

# PHP Programming

## Fundamental

dan

# MySQL

## Fundamental

# Daftar Isi

<b>Daftar Isi .....</b>	<b>2</b>
<b>About the document .....</b>	<b>6</b>
<i>A. Copyright.....</i>	<i>6</i>
<i>B. Disclaimer .....</i>	<i>6</i>
<i>C. Version.....</i>	<i>6</i>
<i>D. Knowledge Required.....</i>	<i>6</i>
<i>E. Feedback and corrections .....</i>	<i>7</i>
<b>About the authors .....</b>	<b>1</b>
<i>Endy Muhardin.....</i>	<i>1</i>
<b>I. Pengantar .....</b>	<b>2</b>
<i>A. Konsep Multi Tier Application .....</i>	<i>2</i>
Client Side Presentation.....	2
Server Side Business Logic .....	2
Back End Storage .....	3
<i>B. Perbedaan Web Based Programming dengan System Programming.....</i>	<i>3</i>
<i>C. Teknologi Alternatif.....</i>	<i>3</i>
CGI Script.....	3
Proprietary API.....	4
ASP.....	4
JavaServlet/JSP.....	4
ColdFusion .....	4
PHP.....	4
<b>II. Hello World.....</b>	<b>5</b>
<i>A. Source code .....</i>	<i>5</i>
<i>B. Compile .....</i>	<i>5</i>
<i>C. Deploy .....</i>	<i>5</i>
<i>D. Error message .....</i>	<i>5</i>
<i>E. Output.....</i>	<i>5</i>
<b>III. Statement dan Comment.....</b>	<b>7</b>
<i>A. Statement .....</i>	<i>7</i>
<i>B. Expression .....</i>	<i>7</i>
<i>C. Comment.....</i>	<i>7</i>
<b>IV. Variabel dan Tipe Data.....</b>	<b>9</b>

A.	<i>Deklarasi variabel</i> .....	9
B.	<i>Inisialisasi variabel</i> .....	9
C.	<i>Tipe data</i> .....	10
D.	<i>Passing Variable</i> .....	11
	URL.....	11
	Form.....	12
	Cookie.....	13
<b>V.</b>	<b>Operators</b> .....	<b>14</b>
A.	<i>Arithmetic operator</i> .....	14
B.	<i>Relational operator</i> .....	15
C.	<i>Logical operator</i> .....	17
D.	<i>Assignment Operator</i> .....	18
	Operator +=.....	18
	Operator -=.....	18
	Operator *=.....	19
	Operator /=.....	19
	Operator %=.....	19
	Operator &=.....	19
	Operator  =.....	20
	Operator ^=.....	20
	Operator .=.....	20
	Operator ++.....	20
	Operator --.....	20
E.	<i>Operator lain-lain</i> .....	21
<b>VI.</b>	<b>Control Flow</b> .....	<b>23</b>
A.	<i>Percabangan</i> .....	23
	if – else.....	23
	switch – case.....	26
B.	<i>Perulangan</i> .....	28
	for.....	28
	while.....	29
C.	<i>Perpindahan</i> .....	30
	break.....	30
	continue.....	31
	return.....	31
	exit.....	32
<b>VII.</b>	<b>Array</b> .....	<b>34</b>
A.	<i>Konsep array</i> .....	34
B.	<i>Inisialisasi array</i> .....	34
C.	<i>Navigasi array</i> .....	35
D.	<i>Fungsi-fungsi array</i> .....	36
	sort.....	36
	asort dan ksort.....	36
<b>VIII.</b>	<b>Function</b> .....	<b>37</b>
A.	<i>Konsep function</i> .....	37

B.	<i>Return value dan parameter</i> .....	37
C.	<i>Deklarasi function</i> .....	37
D.	<i>Implementasi function</i> .....	38
<b>IX.</b>	<b>Variable Lifetime</b> .....	<b>40</b>
A.	<i>Scope</i> .....	40
Local	.....	40
Global	.....	40
B.	<i>Passing variabel</i> .....	40
By Value	.....	40
By Reference	.....	41
<b>X.</b>	<b>Class dan Objects</b> .....	<b>42</b>
A.	<i>Konsep class dan object</i> .....	42
B.	<i>Method dan properties</i> .....	42
<b>XI.</b>	<b>File Handling</b> .....	<b>45</b>
A.	<i>Menulis file</i> .....	46
Buka/buat file	.....	47
Masukkan data	.....	48
Tutup file	.....	49
B.	<i>Menghapus file</i> .....	49
C.	<i>Membuat direktori</i> .....	49
D.	<i>Navigasi direktori</i> .....	49
E.	<i>Menghapus direktori</i> .....	50
F.	<i>Membaca file</i> .....	50
<b>XII.</b>	<b>Database Access</b> .....	<b>52</b>
A.	<i>Sekilas SQL Query</i> .....	54
Query pengelolaan database	.....	54
Query satu tabel	.....	55
Query banyak tabel	.....	56
B.	<i>Membuat koneksi ke database</i> .....	56
Membuka sambungan dengan database server	.....	57
Memilih dan membuka database yang diinginkan	.....	57
Mengirim perintah untuk mengambil/mengubah/menghapus data	.....	57
C.	<i>Mengakses hasil query</i> .....	58
<b>XIII.</b>	<b>String Handling</b> .....	<b>61</b>
A.	<i>Sekilas Regular Expression</i> .....	61
B.	<i>Penggunaan Regular Expression</i> .....	62
<b>XIV.</b>	<b>Session</b> .....	<b>64</b>
A.	<i>No Session</i> .....	64
login.htm	.....	64
loginNoSession.php	.....	65
welcomeNoSession.php	.....	65

displayNoSession.php .....	65
Jalannya aplikasi : .....	66
<i>B. Konsep session</i> .....	68
<i>C. Implementasi : Cookie</i> .....	69
<i>D. Implementasi : Hidden Form</i> .....	72
<i>E. Implementasi : URL Embedding</i> .....	72
<i>F. Implementasi : Session API</i> .....	72

# About the document

## **A. Copyright**

This document is copyrighted (c) 2003 ArtiVisi Intermedia. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, with no Front-Cover Texts, and with no Back-Cover Texts. A copy of the license is available at <http://www.gnu.org/copyleft/fdl.html>

## **B. Disclaimer**

No liability for the contents of this documents can be accepted. Use the concepts, examples and other content at your own risk. As this is a new edition of this document, there may be errors and inaccuracies, that may of course be damaging to your system. Proceed with caution, and although this is highly unlikely, the author(s) do not take any responsibility for that.

You are strongly recommended to take a backup of your system before major installation and backups at regular intervals.

## **C. Version**

Revision : 1.0.1

Date : April 2003

The latest version of this document is available on <http://courseware.artivisi.com>

## **D. Knowledge Required**

This document assume the reader (you) to be fluent in HTML and Web Designing. Basic capabilities such as creating web page, uploading website to internet, and all related skills must be mastered.

## **E. Feedback and corrections**

If you have questions or comments about this document, please feel free to mail us at [support@artivisi.com](mailto:support@artivisi.com). We welcome any suggestions or criticisms.

Thanks.

## About the authors

### Endy Muhardin



Lahir di Jakarta, 8 Juni 1979. Lulus dari SMUN 81 Jakarta Timur pada tahun 1997 dan melanjutkan ke Teknik Industri STT Telkom. Di masa kuliah salah pergaulan sehingga terkena wabah Linux-PHP-MySQL. Selain itu, aktif dalam berbagai organisasi mahasiswa seperti Himpunan Mahasiswa Teknik Industri STT Telkom dan Senat Mahasiswa STT Telkom. Pada tahun 1999 ikut terlibat mendirikan start-up company Telematics Indonesia <<http://www.telematicsindonesia.com>>. Menyelesaikan kuliah pada tahun 2001, meninggalkan Telematics Indonesia yang sedang direstrukturisasi, dan hijrah ke Surabaya.

Saat ini bekerja sebagai trainer di Aptech Computer Education dengan spesialisasi Java, Linux, dan PHP. Selain itu juga aktif sebagai dosen luar di STIKOM Surabaya untuk matakuliah Pemrograman Berorientasi Objek dengan Java dan Aplikasi Internet II.

Tulisan yang dihasilkan antara lain, Modul Pelatihan PHP Programming, Modul Praktikum Java Fundamental, Modul Praktikum Java Distributed, dan berbagai materi lain yang dapat didownload di <http://courseware.artivisi.com>

Aktivitas utama saat ini adalah menginkubasi perusahaan start-up ArtiVisi Intermedia <<http://www.artivisi.com>>.

Penulis dapat dihubungi melalui email [endy@artivisi.com](mailto:endy@artivisi.com)



# I. Pengantar

## A. Konsep Multi Tier Application

Aplikasi internet berbasis web, yang akan kita kembangkan dalam pelatihan ini, adalah salah satu penerapan multi tier application.

Multi tier application adalah aplikasi yang dibagi menjadi beberapa bagian yang menjalankan fungsi masing-masing. Secara umum, ada tiga bagian utama dari multi tier application:

- ❑ Client side presentation
- ❑ Server side business logic
- ❑ Backend storage

### **Client Side Presentation**

Client side presentation mengatur bagaimana aplikasi berinteraksi dengan user. Yang dimaksud dengan interaksi antara lain adalah: bagaimana data ditampilkan, bagaimana fungsi dan fitur aplikasi ditampilkan.

Dalam aplikasi berbasis web, client side presentation dibuat dengan bahasa HTML, CSS, dan JavaScript. Beberapa tool yang digunakan untuk membuat client side presentation diantaranya Microsoft Frontpage, Macromedia Dreamweaver, dan sebagainya.

Client side presentation berbasis web contohnya adalah tampilan aplikasi email yang kita buka dengan browser.

