemukan bagian halaman yang kami inginkan:

```
$pattern = "(\\$[0-9 ]+\\.[0-9]+)";
if (eregi($pattern, $contents, $quote))
{
  echo "$symbol was last sold at: ";
  echo $quote[1];
}
```

Anda dapat menggunakan pendekatan ini untuk berbagai keperluan. Contoh bagus lainnya adalah mengambil informasi cuaca lokal dan menanamkannya di halaman Anda. Pemanfaatan terbaik dari pendekatan ini adalah menggabungkan informasi dari berbagai sumber untuk menambah nilai.

Jika Anda menggunakan sumber informasi luar seperti ini untuk tujuan komersial, sebaiknya periksa sumbernya terlebih dahulu. Ada masalah kekayaan intelektual yang perlu dipertimbangkan dalam beberapa kasus. Jika Anda membuat skrip seperti ini, Anda mungkin ingin melewati beberapa data. Misalnya, jika Anda terhubung ke URL luar, Anda mungkin ingin meneruskan beberapa parameter yang diketik oleh pengguna. Jika Anda melakukan ini, sebaiknya gunakan fungsi `url_encode()`. Fungsi ini akan mengambil string dan mengubahnya ke format yang tepat untuk URL, misalnya, mengubah spasi menjadi tanda tambah. Anda dapat menyebutnya seperti ini:

```
$encodedparameter = url_encode($parameter);
```

**Menggunakan Fungsi Pencarian Jaringan**

PHP menawarkan serangkaian fungsi "pencarian" yang dapat digunakan untuk memeriksa informasi tentang nama host, alamat IP, dan pertukaran email. Misalnya, jika Anda menyiapkan situs direktori seperti Yahoo! saat URL baru dikirimkan, Anda mungkin ingin memeriksa secara otomatis bahwa host URL dan informasi kontak untuk situs tersebut valid. Dengan cara ini, Anda dapat menghemat biaya lebih lanjut saat pengulas datang untuk melihat situs dan menemukan bahwa situs tersebut tidak ada, atau alamat emailnya tidak valid. Daftar 8.2 menunjukkan HTML untuk formulir pengiriman untuk direktori seperti ini.

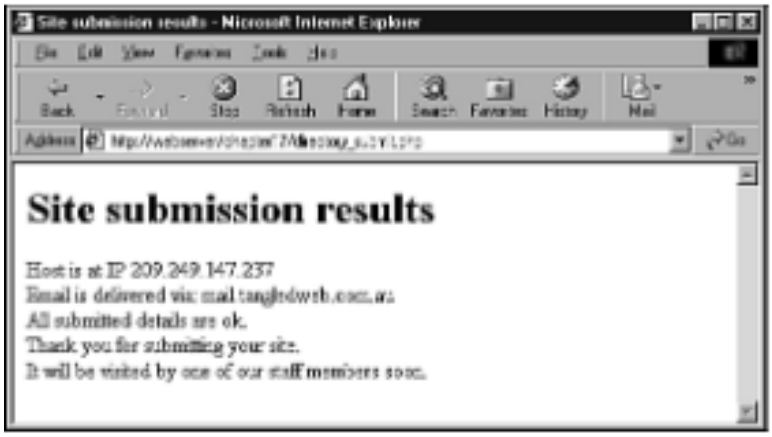
**Daftar 8.2 Directory_submit.html—html untuk formulir pengiriman**

```
<head>
  <title>Submit your site</title>
</head>
<body>
<h1>Submit site</h1>
<form method=post action="directory_submit.php">
URL: <input type=text name="url" size=30 value="http://"><br>
Email contact: <input type=text name="email" size=23><br>
<input type="submit" name="Submit site">
</form>
</body>
</html>
```

Ini adalah formulir yang sangat sederhana—versi yang dirender, dengan beberapa contoh data yang dimasukkan, ditunjukkan pada


**Gambar 8.2** Pengiriman direktori biasanya memerlukan URL Anda dan beberapa detail kontak sehingga administrator direktori dapat memberi tahu Anda saat situs Anda ditambahkan ke direktori.

Saat tombol kirim ditekan, pertama-tama kami ingin memeriksa apakah URL dihosting di mesin sungguhan, dan kedua, apakah bagian host dari alamat email juga ada di mesin sungguhan. Kami telah menulis skrip untuk memeriksa hal-hal ini, dan hasilnya ditunjukkan pada Gambar 8.3.




**Gambar 8.3** Versi skrip ini menampilkan hasil pemeriksaan nama host untuk URL dan alamat email—versi produksi mungkin tidak menampilkan hasil ini, tetapi menarik untuk melihat informasi yang dikembalikan dari pemeriksaan kami.

Skrip yang melakukan pemeriksaan ini menggunakan dua fungsi dari rangkaian fungsi jaringan PHP—`gethostbyname()` dan `getmxrr()`. Skrip lengkapnya ditampilkan dalam Daftar 8.3.

**DAFTAR 8.3 Directory_submit.php—skrip untuk memverifikasi url dan alamat email**

```
<html>
<head>
  <title>Site submission results</title>
</head>
<body>
<h1>Site submission results</h1>
<?
  // Check the URL

  $url = parse_url($url);
  $host = $url[host];
  if(!($ip = gethostbyname($host)))
  {
    echo “Host for URL does not have valid IP”;
    exit;
  }

  echo “Host is at IP $ip <br>”;

  // Check the email address

  $email = explode(“@”, $email);
  $emailhost = $email[1];
  if (!getmxrr($emailhost, $mxhostsarr))
  {
```

```
      echo “Email address is not at valid host”;
      exit;
    }

    echo “Email is delivered via: “;
    foreach ($mxhostsarr as $mx)
      echo “$mx “;

    // If reached here, all ok
    echo “<br>All submitted details are ok.<br>”;
    echo “Thank you for submitting your site.<br>”
         .”It will be visited by one of our staff members soon.”

    // In real case, add to db of waiting sites...
  ?>
  </body>
  </html>
```

Mari kita bahas bagian-bagian menarik dari skrip ini.

Pertama, kita ambil URL dan terapkan fungsi `parse_url()` padanya. Fungsi ini mengembalikan array asosiatif dari berbagai bagian URL. Informasi yang tersedia adalah skema, pengguna, kata sandi, host, port, jalur, kueri, dan fragmen. Biasanya, Anda tidak akan memerlukan semua ini, tetapi berikut ini contoh bagaimana semua ini membentuk URL. Diberikan URL seperti
*http://nobody:secret@bigcompany.com:80/script.php?variable=value#anchor*
nilai dari setiap bagian array akan menjadi:

- `scheme: http://`
- `user: nobody`
- `pass: secret`
- `host: bigcompany.com`
- `port: 80`
- `path: script.php`
- `query: variable=value`
- `fragment: anchor`

Dalam skrip kita, kita hanya menginginkan informasi host, jadi kita menariknya keluar dari array sebagai berikut:

```
$url = parse_url($url);
$host = $url[host];
```

Setelah kita melakukan ini, kita bisa mendapatkan alamat IP dari host tersebut, jika ada di DNS. Kita dapat melakukan ini menggunakan fungsi `gethostbyname()`, yang akan mengembalikan IP jika ada, atau false jika tidak ada:

```
$ip = gethostbyname($host)
```

Anda juga dapat melakukannya dengan cara lain menggunakan fungsi `gethostbyaddr()`, yang mengambil IP sebagai parameter dan mengembalikan nama host. Jika Anda memanggil fungsi-fungsi ini secara berurutan, Anda mungkin akan mendapatkan nama host yang berbeda dari yang Anda gunakan sebelumnya. Ini dapat berarti bahwa situs tersebut menggunakan layanan hosting virtual.

Jika URL valid, kami akan memeriksa alamat email. Pertama, kami membaginya menjadi nama pengguna dan nama host dengan memanggil `explode()`:

```
$email = explode("@", $email);
$emailhost = $email[1];
```

Setelah kami memiliki bagian host dari alamat tersebut, kami dapat memeriksa apakah ada tempat untuk email tersebut menggunakan fungsi `getmxrr()`:

```
getmxrr($emailhost, $mxhostsarr)
```

Fungsi ini mengembalikan kumpulan rekaman MX (*Mail Exchange*) untuk alamat dalam array yang Anda berikan di `$mxhostarr`.

Rekaman MX disimpan di DNS dan dicari seperti nama host. Mesin yang tercantum dalam rekaman MX belum tentu merupakan mesin tempat email tersebut akhirnya akan berakhir. Sebaliknya, itu adalah mesin yang mengetahui ke mana harus mengarahkan email tersebut. (Bisa jadi ada lebih dari satu, oleh karena itu fungsi ini mengembalikan array, bukan string nama host.) Jika kita tidak memiliki rekaman MX di DNS, maka tidak ada tempat untuk email tersebut.

Jika semua pemeriksaan ini baik-baik saja, kita dapat meletakkan data formulir ini di database untuk ditinjau kemudian oleh anggota staf. Selain fungsi yang baru saja kita gunakan, Anda dapat menggunakan fungsi `checkdnsrr()` yang lebih umum, yang mengambil nama host dan mengembalikan true jika ada rekamannya di DNS.

**Menggunakan FTP**

File Transfer Protocol, atau FTP, digunakan untuk mentransfer file antara host di jaringan. Dengan menggunakan PHP, Anda dapat menggunakan `fopen()` dan berbagai fungsi file dengan FTP seperti yang dapat Anda lakukan dengan koneksi HTTP, untuk terhubung ke dan mentransfer file ke dan dari server FTP. Namun, ada juga serangkaian fungsi khusus FTP yang disertakan dengan instalasi PHP standar.

Fungsi-fungsi ini tidak terpasang pada instalasi standar secara default. Untuk menggunakannya di UNIX, Anda perlu menjalankan program konfigurasi PHP dengan opsi `--enable-ftp`, lalu jalankan kembali make. Untuk menggunakan fungsi FTP dengan biner Win32, Anda perlu menambahkan baris

```
extension=php_ftp.dll
```

di bawah bagian "Windows Extensions" pada file php.ini Anda.

## 8.2 MENGGUNAKAN FTP UNTUK MENCADANGKAN FILE

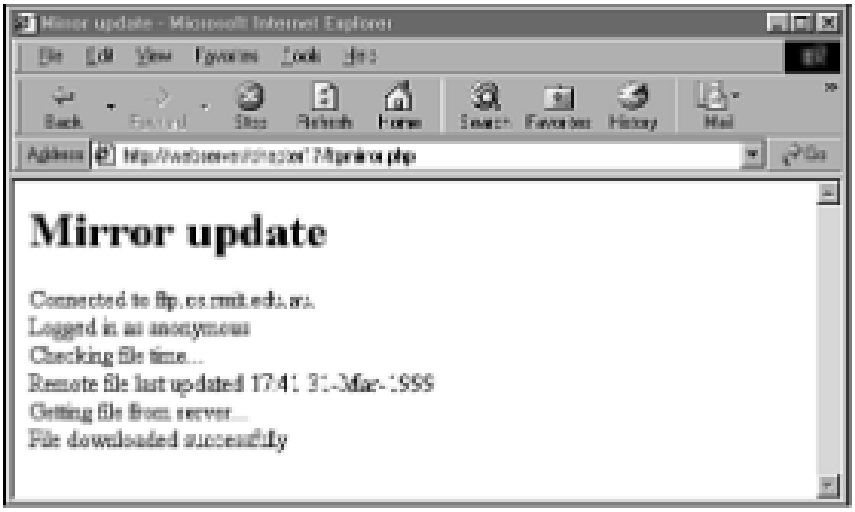
Fungsi FTP berguna untuk memindahkan dan menyalin file dari dan ke host lain. Salah satu penggunaan umum yang dapat Anda lakukan adalah untuk mencadangkan situs Web Anda atau mencerminkan file di lokasi lain. Kita akan melihat contoh sederhana menggunakan fungsi FTP untuk mencerminkan file. Skrip ini ditampilkan dalam Daftar 8.4.

**Daftar 8.4 FTPmirror.php—skrip untuk mengunduh versi baru file dari server FTP**

```
<html>
<head>
  <title>Mirror update</title>
</head>
<body>
<h1>Mirror update</h1>
<?

// set up variables - change these to suit application
$host = “ftp.cs.rmit.edu.au”;
$user = “anonymous”;
$password = “laura@tangledweb.com.au”;
$remotefile = “/pub/tsg/ttssh14.zip”;
$localfile = “$DOCUMENT_ROOT/../writable/ttssh14.zip”;

// connect to host
$conn = ftp_connect(“$host”);
if (!$conn)
{
  echo “Error: Could not connect to ftp server<br>”;
  exit;
}
echo “Connected to $host.<br>”;

// log in to host
@ $result = ftp_login($conn, $user, $pass);
if (!$result)
{
  echo “Error: Could not log on as $user<br>”;
  ftp_quit($conn);
  exit;
}
echo “Logged in as $user<br>”;

// check file times to see if an update is required
echo “Checking file time...<br>”;
if (file_exists($localfile))
{
  $localtime = filemtime($localfile);
```

```
    echo “Local file last updated “;
    echo date(“G:i j-M-Y”, $localtime);
    echo “<br>”;
  }
  else
    $localtime=0;
  $remotetime = ftp_mdtm($conn, $remotefile);
  if (!($remotetime >= 0))
  {
    // This doesn’t mean the file’s not there, server may not support mod time
    echo “Can’t access remote file time.<br>”;
    $remotetime=$localtime+1; // make sure of an update
  }
  else
  {
    echo “Remote file last updated “;
    echo date(“G:i j-M-Y”, $remotetime);
    echo “<br>”;
  }
  if (!($remotetime > $localtime))
  {
    echo “Local copy is up to date.<br>”;
    exit;
  }
  // download file
  echo “Getting file from server...<br>”;
  $fp = fopen ($localfile, “w”);
  if (!$success = ftp_fget($conn, $fp, $remotefile, FTP_BINARY))
  {
    echo “Error: Could not download file”;
    ftp_quit($conn);
    exit;
  }
  fclose($fp);
  echo “File downloaded successfully”;

  // close connection to host
  ftp_quit($conn);

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

Output dari menjalankan skrip ini pada satu kesempatan ditunjukkan pada Gambar 8.4.




**Gambar 8.4** Skrip mirroring FTP memeriksa apakah versi lokal dari sebuah file sudah mutakhir, dan mengunduh versi baru jika belum.

Ini adalah skrip yang cukup umum. Anda akan melihat bahwa skrip ini dimulai dengan menyiapkan beberapa variabel:

```
$host = “ftp.cs.rmit.edu.au”;
$user = “anonymous”;
$password = “laura@tangledweb.com.au”;
$remotefile = “/pub/tsg/ttssh14.zip”;
$localfile = “$DOCUMENT_ROOT/../writable/ttssh14.zip”;
```

Variabel `$host` harus berisi nama server FTP yang ingin Anda sambungkan, dan `$user` dan `$password` sesuai dengan nama pengguna dan kata sandi yang ingin Anda gunakan untuk masuk.

Banyak situs FTP mendukung apa yang disebut login anonim, yaitu nama pengguna yang tersedia secara bebas yang dapat digunakan siapa saja untuk terhubung. Tidak diperlukan kata sandi, tetapi merupakan kesopanan umum untuk menyediakan alamat email Anda sebagai kata sandi sehingga administrator sistem dapat melihat dari mana pengguna mereka berasal. Kami telah mengikuti konvensi ini di sini.

Variabel `$remotefile` berisi jalur ke file yang ingin kami unduh. Dalam kasus ini, kami mengunduh dan mencerminkan salinan lokal Tera Term SSH, klien SSH untuk Windows. (SSH adalah singkatan dari *Secure Shell*. Ini adalah bentuk terenkripsi dari Telnet.) Variabel `$localfile` berisi jalur ke lokasi tempat kami akan menyimpan file yang diunduh di komputer kami. Anda seharusnya dapat mengubah variabel ini untuk menyesuaikan skrip ini dengan tujuan Anda. Langkah-langkah dasar yang kami ikuti dalam skrip ini sama seperti jika Anda ingin secara manual meng-FTP file dari antarmuka baris perintah:

1. Hubungkan ke server FTP jarak jauh.
2. Masuk (baik sebagai pengguna atau anonim).
3. Periksa apakah file jarak jauh telah diperbarui.

4. Jika sudah, unduh.
5. Tutup koneksi FTP. Mari kita bahas masing-masing secara bergantian.

## 8.3 MENGHUBUNGKAN KE SERVER FTP JARAK JAUH

Langkah ini setara dengan mengetik:

```
ftp hostname
```

pada prompt perintah pada platform Windows atau UNIX. Kami menyelesaikan langkah ini dalam PHP dengan kode berikut:

```
$conn = ftp_connect(“$host”);
if (!$conn)
{
  echo “Error: Could not connect to ftp server<br>”;
  exit;
}
echo “Connected to $host.<br>”;
```

Fungsi yang dipanggil di sini adalah `ftp_connect()`. Fungsi ini mengambil nama host sebagai parameter, dan mengembalikan handle ke koneksi, atau false jika koneksi tidak dapat dibuat. Fungsi ini juga dapat mengambil nomor port pada host untuk terhubung sebagai parameter kedua opsional. (Kami tidak menggunakan ini di sini.) Jika Anda tidak menentukan nomor port, port default akan menjadi 21, default untuk FTP.

**Login ke Server FTP**

Langkah berikutnya adalah login sebagai pengguna tertentu dengan kata sandi tertentu. Anda dapat melakukannya menggunakan fungsi `ftp_login()`:

```
@ $result = ftp_login($conn, $user, $pass);
if (!$result)
{
  echo “Error: Could not log on as $user<br>”;
  ftp_quit($conn);
  exit;
}
echo “Logged in as $user<br>”;
```

Fungsi ini mengambil tiga parameter: koneksi FTP (diperoleh dari `ftp_connect()`), nama pengguna, dan kata sandi. Fungsi ini akan mengembalikan true jika pengguna dapat login, dan false jika tidak dapat login. Anda akan melihat bahwa kami meletakkan simbol @ di awal baris untuk menghilangkan kesalahan. Kami melakukan ini karena, jika pengguna tidak dapat login, Anda akan mendapatkan peringatan PHP di jendela browser Anda. Anda dapat menemukan kesalahan seperti yang telah kami lakukan di sini dengan menguji `$result`, dan memberikan

pesan kesalahan Anda sendiri yang lebih mudah digunakan. Perhatikan bahwa jika upaya login gagal, kami sebenarnya menutup koneksi FTP menggunakan

**`ftp_quit()`**—lebih lanjut tentang ini sebentar lagi.

**Memeriksa Waktu Pembaruan File**

Mengingat kita memperbarui salinan lokal suatu berkas, masuk akal untuk memeriksa apakah berkas tersebut perlu diperbarui terlebih dahulu karena Anda tidak ingin mengunduh ulang berkas, khususnya berkas berukuran besar, jika berkas tersebut mutakhir. Ini akan menghindari lalu lintas jaringan yang tidak perlu. Mari kita lihat kode yang melakukan ini.

Pertama, kita periksa apakah kita memiliki salinan lokal berkas tersebut, menggunakan fungsi `file_exists()`. Jika tidak, maka jelas kita perlu mengunduh berkas tersebut. Jika berkas tersebut ada, kita dapatkan waktu modifikasi terakhir berkas tersebut menggunakan fungsi `filemtime()`, dan simpan dalam variabel `$localtime`. Jika tidak ada, kita tetapkan variabel `$localtime` ke 0 sehingga akan menjadi "lebih lama" daripada waktu modifikasi berkas jarak jauh yang memungkinkan:

```
echo “Checking file time...<br>”;
if (file_exists($localfile))
{
  $localtime = filemtime($localfile);
  echo “Local file last updated “;
  echo date(“G:i j-M-Y”, $localtime);
  echo “<br>”;
}
else
  $localtime=0;
```

(Anda dapat membaca lebih lanjut tentang fungsi `file_exists()` dan `filemtime()` di Bab 2 dan Bab 7, “Berinteraksi dengan Sistem Berkas dan Server,” masing-masing.)

Setelah kita memilah waktu lokal, kita perlu mendapatkan waktu modifikasi berkas jarak jauh. Anda dapat memperolehnya menggunakan fungsi `ftp_mdtm()`:

```
$remotetime = ftp_mdtm($conn, $remotefile);
```

Fungsi ini mengambil dua parameter—handle koneksi FTP, dan jalur ke berkas jarak jauh—dan mengembalikan cap waktu UNIX saat berkas terakhir dimodifikasi, atau -1 jika terjadi kesalahan. Tidak semua server FTP mendukung fitur ini, jadi kita mungkin tidak mendapatkan hasil yang berguna dari fungsi tersebut. Dalam kasus ini, kita memilih untuk secara artifisial menetapkan variabel `$remotetime` menjadi “lebih baru” daripada variabel `$localtime` dengan menambahkan 1 ke dalamnya. Ini akan memastikan bahwa upaya dilakukan untuk mengunduh berkas:

```
if (!($remotetime >= 0))
{
  // This doesn't mean the file's not there, server may not support mod time
  echo "Can't access remote file time.<br>";
  $remotetime=$localtime+1; // make sure of an update
}
else
{
  echo "Remote file last updated ";
  echo date("G:i j-M-Y", $remotetime);
  echo "<br>";
}
```

Jika kita memiliki kedua waktu tersebut, kita dapat membandingkannya untuk melihat apakah kita perlu mengunduh berkas tersebut atau tidak:

```
if (!($remotetime > $localtime))
{
  echo "Local copy is up to date.<br>";
  exit;
}
```

**Mengunduh Berkas**

Pada tahap ini, kami akan mencoba mengunduh berkas dari server:

```
echo "Getting file from server...<br>";
$fp = fopen ($localfile, "w");
if (!$success = ftp_fget($conn, $fp, $remotefile, FTP_BINARY))
{
  echo "Error: Could not download file";
  fclose($fp);
  ftp_quit($conn);
  exit;
}
fclose($fp);
echo "File downloaded successfully";
```

Kita membuka berkas lokal menggunakan `fopen()` seperti yang telah kita lihat sebelumnya. Setelah kita melakukan ini, kita memanggil fungsi `ftp_fget()`, yang mencoba mengunduh berkas dan menyimpannya dalam berkas lokal. Fungsi ini mengambil empat parameter. Tiga parameter pertama bersifat langsung—koneksi FTP, handle berkas lokal, dan jalur ke berkas jarak jauh. Parameter keempat adalah mode FTP.

Ada dua mode untuk transfer FTP, ASCII dan biner. Mode ASCII digunakan untuk mentransfer file teks (yaitu, file yang hanya terdiri dari karakter ASCII), dan mode biner, digunakan untuk mentransfer semua yang lain. Pustaka FTP PHP dilengkapi dengan dua

konstanta yang telah ditetapkan sebelumnya, FTP_ASCII dan FTP_BINARY, yang mewakili kedua mode ini. Anda perlu memutuskan mode mana yang sesuai dengan jenis file Anda, dan meneruskan konstanta yang sesuai ke `ftp_fget()` sebagai parameter keempat. Dalam kasus ini, kami mentransfer file zip, jadi kami menggunakan mode FTP_BINARY.

Fungsi `ftp_fget()` mengembalikan true jika semuanya berjalan lancar, atau false jika terjadi kesalahan. Kami menyimpan hasilnya di `$success`, dan memberi tahu pengguna bagaimana hasilnya. Setelah unduhan dicoba, kami menutup file lokal menggunakan fungsi `fclose()`.

Sebagai alternatif untuk `ftp_fget()`, kami dapat menggunakan `ftp_get()`, yang memiliki prototipe berikut:

```
int ftp_get (int ftp_connection, string localfile_path,
          string remotefile_path, int mode)
```

Fungsi ini bekerja dengan cara yang hampir sama seperti `ftp_fget()`, tetapi tidak mengharuskan file lokal dibuka. Anda meneruskannya dengan nama file sistem dari file lokal yang ingin Anda tulis, bukan dengan handle file. Perlu dicatat bahwa tidak ada padanan untuk perintah FTP mget, yang dapat digunakan untuk mengunduh beberapa file sekaligus. Sebaliknya, Anda harus melakukan beberapa panggilan ke `ftp_fget()` atau `ftp_get()`.

**Menutup Koneksi**

Setelah kita selesai dengan koneksi FTP, Anda harus menutupnya menggunakan fungsi `ftp_quit()`:

```
ftp_quit($conn);
```

Anda harus meneruskan handle untuk koneksi FTP ke fungsi ini.

**Mengunggah File**

Jika Anda ingin melakukannya dengan cara lain, yaitu menyalin file dari server Anda ke mesin jarak jauh, Anda dapat menggunakan dua fungsi yang pada dasarnya merupakan kebalikan dari `ftp_fget()` dan `ftp_get()`. Fungsi-fungsi ini disebut `ftp_fput()` dan `ftp_put()`. Keduanya memiliki prototipe berikut:

```
int ftp_fput (int ftp_connection, string remotefile_path, int fp, int mode)
int ftp_put (int ftp_connection, string remotefile_path,
              string localfile_path, int mode)
```

Parameternya sama seperti padanan _get.

**Menghindari Batas Waktu**

Satu masalah yang mungkin Anda hadapi saat mengirim file melalui FTP adalah melebihi waktu eksekusi maksimum. Anda akan tahu apakah ini terjadi karena PHP akan memberi Anda pesan kesalahan. Ini kemungkinan besar terjadi jika server Anda berjalan pada jaringan yang lambat atau padat, atau jika Anda mengunduh file besar, seperti klip film.

Nilai default waktu eksekusi maksimum untuk semua skrip PHP ditetapkan dalam file php.ini. Secara default, ditetapkan pada 30 detik. Ini dirancang untuk menangkap skrip yang berjalan di luar kendali. Namun, saat Anda mengirim file melalui FTP, jika tautan Anda ke seluruh dunia lambat, atau jika filenya besar, transfer file bisa memakan waktu lebih lama dari ini. Untungnya, kita dapat mengubah waktu eksekusi maksimum untuk skrip tertentu menggunakan fungsi `set_time_limit()`. Memanggil fungsi ini akan menyetel ulang jumlah detik maksimum skrip yang diizinkan untuk berjalan, dimulai dari waktu fungsi dipanggil. Misalnya, jika Anda memanggil `set_time_limit(90);` maka skrip akan dapat berjalan selama 90 detik lagi sejak fungsi tersebut dipanggil.

## 8.4 MENGGUNAKAN FUNGSI FTP LAINNYA

Ada sejumlah fungsi FTP lain yang berguna di PHP. Fungsi `ftp_size()` dapat memberi tahu Anda ukuran file di server jarak jauh. Fungsi ini memiliki prototipe berikut:

```
int ftp_size(int ftp_connection, string remotefile_path)
```

Fungsi ini mengembalikan ukuran file jarak jauh dalam byte, atau -1 jika terjadi kesalahan. Fungsi ini tidak didukung oleh semua server FTP.

Salah satu penggunaan `ftp_size()` yang praktis adalah untuk menentukan waktu eksekusi maksimum yang harus ditetapkan untuk transfer tertentu. Mengingat ukuran file dan kecepatan koneksi Anda, Anda dapat memperkirakan berapa lama transfer seharusnya berlangsung, dan menggunakan fungsi `set_time_limit()` yang sesuai.

Anda dapat memperoleh dan menampilkan daftar file dalam direktori di server FTP jarak jauh dengan kode berikut:

```
$listing = ftp_nlist($conn, “$directory_path”);
foreach ($listing as $filename)
  echo “$filename <br>”;
```

Kode ini menggunakan fungsi `ftp_nlist()` untuk mendapatkan daftar nama file dalam direktori tertentu. Mengenai fungsi FTP lainnya, hampir semua hal yang dapat Anda lakukan dari baris perintah FTP dapat Anda lakukan dengan fungsi FTP. Anda dapat menemukan fungsi spesifik yang sesuai dengan setiap perintah FTP dalam manual daring PHP di *http://php.net/manual/ref.ftp.php*. Pengecualian adalah mget (multiple get), tetapi Anda dapat menggunakan `ftp_nlist()` untuk mendapatkan daftar file dan kemudian mengambilnya sesuai kebutuhan.

**Komunikasi Jaringan Umum dengan Curl**

PHP (dari versi 4.0.2 dan seterusnya) memiliki serangkaian fungsi yang bertindak sebagai antarmuka ke cURL, fungsi pustaka URL Klien dari libcurl, yang ditulis oleh Daniel Stenberg. Sebelumnya dalam bab ini, kita melihat penggunaan fungsi `fopen()` dan fungsi pembacaan berkas untuk membaca dari berkas jarak jauh menggunakan HTTP. Ini adalah

batasan dari apa yang dapat Anda lakukan dengan `fopen()`. Kita juga telah melihat cara membuat koneksi FTP menggunakan fungsi FTP.

Fungsi cURL memungkinkan Anda membuat koneksi menggunakan FTP, HTTP, HTTPS, Gopher, Telnet, DICT, FILE, dan LDAP. Anda juga dapat menggunakan sertifikat untuk HTTPS, mengirim parameter HTTP POST dan HTTP GET, mengunggah berkas melalui unggahan FTP atau unggahan HTTP, bekerja melalui proksi, mengatur kuki, dan melakukan autentikasi pengguna HTTP sederhana.

Dengan kata lain, hampir semua jenis koneksi jaringan yang ingin Anda buat dapat dilakukan menggunakan cURL. Untuk menggunakan cURL dengan PHP, Anda perlu mengunduh libcurl, mengompilasinya, dan menjalankan skrip konfigurasi PHP dengan opsi `--with-curl=[path]`. Direktori di path harus berisi direktori lib dan include di sistem Anda. Anda dapat mengunduh pustaka dari *http://curl.haxx.se/*.

Ketahui bahwa Anda memerlukan versi cURL mulai dari 7.0.2-beta dan seterusnya untuk bekerja dengan PHP. Hanya ada beberapa fungsi sederhana yang harus dikuasai untuk menggunakan kekuatan cURL. Prosedur umum untuk menggunakannya adalah

1. Siapkan sesi cURL dengan panggilan ke fungsi `curl_init()`.
2. Tetapkan parameter apa pun untuk transfer dengan panggilan ke fungsi `curl_setopt()`. Di sinilah Anda menetapkan opsi seperti URL untuk terhubung, parameter apa pun untuk dikirim ke URL tersebut, atau tujuan output dari URL.
3. Ketika semuanya sudah disiapkan, panggil `curl_exec()` untuk benar-benar membuat koneksi.
4. Tutup sesi cURL dengan memanggil `curl_close()`.

Satu-satunya hal yang berubah dengan aplikasi adalah URL yang Anda hubungkan dan parameter yang Anda tetapkan dengan `curl_opt()`. Ada banyak parameter yang dapat ditetapkan. Beberapa aplikasi cURL yang umum adalah;

- Mengunduh halaman dari server yang menggunakan HTTPS (karena `fopen()` tidak dapat digunakan untuk tujuan ini)
- Menghubungkan ke skrip yang biasanya mengharapkan data dari formulir HTML menggunakan POST
- Menulis skrip untuk mengirim beberapa set data uji ke skrip Anda dan memeriksa hasilnya

Kita akan mempertimbangkan contoh pertama—ini adalah aplikasi sederhana yang tidak dapat dilakukan dengan cara lain. Contoh ini, yang ditunjukkan dalam Daftar 8.5, akan terhubung ke Equifax Secure Server melalui HTTPS, dan menulis file yang ditemukannya di sana ke file di server Web kita.

**Daftar 8.5 Https-curl.php—skrip untuk membuat koneksi https**

```
<?
echo “<h1>HTTPS transfer with cURL</h1>”;
$outputfile = “$DOCUMENT_ROOT/../writable/equifax.html”;
$fp = fopen($outputfile, “w”);
echo “Initializing cURL session...<br>”;
```

```
$ch = curl_init();
echo “Setting cURL options...<br>”;
curl_setopt ($ch, CURLOPT_URL, “https://equifaxsecure.com”);
curl_setopt ($ch, CURLOPT_FILE, $fp);
echo “Executing cURL session...<br>”;
curl_exec ($ch);
echo “Ending cURL session...<br>”;
curl_close ($ch);
fclose($fp);
?>
```

Mari kita bahas skrip ini. Kita mulai dengan membuka file lokal menggunakan `fopen()`. Di sinilah kita akan menyimpan halaman yang kita transfer dari koneksi aman. Setelah selesai, kita perlu membuat sesi cURL menggunakan fungsi `curl_init()`:

```
$ch = curl_init();
```

Fungsi ini mengembalikan handle untuk sesi cURL. Anda dapat memanggilnya seperti ini, tanpa parameter, atau secara opsional Anda dapat meneruskannya dengan string yang berisi URL untuk terhubung. Anda juga dapat mengatur URL menggunakan fungsi `curl_setopt()`, yang telah kita lakukan dalam kasus ini:

```
curl_setopt ($ch, CURLOPT_URL, “https://equifaxsecure.com”);
curl_setopt ($ch, CURLOPT_FILE, $fp);
```

Fungsi `curl_setopt()` menggunakan tiga parameter. Yang pertama adalah handle sesi, yang kedua adalah nama parameter yang akan ditetapkan, dan yang ketiga adalah nilai yang ingin Anda tetapkan untuk parameter tersebut.

Dalam kasus ini, kami menetapkan dua opsi. Yang pertama adalah URL yang ingin kami sambungkan. Ini adalah parameter CURLOPT_URL. Yang kedua adalah berkas tempat kami ingin data dari koneksi tersebut berada. Jika Anda tidak menentukan berkas, data dari koneksi tersebut akan masuk ke keluaran standar—biasanya peramban. Dalam kasus ini, kami telah menetapkan handle berkas dari berkas keluaran yang baru saja kami buka. Saat opsi ditetapkan, kami memberi tahu cURL untuk benar-benar membuat koneksi:

```
curl_exec ($ch);
```

Di sini, ini akan membuka koneksi ke URL yang telah kita tentukan, mengunduh halaman, dan menyimpannya dalam file yang ditunjuk oleh `$fp`. Setelah koneksi dibuat, kita perlu menutup sesi cURL, dan menutup file yang kita tulis:

```
curl_close ($ch);
fclose($fp);
```

# BAB 9
# MENGELOLA TANGGAL DAN WAKTU

Dalam bab ini, kita akan membahas pengecekan dan pemformatan tanggal dan waktu serta konversi antarformat tanggal. Hal ini khususnya penting saat mengonversi antara format tanggal MySQL dan PHP, format tanggal UNIX dan PHP, dan tanggal yang dimasukkan oleh pengguna dalam formulir HTML. Kita akan membahas;

- Mendapatkan tanggal dan waktu dalam PHP
- Mengonversi antara format tanggal PHP dan MySQL
- Menghitung tanggal
- Menggunakan fungsi kalender

## 9.1 MENDAPATKAN TANGGAL DAN WAKTU DARI PHP

Di Bab 1, “Pengetahuan Singkat PHP,” kita membahas penggunaan fungsi `date()` untuk mendapatkan dan memformat tanggal dan waktu dari PHP. Kita akan membahasnya dan beberapa fungsi tanggal dan waktu PHP lainnya secara lebih rinci sekarang.

**Menggunakan Fungsi `date()`**

Seperti yang mungkin Anda ingat, fungsi `date()` mengambil dua parameter, salah satunya opsional. Yang pertama adalah string format, dan yang kedua, opsional adalah cap waktu UNIX. Jika Anda tidak menentukan cap waktu, maka `date()` akan menggunakan tanggal dan waktu saat ini secara default. Fungsi ini akan mengembalikan string berformat yang mewakili tanggal yang sesuai. Pemanggilan fungsi date yang umum adalah;

```
echo date(“jS F Y”);
```

Ini akan menghasilkan tanggal dengan format “27 Agustus 2000”. Kode format yang diterima oleh date tercantum dalam Tabel 9.1.

**Tabel 9.1 Kode format untuk fungsi `date()` PHP**

| *Kode* | *Keterangan* |
|---|---|
| a | Pagi atau sore, direpresentasikan sebagai dua karakter huruf kecil, "am" atau "pm". |
| A | Pagi atau sore, direpresentasikan sebagai dua karakter huruf besar, "AM" atau "PM". |
| B | Swatch Internet time, skema waktu universal. |
| d | Hari dalam bulan sebagai angka 2 digit dengan angka nol di depan. Rentangnya dari "01" hingga "31". |
| D | Hari dalam seminggu dalam format teks singkat 3 karakter. Rentangnya dari "Senin" hingga "Minggu". |

| | |
|---|---|
| F | Bulan dalam setahun dalam format teks lengkap. Rentangnya dari "Januari" hingga "Desember". |
| g | Jam dalam format 12 jam tanpa angka nol di depan. Rentangnya dari "1" hingga "12". |
| G | Jam dalam format 24 jam tanpa angka nol di depan. Rentangnya dari "0" hingga "23". |
| h | Jam dalam format 12 jam dengan angka nol di depan. Rentangnya dari "01" hingga "12". |
| H | Jam dalam format 24 jam dengan angka nol di depan. Rentangnya dari "00" hingga "23". |
| i | Menit setelah jam dengan angka nol di depan. Rentangnya dari "00" hingga "59". |
| I | Waktu musim panas, direpresentasikan sebagai nilai Boolean. Nilai ini akan mengembalikan "1" jika tanggalnya dalam waktu musim panas dan "O" jika tidak. |
| j | Hari dalam bulan sebagai angka tanpa angka nol di depan. Rentangnya dari "1" hingga "31". |
| l | Hari dalam seminggu dalam format teks lengkap. Rentangnya dari "Senin" hingga "Minggu". |
| L | Tahun kabisat, direpresentasikan sebagai nilai Boolean. Nilai ini akan mengembalikan "1" jika tanggal tersebut berada di tahun kabisat dan "O" jika tidak berada di tahun kabisat. |
| m | Bulan dalam setahun sebagai angka 2 digit dengan angka nol di depan. Rentangnya dari "01" hingga "12". |
| M | Bulan dalam setahun dalam format teks singkat 3 karakter. Rentangnya dari "Jan" hingga "Des". |
| N | Bulan dalam setahun sebagai angka tanpa angka nol di depan. Rentangnya dari "1" hingga "12". |
| s | Detik setelah menit dengan angka nol di depan. Rentangnya dari "00" hingga "59". |
| S | Akhiran ordinal untuk tanggal dalam format 2 karakter. Bisa berupa "st", "nd", "rd", atau "th", tergantung angka setelahnya. |
| t | Jumlah hari total dalam bulan tanggal tersebut. Rentangnya dari "28" hingga "31". |
| T | Pengaturan zona waktu server dalam format 3 karakter, misalnya, "EST". |
| U | Jumlah total detik dari 1 Januari 1970 hingga saat ini; alias cap waktu UNIX untuk tanggal ini. |
| w | Hari dalam seminggu sebagai satu digit. Rentangnya dari "0" (Minggu) hingga "6" (Sabtu). |
| y | Tahun dalam format 2 digit, misalnya "00". |
| Y | Tahun dalam format 4 digit, misalnya, "2000". |

| | |
|---|---|
| z | Hari dalam setahun sebagai angka. Rentangnya adalah "0" hingga "365". |
| Z | Offset untuk zona waktu saat ini dalam hitungan detik. Rentangnya adalah "-43200" hingga "43200". |

## 9.2 BERURUSAN DENGAN CAP WAKTU UNIX

Parameter kedua pada fungsi `date()` adalah cap waktu UNIX. Jika Anda bertanya-tanya apa sebenarnya maksudnya, sebagian besar sistem UNIX menyimpan waktu dan tanggal saat ini sebagai bilangan bulat 32-bit yang berisi jumlah detik sejak tengah malam, 1 Januari 1970, GMT, yang juga dikenal sebagai Epoch UNIX. Ini mungkin tampak agak esoteris jika Anda tidak mengenalnya, tetapi ini adalah standar.

Cap waktu UNIX adalah cara ringkas untuk menyimpan tanggal dan waktu, tetapi perlu dicatat bahwa cap waktu ini tidak mengalami masalah tahun 2000 (Y2K) yang memengaruhi beberapa format tanggal ringkas atau disingkat lainnya. Namun, jika perangkat lunak Anda masih digunakan pada tahun 2038, akan ada masalah serupa. Karena cap waktu tidak memiliki ukuran tetap, tetapi terikat pada ukuran C long, yang setidaknya 32 bit, solusi yang paling mungkin adalah bahwa pada tahun 2038, kompiler Anda akan menggunakan tipe yang lebih besar.

Bahkan jika Anda menjalankan PHP di server Windows, ini masih merupakan format yang digunakan oleh `date()` dan sejumlah fungsi PHP lainnya. Jika Anda ingin mengonversi tanggal dan waktu ke stempel waktu UNIX, Anda dapat menggunakan fungsi `mktime()`. Fungsi ini memiliki prototipe berikut:

```
int mktime (int hour, int minute, int second, int month,
          int day, int year [, int is_dst])
```

Parameternya cukup jelas, kecuali yang terakhir, is_dst, yang menunjukkan apakah tanggal tersebut berada dalam daylight saving time atau tidak. Anda dapat menyetelnya ke 1 jika iya, 0 jika tidak, atau -1 (nilai default) jika Anda tidak tahu. Ini opsional jadi Anda akan jarang menggunakannya.

Perangkap utama yang harus dihindari dengan fungsi ini adalah bahwa parameternya berada dalam urutan yang cukup tidak intuitif. Urutan tersebut tidak memungkinkan untuk mengabaikan waktu. Jika Anda tidak khawatir tentang waktu, Anda dapat meneruskan 0 ke parameter jam, menit, dan detik. Namun, Anda dapat mengabaikan nilai dari sisi kanan daftar parameter. Jika Anda membiarkan parameter kosong, parameter tersebut akan disetel ke nilai saat ini. Oleh karena itu, panggilan seperti:

```
$timestamp = mktime();
```

akan mengembalikan cap waktu UNIX untuk tanggal dan waktu saat ini. Anda tentu saja juga dapat memperolehnya dengan memanggil `$timestamp = date("U");` Anda dapat memasukkan tahun 2 atau 4 digit ke `mktime()`. Nilai dua digit dari 0 hingga 69 akan ditafsirkan

sebagai tahun 2000 hingga 2069, dan nilai dari 70 hingga 99 akan ditafsirkan sebagai tahun 1970 hingga 1999.

**Menggunakan Fungsi `getdate()`**

Fungsi penentu tanggal lain yang mungkin berguna bagi Anda adalah fungsi `getdate()`. Fungsi ini memiliki prototipe berikut:

```
array getdate (int timestamp)
```

Fungsi ini mengambil stempel waktu sebagai parameter dan mengembalikan array asosiatif yang mewakili bagian-bagian tanggal dan waktu tersebut seperti yang ditunjukkan pada Tabel 9.2.

**Tabel 9.2 Pasangan kunci-nilai array asosiatif dari fungsi `getdate()`**

| Kunci | Nilai |
|---|---|
| second | Detik, numerik |
| minutes | Menit, numerik |
| hours | Jam, numerik |
| mday | Hari dalam bulan, numerik |
| wday | Hari dalam seminggu, numerik |
| mon | Bulan, numerik |
| year | Tahun, numerik |
| yday | Hari dalam setahun, numerik |
| weekday | Hari dalam seminggu, format teks lengkap |
| month | Bulan, format teks lengkap |

**Memvalidasi Tanggal**

Anda dapat menggunakan fungsi checkdate() untuk memeriksa apakah suatu tanggal valid. Fungsi ini khususnya berguna untuk memeriksa tanggal yang dimasukkan pengguna. Fungsi `checkdate()` memiliki prototipe berikut:

```
int checkdate (int month, int day, int year)
```

Fungsi ini akan memeriksa apakah tahun merupakan bilangan bulat valid antara 0 dan 32767, apakah bulan merupakan bilangan bulat antara 1 dan 12, dan apakah hari yang diberikan ada di bulan tersebut. Fungsi ini mempertimbangkan tahun kabisat.

Misalnya, checkdate (9, 18, 1972); akan mengembalikan true sementara checkdate (9, 31, 2000) tidak akan mengembalikan true.

**Mengonversi Antara Format Tanggal PHP dan MySQL**

Tanggal dan waktu di MySQL diambil dengan cara yang sedikit berbeda dari yang Anda harapkan. Waktu bekerja secara relatif normal, tetapi MySQL mengharapkan tanggal dimasukkan tahun terlebih dahulu. Misalnya, tanggal 29 Agustus 2000 dapat dimasukkan

sebagai 2000-08-29 atau 00-08-29. Tanggal yang diambil dari MySQL juga akan berada dalam urutan ini secara default.

Untuk berkomunikasi antara PHP dan MySQL, kita biasanya perlu melakukan konversi tanggal. Ini dapat dilakukan di kedua ujungnya. Saat memasukkan tanggal ke MySQL dari PHP, Anda dapat dengan mudah memasukkannya ke dalam format yang benar menggunakan fungsi `date()` seperti yang ditunjukkan sebelumnya. Satu peringatan kecil adalah Anda harus menggunakan versi hari dan bulan dengan nol di depan untuk menghindari kebingungan MySQL.

Jika Anda memilih untuk melakukan konversi di MySQL, dua fungsi yang berguna adalah `DATE_FORMAT()` dan `UNIX_TIMESTAMP()`. Fungsi `DATE_FORMAT()` bekerja mirip dengan fungsi PHP tetapi menggunakan kode format yang berbeda. Hal yang paling umum yang ingin kita lakukan adalah memformat tanggal dalam format MM-DD-YYYY daripada dalam format YYYY-MM-DD yang asli untuk MySQL. Anda dapat melakukan ini dengan menuliskan query Anda sebagai berikut:

```
SELECT DATE_FORMAT(date_column, '%m %d %Y')
FROM tablename;
```

**Tabel 9.3 Kode format untuk fungsi `date_format()` MySQL**

| ***Kode*** | ***Deskripsi*** |
|---|---|
| %M | Bulan, teks lengkap |
| %W | Nama hari kerja, teks lengkap |
| %D | Tanggal dalam bulan, numerik, dengan sufiks teks (misalnya, 1) |
| %Y | Tahun, numerik, 4 digit |
| %y | Tahun, numerik, 2 digit |
| %a | Nama hari dalam seminggu, 3 karakter |
| %d | Hari dalam bulan, numerik, nol di depan |
| %e | Hari dalam bulan, numerik, tanpa angka nol di depan |
| %m | Bulan, numerik, nol di depan |
| %C | Bulan, numerik, tanpa angka nol di depan |
| %b | Bulan, teks, 3 karakter |
| %j | Hari dalam setahun, numerik |
| %H | Jam, jam 24 jam, nol di depan |
| %k | Jam, jam 24 jam, tanpa angka nol di depan |
| %h or %I | Jam, jam 12 jam, angka nol di depan |
| %l | Jam, jam 12 jam, tanpa angka nol di depan |
| %i | Menit, numerik, nol di depan |
| %r | Waktu, 12 jam (hh:mm:ss [AM\|PM]) |
| %T | Waktu, 24 jam (hh:mm:ss) |
| %S or %s | Detik, angka, nol di depan |
| %p | pagi atau sore |
| %W | Hari dalam seminggu, numerik, dari 0 (Minggu) hingga 6 (Sabtu) |

Kode format %m mewakili bulan sebagai angka 2 digit; %d, hari sebagai angka 2 digit; dan %Y, tahun sebagai angka 4 digit. Ringkasan kode format MySQL yang lebih berguna untuk tujuan ini ditunjukkan pada Tabel 9.3.

Fungsi UNIX_TIMESTAMP bekerja dengan cara yang sama, tetapi mengubah kolom menjadi cap waktu UNIX. Misalnya,

```
SELECT UNIX_TIMESTAMP(date_column)
FROM tablename;
```

akan mengembalikan tanggal yang diformat sebagai stempel waktu UNIX. Anda kemudian dapat melakukannya sesuai keinginan Anda di PHP. Sebagai aturan praktis, gunakan stempel waktu UNIX untuk perhitungan tanggal dan format tanggal standar saat Anda hanya menyimpan atau menampilkan tanggal. Lebih mudah untuk melakukan perhitungan dan perbandingan tanggal dengan stempel waktu UNIX.

**Perhitungan Tanggal**

Cara paling sederhana untuk menghitung lamanya waktu antara dua tanggal di PHP adalah dengan menggunakan perbedaan antara stempel waktu UNIX. Kami telah menggunakan pendekatan ini dalam skrip yang ditunjukkan pada Daftar 9.1.

**Daftar 9.1 Calc_age.PHP—skrip menghitung usia seseorang berdasarkan tanggal lahirnya**

```
<?
  // set date for calculation
  $day = 18;
  $month = 9;
  $year = 1972;

  // remember you need bday as day month and year
  $bdayunix = mktime (“”, “”, “”, $month, $day, $year); // get unix ts for bday
  $nowunix = time(); // get unix ts for today
  $ageunix = $nowunix - $bdayunix; // work out the difference
  $age = floor($ageunix / (365 * 24 * 60 * 60)); // convert from seconds to
         //years

  echo “Age is $age”;
?>
```

Dalam skrip ini, kami telah menetapkan tanggal untuk menghitung usia. Dalam aplikasi nyata, kemungkinan besar informasi ini berasal dari formulir HTML.

Kami mulai dengan memanggil `mktime()` untuk menghitung cap waktu untuk ulang tahun dan waktu saat ini:

```
$bdayunix = mktime (“”, “”, “”, $month, $day, $year);
$nowunix = mktime(); // get unix ts for today
```

Sekarang tanggal-tanggal ini sudah dalam format yang sama, kita tinggal mengurangkannya:

```
$ageunix = $nowunix - $bdayunix;
```

Sekarang, bagian yang agak rumit—mengubah periode waktu ini kembali ke satuan ukuran yang lebih mudah dipahami manusia. Ini bukan cap waktu, melainkan usia orang yang diukur dalam detik. Kita dapat mengubahnya kembali ke tahun dengan membaginya dengan jumlah detik dalam setahun. Kita kemudian membulatkannya ke bawah menggunakan fungsi `floor()` karena seseorang dikatakan berusia, misalnya 20 tahun, hingga akhir tahun kedua puluhnya:

```
$age = floor($ageunix / (365 * 24 * 60 * 60)); // ubah dari detik ke tahun
```

Namun, perlu dicatat bahwa pendekatan ini agak cacat karena dibatasi oleh rentang cap waktu UNIX (umumnya bilangan bulat 32-bit).

## 9.3 MENGGUNAKAN FUNGSI KALENDER

PHP memiliki serangkaian fungsi yang memungkinkan Anda mengonversi antara sistem kalender yang berbeda. Kalender utama yang akan Anda gunakan adalah Gregorian, Julian, dan Hitungan Hari Julian. Kalender Gregorian adalah kalender yang saat ini digunakan oleh sebagian besar negara Barat. Tanggal Gregorian 15 Oktober 1582 setara dengan tanggal 5 Oktober 1582 dalam kalender Julian. Sebelum tanggal tersebut, kalender Julian umum digunakan. Berbagai negara beralih ke kalender Gregorian pada waktu yang berbeda, dan beberapa negara baru melakukannya pada awal abad ke-20.

Meskipun Anda mungkin pernah mendengar kedua kalender ini, Anda mungkin belum pernah mendengar tentang Julian Day Count. Kalender ini mirip dalam banyak hal dengan cap waktu UNIX. Kalender ini menghitung jumlah hari sejak tanggal sekitar 4000 SM. Kalender ini sendiri tidak terlalu berguna, tetapi berguna untuk mengonversi antarformat. Untuk mengonversi dari satu format ke format lain, pertama-tama Anda mengonversi ke Julian Day Count (JD) dan kemudian ke kalender keluaran yang diinginkan.

Untuk menggunakan fungsi-fungsi ini, Anda perlu mengompilasi ekstensi kalender ke dalam PHP. Untuk memberi Anda gambaran tentang fungsi-fungsi ini, pertimbangkan prototipe untuk fungsi-fungsi yang akan Anda gunakan untuk mengonversi dari kalender Gregorian ke kalender Julian:

```
int gregoriantojd (int month, int day, int year) string jdtojulian(int julianday)
```

Untuk mengonversi tanggal, kita perlu memanggil kedua fungsi ini:

```
$jd = gregoriantojd (9, 18, 1582);
echo jdtojulian($jd);
```

Ini menggemakan tanggal Julian dalam format mm/dd/yyyy. Variasi fungsi ini tersedia untuk mengonversi antara kalender Gregorian, Julian, Prancis, dan Yahudi serta stempel waktu UNIX.

# BAB 10
# MEMBUAT GAMBAR

Salah satu hal bermanfaat yang dapat Anda lakukan dengan PHP adalah membuat gambar dengan cepat. PHP memiliki beberapa fungsi informasi gambar bawaan, dan Anda juga dapat menggunakan pustaka GD untuk membuat gambar baru atau memanipulasi gambar yang sudah ada. Bab ini membahas cara menggunakan fungsi gambar untuk memperoleh beberapa efek yang menarik dan bermanfaat.

Bahasan dalam bab ini akan meliputi:

- ☑ Menyiapkan dukungan gambar di PHP
- ☑ Memahami format gambar
- ☑ Membuat gambar
- ☑ Menggunakan teks dan font untuk membuat gambar
- ☑ Menggambar gambar dan data grafik

Secara khusus, kita akan membahas dua contoh: membuat tombol situs Web dengan cepat, dan menggambar diagram batang menggunakan gambar dari basis data MySQL.

## 10.1 MENYIAPKAN DUKUNGAN GAMBAR DI PHP

Dukungan gambar di PHP tersedia melalui pustaka gd, yang tersedia di http://www.boutell.com/gd/. Versi 1.6.2 disertakan dengan PHP 4. Secara default, format PNG didukung. Jika Anda juga ingin bekerja dengan JPEG, Anda perlu mengunduh jpeg-6b, dan mengompilasi ulang gd dengan dukungan jpeg yang disertakan. Anda dapat mengunduhnya dari ftp://ftp.uu.net/graphics/jpeg/ Anda kemudian perlu mengonfigurasi ulang PHP dengan opsi

```
--with-jpeg-dir=/path/to/jpeg-6b
```

dan mengompilasi ulang. Jika Anda ingin menggunakan font TrueType dalam gambar Anda, Anda juga memerlukan pustaka FreeType. Ini juga disertakan dengan PHP 4. Atau, Anda dapat mengunduhnya dari http://www.freetype.org/ Jika Anda ingin menggunakan font PostScript Tipe 1, Anda perlu mengunduh t1lib, tersedia dari ftp://ftp.neuroinformatik.ruhr-uni-bochum.de/pub/software/t1lib/ Anda kemudian perlu menjalankan program konfigurasi PHP dengan

```
--with-t1lib[=path/to/t1lib]
```

### Format Gambar

Perpustakaan GD mendukung format JPEG, PNG, dan WBMP. Tidak lagi mendukung format GIF. Mari kita bahas masing-masing format ini secara singkat.

**JPEG**

JPEG (diucapkan "jay-peg") sebenarnya adalah singkatan dari Joint Photographic Experts Group dan merupakan nama badan standar. Format berkas yang kita maksud ketika kita merujuk ke JPEG sebenarnya disebut JFIF, yang sesuai dengan salah satu standar yang dikeluarkan oleh JPEG. Jika Anda tidak mengenalnya, JPEG biasanya digunakan untuk menyimpan gambar fotografi atau gambar lain dengan banyak warna atau gradasi warna. Format ini menggunakan kompresi lossy, yaitu, untuk memadatkan foto ke dalam berkas yang lebih kecil, sebagian kualitas gambar akan hilang. Karena JPEG seharusnya berisi gambar analog, dengan gradasi warna, mata manusia dapat menoleransi sebagian penurunan kualitas. Format ini tidak cocok untuk gambar garis, teks, atau blok warna solid.

Format berkas ini dianggap sebagai pengganti GIF (*Graphics Interchange Format*) karena alasan yang akan kita bahas sebentar lagi. Situs Web PNG menggambarkannya sebagai "format gambar *turbo-studly* dengan kompresi *lossless*". Karena lossless, format gambar ini cocok untuk gambar yang berisi teks, garis lurus, dan blok warna sederhana seperti judul dan tombol situs Web semua tujuan yang sama dengan yang sebelumnya Anda gunakan untuk GIF. PNG menawarkan kompresi yang lebih baik daripada GIF serta transparansi variabel, koreksi gamma, dan interlacing dua dimensi. Namun, format ini tidak mendukung animasi untuk ini Anda harus menggunakan format ekstensi MNG, yang masih dalam pengembangan.

**WBMP**

WBMP adalah singkatan dari *Wireless Bitmap*. Ini adalah format berkas yang dirancang khusus untuk perangkat nirkabel. Meskipun gd mendukung format ini, saat ini tidak ada fungsi PHP yang memanfaatkan fungsi ini.

**GIF**

GIF adalah singkatan dari *Graphics Interchange Format*. Ini adalah format terkompresi tanpa kehilangan yang banyak digunakan di Web untuk menyimpan gambar yang berisi teks, garis lurus, dan blok warna tunggal. Pertanyaan yang mungkin Anda tanyakan adalah, mengapa gd tidak mendukung GIF? Jawabannya adalah dulunya mendukung, hingga versi 1.3.

Namun, perlu dicatat bahwa pembuat gd melarang Anda menggunakan versi ini dan tidak lagi mendukungnya. Salinan versi GIF ini mungkin tidak akan tersedia selamanya. Ada alasan bagus mengapa gd tidak lagi mendukung GIF. GIF standar menggunakan bentuk kompresi yang dikenal sebagai LZW (*Lempel Ziv Welch*), yang tunduk pada paten yang dimiliki oleh UNISYS. Penyedia program yang membaca dan menulis GIF harus membayar biaya lisensi kepada UNISYS. Misalnya, Adobe telah membayar biaya lisensi untuk produk seperti Photoshop yang digunakan untuk membuat GIF. Pustaka kode tampaknya berada dalam situasi di mana penulis pustaka kode harus membayar biaya, dan, sebagai tambahan, pengguna pustaka juga harus membayar biaya. Jadi, jika Anda menggunakan versi GIF dari pustaka GD di situs Web Anda, Anda mungkin berutang biaya lisensi yang cukup besar kepada UNISYS.

Situasi ini sangat disayangkan karena GIF telah digunakan selama bertahun-tahun sebelum UNISYS memutuskan untuk memberlakukan lisensi. Dengan demikian, format tersebut menjadi salah satu standar untuk Web. Banyak perasaan tidak enak yang muncul tentang paten tersebut di komunitas pengembangan Web.

## 10.2 MEMBUAT GAMBAR

Empat langkah dasar untuk membuat gambar dalam PHP adalah sebagai berikut:

1. Membuat gambar kanvas untuk dikerjakan
2. Menggambar bentuk atau mencetak teks pada kanvas tersebut
3. Mengeluarkan grafik akhir
4. Membersihkan sumber daya

Kita akan mulai dengan melihat skrip pembuatan gambar yang sangat sederhana. Skrip ini ditunjukkan pada Daftar 10.1.

**Daftar 10.1 Simplegraph.Php Mengeluarkan Grafik Garis Sederhana Dengan Label Penjualan**