### **Server Side Business Logic**

Server side business logic, sering disebut juga middle tier, adalah bagian yang bertanggung jawab atas cara kerja aplikasi. Di dalamnya kita mengatur bagaimana fungsi dan fitur aplikasi dapat bekerja dengan baik.

Dalam aplikasi berbasis web, ada beberapa alternatif yang dapat digunakan, ditentukan oleh jenis platform yang digunakan. Alternatif ini akan dijelaskan lebih detail pada bagian selanjutnya.

### **Back End Storage**

Bagian ini mengatur cara penyimpanan data. Penyimpanan data merupakan materi yang cukup kompleks dalam pembangunan aplikasi. Karena kecepatan, keutuhan, dan keamanan data merupakan faktor kritis dalam aplikasi.

Ada banyak solusi database yang tersedia di pasaran. Pada umumnya, database yang digunakan bertipe relasional (Relational Database Management System – RDBMS). Manajemen data dilakukan dengan bahasa SQL (Standard Query Language).

## **B. Perbedaan Web Based Programming dengan System Programming**

Pembuatan aplikasi berbasis web berbeda dengan pembuatan aplikasi berbasis windows (visual programming), misalnya Visual Basic, Delphi, atau KDevelop. Dalam visual programming, kita meningkatkan kecepatan dan kinerja aplikasi dengan mengoptimasi penggunaan memori, manajemen proses, dan pengaturan Input-Output. Pada pemrograman berbasis web, faktor yang menentukan kinerja aplikasi adalah kecepatan akses database dan kecepatan akses jaringan dan internet.

Perbedaan kedua, adalah cara aplikasi berjalan. Pada aplikasi visual, aplikasi dibangun dengan menggunakan tool tertentu, kemudian dikompilasi. Hasilnya dapat langsung digunakan dalam komputer. Aplikasi berbasis web tidak dapat dijalankan langsung di komputer. Untuk menjalankannya, dibutuhkan engine tertentu, dalam hal ini web server.

## **C. Teknologi Alternatif**

Teknologi server side yang akan kita pelajari pada modul ini adalah PHP. Walaupun demikian, perlu diketahui teknologi alternatif yang dapat menjadi bahan pertimbangan.

### **CGI Script**

CGI Script dapat dibuat dengan berbagai bahasa pemrograman, misalnya Perl atau Python. Teknologi ini pernah sangat populer di masa awal berkembangnya web based application. Tetapi saat ini banyak

ditinggalkan orang karena tidak efisien, tidak fleksibel, dan keamanannya rendah.

### **Proprietary API**

Teknologi ini adalah bahasa pemrograman yang disediakan masing-masing webserver, misalnya ISAPI atau NSAPI. Teknologi ini memiliki kelemahan, yaitu hanya dapat berjalan di webserver tertentu, sehingga mengurangi portabilitas.

### **ASP**

Merupakan solusi server side programming dari Microsoft. Teknologi ini banyak digunakan oleh para programmer yang berlatar belakang Visual Basic. Database pasangannya adalah MS SQL Server. Operating system pasangannya adalah Window 2000 Server yang menjalankan webserver Microsoft IIS.

### **JavaServlet/JSP**

Teknologi server side Java. Mempunyai banyak keunggulan dan kemudahan pemrograman. Tetapi karena murni berorientasi objek, banyak programmer pemula kesulitan menggunakannya. Selain itu, Java juga adalah bahasa pemrograman yang relatif rumit bagi pemula. Keunggulannya, sangat portabel. Dapat dipindahkan dengan mudah dari Windows ke Unix dan sebaliknya.

### **ColdFusion**

Bahasa pemrograman ini mirip dengan HTML, menggunakan tag untuk membentuk blok-blok programnya.

### **PHP**

Banyak digunakan oleh programmer berlatar belakang C/C++ karena kemiripan syntaxnya. Open source, karenanya gratis dan bebas. Database pasangannya biasanya MySQL, dijalankan bersama webserver Apache di atas operating system Linux. Semuanya gratis dan bebas.

## II. Hello World

Dalam bagian ini, kita akan membuat kode program sederhana untuk menunjukkan bagaimana aplikasi PHP bekerja.

Kita membutuhkan satu file yang akan kita namakan `welcome.php`

### A. Source code

File `welcome.php` akan berisi tiga baris kode sebagai berikut :

```
<?
  echo("Hallo user .. !");
?>
```

### B. Compile

Kode program pada umumnya akan mengalami proses kompilasi setelah source code selesai dibuat. Tetapi karena PHP adalah interpreted language, kita tidak perlu mengkompilasi kode ini.

### C. Deploy

Untuk mendeploy, copy file `welcome.php` ke dalam folder percobaan yang telah disiapkan pada bagian instalasi di atas.

### D. Error message

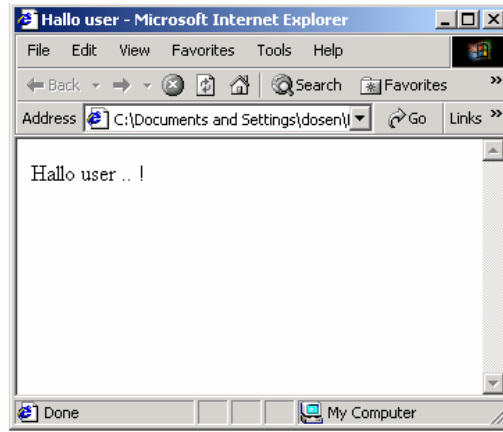
Kode di atas akan mengalami error, karena kita terlalu banyak menulis tanda `)`. Pesan error akan muncul di layar, menunjukkan lokasi error pada kode, dan tipe errornya.

### E. Output

Perbaiki kode `welcome.php` sehingga menjadi seperti berikut ini:

```
<?
echo("Hallo user .. !");
?>
```

kode akan dapat dijalankan dengan baik dan menghasilkan output sebagai berikut :



## III. Statement dan Comment

### A. Statement

Statement adalah satuan perintah dalam PHP. Statement harus diakhiri dengan tanda semicolon/titik-koma (;).

Contoh statement :

```
echo("hello user ... !");
```

Contoh lainnya :

```
echo("4 + 5 = " . 4+5);
```

### B. Expression

Expression adalah satu bagian kecil kode yang akan dihitung hasilnya oleh php. Contoh expression :

```
4 + 5
```

Penggunaan expression :

```
echo("4 + 5 = " . 4+5);
```

### C. Comment

Comment adalah bagian dari kode yang tidak dieksekusi/dijalankan. Comment dibuat untuk memperjelas atau memberi keterangan pada kode program.

Ada dua cara menulis comment : comment satu baris dan comment banyak baris.

Comment satu baris dibuat dengan menggunakan tanda //. Semua statement yang ada di kanan // tidak dijalankan oleh interpreter. Contoh penggunaan:

```
echo("4 + 5 = " . 4+5); // menampilkan hasil 4 + 5
```

Comment banyak baris dibuat dengan menggunakan pasangan /\* dan \*/. Semua tulisan yang dibuat di antara tanda tersebut tidak akan dieksekusi oleh interpreter. Contoh penggunaan :

```
/*  
kode ini akan menampilkan hasil dari  
4 + 5  
*/  
echo("4 + 5 = " . 4+5);
```

## IV. Variabel dan Tipe Data

Variabel digunakan sebagai tempat penyimpanan data sementara. Data yang disimpan dalam variabel akan hilang setelah program selesai dieksekusi. Untuk penyimpanan data yang permanen, kita dapat menyimpan data di database atau di disk. Silahkan mengacu pada Akses Database untuk mendalami penggunaan database, dan Akses File dan Folder untuk penyimpanan data di filesystem.

Variabel di PHP diawali dengan tanda \$.

Untuk dapat menggunakan variabel, ada dua langkah yang harus dilakukan, **deklarasi** dan **inisialisasi**.

### A. Deklarasi variabel

Deklarasi variabel bisa disebut juga memperkenalkan atau mendaftarkan variabel ke dalam program.

Dalam php, deklarasi variabel seringkali digabung dengan inisialisasi.

Variabel dalam PHP dinyatakan dengan awalan \$.

Contoh :

```
$namaPembeli  
$jumlahBarang  
$harga
```

Ada beberapa aturan yang diikuti berkenaan dengan penggunaan nama variabel. Aturan pemberian nama variabel :

- Dimulai dengan tanda \$
- Karakter pertama harus huruf atau garis bawah ( \_ )
- Karakter berikutnya boleh huruf, angka, atau garis bawah.

### B. Inisialisasi variabel

Inisialisasi variabel adalah mengisi nilai untuk pertama kalinya ke dalam variabel. Contoh inisialisasi :



```
$namaDepan = "Endy";  
$namaBelakang = "Muhardin";  
$jumlahBarang = 3;  
$harga = 1000;
```

### C. Tipe data

Dalam bahasa pemrograman yang lain, ada bermacam-macam tipe data, misalnya integer(bilangan bulat), float(bilangan pecahan), char(karakter angka dan huruf), string(kumpulan huruf atau kata), dan berbagai tipe lainnya.

PHP mengenal dua tipe data sederhana; numerik dan literal. Ditambah dengan dua tipe data yang tidak sederhana, yaitu array dan object.

Tipe Numerik dapat menyimpan bilangan bulat. PHP mampu menyimpan data bilangan bulat dengan jangkauan dari -2 milyar sampai +2 milyar. Contoh bilangan bulat: 3, 7, 20.

Selain itu, tipe numerik juga digunakan untuk menyimpan bilangan pecahan

Tipe literal digunakan untuk menyimpan data berupa kumpulan huruf, kata, dan angka.