```
<?
// set up image
   $height = 200;
   $width = 200;
   $im = ImageCreate($width, $height);
   $white = ImageColorAllocate ($im, 255, 255, 255);
   $black = ImageColorAllocate ($im, 0, 0, 0);

// draw on image
   ImageFill($im, 0, 0, $black);
   ImageLine($im, 0, 0, $width, $height, $white);
   ImageString($im, 4, 50, 150, “Sales”, $white);

// output image
   Header (“Content-type: image/png”);
   ImagePng ($im);

// clean up
   ImageDestroy($im);
?>
```

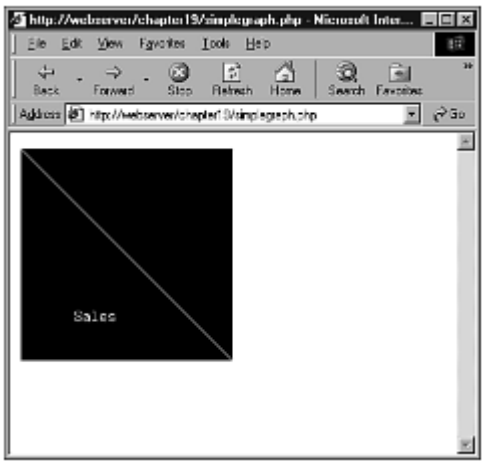
Output dari menjalankan skrip ini ditunjukkan pada Gambar 10.1. Kita akan membahas langkah-langkah pembuatan gambar ini satu per satu.

**Membuat Gambar Kanvas**

Untuk mulai membuat atau mengubah gambar di PHP, Anda perlu membuat pengenal gambar. Ada dua cara dasar untuk melakukannya. Salah satunya adalah membuat kanvas kosong, yang dapat Anda lakukan dengan memanggil fungsi `ImageCreate()`, seperti yang telah kita lakukan dalam skrip ini dengan yang berikut:

```
$im = ImageCreate($width, $height);
```

Anda perlu meneruskan dua parameter ke `ImageCreate()`. Yang pertama adalah lebar gambar baru, dan yang kedua adalah tinggi gambar baru. Fungsi tersebut akan mengembalikan pengenal untuk gambar baru. (Ini bekerja seperti pegangan berkas.)




**Gambar 10.1 Skrip Tersebut Menggambar Latar Belakang Hitam, Lalu Menambahkan Garis Dan Label Teks Untuk Gambar Tersebut.**

Cara alternatif adalah dengan membaca berkas gambar yang sudah ada, yang kemudian dapat Anda filter, ubah ukurannya, atau tambahkan. Anda dapat melakukannya dengan salah satu fungsi `ImageCreateFromPNG()`, `ImageCreateFromJPEG()`, atau `ImageCreateFromGIF()`, tergantung pada format berkas yang Anda gunakan untuk membaca. Masing-masing fungsi ini menggunakan nama berkas sebagai parameter, seperti, misalnya,

```
$im = ImageCreateFromPNG("baseimage.png");
```

Contoh ditunjukkan kemudian dalam bab ini menggunakan gambar yang sudah ada untuk membuat tombol secara otomatis.

**Menggambar atau Mencetak Teks pada Gambar**

Sebenarnya ada dua tahap untuk menggambar atau mencetak teks pada gambar. Pertama, Anda harus memilih warna yang ingin Anda gambar. Seperti yang mungkin sudah Anda ketahui, warna yang akan ditampilkan pada monitor komputer terdiri dari jumlah cahaya merah, hijau, dan biru yang berbeda. Format gambar menggunakan palet warna yang terdiri dari subset tertentu dari semua kemungkinan kombinasi dari tiga warna. Untuk menggunakan warna untuk menggambar dalam gambar, Anda perlu menambahkan warna ini ke palet gambar. Anda harus melakukan ini untuk setiap warna yang ingin Anda gunakan, bahkan hitam dan putih. Anda dapat memilih warna untuk gambar Anda dengan memanggil fungsi `ImageColorAllocate()`. Anda perlu meneruskan pengenal gambar dan nilai merah, hijau, dan biru (RGB) dari warna yang ingin Anda gambar ke dalam fungsi.

Dalam Daftar 10.1, kami menggunakan dua warna: hitam dan putih. Kami mengalokasikannya dengan memanggil

```
$white = ImageColorAllocate ($im, 255, 255, 255);
$black = ImageColorAllocate ($im, 0, 0, 0);
```

Fungsi ini mengembalikan pengenal warna yang dapat kita gunakan untuk mengakses warna tersebut nanti. Kedua, untuk benar-benar menggambar ke dalam gambar, sejumlah fungsi berbeda tersedia, bergantung pada apa yang ingin Anda gambar garis, busur, poligon, atau teks. Fungsi menggambar umumnya memerlukan yang berikut sebagai parameter:

- ✓ Pengenal gambar
- ✓ Koordinat awal dan terkadang akhir dari apa yang ingin Anda gambar
- ✓ Warna yang ingin Anda gambar
- ✓ Untuk teks, informasi fon

Dalam kasus ini, kami menggunakan tiga fungsi menggambar. Mari kita bahas masing-masing secara bergantian. Pertama, kami melukis latar belakang hitam untuk menggambar menggunakan fungsi `ImageFill(): ImageFill($im, 0, 0, $black);`

Fungsi ini mengambil pengenal gambar, koordinat awal area yang akan dicat (x dan y), dan warna yang akan diisi sebagai parameter. Selanjutnya, kita menggambar garis dari pojok kiri atas (0, 0) ke pojok kanan bawah (`$width, $height`) gambar:

```
ImageLine($im, 0, 0, $width, $height, $white);
```

Fungsi ini mengambil pengenal gambar, titik awal x dan y untuk garis, titik akhir, lalu warna, sebagai parameter.

Terakhir, kita menambahkan label ke grafik:

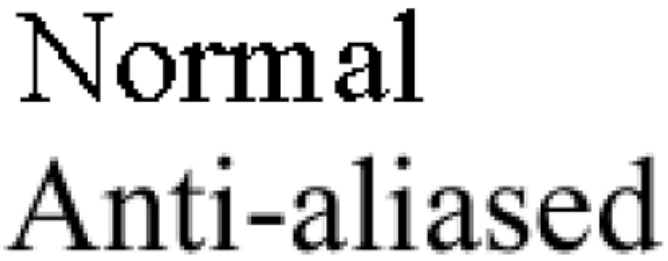
```
ImageString($im, 4, 50, 150, “Sales”, $white);
```

Fungsi `ImageString()` mengambil beberapa parameter yang sedikit berbeda. Prototipe untuk fungsi ini adalah

```
int imagestring (int im, int font, int x, int y, string s, int col)
```

Parameter yang digunakan adalah pengenal gambar, fon, koordinat x dan y untuk mulai menulis teks, teks yang akan ditulis, dan warna. Fon adalah angka antara 1 dan 5. Ini mewakili sekumpulan fon bawaan. Sebagai alternatif, Anda dapat menggunakan fon TrueType, atau fon `PostScript` Tipe 1. Setiap set fon ini memiliki set fungsi yang sesuai. Kita akan menggunakan fungsi `TrueType` pada contoh berikutnya.

Alasan yang tepat untuk menggunakan salah satu set fungsi fon alternatif adalah bahwa teks yang ditulis oleh `ImageString()` dan fungsi terkait, seperti `ImageChar()` (menulis karakter ke gambar) diberi alias. Fungsi `TrueType` dan `PostScript` menghasilkan teks anti-alias. Jika Anda tidak yakin apa perbedaannya, lihat Gambar 10.2. Jika kurva atau garis miring muncul pada huruf, teks alias tampak bergerigi. Kurva atau sudut dicapai dengan menggunakan efek "tangga". Pada gambar anti-alias, ketika terdapat lengkungan atau sudut pada teks, piksel dalam warna antara latar belakang dan warna teks digunakan untuk menghaluskan tampilan teks.




**Gambar 10.2 Teks Normal Tampak Tidak Rata, Terutama Pada Ukuran Huruf Besar. Anti-Aliasing Menghaluskan Lengkungan Dan Sudut Huruf.**

**Mengeluarkan Grafik Akhir**

Dalam contoh ini, kami telah mengeluarkan gambar ke browser. Ini adalah proses dua tahap. Pertama, kami perlu memberi tahu browser Web bahwa kami mengeluarkan gambar, bukan teks atau HTML. Kami melakukannya dengan menggunakan fungsi `Header()` untuk menentukan tipe MIME gambar:

```
Header (“Content-type: image/png”);
```

Biasanya saat Anda mengambil file di browser, tipe MIME adalah hal pertama yang dikirim server Web. Untuk halaman HTML atau PHP (setelah eksekusi), hal pertama yang dikirim adalah

```
Content-type: text/html
```

Ini memberi tahu browser cara menginterpretasikan data yang mengikutinya. Dalam kasus ini, kita ingin memberi tahu browser bahwa kita mengirim gambar, bukan keluaran HTML biasa. Kita dapat melakukannya menggunakan fungsi Header(), yang belum kita bahas.

Fungsi ini mengirim string header HTTP mentah. Aplikasi umum lainnya adalah melakukan pengalihan HTTP. Ini memberi tahu browser untuk memuat halaman yang berbeda, bukan yang diminta. Biasanya digunakan saat halaman telah dipindahkan. Misalnya,

```
Header (“Lokasi: http://www.domain.com/new_home_page.html “);
```

Hal penting yang perlu diperhatikan saat menggunakan fungsi `Header()` adalah fungsi ini tidak dapat dijalankan jika header HTTP telah dikirim untuk halaman tersebut. PHP akan mengirim header HTTP secara otomatis untuk Anda segera setelah Anda mengeluarkan apa pun ke browser. Oleh karena itu, jika Anda memiliki pernyataan gema, atau bahkan spasi sebelum tag PHP pembuka, header akan dikirim, dan Anda akan mendapatkan pesan peringatan dari PHP saat mencoba memanggil `Header()`. Namun, Anda dapat mengirim beberapa header HTTP dengan beberapa panggilan ke fungsi `Header()` dalam skrip yang

sama, meskipun semuanya harus muncul sebelum output apa pun dikirim ke browser. Setelah kami mengirim data header, kami mengeluarkan data gambar dengan panggilan ke

```
ImagePng ($im);
```

Ini mengirimkan output ke browser dalam format PNG. Jika Anda ingin mengirimkannya dalam format yang berbeda, Anda dapat memanggil `ImageJPEG()`—jika dukungan JPEG diaktifkan atau `ImageGIF()` jika Anda memiliki versi gd yang lebih lama. Anda juga perlu mengirim header yang sesuai terlebih dahulu; yaitu, baik

```
Header (“Content-type: image/jpeg”);
```

atau

```
Header (“Content-type: image/gif”);
```

Opsi kedua yang dapat Anda gunakan, sebagai alternatif untuk semua opsi sebelumnya, adalah menulis gambar ke dalam file, bukan ke browser. Anda dapat melakukannya dengan menambahkan parameter opsional kedua ke ImagePNG() (atau fungsi serupa untuk format lain yang didukung):

```
ImagePNG($im, $filename);
```

Ingatlah bahwa semua aturan umum tentang penulisan ke file dari PHP berlaku (misalnya, memiliki izin yang ditetapkan dengan benar).

**Membersihkan**

Setelah selesai menggunakan gambar, Anda harus mengembalikan sumber daya yang telah Anda gunakan ke server dengan menghancurkan pengenal gambar. Anda dapat melakukannya dengan memanggil `ImageDestroy()`:

```
ImageDestroy($im);
```

**Menggunakan Gambar yang Dihasilkan Secara Otomatis di Halaman Lain**

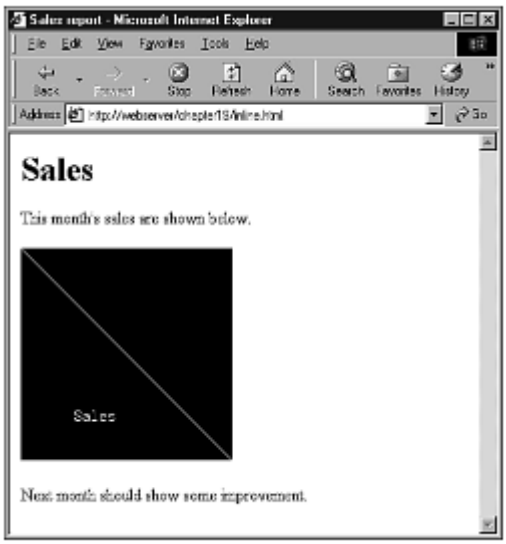
Karena header hanya dapat dikirim satu kali, dan ini adalah satu-satunya cara untuk memberi tahu browser bahwa kita sedang mengirim data gambar, agak sulit untuk menyematkan gambar yang kita buat secara otomatis di halaman biasa. Tiga cara yang dapat Anda lakukan adalah sebagai berikut:

1. Anda dapat membuat seluruh halaman terdiri dari keluaran gambar, seperti yang kita lakukan pada contoh sebelumnya.
2. Anda dapat menulis gambar ke file seperti yang disebutkan sebelumnya, lalu merujuknya dengan tag <IMG> normal.
3. Anda dapat meletakkan skrip produksi gambar di tag gambar.

Kita telah membahas metode 1 dan 2. Mari kita lihat sekilas metode 3. Untuk menggunakan metode ini, Anda menyertakan gambar sebaris dalam HTML dengan memiliki tag gambar seperti berikut:

```
<img src=”simplegraph.php” height=200 width=200 alt=”Sales going down”>
```

Daripada langsung memasukkan PNG, JPEG, atau GIF, masukkan skrip PHP yang menghasilkan gambar dalam tag SRC. Gambar ini akan diambil dan output akan ditambahkan sebaris, seperti yang ditunjukkan pada Gambar 10.3.




**Gambar 10.3** Gambar Sebaris Yang Diproduksi Secara Dinamis Tampak Sama Seperti Gambar Biasa Bagi Pengguna Akhir.

## 10.3 MENGGUNAKAN TEKS DAN FONT UNTUK MEMBUAT GAMBAR

Kita akan melihat contoh yang lebih rumit. Akan berguna untuk dapat membuat tombol atau gambar lain untuk situs Web Anda secara otomatis. Anda dapat membuat tombol sederhana berdasarkan persegi panjang warna latar belakang menggunakan teknik yang telah kita bahas.

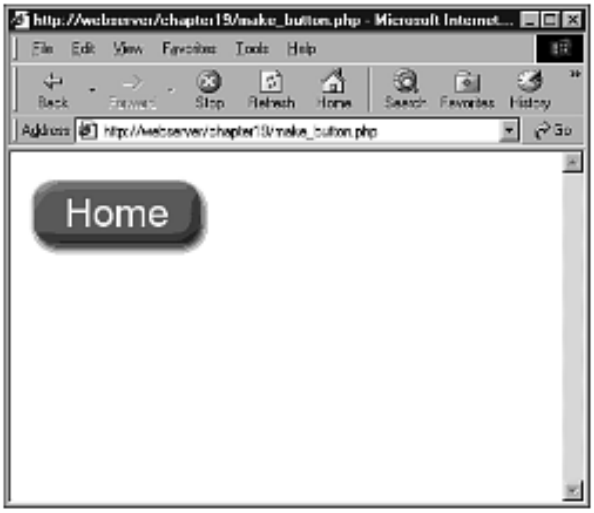
Namun, dalam contoh ini, kita akan membuat tombol menggunakan templat tombol kosong yang memungkinkan kita memiliki fitur seperti tepi miring dan sebagainya, yang jauh lebih mudah dibuat menggunakan Photoshop, GIMP, atau alat grafis lainnya. Dengan pustaka gambar di PHP, kita dapat memulai dengan gambar dasar dan menggambar di atasnya.

Kami juga akan menggunakan font TrueType sehingga kami dapat menggunakan teks anti-alias. Fungsi font TrueType memiliki kekhasannya sendiri, yang akan kami bahas. Proses dasarnya adalah mengambil beberapa teks dan membuat tombol dengan teks tersebut di atasnya. Teks akan dipusatkan secara horizontal dan vertikal pada tombol, dan akan ditampilkan dalam ukuran font terbesar yang sesuai dengan tombol. Kami telah membuat

antarmuka untuk generator tombol untuk pengujian dan eksperimen. Antarmuka ini ditunjukkan pada Gambar 10.4.

Anda dapat menggunakan jenis antarmuka ini untuk program guna membuat situs Web secara otomatis. Anda juga dapat memanggil skrip yang kami tulis dalam mode sebaris, untuk membuat semua tombol situs Web secara langsung! Keluaran khas dari skrip ditunjukkan pada Gambar 10.5.


**Gambar 10.4 Bagian Depan Memungkinkan Pengguna Memilih Warna Tombol Dan Mengetik Teks Yang Diinginkan.**




**Gambar 10.5 Tombol Yang Dibuat Oleh Skrip Make_Button.Php.**

Tombol dibuat oleh skrip yang disebut *make_button.php*. Skrip ini ditampilkan dalam Daftar 10.2.

**Daftar 10.2** *make_button.php*. Skrip Ini Dapat Dipanggil dari Formulir di *design_button.html* atau dari Dalam Tag Gambar HTML

```
<?
// check we have the appropriate variable data
// variables are button-text and color
if (empty($button_text) || empty($color))
{
  echo “Could not create image - form not filled out correctly”;
  exit;
}

// create an image of the right background and check size
$im = imagecreatefrompng (“$color-button.png”);

$width_image = ImageSX($im);
$height_image = ImageSY($im);

// Our images need an 18 pixel margin in from the edge image
$width_image_wo_margins = $width_image - (2 * 18);
$height_image_wo_margins = $height_image - (2 * 18);

// Work out if the font size will fit and make it smaller until it does
// Start out with the biggest size that will reasonably fit on our buttons
$font_size = 33;

do
{
  $font_size--;

  // find out the size of the text at that font size
  $bbox=imagettfbbox ($font_size, 0, “arial.ttf”, $button_text);

  $right_text = $bbox[2]; // right co-ordinate
  $left_text = $bbox[0]; // left co-ordinate
  $width_text = $right_text - $left_text; // how wide is it?
  $height_text = abs($bbox[7] - $bbox[1]); // how tall is it?