Tipe boolean, yang dikenal dalam bahasa program yang lainnya, tidak ada dalam PHP. Untuk menguji benar salah (true false), kita menggunakan tipe data yang tersedia. FALSE dapat digantikan oleh integer 0, double 0.0 atau string kosong, yaitu "". Selain nilai itu, semua dianggap TRUE.

Variabel dapat digunakan untuk menyimpan berbagai jenis data. Misalnya data numerik yang dapat dioperasikan secara matematika. Contoh :

```
$jumlahBarang = 3;  
$harga = 1000;  
$pembayaran = $jumlahBarang * $harga;
```

pada contoh di atas, variabel pembayaran akan menyimpan nilai 3000.

Sedangkan data non numerik (disebut juga data literal) tidak dapat dioperasikan secara matematika. Contoh :

```
$nama = $namaDepan + $namaBelakang;
```

variabel nama akan menyimpan gabungan dari dua variabel, yaitu "Endy Muhardin".

Secara umum, data literal ditandai dengan pasangan " dan ". Data numerik tidak dikelilingi oleh " dan ". Tetapi biasanya PHP akan secara otomatis mengubah tipe data sesuai kebutuhan. Contoh :

```
$jalan = "Gubeng Kertajaya";  
$noRumah = 29;  
$blok = 4c;  
$jumlahPeghuni = 3;
```

```
$alamat = $jalan + $noRumah;  
$hasil = $noRumah + $jumlahPeghuni;  
$hasilAneh = $blok + $noRumah;
```

Pada sampel kode di atas, variabel alamat akan menyimpan nilai Gubeng Kertajaya 29. PHP secara otomatis mengubah tipe data variabel noRumah (numerik) menjadi literal. Variabel alamat akan bertipe literal.

Variabel hasil akan menyimpan nilai 32, yaitu penjumlahan dari 29 dan 3.

Perhatikan, konversi otomatis ini kadang berjalan secara tidak semestinya. Ini dapat dilihat dari variabel hasilAneh yang akan menyimpan nilai 7. PHP mengambil nilai numerik dari variabel blok, yaitu 4, kemudian menambahkannya dengan isi variabel jumlahPeghuni. Hasil akhirnya adalah 4 + 3, yaitu 7.

#### D. Passing Variable

Variabel dapat di-passing atau diteruskan ke halaman web berikutnya yang diakses user. Ada beberapa teknik untuk meneruskan variabel, diantaranya :

- Melalui URL
- Melalui Form
- Melalui Cookie

##### URL

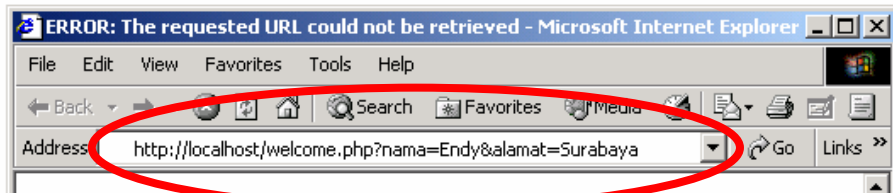
Variabel diteruskan melalui URL dengan format sbb

```
[alamat web]?var1=nilai1&var2=nilai2
```

Misalnya, untuk memberikan variabel \$nama berisi "Endy" dan \$alamat = "Surabaya" ke welcome.php, kita akan menulis :

```
welcome.php?nama=Endy&alamat=Surabaya
```

Di kolom address pada explorer. Untuk lebih jelasnya, perhatikan gambar berikut.



Variabel ini dapat diakses di script welcome.php dengan cara sebagai berikut :

```
<?
    echo("Variabel \$user berisi : $user");
    echo("Variabel \$alamat berisi : $alamat");
?>
```

### Form

Cara lain untuk mengirim kedua variabel tersebut adalah dengan menggunakan form dengan kode sebagai berikut :

```
<html>

<head>
<title>Passing Variable</title>
</head>

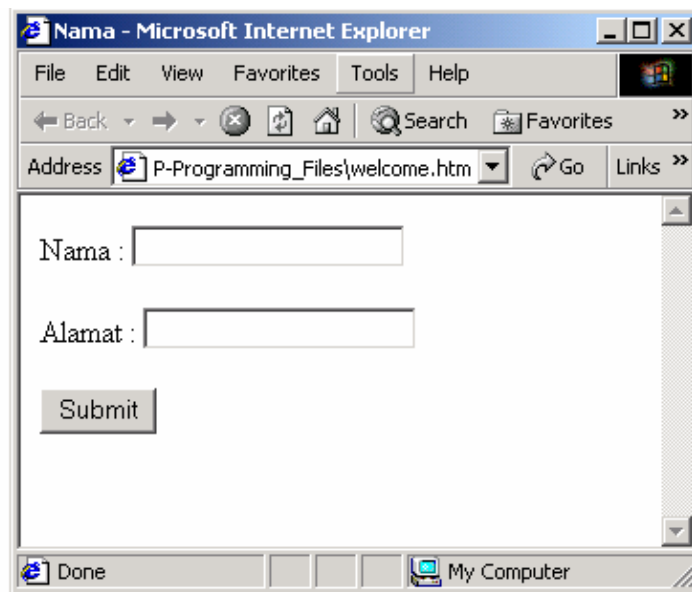
<body>

<form method="POST" action="welcome.php">
    <p>Nama : <input type="text" name="nama"></p>
    <p>Alamat : <input type="text" name="alamat"></p>
    <p><input type="submit" value="Submit" name="B1"></p>
</form>

</body>

</html>
```

dan tampilan sebagai berikut :



form tersebut akan diproses oleh file welcome.php

### **Cookie**

Penggunaan cookie akan dibahas pada bagian tentang session.

## V. Operators

Operator digunakan untuk memanipulasi nilai suatu variabel. Variabel yang nilainya dimodifikasi oleh operator disebut operand. Contoh penggunaan operator misalnya  $13 - 3$ . 13 dan 3 adalah operand. Tanda "-" disebut operator.

Untuk kemudahan penjelasan, operator diklasifikasikan menjadi :

- ❑ Arithmetic Operator
- ❑ Assignment Operator
- ❑ Comparison Operator
- ❑ Logical Operator
- ❑ Lain-lain

### A. Arithmetic operator

Arithmetic Operator digunakan untuk melakukan perhitungan matematika. Misalnya

```
$a = 5 + 3;
```

Operator "+" berfungsi untuk menambahkan kedua operand (5 dan 3).

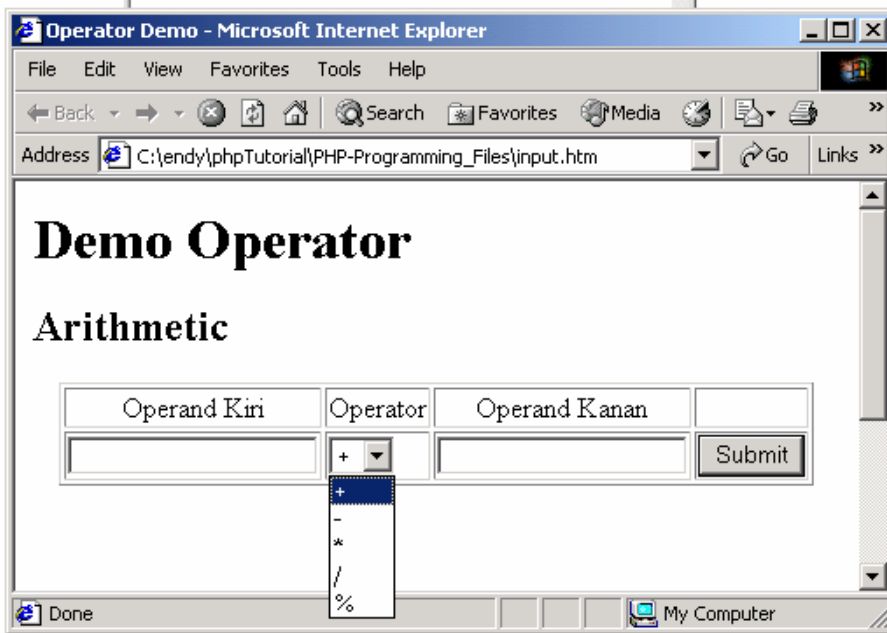
Ada beberapa arithmetic operator, yaitu :

- ❑ + : penjumlahan
- ❑ - : pengurangan
- ❑ \* : perkalian
- ❑ / : pembagian
- ❑ % : nilai sisa pembagian

Contoh penggunaan :

Buatlah dua file berikut :

operator.htm



Komponen	Nama Variabel
TextField kiri	operand1
TextField kanan	operand2
Operator	op
Action	hasilArithmetic.php

operator.php

```
<?
    $perhitungan = $operand1.$operator.$operand2;
    eval("\$hasil = $perhitungan;");
    echo("Hasil Perhitungan : ");
    echo("<b>");
    echo($hasil);
    echo("</b>");
?>
```

## B. Relational operator

Relational operator digunakan untuk membandingkan nilai dari dua operand. Hasil perbandingan dinyatakan dalam nilai boolean. TRUE berarti benar, dan FALSE berarti salah.