} while ( $font_size>8 &&
          ( $height_text>$height_image_wo_margins ||
            $width_text>$width_image_wo_margins )
          );

if ( $height_text>$height_image_wo_margins ||
     $width_text>$width_image_wo_margins )
{
  // no readable font size will fit on button
  echo “Text given will not fit on button.<BR>”;
```

```
}
else
{
  // We have found a font size that will fit
  // Now work out where to put it

  $text_x = $width_image/2.0 - $width_text/2.0;
  $text_y = $height_image/2.0 - $height_text/2.0 ;

  if ($left_text < 0)
    $text_x += abs($left_text); // add factor for left overhang

  $above_line_text = abs($bbox[7]); // how far above the baseline?
  $text_y += $above_line_text; // add baseline factor

  $text_y -= 2; // adjustment factor for shape of our template

  $white = ImageColorAllocate ($im, 255, 255, 255);
  ImageTTFText ($im, $font_size, 0, $text_x, $text_y, $white, “arial.ttf”,
        $button_text);

  Header (“Content-type: image/png”);
  ImagePng ($im);
}

ImageDestroy ($im);
?>
```

Ini adalah salah satu skrip terpanjang yang pernah kita lihat sejauh ini. Mari kita bahas bagian demi bagian. Kita mulai dengan beberapa pemeriksaan kesalahan dasar, lalu menyiapkan kanvas tempat kita akan bekerja.

**Menyiapkan Kanvas Dasar**

Dalam Daftar 10.2, daripada memulai dari awal, kita akan mulai dengan gambar yang sudah ada untuk tombol tersebut. Kita memiliki tiga pilihan warna pada tombol dasar: merah (`red-button.png`), `hijau` (`green-button.png`), dan biru (`blue-button.png`). Warna yang dipilih pengguna disimpan dalam variabel `$color` dari formulir. Kita mulai dengan menyiapkan pengenal gambar baru berdasarkan tombol yang sesuai:

```
$im = imagecreatefrompng (“$color-button.png”);
```

Fungsi `ImageCreateFromPNG()` mengambil nama file PNG sebagai parameter, dan mengembalikan pengenal gambar baru untuk gambar yang berisi salinan PNG tersebut. Perhatikan bahwa ini tidak mengubah PNG dasar dengan cara apa pun. Kita dapat menggunakan fungsi `ImageCreateFromJPEG()` dan `ImageCreateFromGIF()` dengan cara yang sama jika dukungan yang sesuai telah terpasang.

**Memasang Teks pada Tombol**

Kita memiliki beberapa teks yang diketik oleh pengguna yang disimpan dalam variabel `$button_text`. Yang ingin kita lakukan adalah mencetaknya pada tombol dengan ukuran font terbesar yang sesuai. Kita melakukan ini dengan iterasi, atau secara tegas, dengan uji coba berulang. Kita mulai dengan menyiapkan beberapa variabel yang relevan. Dua variabel pertama adalah tinggi dan lebar gambar tombol:

```
$width_image = ImageSX($im);
$height_image = ImageSY($im);
```

Dua yang kedua mewakili margin dari tepi tombol. Gambar tombol kita miring, jadi kita perlu menyisakan ruang untuk itu di sekitar tepi teks. Jika Anda menggunakan gambar yang berbeda, angka ini akan berbeda! Dalam kasus kita, margin di setiap sisi sekitar 18 piksel.

```
$width_image_wo_margins = $width_image - (2 * 18);
$height_image_wo_margins = $height_image - (2 * 18);
```

Kita juga perlu mengatur ukuran font awal. Kita mulai dengan 32 (sebenarnya 33, tetapi kita akan menguranginya sebentar lagi) karena ini adalah font terbesar yang akan muat di tombol:

```
$font_size = 33;
```

Sekarang kita mengulang, mengurangi ukuran font di setiap iterasi, hingga teks yang dikirimkan akan muat di tombol dengan wajar:

```
do
{
  $font_size--;

  // find out the size of the text at that font size
  $bbox=imagettfbbox ($font_size, 0, “arial.ttf”, $button_text);
  $right_text = $bbox[2]; // right co-ordinate
  $left_text = $bbox[0]; // left co-ordinate
  $width_text = $right_text - $left_text; // how wide is it?
  $height_text = abs($bbox[7] - $bbox[1]); // how tall is it?
  } while ( $font_size>8 &&

( $height_text>$height_image_wo_margins ||
$width_text>$width_image_wo_margins )

);
```

Kode ini menguji ukuran teks dengan melihat apa yang disebut kotak pembatas teks. Kita melakukannya menggunakan fungsi `ImageGetTTFBBox()`, yang merupakan salah satu fungsi

fon TrueType. Setelah mengetahui ukurannya, kita akan mencetak pada tombol menggunakan fon TrueType dan fungsi `ImageTTFText()`. Kotak pembatas teks adalah kotak terkecil yang dapat Anda gambar di sekeliling teks. Contoh kotak pembatas ditunjukkan pada Gambar 19.6.


**Gambar 110.6 Koordinat Kotak Pembatas Diberikan Relatif Terhadap Garis Dasar. Titik Asal Koordinat Ditunjukkan Di Sini Sebagai (0,0).**

Untuk mendapatkan dimensi kotak, kita memanggil `$bbox=imagettfbbox ($font_size, 0, "arial.ttf", $button_text)`; Panggilan ini mengatakan, "Untuk ukuran font yang diberikan `$font_size`, dengan teks miring pada sudut nol derajat, menggunakan font TrueType Arial, beri tahu saya dimensi teks dalam `$button_text`." Perlu dicatat bahwa Anda sebenarnya perlu meneruskan jalur ke file yang berisi font ke dalam fungsi. Dalam kasus ini, file tersebut berada di direktori yang sama dengan skrip (default), jadi kami belum menentukan jalur yang lebih panjang.

Fungsi mengembalikan array yang berisi koordinat sudut kotak pembatas. Isi array ditunjukkan pada Tabel 10.1.

**Tabel 10.1 Isi Array Kotak Pembatas**

| *Indeks Array* | *Konten* |
|---|---|
| 0 | Koordinat X, pojok kiri bawah |
| 1 | Koordinat Y, pojok kiri bawah |
| 2 | Koordinat X, pojok kanan bawah |
| 3 | Koordinat Y, pojok kanan bawah |
| 4 | Koordinat X, pojok kanan atas |
| 5 | Koordinat Y, pojok kanan atas |
| 6 | Koordinat X, sudut kiri atas |
| 7 | Koordinat Y, sudut kiri atas |

Untuk mengingat apa isi array tersebut, cukup ingat bahwa penomoran dimulai di sudut kiri bawah kotak pembatas dan berputar berlawanan arah jarum jam. Ada satu hal yang rumit tentang nilai yang dikembalikan dari fungsi `ImageTTFBBox()`. Nilai tersebut adalah nilai koordinat, yang ditetapkan dari titik asal. Namun, tidak seperti koordinat untuk gambar, yang ditetapkan relatif terhadap sudut kiri atas, nilai tersebut ditetapkan relatif terhadap garis dasar.

Lihat Gambar 10.6 lagi. Anda akan melihat bahwa kami telah menggambar garis di sepanjang bagian bawah sebagian besar teks. Ini dikenal sebagai garis dasar. Beberapa huruf menggantung di bawah garis dasar, seperti y dalam contoh ini. Ini disebut descender. Sisi kiri garis dasar ditetapkan sebagai titik asal pengukuran yaitu, koordinat X 0 dan koordinat Y 0.

Koordinat di atas garis dasar memiliki koordinat X positif dan koordinat di bawah garis dasar memiliki koordinat X negatif.

Selain itu, teks mungkin sebenarnya memiliki nilai koordinat yang berada di luar kotak pembatas. Misalnya, teks mungkin sebenarnya dimulai pada koordinat X -1. Semua ini menyimpulkan bahwa diperlukan kehati-hatian saat melakukan perhitungan dengan angka-angka ini. Kami menghitung lebar dan tinggi teks sebagai berikut:

```
$right_text = $bbox[2]; // right co-ordinate
$left_text = $bbox[0]; // left co-ordinate
$width_text = $right_text - $left_text; // how wide is it?
$height_text = abs($bbox[7] - $bbox[1]); // how tall is it?
```

Setelah kita memiliki ini, kita menguji kondisi loop:

```
} while ( $font_size>8 &&
          ( $height_text>$height_image_wo_margins ||
            $width_text>$width_image_wo_margins )
          );
```

Kami menguji dua set kondisi di sini. Yang pertama adalah font masih dapat dibaca tidak ada gunanya membuatnya jauh lebih kecil dari 8 poin karena tombolnya menjadi terlalu sulit dibaca. Set kondisi kedua menguji apakah teks akan muat di dalam ruang gambar yang kami miliki. Selanjutnya, kami memeriksa untuk melihat apakah perhitungan iteratif kami menemukan ukuran font yang dapat diterima atau tidak, dan melaporkan kesalahan jika tidak:

```
if ( $height_text>$height_image_wo_margins ||
     $width_text>$width_image_wo_margins )
{
  // no readable font size will fit on button
  echo “Text given will not fit on button.<BR>”;
}
```

### 10.4 MEMPOSISIKAN TEKS

Jika semuanya baik-baik saja, selanjutnya kita tentukan posisi dasar untuk awal teks. Ini adalah titik tengah ruang yang tersedia.

```
$text_x = $width_image/2.0 - $width_text/2.0;
$text_y = $height_image/2.0 - $height_text/2.0 ;
```

Karena adanya komplikasi dengan sistem koordinat relatif dasar, kita perlu menambahkan beberapa faktor koreksi:

```
if ($left_text < 0)
```

```
    $text_x += abs($left_text); // add factor for left overhang

$above_line_text = abs($bbox[7]); // how far above the baseline?
$text_y += $above_line_text; // add baseline factor

$text_y -= 2; // adjustment factor for shape of our template
```

Faktor koreksi ini memungkinkan garis dasar dan sedikit penyesuaian karena gambar kita agak "berat di bagian atas".

**Menuliskan Teks pada Tombol**

Setelah itu, semuanya berjalan lancar. Kita atur warna teks, yaitu putih:

```
$white = ImageColorAllocate ($im, 255, 255, 255);
```

Kita kemudian dapat menggunakan fungsi `ImageTTFText()` untuk benar-benar menggambar teks pada tombol:

```
ImageTTFText ($im, $font_size, 0, $text_x, $text_y, $white, "arial.ttf",
$button_text);
```

Fungsi ini membutuhkan banyak parameter. Secara berurutan, parameter tersebut adalah pengenal gambar, ukuran font dalam poin, sudut yang kita inginkan untuk menggambar teks, koordinat awal X dan Y teks, warna teks, berkas font, dan, terakhir, teks sebenarnya yang akan ditambahkan pada tombol.

**Menyelesaikan**

Akhirnya, kita dapat menampilkan tombol tersebut ke browser:

```
Header ("Content-type: image/png"); ImagePng ($im);
```

Kemudian saatnya untuk membersihkan sumber daya dan mengakhiri skrip:

```
ImageDestroy ($im);
```

Itu saja! Jika semuanya berjalan lancar, kita sekarang akan memiliki tombol di jendela browser yang tampak mirip dengan yang Anda lihat pada Gambar 19.5.

**Menggambar Gambar dan Membuat Grafik Data**

Pada aplikasi terakhir, kita melihat gambar dan teks yang ada. Kita belum melihat contoh dengan gambar, jadi kita akan melakukannya sekarang. Dalam contoh ini, kita akan menjalankan jajak pendapat di situs Web kita untuk menguji siapa yang akan dipilih pengguna dalam pemilihan fiktif. Kita akan menyimpan hasil jajak pendapat dalam basis data MySQL, dan menggambar diagram batang dari hasil tersebut menggunakan fungsi gambar. Fungsi-fungsi ini terutama digunakan untuk membuat grafik. Anda dapat membuat grafik data apa pun yang Anda inginkan penjualan, kunjungan ke situs web, atau apa pun yang Anda suka.

Untuk contoh ini, kami telah menghabiskan beberapa menit untuk menyiapkan basis data MySQL yang disebut poll. Basis data ini berisi satu tabel yang disebut `poll_results`, yang menyimpan nama kandidat di kolom kandidat, dan jumlah suara yang mereka terima di kolom `num_votes`. Kami juga telah membuat pengguna untuk basis data ini yang disebut poll, dengan kata sandi poll. Proses penyiapan ini memakan waktu sekitar lima menit, dan Anda dapat melakukannya dengan menjalankan skrip SQL yang ditunjukkan pada Daftar 10.3. Anda dapat melakukannya dengan menyalurkan skrip melalui login root menggunakan

```
mysql -u root -p < pollsetup.sql
```

Tentu saja, Anda juga dapat menggunakan login pengguna mana pun dengan hak istimewa MySQL yang sesuai.

**Daftar 10.3** pollsetup.sql Menyiapkan Basis Data Polling

```
create database poll;
use poll;
create table poll_results (
  candidate varchar(30),
  num_votes int
);
insert into poll_results values
  ('John Smith', 0),
  ('Mary Jones', 0),
  ('Fred Bloggs', 0)
;
grant all privileges
on poll.*
to poll@localhost
identified by 'poll';
```

Basis data ini berisi tiga kandidat. Kami menyediakan antarmuka pemungutan suara melalui halaman yang disebut vote.html. Kode untuk halaman ini ditampilkan dalam Daftar 10.4.

Daftar 10.4 Vote.Html—Pengguna Dapat Memberikan Suara Mereka Di Sini

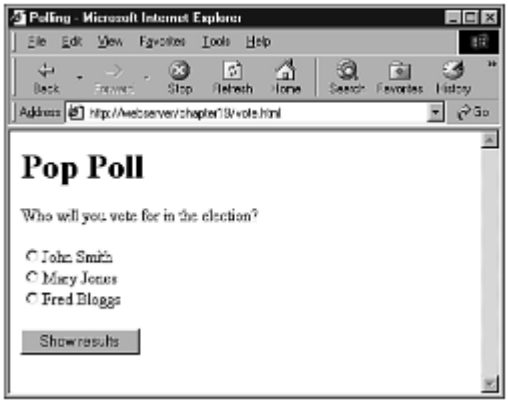
```
<html>
<head>
    <title>Polling</title>
<head>
<body>
<h1>Pop Poll</h1>
<p>Who will you vote for in the election?</p>
<form method=post action="show_poll.php">
<input type=radio name=vote value="John Smith">John Smith<br>
<input type=radio name=vote value="Mary Jones">Mary Jones<br>
```

```
<input type=radio name=vote value=”Fred Bloggs”>Fred Bloggs<br><br>
<input type=submit value=”Show results”>
</form>
</body>
```

Output dari halaman ini ditunjukkan pada Gambar 10.7.

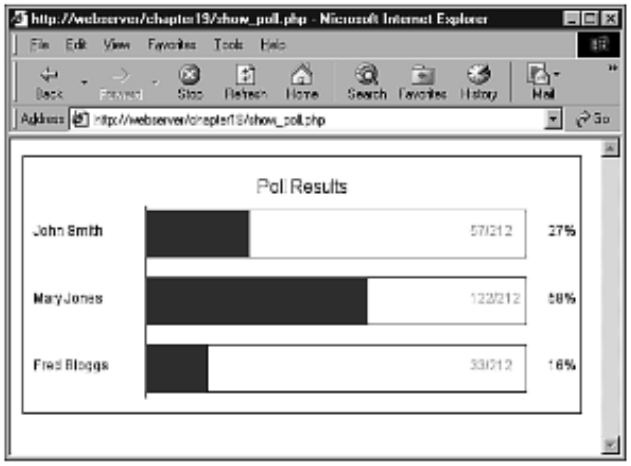
Ide umumnya adalah, saat pengguna mengklik tombol tersebut, kami akan menambahkan suara mereka ke basis data, mengeluarkan semua suara dari basis data, dan menggambar diagram batang dari hasil terkini. Output umum setelah beberapa suara diberikan ditunjukkan pada Gambar 10.8.

Skrip yang menghasilkan gambar ini cukup panjang. Kami telah membaginya menjadi empat bagian, dan kami akan membahas setiap bagian secara terpisah.




**Gambar 10.7 Pengguna Dapat Memberikan Suara Mereka Di Sini, Dan Mengklik Tombol Kirim Akan Menunjukkan Kepada Mereka Hasil Jajak Pendapat Terkini.**

Sebagian besar skrip tersebut familier; kami telah melihat banyak contoh MySQL yang mirip dengan ini. Kami telah melihat cara melukis kanvas latar belakang dengan warna solid, dan cara mencetak label teks di atasnya.




**Gambar 10.8 Hasil Pemungutan Suara Dibuat Dengan Menggambar Serangkaian Garis, Persegi Panjang, Dan Item Teks Ke Kanvas.**

Bagian baru skrip ini terkait dengan menggambar garis dan persegi panjang. Kami akan memfokuskan perhatian pada bagian-bagian ini. Bagian 1 (dari skrip empat bagian ini) ditampilkan dalam Daftar 10.5.1.

**Daftar 10.5.1 Showpoll.Php Bagian 1 Memperbarui Basis Data Pemungutan Suara Dan Mengambil Hasil Baru**

```
<?
/*****************************************
Database query to get poll info
*****************************************/
// log in to database
if (!$db_conn = @mysql_connect("localhost", "poll", "poll"))
{
  echo "Could not connect to db<br>";
  exit;
};
@mysql_select_db("poll");

if (!empty($vote)) // if they filled the form out, add their vote
{
  $vote = addslashes($vote);
  $query = "update poll_results
       set num_votes = num_votes + 1
       where candidate = '$vote'";
  if(!($result = @mysql_query($query, $db_conn)))
  {
    echo "Could not connect to db<br>";
    exit;
  }
};

// get current results of poll, regardless of whether they voted
$query = "select * from poll_results";
if(!($result = @mysql_query($query, $db_conn)))
{
  echo "Could not connect to db<br>";
  exit;
}
$num_candidates = mysql_num_rows($result);

// calculate total number of votes so far
$total_votes=0;
while ($row = mysql_fetch_object ($result))
{
  $total_votes += $row->num_votes;
```

```
}
mysql_data_seek($result, 0); // reset result pointer
```

Bagian 1, yang ditunjukkan pada Daftar 10.5.1, terhubung ke basis data MySQL, memperbarui suara sesuai dengan yang diketik pengguna, dan mendapatkan suara baru. Setelah kita memiliki informasi tersebut, kita dapat mulai membuat kalkulasi untuk menggambar grafik. Bagian 2 ditunjukkan pada Daftar 10.5.2.

**Daftar 10.5.2 Showpoll.Php Bagian 2 Menyiapkan Semua Variabel Untuk Menggambar**

```
/****************************************
  Initial calculations for graph
****************************************/
// set up constants
$width=500;          // width of image in pixels - this will fit in 640x480
$left_margin = 50;   // space to leave on left of image
$right_margin= 50;   // ditto right
$bar_height = 40;
$bar_spacing = $bar_height/2;
$font = “arial.ttf”;
$title_size= 16; // point
$main_size= 12; // point
$small_size= 12; // point
$text_indent = 10; // position for text labels on left

// set up initial point to draw from
$x = $left_margin + 60; // place to draw baseline of the graph
$y = 50; // ditto
$bar_unit = ($width-($x+$right_margin)) / 100; // one “point” on the graph

// calculate height of graph - bars plus gaps plus some margin
$height = $num_candidates * ($bar_height + $bar_spacing) + 50;
```

Bagian 2 menyiapkan beberapa variabel yang akan kita gunakan untuk menggambar grafik.

Menentukan nilai untuk variabel semacam ini bisa jadi membosankan, tetapi sedikit pemikiran sebelumnya tentang seperti apa tampilan akhir gambar yang Anda inginkan akan membuat proses menggambar jauh lebih mudah. Nilai yang kita gunakan di sini diperoleh dengan membuat sketsa efek yang diinginkan pada selembar kertas dan memperkirakan proporsi yang dibutuhkan. Variabel `$width` adalah lebar total kanvas yang akan kita gunakan. Kita juga mengatur margin kiri dan kanan (masing-masing dengan `$left_margin` dan `$right_margin`); "ketebalan" dan jarak antar batang (`$bar_height` dan `$bar_spacing`); serta font, ukuran font, dan posisi label (`$font, $title_size, $main_size, $small_size,` dan `$text_indent`).

Dengan nilai dasar ini, kita kemudian dapat membuat beberapa perhitungan. Kita ingin menggambar garis dasar tempat semua batang membentang. Kita dapat menentukan posisi

untuk garis dasar ini dengan menggunakan margin kiri ditambah kelonggaran untuk label teks untuk koordinat X, dan sekali lagi perkiraan dari sketsa kita untuk koordinat Y. Kita juga menentukan dua nilai penting: pertama, jarak pada grafik yang mewakili satu unit:

```
$bar_unit = ($width-($x+$right_margin)) / 100; // one “point” on the graph
```

Ini adalah panjang maksimum batang dari garis dasar ke margin kanan dibagi dengan 100 karena grafik kita akan menunjukkan nilai persentase.
Nilai kedua adalah tinggi total yang kita perlukan untuk kanvas:

```
$height = $num_candidates * ($bar_height + $bar_spacing) + 50;
```

Ini pada dasarnya adalah tinggi per batang dikalikan jumlah batang, ditambah jumlah tambahan untuk judul. Bagian 3 ditunjukkan dalam Daftar 10.5.3.

Daftar 10.5.3 Showpoll.Php Bagian 3 Menyiapkan Grafik, Siap Untuk Menambahkan Data

```
/*****************************************
Set up base image
*****************************************/
// create a blank canvas
$im = imagecreate($width,$height);

// Allocate colors
$white=ImageColorAllocate($im,255,255,255);
$blue=ImageColorAllocate($im,0,64,128);
$black=ImageColorAllocate($im,0,0,0);
$pink = ImageColorAllocate($im,255,78,243);
$text_color = $black;
$percent_color = $black;
$bg_color = $white;
$line_color = $black;
$bar_color = $blue;
$number_color = $pink;

// Create “canvas” to draw on
ImageFilledRectangle($im,0,0,$width,$height,$bg_color);

// Draw outline around canvas
ImageRectangle($im,0,0,$width-1,$height-1,$line_color);

// Add title
$title = “Poll Results”;
$title_dimensions = ImageTTFBBox($title_size, 0, $font, $title);
$title_length = $title_dimensions[2] - $title_dimensions[0];
$title_height = abs($title_dimensions[7] - $title_dimensions[1]);
```

```
$title_above_line = abs($title_dimensions[7]);
$title_x = ($width-$title_length)/2; // center it in x
$title_y = ($y - $title_height)/2 + $title_above_line; // center in y gap
ImageTTFText($im, $title_size, 0, $title_x, $title_y,
       $text_color, $font, $title);

// Draw a base line from a little above first bar location
// to a little below last
ImageLine($im, $x, $y-5, $x, $height-15, $line_color);
```

Pada Bagian 3, kita menyiapkan gambar dasar, mengalokasikan warna, lalu mulai menggambar grafik. Kita mengisi latar belakang untuk grafik kali ini menggunakan

```
ImageFilledRectangle($im,0,0,$width,$height,$bg_color);
```

Fungsi `ImageFilledRectangle()`, seperti yang mungkin Anda bayangkan, menggambar persegi panjang yang terisi. Parameter pertama, seperti biasa, adalah pengenal gambar. Kemudian kita harus meneruskannya koordinat X dan Y dari titik awal dan titik akhir persegi panjang. Ini masing-masing sesuai dengan sudut kiri atas dan sudut kanan bawah. Dalam kasus ini, kita mengisi seluruh kanvas dengan warna latar belakang, yang merupakan parameter terakhir, dan warnanya putih. Kemudian kita memanggil;

```
ImageRectangle($im,0,0,$width-1,$height-1,$line_color);
```

untuk menggambar garis luar hitam di sekeliling tepi kanvas. Fungsi ini menggambar persegi panjang yang diberi garis luar, bukan yang terisi. Parameternya sama. Perhatikan bahwa kita telah menggambar persegi panjang tersebut ke `$width-1` dan `$height-1` kanvas dengan lebar dan tinggi berkisar dari (0, 0) ke nilai-nilai ini. Jika kita menggambarnya ke $width dan `$height`, persegi panjang tersebut akan berada di luar area kanvas.

Kita menggunakan logika dan fungsi yang sama seperti yang kita lakukan pada skrip terakhir kita untuk memusatkan dan menulis judul pada grafik. Terakhir, kita menggambar garis dasar untuk batang-batang dengan

```
ImageLine($im, $x, $y-5, $x, $height-15, $line_color);
```

Fungsi `ImageLine()` menggambar garis pada gambar yang kita tentukan (`$im`) dari satu set koordinat (`$x, $y-5`) ke yang lain (`$x, $height-15`), dalam warna yang ditentukan oleh $line_color. Dalam kasus ini, kita menggambar garis dasar dari sedikit di atas tempat kita ingin menggambar batang pertama, hingga sedikit di atas bagian bawah kanvas.

Kita sekarang siap untuk mengisi data pada grafik. Bagian 4 ditunjukkan pada Daftar 10.5.4.

**Daftar 10.5.4 Showpoll.Php Bagian 4 Menggambar Data Aktual Pada Grafik Dan Menyelesaikannya**

```
/*****************************************
Draw data into graph
*****************************************/
// Get each line of db data and draw corresponding bars
while ($row = mysql_fetch_object ($result))
{
  if ($total_votes > 0)
    $percent = intval(round(($row->num_votes/$total_votes)*100));
  else
    $percent = 0;

  // display percent for this value
  ImageTTFText($im, $main_size, 0, $width-30, $y+($bar_height/2),
        $percent_color, $font, $percent.”%”);
  if ($total_votes > 0)
    $right_value = intval(round(($row->num_votes/$total_votes)*100));
  else
    $right_value = 0;

  // length of bar for this value
  $bar_length = $x + ($right_value * $bar_unit);

  // draw bar for this value
    ImageFilledRectangle($im, $x, $y-2, $bar_length, $y+$bar_height,
    $bar_color);

  // draw title for this value
  ImageTTFText($im, $main_size, 0, $text_indent, $y+($bar_height/2),
      $text_color, $font, “$row->candidate”);

  // draw outline showing 100%
  ImageRectangle($im, $bar_length+1, $y-2,
      ($x+(100*$bar_unit)), $y+$bar_height, $line_color);

  // display numbers
  ImageTTFText($im, $small_size, 0, $x+(100*$bar_unit)-50,
      $y+($bar_height/2), $number_color, $font, $row-
      >num_votes.”/”.$total_votes);

  // move down to next bar
  $y=$y+($bar_height+$bar_spacing);
}

/*****************************************
Display image
```

```
*****************************************/
Header("Content-type: image/png");
ImagePng($im);

/*****************************************
  Clean up
*****************************************/
ImageDestroy ($im);
?>
```

Bagian 4 menelusuri kandidat dari basis data satu per satu, menghitung persentase suara, dan menggambar garis dan label untuk setiap kandidat.

Sekali lagi kita menambahkan label menggunakan ImageTTFText(). Kita menggambar garis sebagai persegi panjang terisi menggunakan ImageFilledRectangle():

```
ImageFilledRectangle($im, $x, $y-2, $bar_length, $y+$bar_height, $bar_color);
```

Kita menambahkan garis luar untuk tanda 100% menggunakan ImageRectangle():

```
ImageRectangle($im, $bar_length+1, $y-2,
                  ($x+(100*$bar_unit)), $y+$bar_height, $line_color);
```

Setelah kita menggambar semua garis, kita kembali mengeluarkan gambar menggunakan `ImagePNG()`, dan membersihkannya sendiri menggunakan `ImageDestroy()`. Ini adalah skrip yang cukup panjang, tetapi dapat dengan mudah disesuaikan dengan kebutuhan Anda, atau untuk membuat jajak pendapat secara otomatis melalui antarmuka. Satu fitur penting yang tidak ada pada skrip ini adalah mekanisme anti-kecurangan. Pengguna akan segera menyadari bahwa mereka dapat memberikan suara berulang kali dan membuat hasilnya tidak berarti.

**Fungsi Gambar Lainnya**

Selain fungsi gambar yang telah kita gunakan dalam bab ini, ada fungsi yang memungkinkan Anda menggambar garis lengkung (`ImageArc()`) dan poligon (`ImagePolygon()`), serta variasi dari yang telah kita gunakan di sini. Selalu mulai dengan membuat sketsa apa yang ingin Anda gambar, lalu Anda dapat membuka manual untuk fungsi tambahan yang mungkin Anda perlukan.

# BAB 11
# MENGGUNAKAN KONTROL SESI DALAM PHP

***Apa Itu Kontrol Sesi***

Anda mungkin pernah mendengar bahwa "HTTP adalah protokol tanpa status." Artinya, protokol tersebut tidak memiliki cara bawaan untuk mempertahankan status antara dua transaksi. Ketika pengguna meminta satu halaman, diikuti oleh halaman lain, HTTP tidak menyediakan cara bagi kita untuk mengetahui bahwa kedua permintaan tersebut berasal dari pengguna yang sama.

Ide kontrol sesi adalah untuk dapat melacak pengguna selama satu sesi di situs Web. Jika kita dapat melakukan ini, kita dapat dengan mudah mendukung proses masuk pengguna dan menampilkan konten sesuai dengan tingkat otorisasi atau preferensi pribadinya. Kita dapat melacak perilaku pengguna. Kita dapat menerapkan keranjang belanja. Dalam versi PHP sebelumnya, kontrol sesi didukung melalui PHPLib, PHP Base Library, yang masih merupakan perangkat yang berguna. Sejak versi 4, PHP menyertakan fungsi kontrol sesi asli. Secara konseptual, fungsi tersebut mirip dengan PHPLib, tetapi PHPLib menawarkan beberapa fungsi tambahan. Jika Anda merasa bahwa fungsi asli tersebut tidak sepenuhnya memenuhi kebutuhan Anda, Anda mungkin ingin mencobanya.

## 11.1 FUNGSIONALITAS SESI DASAR

Sesi dalam PHP digerakkan oleh ID sesi unik, angka acak kriptografi. ID sesi ini dibuat oleh PHP dan disimpan di sisi klien selama masa pakai sesi. ID tersebut dapat disimpan di komputer pengguna dalam cookie, atau diteruskan melalui URL. ID sesi berfungsi sebagai kunci yang memungkinkan Anda mendaftarkan variabel tertentu sebagai apa yang disebut variabel sesi. Isi variabel ini disimpan di server. ID sesi adalah satu-satunya informasi yang terlihat di sisi klien. Jika, pada saat koneksi tertentu ke situs Anda, ID sesi terlihat baik melalui cookie atau URL, Anda dapat mengakses variabel sesi yang disimpan di server untuk sesi tersebut. Secara default, variabel sesi disimpan dalam file datar di server. (Anda dapat mengubahnya untuk menggunakan basis data jika Anda ingin menulis fungsi Anda sendiri lebih lanjut tentang ini di bagian “Mengonfigurasi Kontrol Sesi.”) Anda mungkin pernah menggunakan situs web yang menyimpan ID sesi di URL. Jika ada serangkaian data yang tampak acak di URL Anda, kemungkinan besar itu adalah semacam kontrol sesi. Cookie adalah solusi berbeda untuk masalah mempertahankan status di sejumlah transaksi sambil tetap memiliki URL yang tampak bersih.

**Apa Itu Cookie?**

Cookie adalah sepotong kecil informasi yang dapat disimpan oleh skrip di komputer sisi klien. Anda dapat menyetel cookie di komputer pengguna dengan mengirimkan header HTTP yang berisi data dalam format berikut:

```
Set-Cookie: NAME=VALUE; [expires=DATE;] [path=PATH;]
```

```
[domain=DOMAIN_NAME;] [secure]
```

Ini akan membuat kuki bernama NAME dengan nilai VALUE. Semua parameter lainnya bersifat opsional. Kolom expires menetapkan tanggal saat kuki tidak lagi relevan. (Perhatikan bahwa jika tidak ada tanggal kedaluwarsa yang ditetapkan, kuki tersebut secara efektif bersifat permanen kecuali dihapus secara manual oleh Anda atau pengguna.) Bersama-sama, path dan domain dapat digunakan untuk menentukan URL atau URL yang relevan dengan kuki. Kata kunci secure berarti bahwa kuki tidak akan dikirim melalui koneksi HTTP biasa. Saat peramban terhubung ke URL, peramban akan mencari kuki yang tersimpan secara lokal terlebih dahulu. Jika ada yang relevan dengan URL yang terhubung, kuki tersebut akan dikirimkan kembali ke server.

**Menetapkan Cookie dari PHP**

Anda dapat menetapkan kuki secara manual di PHP menggunakan fungsi `setcookie()`. Fungsi ini memiliki prototipe berikut:

```
int setcookie (string name [, string value [, int expire [, string path [,
  string domain [, int secure]]]]])
```

Parameter tersebut sama persis dengan yang ada di header Set-Cookie yang disebutkan sebelumnya. Jika Anda menetapkan cookie sebagai

```
setcookie (“mycookie”, “value”);
```

ketika pengguna mengunjungi halaman berikutnya di situs Anda (atau memuat ulang halaman saat ini), Anda akan memiliki akses ke variabel bernama `$mycookie` yang berisi nilai “value”. Anda juga dapat mengaksesnya melalui

```
$HTTP_COOKIE_VARS[“mycookie”].
```

Anda dapat menghapus cookie dengan memanggil `setcookie()` lagi dengan nama cookie yang sama tetapi tanpa nilai. Jika Anda menyetel cookie dengan parameter lain (seperti URL tertentu atau tanggal kedaluwarsa), Anda harus mengirim parameter yang sama lagi, atau Anda tidak akan dapat menghapus cookie. Anda juga dapat menyetel cookie secara manual melalui fungsi `Header()` dan sintaksis cookie yang diberikan sebelumnya. Salah satu kiatnya adalah header cookie harus dikirim sebelum header lainnya, atau header tersebut tidak akan berfungsi.

**Menggunakan Cookie dengan Sesi**

Cookie memiliki beberapa masalah terkait: Beberapa browser tidak menerima cookie, dan beberapa pengguna mungkin telah menonaktifkan cookie di browser mereka. Ini adalah salah satu alasan sesi PHP menggunakan metode cookie/URL ganda. (Kita akan membahas lebih lanjut tentang ini sebentar lagi.) Saat Anda menggunakan sesi PHP, Anda tidak perlu menyetel cookie secara manual. Fungsi sesi akan mengurusnya untuk Anda. Anda dapat

menggunakan fungsi `session_get_cookie_params()` untuk melihat konten cookie yang disetel oleh kontrol sesi. Ia mengembalikan array asosiatif yang berisi elemen lifetime, path, dan domain.
Anda juga dapat menggunakan

```
session_set_cookie_params($lifetime, $path, $domain);
```

untuk menyetel parameter cookie sesi.

## 11.2 MENYIMPAN ID SESI

PHP akan menggunakan cookie secara default dengan sesi. Jika memungkinkan, cookie akan disetel untuk menyimpan ID sesi. Metode lain yang dapat digunakan adalah menambahkan ID sesi ke URL. Anda dapat menyetelnya agar terjadi secara otomatis jika Anda mengompilasi PHP dengan opsi *--enable-trans-sid*.

Atau, Anda dapat menyematkan ID sesi secara manual di tautan sehingga dapat diteruskan. ID sesi disimpan dalam SID konstan. Untuk meneruskannya secara manual, Anda menambahkannya di akhir tautan yang mirip dengan parameter GET:

```
<A HREF=”link.php?<?=SID?>”>
```

Umumnya lebih mudah untuk mengompilasi dengan *--enable-trans-sid*, jika memungkinkan. Perhatikan juga bahwa konstanta SID hanya akan berfungsi seperti ini jika Anda telah mengonfigurasi PHP dengan *--enable-track-vars*.

**Mengimplementasikan Sesi Sederhana**

Langkah-langkah dasar penggunaan sesi adalah

- Memulai sesi
- Mendaftarkan variabel sesi
- Menggunakan variabel sesi
- Membatalkan pendaftaran variabel dan menghapus sesi

Perhatikan bahwa langkah-langkah ini tidak harus semuanya terjadi dalam skrip yang sama, dan beberapa di antaranya akan terjadi dalam beberapa skrip. Mari kita bahas masing-masing langkah ini secara bergantian.

**Memulai Sesi**

Sebelum Anda dapat menggunakan fungsionalitas sesi, Anda perlu benar-benar memulai sesi. Ada tiga cara untuk melakukannya. Yang pertama, dan paling sederhana, adalah memulai skrip dengan panggilan ke fungsi `session_start()`:

```
session_start();
```

Fungsi ini memeriksa untuk melihat apakah sudah ada ID sesi saat ini. Jika tidak, ia akan membuat satu. Jika sudah ada, pada dasarnya variabel sesi yang terdaftar akan dimuat sehingga Anda dapat menggunakannya. Ada baiknya untuk memanggil `session_start()` di

awal semua skrip yang menggunakan sesi. Kedua, sesi akan dimulai saat Anda mencoba mendaftarkan variabel sesi (lihat bagian berikutnya). Cara ketiga untuk memulai sesi adalah dengan mengatur PHP agar memulainya secara otomatis saat seseorang mengunjungi situs Anda. Anda dapat melakukannya dengan opsi `session.auto_start` di file php.ini—kita akan membahasnya saat membahas konfigurasi.

**Mendaftarkan Variabel Sesi**

Agar variabel dapat dilacak dari satu skrip ke skrip lain, Anda perlu mendaftarkannya dengan panggilan ke `session_register()`. Misalnya, untuk mendaftarkan variabel `$myvar`, Anda dapat menggunakan kode berikut

```
$myvar = 5; session_register("myvar");
```

Perhatikan bahwa Anda perlu meneruskan string yang berisi nama variabel ke `session_register()`. String ini tidak boleh menyertakan simbol $. Ini akan merekam nama variabel dan melacak nilainya. Variabel akan dilacak hingga sesi berakhir, atau hingga Anda membatalkan pendaftarannya secara manual. Anda dapat mendaftarkan lebih dari satu variabel sekaligus dengan memberikan daftar nama variabel yang dipisahkan koma; misalnya

```
session_register("myvar1", "myvar2");
```

**Menggunakan Variabel Sesi**

Untuk membawa variabel sesi ke dalam cakupan sehingga dapat digunakan, Anda harus terlebih dahulu memulai sesi menggunakan salah satu opsi yang dijelaskan sebelumnya. Anda kemudian dapat mengakses variabel tersebut. Jika Anda mengaktifkan `register_globals`, Anda dapat mengaksesnya melalui nama bentuk pendeknya; misalnya, `$myvar`. Jika Anda tidak mengaktifkannya, Anda dapat mengakses variabel melalui array asosiatif `$HTTP_SESSION_VARS` sebagai, misalnya, `$HTTP_SESSION_VARS["myvar"].`

Variabel sesi tidak dapat ditimpa oleh data GET atau POST, yang merupakan fitur keamanan yang baik, tetapi perlu diingat saat membuat kode. Di sisi lain, Anda perlu berhati-hati saat memeriksa apakah variabel sesi telah ditetapkan (melalui, misalnya, `isset()` atau `empty()`). Ingatlah bahwa variabel dapat ditetapkan oleh pengguna melalui GET atau POST. Anda dapat memeriksa variabel untuk melihat apakah itu adalah variabel sesi terdaftar dengan memanggil fungsi `session_is_registered()`. Anda memanggil fungsi ini seperti ini:

```
$result = session_is_registered("myvar");
```

Ini akan memeriksa apakah `$myvar` adalah variabel sesi terdaftar dan mengembalikan true atau false. Atau, Anda juga dapat memeriksa array `$HTTP_SESSION_VARS` untuk mengetahui keberadaan variabel.

**Menghapus Pendaftaran Variabel dan Memusnahkan Sesi**

Setelah selesai dengan variabel sesi, Anda dapat menghapus pendaftarannya menggunakan fungsi `session_unregister()`, sebagai berikut:

```
session_unregister(“myvar”);
```

Sekali lagi, fungsi ini memerlukan nama variabel yang ingin Anda hapus pendaftarannya sebagai string, dan tanpa simbol $. Fungsi ini hanya dapat menghapus pendaftaran satu variabel sesi pada satu waktu (tidak seperti `session_register()`). Namun, Anda dapat menggunakan `session_unset()` untuk menghapus pendaftaran semua variabel sesi saat ini. Setelah selesai dengan sesi, pertama-tama Anda harus membatalkan pendaftaran semua variabel, lalu memanggil