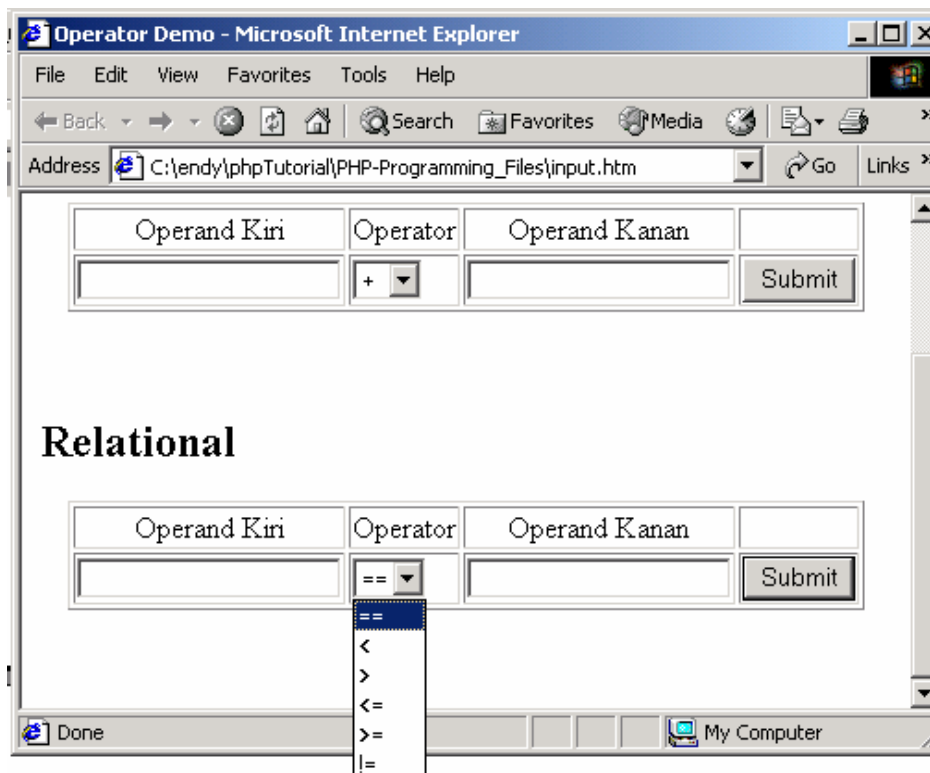
Beberapa jenis relational operator :

- ❑ == : memeriksa apakah operand kanan bernilai sama dengan operand kiri
- ❑ > : memeriksa apakah operand kiri bernilai lebih besar daripada operand kanan

- < : memeriksa apakah operand kiri bernilai lebih kecil dengan operand kanan
- >= : memeriksa apakah operand kiri bernilai lebih besar atau sama dengan operand kanan
- <= : memeriksa apakah operand kiri bernilai lebih kecil atau sama dengan operand kanan
- != : memeriksa apakah operand kanan tidak bernilai sama dengan operand kiri

Untuk mengetahui cara penggunaan relational operator, buatlah contoh seperti petunjuk berikut.

Tambahkan operator.htm sehingga menjadi seperti gambar di bawah:



Komponen	Nama Variabel
TextField kiri	operand1
TextField kanan	operand2
Operator	Op
Action	hasilRelational.php

Buat file operator.php seperti ini :

```

<?
    $perhitungan = $operand1.$op.$operand2;
    $hasil = eval($perhitungan);

    echo("Hasil Perhitungan : ");
    echo("<b>");
    echo($hasil);
    echo("</b>");
?>

```

**C. Logical operator**

Logical Operator digunakan untuk membandingkan dua nilai variabel yang bertipe boolean. Hasil yang didapat dari penggunaan logical operator adalah boolean.

Tabel logika berikut digunakan sebagai pedoman perhitungan

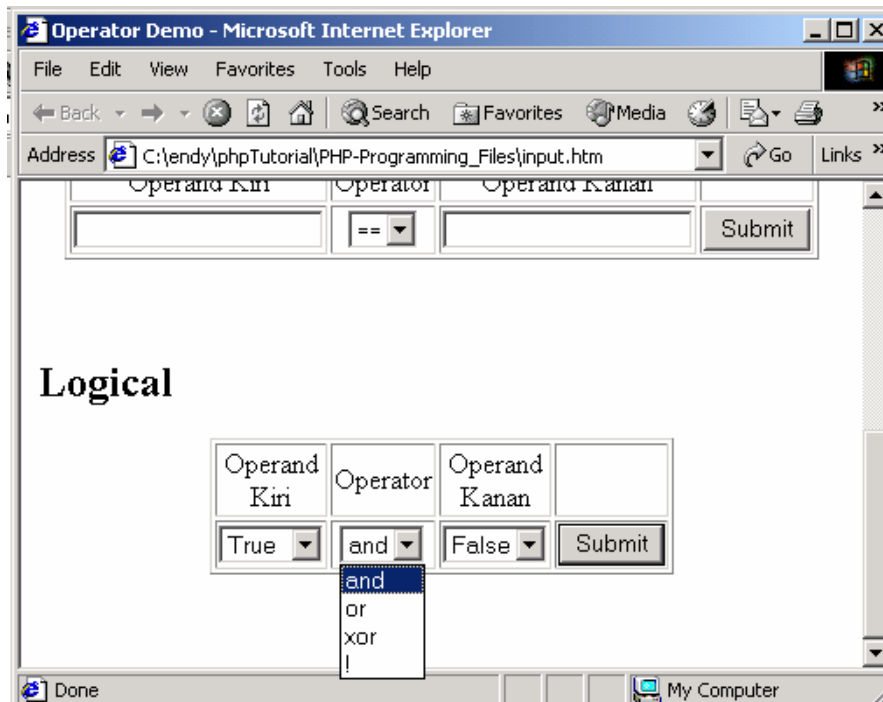
Operand kiri	Operator	Nama	Operand kanan	Hasil
TRUE	&&	And	TRUE	TRUE
TRUE	&&	And	FALSE	FALSE
FALSE	&&	And	TRUE	FALSE
FALSE	&&	And	FALSE	FALSE
TRUE		Or	TRUE	TRUE
TRUE		Or	FALSE	TRUE
FALSE		Or	TRUE	TRUE
FALSE		Or	FALSE	FALSE
TRUE	Xor	Exclusive Or	TRUE	FALSE
TRUE	Xor	Exclusive Or	FALSE	TRUE
FALSE	Xor	Exclusive Or	TRUE	TRUE
FALSE	Xor	Exclusive Or	FALSE	FALSE
!	Not		TRUE	FALSE
!	Not		FALSE	TRUE

Untuk melihat pemakaian logical operator, ikuti petunjuk di bawah ini.

File operator.php masih sama seperti di atas.

File operator.htm menjadi seperti di bawah ini :





#### D. Assignment Operator

Assignment operator digunakan untuk memberi/mengisi nilai ke dalam variabel tertentu. Contoh sederhana :

```
$nama = "endy";
```

Pada contoh di atas, operator "=" digunakan untuk mengisi nilai "endy" ke dalam variabel nama.

Selain operator "=", ada beberapa assignment operator yang lainnya, seperti dapat dilihat pada penjelasan berikut :

##### **Operator +=**

Penjelasan :

Menambahkan nilai pada variabel

Contoh :

```
$a += 3;
```

sama dengan

```
$a = $a + 3;
```

##### **Operator -=**

Penjelasan :

Mengurangi nilai pada variabel

Contoh :

```
$a -= 3;
```

sama dengan

```
$a = $a - 3;
```

### **Operator \*=**

Penjelasan :

Mengalikan variabel dengan bilangan tertentu

Contoh :

```
$a *= 3;
```

sama dengan

```
$a = $a * 3;
```

### **Operator /=**

Penjelasan :

Membagi variabel dengan bilangan tertentu

Contoh :

```
$a /= 3;
```

sama dengan

```
$a = $a / 3;
```

### **Operator %=**

Penjelasan :

Mencari sisa hasil bagi variabel dengan bilangan tertentu

Contoh :

```
$a %= 3;
```

sama dengan

```
$a = $a % 3;
```

### **Operator &=**

Penjelasan :

Melakukan operasi logical AND pada variabel

Contoh :

```
$a &= TRUE;
```

sama dengan

```
$a = $a & TRUE;
```

### **Operator |=**

Penjelasan :

Melakukan operasi logical OR pada variabel

Contoh :

```
$a |= FALSE;
```

sama dengan

```
$a = $a | FALSE;
```

### **Operator ^=**

Penjelasan :

Melakukan operasi bitwise xor pada variabel

Contoh :

```
$a ^= 3;
```

sama dengan

```
$a = $a ^ 3;
```

### **Operator .=**

Penjelasan :

Menambahkan String pada variabel

Contoh :

```
$a .= "rudi";
```

sama dengan

```
$a = $a . "rudi";
```

### **Operator ++**

Penjelasan :

Menambahkan nilai satu pada variabel

Contoh :

```
$a ++ ;
```

sama dengan

```
$a = $a + 1;
```

### **Operator --**

Penjelasan :

Mengurangi nilai satu pada variabel

Contoh :

```
$a -- ;
```

sama dengan

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

## E. Operator lain-lain

- Operator penggabung String

Pada PHP, string digabungkan dengan operator . (titik).

Contoh penggunaan :

```
$string1 = "Hello";  
$string2 = "World";  
echo($string1." ".$string2);
```

akan menampilkan :

```
Hello World
```

- Operator percabangan

Percabangan pada umumnya dilakukan dengan struktur if-else, seperti pada contoh berikut:

```
if($user == "endy"){  
    echo("Welcome Endy");  
}else{  
    echo("Wrong username");  
}
```

Hal yang sama dapat dilakukan dengan cara :

```
echo($user == endy ? "Welcome Endy" : "Wrong username");
```

Perhatikan tanda ? dan :

PHP memeriksa apakah pernyataan di sebelah kiri "?" benar atau salah. Apabila benar, pernyataan di sebelah kiri tanda ":" dieksekusi. Bila salah, pernyataan di sebelah kanan ":" dieksekusi.

Lebih jelas tentang percabangan dapat dipelajari dalam bagian Control Flow.

- Operator Error Suppression

PHP menampilkan pesan error apabila built-in function (function yang disediakan PHP) mengalami error. Misalnya tidak bisa membuka file, tidak bisa mengakses database, dan lainnya.