```
session_destroy();
```

untuk membersihkan ID sesi.

**Contoh Sesi Sederhana**

Beberapa hal ini mungkin tampak sedikit abstrak, jadi mari kita lihat contohnya. Kita akan menerapkan satu set yang terdiri dari tiga halaman. Pada halaman pertama, kita akan memulai sesi dan mendaftarkan variabel `$sess_var`. Kode untuk melakukan ini ditunjukkan pada Daftar 11.1.

Daftar 11.1 Page1.Php—Memulai Sesi Dan Mendaftarkan Variabel

```
<?

    session_start();
    session_register(“sess_var”);

    $sess_var = “Hello world!”;

    echo “The content of \$sess_var is $sess_var<br>”;

?>
<a href = “page2.php”>Next page</a>
```

Kita telah mendaftarkan variabel dan menetapkan nilainya. Output dari skrip ini ditunjukkan pada Gambar 11.1.

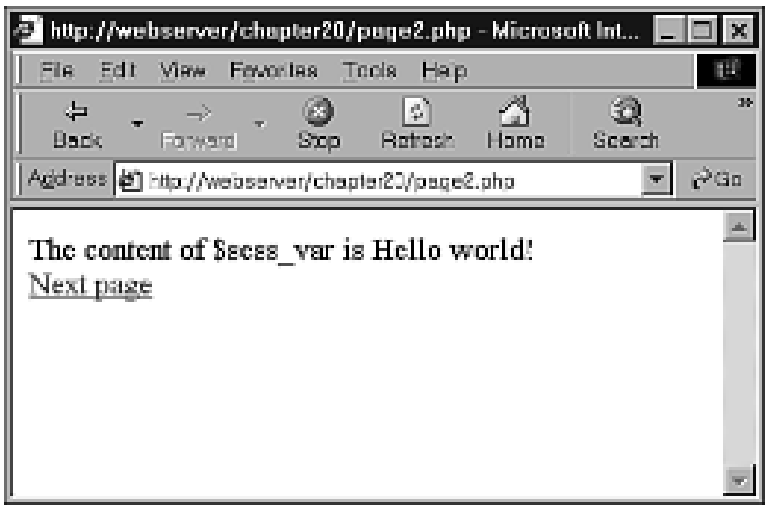

**Gambar 11.1 Nilai Awal Variabel Sesi Ditunjukkan Oleh Page1.Php.**

Perhatikan bahwa kita mengubah nilai setelah variabel didaftarkan. Kita juga dapat melakukan yang sebaliknya menetapkan nilai dan kemudian mendaftarkan variabel. Nilai akhir variabel pada halaman adalah nilai yang akan tersedia pada halaman berikutnya. Di akhir skrip, variabel sesi diserialisasikan, atau dibekukan, hingga dimuat ulang melalui panggilan berikutnya ke `session_start()`. Oleh karena itu, kita memulai skrip berikutnya dengan memanggil `session_start()`. Skrip ini ditunjukkan pada Daftar 11.2.

**Daftar 11.2** Page2.Php Mengakses Variabel Sesi Dan Membatalkan Pendaftarannya

```
<?
    session_start();
    echo “The content of \$sess_var is $sess_var<br>”;
    session_unregister(“sess_var”);
?>
<a href = “page3.php”>Next page</a>
```

Setelah memanggil `session_start()`, variabel $sess_var tersedia dengan nilai yang disimpan sebelumnya, seperti yang dapat Anda lihat pada Gambar 11.2.



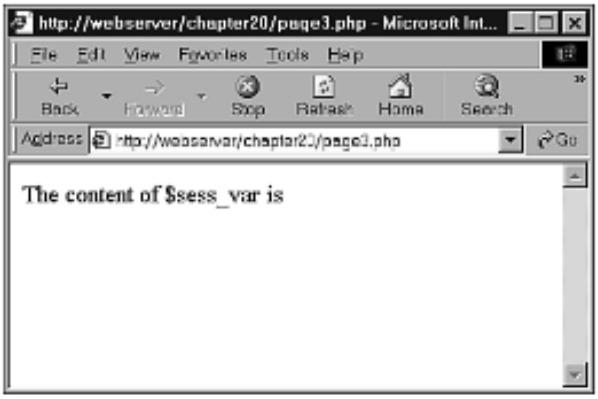

**Gambar 11.2** Nilai Variabel Sesi Telah Diteruskan Melalui Id Sesi Ke Page2.Php.

Setelah menggunakan variabel, kita memanggil `session_unregister()` untuk membatalkan pendaftaran variabel. Dengan cara ini, sesi masih ada, tetapi variabel `$sess_var` bukan lagi variabel yang terdaftar. Akhirnya kita meneruskan ke page3.php, skrip terakhir dalam contoh kita. Kode untuk skrip ini ditunjukkan dalam Daftar 11.3.

Daftar 11.3 Page3.Php Mengakhiri Sesi

```
<?
    session_start();
    echo “The content of \$sess_var is $sess_var<br>”;
    session_destroy();
?>
```

Seperti yang dapat Anda lihat pada Gambar 11.3, kita tidak lagi memiliki akses ke nilai persisten $sess_var.




**Gambar 11.3 Variabel Yang Dihapus Pendaftarannya Tidak Lagi Tersedia**.

Kita akhiri dengan memanggil `session_destroy()` untuk membuang ID sesi.

## 11.3 MENGONFIGURASI KONTROL SESI

Terdapat serangkaian opsi konfigurasi untuk sesi yang dapat Anda atur dalam berkas php.ini. Beberapa opsi yang lebih berguna, dan deskripsi masing-masing, ditunjukkan pada Tabel 11.1.

**Tabel 11.1 Opsi Konfigurasi Sesi**

| ***Nama Optional*** | ***Default*** | ***Efek*** |
|---|---|---|
| `session.auto_start` | 0 (Disabled) | Secara otomatis memulai sesi. |
| `session. cache_expire` | 180 | Mengatur jangka waktu aktif untuk halaman sesi yang di-cache, dalam menit. |
| `session.cookie_domain` | Tidak ada | Domain yang akan ditetapkan dalam cookie sesi. |

| | | |
|---|---|---|
| session.cookie_lifetime | 0 | Berapa lama cookie ID sesi akan bertahan di komputer pengguna. Nilai default, 0, akan bertahan hingga browser ditutup. |
| session.cookie_path | / | Jalur untuk ditetapkan dalam kuki sesi. |
| session.name | PHPSESSID | Nama sesi yang digunakan sebagai nama kuki pada sistem pengguna. |
| session.save_handler | Files | Menentukan tempat penyimpanan data sesi. Anda dapat mengaturnya agar mengarah ke database, tetapi Anda harus menulis fungsi Anda sendiri. |
| session.save_path | /tmp | Jalur tempat data sesi disimpan. Secara lebih umum, argumen yang diteruskan ke penyimpanan ditangani dan didefinisikan oleh session.save_handler. |
| session. use_cookies | 1 (Enabled) | Mengonfigurasi sesi untuk menggunakan cookie di sisi klien. |

**Menerapkan Autentikasi dengan Kontrol Sesi**

Terakhir, kita akan melihat contoh yang lebih substansial menggunakan kontrol sesi. Mungkin penggunaan kontrol sesi yang paling umum adalah untuk melacak pengguna setelah mereka diautentikasi melalui mekanisme login. Dalam contoh ini, kita akan menggabungkan autentikasi dari basis data MySQL dengan penggunaan sesi untuk menyediakan fungsionalitas ini.


**Gambar 11.4 Karena Pengguna Belum Masuk, Tunjukkan Halaman Masuk.**

Contoh tersebut terdiri dari tiga skrip sederhana. Yang pertama, authmain.php, menyediakan formulir login dan autentikasi untuk anggota situs Web kita. Yang kedua, members_only.php, menampilkan informasi hanya kepada anggota yang telah berhasil

login. Yang ketiga, `logout.php`, mengeluarkan anggota. Untuk memahami cara kerjanya, lihat Gambar 11.4. Ini adalah halaman awal yang ditampilkan oleh

`authmain.php`.

Halaman ini menyediakan tempat bagi pengguna untuk masuk. Jika pengguna mencoba mengakses bagian Anggota tanpa masuk terlebih dahulu, pengguna akan mendapatkan pesan yang ditunjukkan pada Gambar 11.5.


**Gambar 11.5 Pengguna Yang Belum Masuk Tidak Dapat Melihat Konten Situs; Pesan Ini Akan Ditampilkan Sebagai Gantinya.**

Namun, jika pengguna masuk terlebih dahulu (dengan nama pengguna: `testuser` dan kata sandi: `test123`) lalu mencoba melihat halaman Anggota, pengguna akan mendapatkan output yang ditunjukkan pada Gambar 1.6.


**Gambar 11.6 Setelah Pengguna Masuk, Ia Dapat Mengakses Area Anggota.**

Mari kita lihat kode untuk aplikasi ini. Sebagian besar kode ada di `authmain.php`. Skrip ini dapat dilihat di Daftar 11.4. Kita akan membahasnya sedikit demi sedikit.

**Daftar 11.4** Authmain.Php Bagian Utama Aplikasi Autentikasi

```
<?
session_start();

if ($userid && $password)
{
  // if the user has just tried to log in

  $db_conn = mysql_connect("localhost", "webauth", "webauth");
  mysql_select_db("auth", $db_conn);
  $query = "select * from auth " ."where name='$userid' "
        ." and pass=password('$password')";
  $result = mysql_query($query, $db_conn);
  if (mysql_num_rows($result) >0 )
  {
    // if they are in the database register the user id
    $valid_user = $userid;
    session_register("valid_user");
  }
}
?>
<html>
<body>
<h1>Home page</h1>
<?
```

```
    if (session_is_registered("valid_user"))
    {
      echo "You are logged in as: $valid_user <br>";
      echo "<a href=\"logout.php\">Log out</a><br>";
    }
    else
    {
      if (isset($userid))
      {
        // if they've tried and failed to log in
        echo "Could not log you in";
      }
      else
      {
        // they have not tried to log in yet or have logged out
        echo "You are not logged in.<br>";
      }

      // provide form to log in
      echo "<form method=post action=\"authmain.php\">";
      echo "<table>";
      echo "<tr><td>Userid:</td>";
      echo "<td><input type=text name=userid></td></tr>";
      echo "<tr><td>Password:</td>";
      echo "<td><input type=password name=password></td></tr>";
      echo "<tr><td colspan=2 align=center>";
      echo "<input type=submit value=\"Log in\"></td></tr>";
      echo "</table></form>";
    }
  ?>
  <br>
  <a href="members_only.php">Members section</a>
  </body>
  </html>
```

Beberapa logika yang cukup rumit ada dalam skrip ini karena skrip ini menampilkan formulir login, dan juga merupakan tindakan dari formulir tersebut.

Aktivitas skrip ini berputar di sekitar variabel sesi `$valid_user`. Ide dasarnya adalah jika seseorang berhasil login, kita akan mendaftarkan variabel sesi yang disebut `$valid_user` yang berisi userid-nya. Hal pertama yang kita lakukan dalam skrip ini adalah memanggil `session_start()`. Ini akan memuat variabel sesi `$valid_user` jika telah terdaftar. Pada bagian pertama skrip, tidak ada kondisi if yang akan berlaku dan pengguna akan gagal hingga akhir skrip, di mana kita memberi tahu dia bahwa dia tidak login dan memberinya formulir untuk melakukannya:

```
echo “<form method=post action=\”authmain.php\”>”;
echo “<table>”;
echo “<tr><td>Userid:</td>”;
echo “<td><input type=text name=userid></td></tr>”;
echo “<tr><td>Password:</td>”;
echo “<td><input type=password name=password></td></tr>”;
echo “<tr><td colspan=2 align=center>”;
echo “<input type=submit value=\”Log in\”></td></tr>”;
echo “</table></form>”;
```

Saat dia menekan tombol kirim pada formulir, skrip ini dipanggil kembali dan kita mulai lagi dari awal. Kali ini, kita akan memiliki `userid` dan `password` untuk diautentikasi, yang disimpan sebagai `$userid` dan `$password`. Jika variabel ini ditetapkan, kita masuk ke blok autentikasi:

```
if ($userid && $password)
{
  // if the user has just tried to log in

  $db_conn = mysql_connect(“localhost”, “webauth”, “webauth”);
  mysql_select_db(“auth”, $db_conn);
  $query = “select * from auth “
           .”where name=’$userid’ “
           .” and pass=password(‘$password’)”;
  $result = mysql_query($query, $db_conn);
```

Kami terhubung ke basis data MySQL dan memeriksa userid dan kata sandi. Jika keduanya cocok di basis data, kami mendaftarkan variabel `$valid_user` yang berisi userid untuk pengguna tertentu ini, sehingga kami tahu siapa yang login lebih lanjut.

```
if (mysql_num_rows($result) >0 )
  {
    // if they are in the database register the user id
    $valid_user = $userid;
    session_register(“valid_user”);
  }
}
```

Karena sekarang kita tahu siapa dia, kita tidak perlu menunjukkan formulir login lagi kepadanya. Sebagai gantinya, kita akan memberi tahu dia bahwa kita tahu siapa dia, dan memberinya pilihan untuk keluar:

```
if (session_is_registered(“valid_user”))
{
  echo “You are logged in as: $valid_user <br>”;
  echo “<a href=\”logout.php\”>Log out</a><br>”;
```

```
}
```

Jika kita mencoba untuk login dan gagal karena suatu alasan, kita akan memiliki userid tetapi bukan variabel `$valid_user`, sehingga kita dapat memberinya pesan kesalahan:

```
if (isset($userid))
{
  // if they've tried and failed to log in
  echo "Could not log you in";
}
```

Sekian untuk skrip utama. Sekarang, mari kita lihat halaman Anggota. Kode untuk skrip ini ditampilkan dalam Daftar 11.5.

**Daftar 11.5** Members_Only.Phpn Kode Untuk Bagian Anggota Di Situs Web Kami Memeriksa Pengguna Yang Valid

```
<?
  session_start();

  echo "<h1>Members only</h1>";

  // check session variable

  if (session_is_registered("valid_user"))
  {
    echo "<p>You are logged in as $valid_user.</p>";
    echo "<p>Members only content goes here</p>";
  }


  else
  {
    echo "<p>You are not logged in.</p>";
    echo "<p>Only logged in members may see this page.</p>";
  }

  echo "<a href=\"authmain.php\">Back to main page</a>";
?>
```

Kode ini sangat sederhana. Yang dilakukannya hanyalah memulai sesi, dan memeriksa apakah sesi saat ini berisi pengguna terdaftar menggunakan fungsi `session_registered_user()`. Jika pengguna masuk, kami menunjukkan kepadanya konten anggota; jika tidak, kami memberi tahunya bahwa dia tidak berwenang.

Terakhir, kami memiliki skrip `logout.php` yang mengeluarkan pengguna dari sistem. Kode untuk skrip ini ditunjukkan dalam Daftar 11.6.

**Daftar 11.6** `Logout.Php` Skrip Ini Membatalkan Pendaftaran Variabel Sesi Dan Menghancurkan Sesi

```
<?
  session_start();

  $old_user = $valid_user; // store to test if they *were* logged in
  $result = session_unregister("valid_user");
  session_destroy();
?>
<html>
<body>
<h1>Log out</h1>
<?
  if (!empty($old_user))
  {
    if ($result)
    {
      // if they were logged in and are not logged out
      echo "Logged out.<br>";
    }
    else
    {
      // they were logged in and could not be logged out
      echo "Could not log you out.<br>";
    }
  }
  else
  {
    // if they weren't logged in but came to this page somehow
    echo "You were not logged in, and so have not been logged out.<br>";
  }
?>
```

Kodenya sangat sederhana, tetapi kami melakukan sedikit kerja yang rumit. Kami memulai sesi, menyimpan nama pengguna lama pengguna, membatalkan pendaftaran variabel pengguna yang valid, dan menghapus sesi. Kemudian, kami memberikan pesan kepada pengguna yang akan berbeda jika pengguna tersebut keluar dari akun, tidak dapat keluar dari akun, atau tidak masuk ke akun sejak awal. Rangkaian skrip sederhana ini akan menjadi dasar bagi banyak pekerjaan yang akan kami lakukan di bab-bab berikutnya.

# BAB 12
# FITUR BERGUNA LAINNYA

## 12.1 MENGGUNAKAN TANDA KUTIP AJAIB

Anda mungkin telah memperhatikan bahwa Anda perlu berhati-hati saat menggunakan simbol tanda kutip (‘ dan “) dan garis miring terbalik (\) dalam string. PHP akan menjadi bingung dengan pernyataan string yang dicoba seperti

```
echo “color = “#FFFFFF””;
```

dan memberikan kesalahan penguraian. Untuk menyertakan tanda kutip di dalam string, gunakan jenis tanda kutip yang berbeda dari tanda kutip yang melampirkan string. Misalnya

```
echo “color = ‘#FFFFFF’”;
```

atau

```
echo ‘color = “#FFFFFF”’;
```

keduanya akan valid. Masalah yang sama terjadi dengan input pengguna, serta input dan output ke, atau dari, program lain. Mencoba menjalankan query mysql seperti

```
insert into company values (‘Bob’s Auto Parts’);
```

akan menghasilkan kebingungan serupa di parser MySQL. Kita telah melihat penggunaan `addslashes()` dan `stripslashes()` yang akan menghilangkan karakter tanda kutip tunggal, tanda kutip ganda, garis miring terbalik, dan NUL.

PHP memiliki kemampuan yang berguna untuk menambahkan dan menghapus garis miring secara otomatis atau ajaib untuk Anda. Dengan dua pengaturan dalam berkas php.ini, Anda dapat mengaktifkan atau menonaktifkan tanda kutip ajaib untuk `GET, POST`, data `Cookie`, dan untuk sumber lainnya. Nilai dari direktif `magic_quotes_gpc` mengontrol apakah tanda kutip ajaib digunakan untuk operasi `GET, POST`, dan `Cookie`.

Dengan `magic_quotes_gpc` aktif, jika seseorang mengetik “*Bob’s Auto Parts*” ke dalam formulir di situs Anda, skrip Anda akan menerima “*Bob\’s Auto Parts*” karena tanda kutip akan di-escape untuk Anda. Fungsi `get_magic_quotes_gpc()` mengembalikan 1 atau 0, yang memberi tahu Anda nilai `magic_quotes_gpc` saat ini. Ini sangat berguna untuk pengujian jika Anda perlu menghapus garis miring() dari data yang diterima dari pengguna.

Nilai magic_quotes_runtime, mengontrol apakah tanda kutip ajaib digunakan oleh fungsi yang mendapatkan data dari basis data dan berkas. Untuk mendapatkan nilai magic_quotes_runtime, gunakan fungsi get_magic_quotes_runtime(). Fungsi ini mengembalikan 1 atau 0. Magic quoting dapat diaktifkan untuk skrip tertentu menggunakan fungsi set_magic_quotes_runtime().

**Mengevaluasi String: eval()**

Fungsi eval() akan mengevaluasi string sebagai kode PHP. Misalnya,

```
eval ( “echo ‘Hello World’;” );
```

akan mengambil konten string dan mengeksekusinya. Baris ini akan menghasilkan output yang sama seperti

```
echo ‘Hello World’;
```

Ada berbagai kasus di mana eval() dapat berguna. Anda mungkin ingin menyimpan blok kode dalam database, dan mengambil serta eval() blok tersebut di lain waktu. Anda mungkin ingin membuat kode dalam satu loop, lalu menggunakan eval() untuk mengeksekusinya. Anda dapat menggunakan eval() untuk memperbarui atau mengoreksi kode yang ada. Jika Anda memiliki banyak koleksi skrip yang memerlukan perubahan yang dapat diprediksi, akan mungkin (tetapi tidak efisien) untuk menulis skrip yang memuat skrip lama ke dalam string, menjalankan regexp untuk membuat perubahan, lalu menggunakan eval() untuk menjalankan skrip yang dimodifikasi. Bahkan dapat dibayangkan bahwa orang yang sangat percaya di suatu tempat mungkin ingin mengizinkan kode PHP dimasukkan ke dalam browser dan dijalankan di servernya.

**Menghentikan Eksekusi: die dan exit**

Sejauh ini dalam buku ini kita telah menggunakan konstruksi bahasa *exit* untuk menghentikan eksekusi skrip. Seperti yang mungkin Anda ingat, konstruksi ini muncul pada baris tersendiri, seperti ini:

```
exit;
```

Konstruksi ini tidak mengembalikan apa pun. Untuk penghentian yang sedikit lebih berguna, kita dapat menggunakan die(). Konstruksi bahasa ini dapat digunakan untuk mengeluarkan pesan kesalahan atau menjalankan fungsi sebelum menghentikan skrip. Ini akan familier bagi programmer Perl. Anda dapat menggunakannya sendiri, dengan cara yang mirip dengan exit:

```
die(“Script ending now”);
```

Lebih umum, konstruksi ini ditulis dengan pernyataan yang mungkin gagal, seperti membuka file atau menghubungkan ke basis data:

```
mysql_query($query) atau die(“Could not execute query”);
```

Daripada hanya mencetak pesan kesalahan, Anda dapat memanggil satu fungsi terakhir sebelum skrip berakhir:

```
function err_msg()
{
  echo “MySQL error was: “;
  echo mysql_error();
}

mysql_query($query) or die(err_msg());
```

Ini dapat berguna sebagai cara memberi pengguna beberapa alasan mengapa skrip gagal. Atau, Anda dapat mengirim email kepada diri sendiri sehingga Anda tahu jika terjadi kesalahan besar, atau menambahkan kesalahan ke berkas log.

## 12.2 SERIALISASI

Serialisasi adalah proses mengubah apa pun yang dapat Anda simpan dalam variabel atau objek PHP menjadi aliran byte yang dapat disimpan dalam basis data atau diteruskan melalui URL dari satu halaman ke halaman lainnya. Tanpa ini, sulit untuk menyimpan atau meneruskan seluruh konten array atau objek. Kegunaannya telah berkurang sejak diperkenalkannya kontrol sesi. Serialisasi data terutama digunakan untuk jenis hal yang sekarang Anda gunakan untuk kontrol sesi. Faktanya, fungsi kontrol sesi membuat serial variabel sesi untuk menyimpannya di antara permintaan HTTP.

Namun, Anda mungkin masih ingin menyimpan array atau objek PHP dalam file atau basis data. Jika Anda ingin melakukannya, ada dua fungsi yang perlu Anda ketahui cara penggunaannya: `serialize()` dan `unserialize()`. Anda dapat memanggil fungsi `serialize()` sebagai berikut:

```
$serial_object = serialize($my_object);
```

Jika Anda ingin mengetahui apa yang sebenarnya dilakukan serialisasi, lihat apa yang dikembalikan oleh serialize. Serialisasi mengubah konten objek atau array menjadi string. Sebagai contoh, kita dapat melihat output dari menjalankan serialize pada objek karyawan sederhana, yang didefinisikan dan dibuat dengan cara berikut:

```
class employee
{
  var $name;
  var $employee_id;
};
```

```
$this_emp = new employee;
$this_emp->name = “Fred”;
$this_emp->employee_id = 5324;
```

Jika kita melakukan serialisasi ini dan menampilkannya di browser, outputnya adalah

```
O:8:”employee”:2:{s:4:”name”;s:4:”Fred”;s:11:”employee_id”;i:5324;}
```

Cukup mudah untuk melihat hubungan antara data objek asli dan data serialisasi. Karena data serialisasi hanya berupa teks, Anda dapat menuliskannya ke database atau apa pun yang Anda suka. Ketahuilah bahwa Anda harus menambahkan garis miring() ke data apa pun sebelum menuliskannya ke database, seperti biasa. Anda dapat melihat perlunya hal ini dengan memperhatikan tanda kutip pada string serialisasi sebelumnya. Untuk mendapatkan kembali objek, panggil `unserialize()`:

```
$new_object = unserialize($serial_object);
```

Tentunya, jika Anda memanggil `addslashes()` sebelum memasukkan objek ke database, Anda perlu memanggil `stripslashes()` sebelum membatalkan serialisasi string.

## 12.3 MENDAPATKAN INFORMASI MENGENAI LINGKUNGAN PHP

Sejumlah fungsi dapat digunakan untuk mencari tahu informasi tentang cara PHP dikonfigurasi.

**Mencari Tahu Ekstensi Apa yang Dimuat**

Anda dapat dengan mudah melihat set fungsi apa yang tersedia, dan fungsi apa yang tersedia di setiap set tersebut menggunakan fungsi `get_loaded_extensions()` dan `get_extension_funcs()`. Fungsi `get_loaded_extensions()` mengembalikan array dari semua set fungsi yang saat ini tersedia untuk PHP. Jika diberi nama set fungsi atau ekstensi tertentu, `get_extension_funcs()` mengembalikan array dari fungsi-fungsi dalam set tersebut. Skrip dalam Daftar 12.1 mencantumkan semua fungsi yang tersedia untuk instalasi PHP Anda dengan menggunakan kedua fungsi ini.

**DAFTAR 12.1** *`list_functions.php`* Skrip Ini Mencantumkan Semua Ekstensi yang Tersedia untuk PHP, dan Dengan Setiap Ekstensi, Menyediakan Daftar Fungsi Berpoin dalam Ekstensi Itu

```
<?
  echo “Function sets supported in this install are:<br>”;
  $extensions = get_loaded_extensions();
  foreach ($extensions as $each_ext)
  {
    echo “$each_ext <br>”;
    echo “<ul>”;
    $ext_funcs = get_extension_funcs($each_ext);
```

```
      foreach($ext_funcs as $func)
    {
        echo “<li> $func”;
      }
      echo “</ul>”;
    }
?>
```

Perhatikan bahwa fungsi `get_loaded_extensions()` tidak mengambil parameter apa pun, dan fungsi `get_extension_funcs()` mengambil nama ekstensi sebagai satu-satunya parameternya. Informasi ini dapat membantu jika Anda mencoba mengetahui apakah Anda telah berhasil memasang ekstensi.

**Mengidentifikasi Pemilik Skrip**

Anda dapat mengetahui pengguna yang memiliki skrip yang sedang dijalankan dengan memanggil fungsi `get_current_user()`, sebagai berikut:

```
echo get_current_user();
```

Ini terkadang dapat berguna untuk memecahkan masalah izin.

**Mencari Tahu Kapan Skrip Dimodifikasi**

Menambahkan tanggal modifikasi terakhir ke setiap halaman di situs adalah hal yang cukup populer untuk dilakukan. Anda dapat memeriksa tanggal modifikasi terakhir dari skrip dengan fungsi `getlastmod()` (perhatikan tidak adanya garis bawah pada nama fungsi), sebagai berikut:

```
echo date(“g:i a, j M Y”,getlastmod());
```

Fungsi ini mengembalikan stempel waktu UNIX, yang dapat kita masukkan ke `date()` seperti yang telah kita lakukan di sini, untuk menghasilkan tanggal yang dapat dibaca manusia.

**Memuat Ekstensi Secara Dinamis**

Anda sebenarnya dapat memuat pustaka ekstensi saat runtime, jika tidak dikompilasi, menggunakan fungsi `dl()`. Fungsi ini mengharapkan nama file yang berisi pustaka sebagai parameter. Di bawah UNIX, ini akan menjadi nama file yang diakhiri dengan .so; di bawah Windows, ini akan diakhiri dengan .dll. Contoh panggilan ke `dl()` adalah

```
dl(“php_ftp.dll”);
```

Ini akan memuat ekstensi FTP secara dinamis (pada mesin Windows). Anda tidak boleh menentukan direktori tempat file berada: Sebaliknya, Anda harus mengonfigurasi ini dalam file php.ini. Direktif yang disebut `extension_dir` akan menentukan direktori tempat PHP akan mencari pustaka untuk dimuat secara dinamis. Jika Anda mengalami masalah saat memuat ekstensi secara dinamis, periksa juga file php.ini Anda untuk direktif `enable_dl`. Jika tidak aktif, Anda tidak akan dapat memuat ekstensi secara dinamis. Terutama jika mesin yang

Anda gunakan bukan milik Anda, ini mungkin dinonaktifkan karena alasan keamanan. Anda juga tidak akan dapat menggunakan dl() jika PHP berjalan dalam mode aman.

## 12.3 MENGUBAH RUNTIME ENVIRONMENT UNTUK SEMENTARA

Anda dapat melihat direktif yang ditetapkan dalam file php.ini, atau mengubahnya selama masa pakai skrip tunggal. Ini dapat sangat berguna, misalnya, bersama dengan direktif `max_execution_time` jika Anda tahu skrip Anda akan memerlukan waktu untuk dijalankan. Anda dapat mengakses dan mengubah direktif menggunakan fungsi kembar `ini_get()` dan `ini_set().` Daftar 12.2 menunjukkan skrip sederhana yang menggunakan fungsi-fungsi ini.

**Daftar 12.2** Iniset.Php Skrip Ini Mengatur Ulang Variabel Dari File Php.Ini

```
<?

  $old_max_execution_time = ini_set("max_execution_time", 120);
  echo "old timeout is $old_max_execution_time <br>";
  $max_execution_time = ini_get("max_execution_time");
  echo "new timeout is $max_execution_time <br>";

?>
```

Fungsi `ini_set()` mengambil dua parameter. Yang pertama adalah nama arahan konfigurasi dari php.ini yang ingin kita ubah, dan yang kedua adalah nilai yang ingin kita ubah. Fungsi ini mengembalikan nilai arahan sebelumnya. Dalam kasus ini, kita menyetel ulang nilai dari waktu maksimum default 30 detik untuk menjalankan skrip menjadi 120 detik. Fungsi `ini_get()` hanya memeriksa nilai arahan konfigurasi tertentu. Nama arahan harus diteruskan ke sana sebagai string. Di sini kita hanya menggunakannya untuk memeriksa apakah nilainya benar-benar berubah.

**Penyorotan Sumber**

PHP dilengkapi dengan penyorot sintaks bawaan, mirip dengan banyak IDE. Secara khusus, penyorot ini berguna untuk berbagi kode dengan orang lain, atau menyajikannya untuk didiskusikan di halaman Web. Fungsi `show_source()` dan `highlight_file()` adalah sama. (Fungsi `show_source()` sebenarnya adalah alias untuk `highlight_file()`.) Kedua fungsi ini menerima nama file sebagai parameter. (Berkas ini harus berupa berkas PHP, jika tidak, Anda tidak akan mendapatkan hasil yang berarti.) Misalnya,

```
show_source("list_functions.php");
```

Berkas akan ditampilkan di peramban dengan teks yang disorot dalam berbagai warna, tergantung pada apakah itu berupa string, komentar, kata kunci, atau HTML. Output dicetak pada warna latar belakang. Konten yang tidak sesuai dengan salah satu kategori ini dicetak dalam warna default. Fungsi `highlight_string()` bekerja dengan cara yang sama, tetapi mengambil string sebagai parameter, dan mencetaknya ke peramban dalam format yang

disorot sintaksis. Anda dapat mengatur warna untuk penyorotan sintaksis di berkas php.ini Anda. Bagian yang Anda cari tampak seperti ini:

```
; Colors for Syntax Highlighting mode
highlight.string = #DD0000
highlight.comment = #FF8000
highlight.keyword = #007700
highlight.bg = #FFFFFF
highlight.default = #0000BB
highlight.html = #000000
```

Warnanya dalam format HTML RGB standar.

# DAFTAR PUSTAKA


Ahmed, F. (2023). *JWT and OAuth2 in PHP*. AuthPress.

Ali, N. (2022). *Composer & Autoload PSR-4 in PHP*. PHP Experts Press.

Author, A. A. (2023). Modern PHP: New Features and Good Practices. *PHP: The Right Way*.

Baker, V. (2023). *PSR-3 and Logging Standards in PHP*. PHP Standards Group.

Ballad, T., & Ballad, W. (2021). *Securing PHP Web Applications*. Addison-Wesley Professional.

Brown, K. (2021). *OOP in PHP: Clean Code and Architecture*. TechWorld.

Calzavara, S., Focardi, R., Grimm, N., Maffei, M., & Tempesta, M. (2020, January 28). *Language-Based Web Session Integrity*. arXiv preprint.

Chan, J. (2020, June 23). *PHP: Learn PHP in One Day and Learn It Well: PHP for Beginners with Hands-on Project*. Independently published. ReactDOM+1Reddit+1

Chen, L. (2024). *CodeIgniter 4 RESTful APIs*. CI Press.

CloudDevs. (2020). *Understanding PHP Security: Best Practices for Developers*. CloudDevs blog.

Cloudways. (2021, December 8). *Best PHP Security Tips You Should Know*. Blog post. The European Business Review+2Cloudways+2Cloudways+2

CodeX (Mayur Koshti). (2020). *Secure Coding Practices for PHP Developers*. Medium blog. Reddit+15medium.com+15clouddevs.com+15

Crudu, V. (2023). *Modern PHP Patterns and Practices*. TechPress.

Daillac. (2023). *PHP Best Practices in 2023 for Secure Web Development*. Daillac Blog.

Das, R. (2023). *Eloquent ORM in Laravel*. ORM Publishing.

Davis, T. (2023). *Redis & OPcache for PHP Performance*. CachePress.

Edwards, J. (2024). *Web App Security: HTTPS/TLS for PHP*. SecureConn Press.

European Business Review. (2020). *Top 10 PHP Security Best Practices*. European Business Review. The European Business Review

Gómez, R. (2023). *Docker for PHP Developers*. Container Books.

Grant, C. (2023). *Modern PHP Frameworks Comparison (Laravel, Symfony, CI4)*. DevTrends Publishing

Habr. (2020). *PHP Best Practices to Follow in 2020*. Habr.com. habr.com

Haidar, B. (2024, Jan–Feb). Real-time communication in PHP Laravel: Part 2. *CODE Magazine*.

Jahanshahi, R., Doupé, A., & Egele, M. (2020, June 22). *You shall not pass: Mitigating SQL Injection Attacks on Legacy Web Applications*. arXiv preprint.

JetBrains Blog. (2025, February 5). The State of PHP 2024 – Expert Review.

JetBrains. (2023). *The State of Developer Ecosystem 2023: PHP Trends*. JetBrains report.

Johnson, E. (2022). *Deploying PHP Apps with GitHub Actions*. CI Press.

Kim, S. (2023). *Testing PHP with PHPUnit and Mockery*. QA Press.

Kimura, T. (2022). *Database Security & PHP PDO*. SecureDB Press.

Lee, M. (2022). *PHP Performance Optimization*. PerfPress.

Lopez, H. (2022). *WebSockets and Real-time PHP*. LiveWeb Press.

Madoff, P. (2022). *Learn PHP & MySQL – Zero to Hero Programming Crash Course*. ABC Publisher.

Miller, A. (2023). *Symfony 6: From Beginner to Pro*. Symfony Press.

Müller, S. (2024). *Static Analysis: PHPStan & Psalm*. CodeQuality Press.

Nguyen, P. (2023). *Building REST APIs with PHP*. API Press.

O'Neill, D. (2024). *Middleware & Policy in Laravel*. SecureWeb Press.

Orbitwebtech. (2022). What's New in PHP 8.2: Latest Features and Update.

Packt Publishing. (2022). *PHP Web Development with Laminas*.

Packt Publishing. (2024). *Mastering Laravel 10*.

Patel, R. (2024). *Form Handling & Validation in PHP*. InputPress.

Patel, S. (2024). *CI/CD for PHP Projects*. DevOps Publishing.

PHP Architect. (2022, October). What's changed in PHP 8.2.

PHP.Watch. (2022, December 8). PHP 8.2 Released: features & improvements. *PHP.Watch*.

Rabheru, R., Hanif, H., & Maffeis, S. (2020, December 16). *A Hybrid Graph Neural Network Approach for Detecting PHP Vulnerabilities*. arXiv preprint

Ravoof, S. (2024, August 6). What's new in PHP 8.2 — New Features, Deprecations, & Changes. *Kinsta Blog*.

Reichart, F. (2023). Why You Should Use Laravel in 2024. *Medium*.

Robinson, P. (2022). *Session & Cookie Management in PHP*. StatePress.

Rossi, G. (2024). *Security in PHP: Mitigating XSS, CSRF, SQLi*. SecureDev.

RunCloud. (2023). *PHP Security – best practices to secure your web app in 2025*.

Sánchez, M. (2023). *Content Security Policy in PHP Applications*. WebSec Press.

Smith, J. (2022). *PHP Unit Testing with PHPUnit*. DevBooks.

Snyk. (2024). Getting started with PHP static analysis in 2024. *Snyk Blog*.

Snyk. (2024). What you should know about PHP code security. *Snyk Blog*.

Stitcher.io. (2022, December 8). What's new in PHP 8.2.

Stone, B. (2024). *Logging in PHP: Monolog & Sentry*. LogPress.

Tatroe, K., MacIntyre, P., & Lerdorf, R. (2020, April). *Programming PHP: Creating Dynamic Web Pages* (4th ed.). O'Reilly Media. Reddit+3booksoncode.com+3Tech Hyme+3

VPSBG.eu (2022). PHP 8.2 announced – new and deprecated features.

Welling, L., & Thompson, L. (2021). *PHP & MySQL Web Development* (5th ed.). XYZ Publisher.

Zend. (2022). *PHP Security Best Practices for Critical Apps*. Zend Blog.

Zend. (2023). PHP 8.2: New Features, Changes, and Deprecations. *Zend Blog*.

Zhang, Y. (2023). *File Upload & Security in PHP*. SecureFile Publishing.

# PENGEMBANGAN
# WEB PHP
## (Hypertext Preprocessor) Language

**Dr. Budi Raharjo, S.Kom, M.Kom, MM.**

### BIODATA PENULIS

Dr. Budi Raharjo, S.Kom, M.Kom, MM lahir di Semarang, tanggal 22 Februari 1985. Beliau adalah Alumni dari Universitas Bina Nusantara (BINUS University) Jakarta dan juga alumni Universitas Kristen Satya wacana (UKSW) Salatiga. Dr. Budi Raharjo telah menjadi Dosen pada Universitas STEKOM pada mata kuliah Kepemimpinan (Leadership), mata kuliah Pengantar Akuntansi, Manajemen Proses, Manajemen Akuntansi dan Manajemen Resiko Bisnis. Selain sebagai dosen Universitas STEKOM, Dr. Budi Raharjo, M.Kom, MM juga mempunyai bisnis sendiri dalam bidang perhotelan dan juga sebagai wirausaha dalam bidang pemasok unggas (ayam) beku, ke berbagai kota besar, khususnya Jakarta dan sekitarnya.

Pengalaman beliau berwirausaha menjadi bekal utama dalam penulisan buku ajar yang diterbitkan oleh Yayasan Prima Agus Teknik (YPAT) Semarang. Oleh sebab itu bukunya berisi langkah langkah praktis yang mudah diikuti oleh para mahasiswa, saat mahasiswa mengikuti proses perkuliahan pada Universitas Sains dan Teknologi Komputer (Universitas STEKOM). Memiliki Jabatan Akademik Lektor 300 dan Menjabat sebagai Wakil Rektor 1 bidang (Akademik) di kampus Universitas STEKOM Semarang.

YAYASAN PRIMA AGUS TEKNIK

**PENERBIT :**
YAYASAN PRIMA AGUS TEKNIK
Jl. Majapahit No. 605 Semarang
Telp. (024) 6723456. Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

ISBN 978-634-7227-21-8 (PDF)
9 786347 227218