Pada saat pembuatan aplikasi, pesan error ini sangat membantu dalam menyelesaikan dan memperbaiki kesalahan pemrograman. Tetapi, pada saat aplikasi selesai dibuat dan digunakan secara umum, pesan error ini akan mengganggu pengguna.

Untuk mematikan pesan error tersebut, kita menggunakan operator @.

Contoh penggunaan :

```
@chdir("temp");
```

Pada kondisi normal, function chdir akan menimbulkan pesan error apabila direktori temp tidak ditemukan atau tidak dapat diakses. Dengan menggunakan operator @, PHP akan "diam saja" apabila direktori temp tidak ditemukan atau tidak dapat diakses.

## VI. Control Flow

Control flow dalam bahasa Indonesia dapat diartikan sebagai aliran kendali. Maksud sebenarnya dari control flow adalah bagaimana urutan eksekusi perintah di dalam program.

Misalnya, dalam function :

```
function testFlow()
{
    int a = 5;
    echo(a);
}
```

Perintah pertama yang dijalankan adalah mengisi nilai 5 ke dalam variabel a.

Perintah kedua yang dijalankan adalah menampilkan nilai yang tersimpan dalam variabel a (dalam hal ini 5) ke browser.

Control flow di atas merupakan sebuah contoh sederhana. Beberapa control flow yang tersedia dalam PHP :

- Percabangan (branching)
- Perulangan (looping)
- Perpindahan (jumping)

### A. Percabangan

Percabangan, atau sering disebut juga dengan istilah decision-making, memungkinkan aplikasi untuk memeriksa isi suatu variabel atau hasil perhitungan ekspresi dan mengambil tindakan yang sesuai. Ada dua jenis percabangan, dipilih berdasarkan kriteria pemeriksaan dan jumlah pilihan yang tersedia.

#### **if – else**

Konstruksi if-else dapat dijelaskan sebagai berikut :

```
if(condition)
{
    // statement 1 goes here
}
else
{
    // statemant 2 goes here
}
// statement 3 goes here
```

Aliran program :

1. Condition akan diperiksa
2. Bila bernilai true, statement 1 akan dijalankan
3. Bila bernilai false, statement 2 akan dijalankan
4. Statement 3 dijalankan

### **if - elseif - else**

Untuk pilihan yang lebih dari dua, PHP menyediakan konstruksi if-elseif-else.

```
if(condition1)
{
    // statement 1
}
elseif(condition2)
{
    // statement 2
}
else
{
    // statement 3
}
// statement 4
```

Aliran program :

Ada 3 kemungkinan aliran program :

- Apabila condition 1 bernilai true :
  1. Statement 1 dijalankan
  2. Statement 4 dijalankan
- Apabila condition 1 bernilai false, dan condition 2 bernilai true :
  1. Statement 2 dijalankan
  2. Statement 4 dijalankan

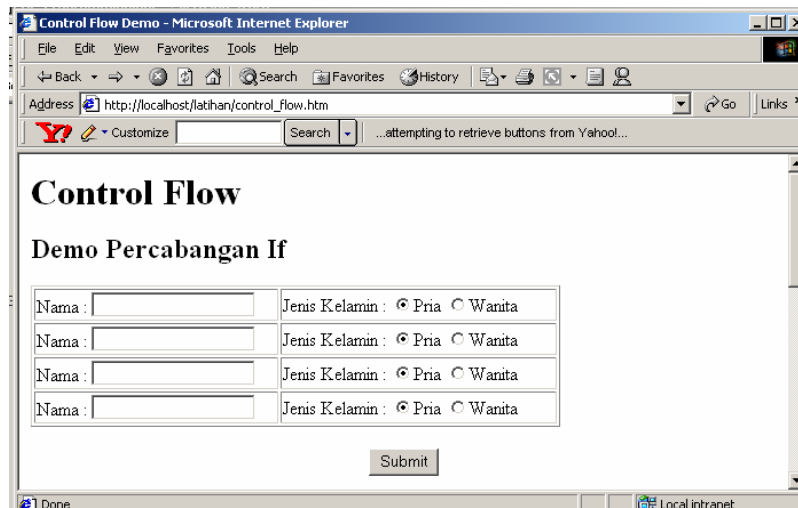
□ Apabila condition 1 dan condition 2 bernilai false :

1. Statement 3 dijalankan

2. Statement 4 dijalankan

Contoh penggunaan if-else dapat dilihat dengan mengikuti contoh di bawah. Buatlah dua file, control\_flow.htm dan ifDemo.php.

control\_flow.htm mempunyai tampilan sebagai berikut :



ifDemo.php berisi listing kode sebagai berikut :



```

<?
// ifDemo.php
// digunakan bersama control_flow.htm
$kelabu = "#303030";
$putih = "#FFFFFF";

if($kelamin1 == "Pria"){
    $latar1 = $kelabu;
}else{
    $latar1 = $putih;
}

if($kelamin2 == "Pria"){
    $latar2 = $kelabu;
}else{
    $latar2 = $putih;
}

if($kelamin3 == "Pria"){
    $latar3 = $kelabu;
}else{
    $latar3 = $putih;
}

if($kelamin4 == "Pria"){
    $latar4 = $kelabu;
}else{
    $latar4 = $putih;
}

$output = "
<table border=1>
    <tr>
        <td background=$latar1>$nama1</td>
    </tr>
    <tr>
        <td background=$latar2>$nama2</td>
    </tr>
    <tr>
        <td background=$latar3>$nama3</td>
    </tr>
    <tr>
        <td background=$latar4>$nama4</td>
    </tr>
</table>
";
echo($output);
?>

```

### switch – case

konstruksi switch dapat dijelaskan sebagai berikut :

```

switch(a){
  case 1;
    // statement 1 goes here
    break;
  case 2;
    // statement 2 goes here
    break;
  case 3;
    // statement 3 goes here
    break;
  default;
    // statement 4 goes here
    break;
}
// statement 5 goes here

```

Aliran program :

1. Variabel a diperiksa
2. Statement dieksekusi
  - a) Apabila a == 1, statement 1 dijalankan
  - b) Apabila a == 2, statement 2 dijalankan
  - c) Apabila a == 3, statement 3 dijalankan
  - d) Apabila a tidak memenuhi 2a - 2c, statement 4 dijalankan
3. Statement 5 dijalankan

Keyword break memegang peranan penting di sini. Fungsinya adalah mencegah fall-through, bandingkan dengan program berikut(break di baris ke 5 dihilangkan)

```

switch(a){
  case 1;
    // statement 1 goes here
  case 2;
    // statement 2 goes here
    break;
  case 3;
    // statement 3 goes here
    break;
  default;
    // statement 4 goes here
    break;
}

```

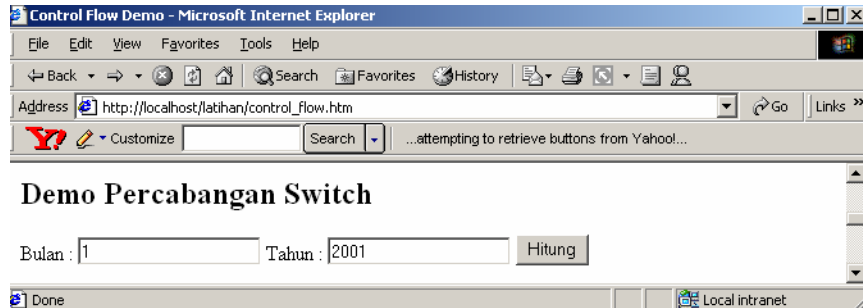
Aliran program :

1. variabel a diperiksa
  - 2a. Apabila a == 1, statement 1 dijalankan, kemudian menjalankan statement 2.**
  - 2b. Apabila a == 2, statement 2 dijalankan

- 2c. Apabila a == 3, statement 3 dijalankan
- 2d. Apabila a tidak memenuhi 2a - 2c, statement 4 dijalankan
- 3. Statement 5 dijalankan

Perbedaan ada pada langkah 2a. Bandingkan dengan listing pertama.

Untuk melihat contoh penggunaan switch – case, tambahkan tampilan pada control\_flow.htm menjadi seperti gambar di bawah.



Buat file switchDemo.php sebagai berikut :

```
<?
switch($bulan){
    case 1 :
    case 3 :
    case 5 :
    case 7 :
    case 8 :
    case 10 :
    case 12 :
        $hari = 31;
        break;
    case 4 :
    case 6 :
    case 8 :
    case 11 :
        $hari = 30;
        break;
    case 2 :
        if(($tahun%4) == 0){
            $hari = 29;
        }else{
            $hari = 28;
        }
    }
}

echo("<h2>Jumlah hari pada bulan $bulan tahun $tahun =
$hari hari</h2>");
?>
```

## B. Perulangan

### for

Looping dengan for disebut juga determinate loop, artinya looping yang jumlah pengulangannya (iterasi) telah ditentukan di awal looping.

Ada beberapa bagian penting dari for loop:

- ❑ Initialization expression
- ❑ Stop condition
- ❑ Iterative expression
- ❑ Loop body

Initialization Expression dijalankan satu kali, pada saat looping dimulai. Biasanya bagian ini digunakan untuk menginisialisasi counter(penghitung).

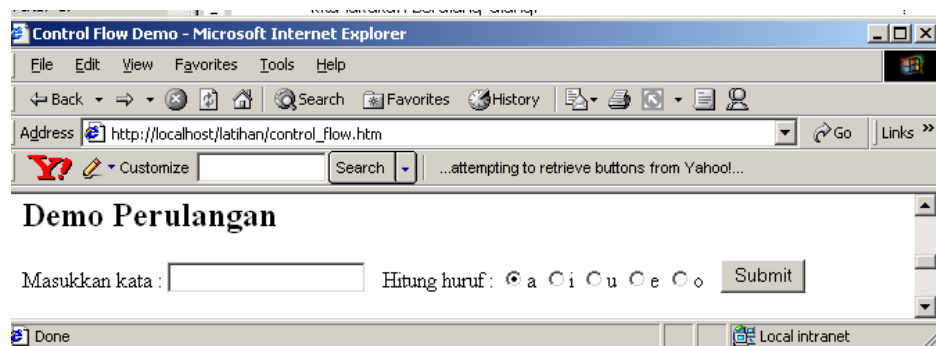
Stop condition diperiksa nilainya sebelum setiap iterasi dieksekusi. Apabila condition bernilai false, iterasi dihentikan.

Iterative expression dilakukan setelah iterasi dieksekusi. Bagian ini biasanya digunakan untuk menambah nilai counter.

Loop body dieksekusi sekali setiap iterasi, merupakan perintah yang ingin kita lakukan berulang-ulang.

Contoh kode di atas akan menampilkan tulisan Hello World di browser sebanyak 10 kali.

Untuk melihat contoh penggunaan for, tambahkan tampilan pada control\_flow.htm menjadi seperti gambar di bawah.



Pasang baris kode berikut pada forDemo.php

```
<?
$jumlah = 0;
for($i=0; $i<strlen($kata); $i++){
    if(substr($kata, $i, 1) == $huruf){
        $jumlah ++;
    }
}
?>
```

### **while**

while loop juga dikenal dengan istilah indeterminate loop, artinya jumlah loopingnya tidak ditentukan pada awal looping. while loop lebih sederhana daripada for loop, karena cuma memiliki dua bagian:

Stop Condition

Loop body

Stop condition diperiksa sebelum tiap iterasi dilaksanakan. Selama stop condition bernilai true, perintah dalam loop body akan dilakukan berulang-ulang. Iterasi akan dihentikan apabila stop condition bernilai false.

Sama seperti pada for loop, loop body dilaksanakan satu kali setiap iterasi.

Loop di atas akan berjalan terus tanpa henti, karena tidak ada perintah yang mengubah nilai stop condition.

do-while loop merupakan modifikasi dari while loop. Bentuknya dapat dilihat pada sampel kode berikut:

```
do{
    // some statement
}
while (a == true)
```

untuk melihat contoh penggunaan while, gunakan form yang sama dengan forDemo.

Tambahkan baris kode berikut pada whileDemo.php

```
<?
$jumlah = 0;
$i = 0;
while($i<strlen($kata)){
    if(substr($kata, $i, 1) == $huruf){
        $jumlah ++;
        $i++;
    }
}
?>
```

## C. Perpindahan

### **break**

Break digunakan dalam looping untuk menghentikan suatu loop. Untuk lebih jelasnya, perhatikan kode berikut :

```

<?
// melakukan break pada $i == 2
for($i = 0; $i<5; $i++){
    if($i == 2){
        break;
    }
    echo("Nilai i : $i <br>");
}
echo("Loop Selesai");
?>

```

kode tersebut akan melakukan break pada saat i bernilai 2, sehingga akan menghasilkan output sebagai berikut :

```

Nilai i : 0
Nilai i : 1
Loop Selesai

```

### **continue**

continue berfungsi untuk melewati satu iterasi/putaran dalam rangkaian loop. Untuk lebih jelasnya, kode di atas akan kita modifikasi.

```

<?
// melakukan continue pada $i == 2
for($i = 0; $i<5; $i++){
    if($i == 2){
        continue;
    }
    echo("Nilai i : $i <br>");
}
echo("Loop Selesai");
?>

```

kode tersebut akan melakukan break pada saat i bernilai 2, sehingga akan menghasilkan output sebagai berikut :

```

Nilai i : 0
Nilai i : 1
Nilai i : 3
Nilai i : 4
Loop Selesai

```

### **return**

perintah return berfungsi untuk memerintahkan kode untuk keluar dari fungsi. Fungsi akan kita pelajari dengan lebih mendalam pada bagian selanjutnya. Untuk sementara, fungsi cuma akan digunakan untuk menjelaskan return. Perhatikan kode di bawah :

```

<?
function testReturn(){
    for($i = 0; $i<5; $i++){
        // melakukan return pada $i == 2
        if($i == 2){
            return;
        }
        echo("Nilai i : $i <br>");
    }
    echo("Loop Selesai");
}

// jalankan function
testReturn();
echo("Function selesai");
?>

```

outputnya adalah :

```

Nilai i : 0
Nilai i : 0
Function Selesai

```

Perhatikan bahwa Loop Selesai tidak dijalankan. Ini menunjukkan bahwa setelah return dieksekusi, program langsung keluar dari function dan mengeksekusi perintah setelah function, yaitu

```

echo("Function Selesai");

```

### exit

exit digunakan untuk menghentikan keseluruhan script php. Untuk lebih jelasnya, kita akan modifikasi kode di atas menjadi seperti berikut :

```

<?
function testExit(){
    for($i = 0; $i<5; $i++){
        // melakukan return pada $i == 2
        if($i == 2){
            exit;
        }
        echo("Nilai i : $i <br>");
    }
    echo("Loop Selesai");
}

// jalankan function
testExit();
echo("Function selesai");
?>

```

outputnya adalah :

```

Nilai i : 0
Nilai i : 0
Function Selesai

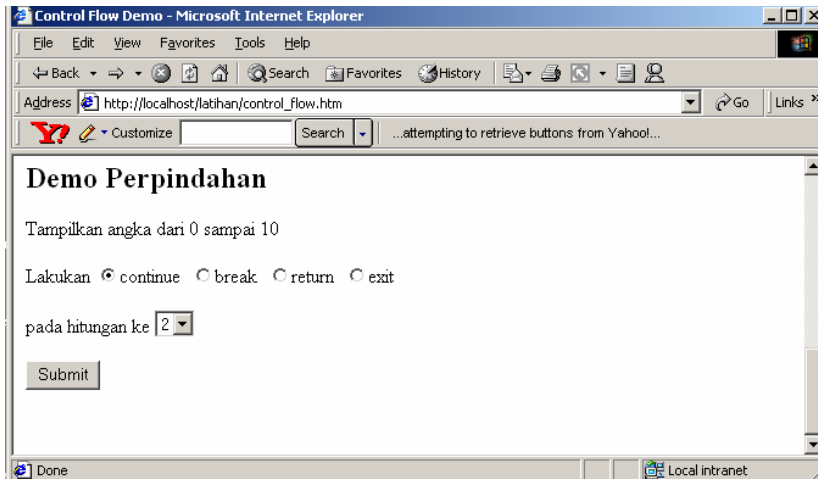
```

Perhatikan bahwa baris

```
echo("Function selesai");
```

Tidak dijalankan.

Untuk melihat contoh penggunaan break, continue, return, dan exit; tambahkan tampilan pada control\_flow.htm menjadi seperti gambar di bawah.



Tambahkan baris kode berikut pada perpindahan.php

```
<?
/*
variabel yang dibutuhkan
$mark -> tempat dilakukan perpindahan
$perintah -> perintah pindah : continue, break, return, exit
*/
function execute($tanda, $perintah){
for($i=0; $i<11; $i++){
    if($i == $tanda){
        if($perintah == "continue"){
            continue;
        }elseif($perintah == "break"){
            break;
        }
        elseif($perintah == "return"){
            return;
        }elseif($perintah == "exit"){
            exit;
        }
    }
    echo($i."<br>");
}
echo("Looping Selesai<br>");
}

execute($tanda, $perintah);
echo("Function execute selesai<br>");
?>
```



## VII. Array

### A. Konsep array

Array adalah kumpulan beberapa data yang disimpan dalam satu variabel. Jadi, berbeda dengan variabel sebelumnya yang hanya menyimpan satu nilai saja, array dapat menampung lebih dari satu nilai.

Sebagai contoh, kita akan menyimpan data teman-teman kita dalam satu variabel `$teman`. Ada tiga orang teman yang akan kita masukkan ke dalam variabel tersebut. Secara visual, dapat digambarkan sebagai berikut :

Charlie	Ani	Budi
---------	-----	------

`$teman`

Tiap nilai dalam array (disebut *elemen*) diakses dengan menggunakan index. PHP mengenal dua macam index, yaitu index numerik dan index asosiatif. Index numerik menggunakan angka untuk menandai tiap elemen, sedangkan index asosiatif memberi nama untuk masing-masing elemen. Penggunaan index akan dibahas lebih detail di bawah.

### B. Inisialisasi array

Inisialisasi (mengisi nilai) array `$teman` di atas dilakukan dengan cara sebagai berikut :

```
$teman = array("Charlie", "Ani", "Budi");
```

Kode di atas akan menjadikan `$teman` array berindex numerik. Selain dengan cara di atas, kita juga dapat membuat array yang sama dengan cara :

```
$teman[0] = "Charlie";  
$teman[1] = "Ani";  
$teman[2] = "Charlie";
```

hasilnya adalah array berindeks numerik yang dapat digambarkan sebagai berikut :

Charlie	Ani	Budi
0	1	2

`$teman`

untuk membuat array berindex asosiatif, kita membutuhkan label atau nama untuk masing-masing elemen. Contoh di atas akan dikembangkan agar dapat menyimpan nomer telepon masing-masing teman. Gambaran visualnya adalah sebagai berikut :

Charlie	Ani	Budi
123	528	456

\$rekan

pada gambar di atas, nama masing-masing teman akan dijadikan label untuk nilai nomer telepon yang disimpan. Kode programnya adalah sebagai berikut :

```
$rekan = array ("Charlie"=>123, "Ani"=>528, "Budi"=>456);
```

atau

```
$rekan = array("Charlie"=>123);  
$rekan ["Ani"] = 528;  
$rekan ["Budi"] = 456;
```

### C. Navigasi array

Navigasi (mengakses nilai) array dapat dilakukan dengan looping. Misalnya untuk menampilkan isi array berindex numerik di atas, kita akan menggunakan kode seperti berikut :

```
for($i=0; $i<length($teman);$i++){  
    echo("Array \">$teman index ke [$i] adalah $teman[$i]");  
    echo("<br>");  
}
```

Untuk mengakses array berindex asosiatif, kita tidak dapat menggunakan loop seperti di atas, karena label/index elemen disimpan dalam bentuk string.

Loop untuk mengakses array berindex asosiatif dapat dilihat pada sampel kode di bawah :

```
while(list($index, $nilai) = each($rekan)){  
    echo("Array \ $rekan index $index berisi nilai $nilai");  
    echo("<br>");  
}
```

ada dua fungsi yang terlibat di atas, `each` dan `list`. Fungsi `each` berguna untuk mengakses masing-masing elemen dalam array. Elemen yang dihasilkan adalah pasangan `key` dan `value`, dimana `key` adalah

indexnya dan `value` adalah isinya. Pasangan tersebut akan dipisahkan oleh fungsi `list` dan dimasukkan ke dalam variabel `$index` dan `$nilai`.

#### D. Fungsi-fungsi array

##### **sort**

Elemen-elemen dalam array dapat diurutkan dengan menggunakan fungsi `sort()`. Contoh penggunaannya adalah sebagai berikut :

```
sort($teman);
for($i=0; $i<length($teman);$i++){
    echo("Array \ $teman index ke [$i] adalah $teman[$i]");
    echo("<br>");
}
```

akan menghasilkan output :

```
Array teman index ke 0 adalah Ani
Array teman index ke 1 adalah Budi
Array teman index ke 2 adalah Charlie
```

##### **asort dan ksort**

`asort` digunakan untuk mengurutkan asosiatif array menurut isinya.

Contoh penggunaan :

```
asort($rekan);
while(list($index, $nilai) = each($rekan)){
    echo("Array \ $rekan index $index berisi nilai $nilai");
    echo("<br>");
}
```

akan menghasilkan output :

```
Array $rekan index Charlie berisi nilai 123
Array $rekan index Budi berisi nilai 456
Array $rekan index Ani berisi nilai 528
```

`ksort` digunakan untuk mengurutkan asosiatif array menurut indexnya.

Contoh penggunaan :

```
ksort($rekan);
while(list($index, $nilai) = each($rekan)){
    echo("Array \ $rekan index $index berisi nilai $nilai");
    echo("<br>");
}
```

akan menghasilkan output :

```
Array $rekan index Ani berisi nilai 528
Array $rekan index Budi berisi nilai 456
Array $rekan index Charlie berisi nilai 123
```

## VIII. Function

### A. Konsep function

Function adalah kumpulan beberapa statement yang dibuat dengan tujuan menyelesaikan satu tugas tertentu.

### B. Return value dan parameter

Perhatikan kode berikut :

```
function add($a, $b){  
    return $a + $b;  
}
```

Function sederhana di atas akan menerima masukan berupa dua angka. Kemudian kedua angka tersebut akan dijumlahkan, dan hasilnya dikembalikan kepada pemanggil function.

Nilai yang dikembalikan tersebut disebut `return value`. Sedangkan nilai yang dimasukkan ke dalam function (`$a` dan `$b`) disebut `parameter`.

### C. Deklarasi function

Ada beberapa hal utama yang perlu diperhatikan dalam deklarasi function pada PHP.

- nama function
- parameter
- function body

Contoh function:

```
1  function addNumber($x, $y)  
2  {  
3      z = x + y;  
4      echo(z);  
5  }  
6  
7  function jumlahkanlah(int x, int y)  
8  {  
9      z = x + y;  
10     return z;  
11 }
```

Perhatikan contoh function `addNumber` di atas.

Baris satu merupakan deklarasi function. deklarasi berisi :

- keyword function
- nama function
- parameter

Parameter adalah nilai yang dimasukkan ke dalam function untuk diproses sehingga menghasilkan output.

Nama function ditentukan dengan memenuhi aturan sebagai berikut :

- ❑ Tidak boleh sama dengan function yang telah ada dalam PHP.
- ❑ Hanya boleh terdiri dari huruf, angka, dan garis bawah (underscore)
- ❑ Tidak boleh diawali dengan angka

```
function 4uOnly(){} // tidak boleh
```

bahasa pemrograman lain mendukung fasilitas overloading, yaitu sebuah function dapat memiliki nama yang sama dan hasil yang berbeda, dengan syarat parameternya berbeda. PHP tidak mendukung overloading. Jadi, kita tidak dapat menggunakan nama yang telah digunakan sebelumnya.

#### D. Implementasi function

Contoh function:

```
1  function addNumber($x, $y)
2  {
3      z = x + y;
4      echo(z);
5  }
6
7  function jumlahkanlah(int x, int y)
8  {
9      z = x + y;
10     return z;
11 }
```

Perhatikan contoh function addNumber di atas.

Perhatikan baris 2 sampai 4.

- ❑ Function body dibatasi oleh sepasang { dan }
- ❑ Function body berisi instruksi yang harus dilakukan komputer untuk menghasilkan output yang diinginkan.
- ❑ Baris 3 menyuruh komputer untuk membuat satu variabel bernama z yang isinya adalah hasil penjumlahan x dan y.
- ❑ x dan y didapat dari input yang diberikan user.

- Baris 4 menyuruh komputer untuk menampilkan hasil perhitungan ke layar.
- Bedakan dengan baris 4 pada function jumlahkanlah yang memerintahkan komputer untuk menampilkan hasil perhitungan di layar

Contoh penggunaan function

```
hasil = jumlahkanlah(4,5);  
addNumber(4,5);
```

# IX. Variable Lifetime

## A. Scope

Scope, lifetime, visibility adalah berbagai istilah yang sama, menyatakan di mana variabel dapat digunakan dalam program. PHP mengenal dua macam scope, yaitu local dan global. Variabel local hanya dapat digunakan dalam blok tempatnya dideklarasikan. Variabel global dapat digunakan di mana saja dalam program setelah dideklarasikan dan diinisialisasi.

### Local

Untuk lebih memahami scope local, perhatikan script di bawah :

```
<?
function testVar(){
    $a = 3;
}
echo($a);           // error - variabel $a tidak dikenali
?>
```

Variabel `$a` hanya berlaku di dalam function `testVar`, sehingga tidak dapat diakses di luar function.

### Global

Variabel global dideklarasikan di luar function dan dapat digunakan di mana saja dalam program. Variabel global tidak berlaku di dalam function kecuali dipanggil dengan keyword `global`. Contoh penggunaan :

```
<?
$a = 4;
function testVar2(){
    echo($a);       // -- tidak menghasilkan apa-apa
    global $a;
    echo($a);       // -- menampilkan 4
}
?>
```

## B. Passing variabel

### By Value

Variabel dipassing (dimasukkan) ke dalam function dengan cara pass by value. Pass by value membuat kopi dari variabel yang asli. Dengan demikian variabel asli tidak terpengaruh. Untuk lebih jelasnya, perhatikan contoh berikut :

```
function tambahSatu($angka){
    $angka++;
}
```

Function tersebut akan digunakan sebagai berikut :

```
<?
$a = 7;
tambahSatu($a);
echo("Nilai a = " . $a);
?>
```

kode di atas akan menghasilkan output

```
Nilai a = 7
```

karena variabel angka yang ada dalam function berakhir umurnya pada saat function selesai dijalankan. Dan variabel a yang asli tetap bernilai 7.

Untuk menghasilkan efek yang diinginkan, kita dapat menggunakan pass by reference.

### **By Reference**

Pada pass by reference, kita memasukkan variabel yang asli ke dalam function. Teknik ini dilakukan dengan cara sebagai berikut :

```
function tambahSatu(&$angka){
    $angka++;
}
```

Function tersebut akan digunakan sama seperti contoh di atas :

```
<?
$a = 7;
tambahSatu($a);
echo("Nilai a = " . $a);
?>
```

kode di atas akan menghasilkan output

```
Nilai a = 8
```



## X. Class dan Objects

Aplikasi perangkat lunak dibuat untuk menyelesaikan masalah dalam kehidupan nyata. Dalam proses perancangannya, ada beberapa pendekatan yang dipakai.

- ❑ Sekuensial Programming
- ❑ Structured Programming
- ❑ Object Oriented Programming

### A. Konsep class dan object

Untuk dapat memahami class dan object, kita akan memvisualisasikan aplikasi address book.

Dalam sebuah address book, komponen utama yang terlibat adalah kontak

Komponen kontak disebut class.

Class adalah definisi (cara menggambarkan) suatu benda.

Object adalah benda nyata yang ada dalam session pelatihan, diantaranya:

Kontak : Charlie, Budi, Ani

Hubungan tersebut dapat dinyatakan dalam istilah teknis sebagai berikut:

Object merupakan instance dari class.

Class didefinisikan dengan kode sebagai berikut :

```
<?
class Contact{
}
?>
```

object dari class Contact dibuat dengan kode sebagai berikut :

```
$ani = new Contact();
$budi = new Contact();
$charlie = new Contact();
```

### B. Method dan properties

Lebih mendetail tentang class Contact. Semua contact, baik Ani, Budi, maupun Charlie, mempunyai karakteristik yang sama. Mereka semua memiliki nama lengkap, nomer telepon, dan alamat. Karakteristik ini

dikenal dengan istilah *properties*. Properties diterjemahkan ke dalam kode menjadi :

```
<?
class Contact{
    var $namaLengkap;
    var $telp;
    var $alamat;
}
?>
```

Tiap object memiliki nilai yang berbeda-beda untuk masing-masing properties. Hal ini juga sering dikatakan : object memiliki state yang berbeda satu dengan lainnya. Sampel kode yang menggambarkan kondisi tersebut adalah :

```
$sani = new Contact();
$sani->namaLengkap = "Ani Malia";
$sani->telp = 528;
$sani->alamat = "Ragunan";
```

```
$budi = new Contact();
$budi->namaLengkap = "Budi Man";
$budi->telp = 456;
$budi->alamat = "Bandung";
```

```
$charlie = new Contact();
$charlie->namaLengkap = "Charlie Charmless";
$charlie ->telp = 123;
$charlie ->alamat = "USA";
```

selain menyimpan karakteristik, class juga dapat melakukan aktivitas. Ini disebut dengan method. Misalnya, class Contact dapat melakukan aktivitas memanggil contact lain. Konsep ini diterjemahkan menjadi kode sebagai berikut :

```
<?
class Contact{
    var $namaLengkap;
    var $telp;
    var $alamat;

    function panggil($contact){
        echo("Memanggil kontak "+$contact->namaLengkap);
    }
}
?>
```

kode tersebut dieksekusi sebagai berikut :

```
$sani = new Contact();  
$sani->namaLengkap = "Ani Malia";  
$budi = new Contact();  
$budi->panggil($sani);
```

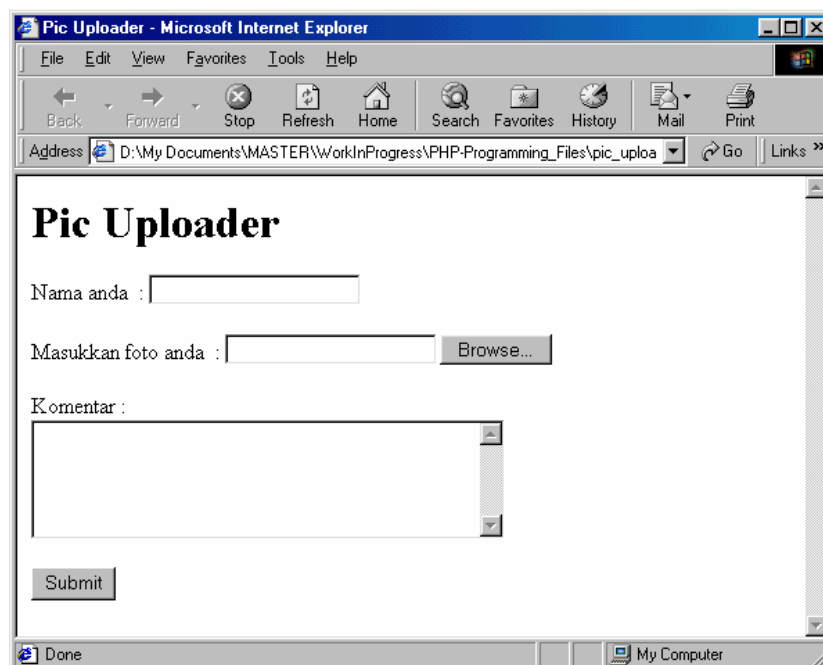
dan menghasilkan output sebagai berikut :

```
Memanggil Ani Malia
```

## XI. File Handling

Pada bagian ini, kita akan mempelajari bagaimana script php berinteraksi dengan file. Sebagai contoh, kita akan meneruskan aplikasi contact manager dengan memungkinkan pengguna untuk mengupload file foto beserta komentarnya. Setelah foto berhasil diupload, kita akan memberikan halaman untuk mengakses foto dan komentar tersebut.

Sebelumnya, form HTML yang dibutuhkan harus disiapkan dengan bentuk sebagai berikut :



The image shows a screenshot of a Microsoft Internet Explorer browser window titled "Pic Uploader - Microsoft Internet Explorer". The address bar shows the path "D:\My Documents\MASTER\WorkInProgress\PHP-Programming\_Files\pic\_uploa". The main content area displays a form titled "Pic Uploader" with the following elements:

- A text input field labeled "Nama anda :".
- A text input field labeled "Masukkan foto anda :" followed by a "Browse..." button.
- A text area labeled "Komentar :".
- A "Submit" button at the bottom left.

The browser's status bar at the bottom shows "Done" and "My Computer".

form tersebut harus dapat handle file upload. Listing kode upload.htm adalah sebagai berikut :



Form tersebut, setelah diisi dan disubmit, akan diterima oleh server dan menghasilkan beberapa variabel berikut :

Nama Variabel	Keterangan	Isi
\$nama	Berisi tulisan yang diisikan user di textfield nama anda	Endy Muhardin
\$comment	Berisi tulisan yang diisikan user di textarea komentar	Ini foto saya waktu kecil
\$pic	Berisi file gambar yang diupload, disimpan sementara di server	--
\$pic_name	Berisi nama file yang diupload	Coffee Bean.bmp
\$pic_size	Berisi ukuran file yang diupload	16.6 kB

Informasi di atas dapat kita tampilkan dalam script upload.php dengan kode sebagai berikut :

```
echo("Nama File User = $pic_name <br>");  
echo("Nama File User di server= $pic <br>");  
echo("Ukuran File User = $pic_size <br>");  
echo("Jenis File User = $pic_type <br>");
```

File yang akan kita tulis ke harddisk berada dalam variabel \$pic. File ini akan kita copy untuk disimpan secara permanen dalam disk. Untuk itu, kita akan menambah kode berikut pada upload.php :

```
$fileServer = "upload/$nama/$pic_name";  
copy($pic, $fileServer);
```

Selain file gambar yang diupload, kita juga akan menuliskan komentar yang ada di textarea ke dalam file, dengan nama yang sama dengan nama gambar, diakhiri dengan extension txt untuk menandai bahwa file tersebut mempunyai format text.

Untuk menulis ke dalam file, lakukan langkah-langkah berikut :

#### **Buka/buat file**

Untuk dapat mengakses dan mengubah isi file, kita membutuhkan sebuah file descriptor. File descriptor adalah suatu variabel yang digunakan untuk mewakili file tertentu.

File descriptor didapat dengan menggunakan fungsi php : fopen, untuk membuka file.

File yang akan kita buat akan diberi nama sesuai dengan nama gambar.

Tambahkan kode berikut pada upload.php

```
$fileComment = "/upload/$nama/$pic_name-comment.txt";  
$fp = fopen($fileComment, "w");
```

seperti kita lihat pada contoh di atas, kita menggunakan fungsi fopen. Fungsi ini membutuhkan dua masukan (parameter), yaitu : nama file, dan jenis akses.

- ❑ Nama File : bertipe string, merupakan nama file yang akan dibuka. Harus berada dalam folder yang sama dengan tempat script php yang memanggilnya.
- ❑ Jenis Akses : bertipe string, menentukan perlakuan yang diperbolehkan terhadap file yang dibuka. Ada beberapa jenis akses:
  - ❑ Read : disimbolkan dengan huruf r. File yang dibuka hanya dapat dibaca dan tidak boleh ditulisi. File yang dibuka dengan akses r akan menimbulkan pesan error kalau ada usaha untuk menulis ke dalamnya.
  - ❑ Write : disimbolkan dengan huruf w. File yang dibuka hanya dapat ditulis dan tidak bisa dibaca. Penulisan terhadap file akan **menghapus** isi yang sebelumnya.
  - ❑ Read dan write : disimbolkan dengan r+. File yang dibuka dapat dibaca dan juga ditulisi. Penulisan terhadap file akan ditambahkan pada bagian awal file.
  - ❑ Write dan read : disimbolkan dengan w+. Apabila file sudah ada, isinya akan dihapus.
  - ❑ Menambahkan (append) : disimbolkan dengan a. Membuka (membuat bila belum ada) dan menulis di awal file.
  - ❑ Menambahkan (append) dan membaca: disimbolkan dengan a+.  
Membuka (membuat bila belum ada) dan menulis di awal file.
  - ❑ Mode binary. Disimbolkan dengan b. Digunakan pada filesystem windows yang membedakan file text dan binary.

#### **Masukkan data**

Setelah file berhasil dibuka/dibuat, masukkan data ke dalamnya. Perintahnya adalah sebagai berikut :

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

### Tutup file

Setelah itu, tutup file.

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

## B. Menghapus file

File yang diupload pada pembahasan di atas telah tersimpan secara permanen di harddisk server di folder `upload/endymuhardin/Coffee Bean.bmp`. Dengan demikian, kita dapat menghapus file yang disimpan sementara oleh server. File tersebut kita hapus dengan menggunakan perintah `unlink()` atau `delete()`. Tambahkan baris berikut pada `upload.php`.

```
unlink($pic);
```

## C. Membuat direktori

Pembaca yang teliti akan menyadari bahwa nama folder pada contoh di atas disesuaikan dengan nama user yang mengupload foto. Karena kita sebagai programmer tidak dapat meramalkan nama user yang akan mengupload foto, kita tidak dapat menyediakan folder yang sesuai dengan kebutuhan tersebut. Dengan demikian, kita harus membuat folder secara dynamic. Membuat direktori atau folder baru bukan hal yang sulit, tambahkan baris berikut pada `upload.php` :

```
$oldmask = umask(0);  
mkdir("upload/$nama", 0777);  
umask($oldmask);
```

direktori akan dibuat dengan nama folder sesuai dengan nama user yang dimasukkan pada form dengan permission 777. Arti dari permission 777 dan `umask` tidak dibahas dalam tutorial ini.

## D. Navigasi direktori

Selanjutnya, kita akan menampilkan foto yang telah diupload oleh user. Untuk itu, kita harus membuka folder yang berisi file, melihat daftar file yang ada, dan menampilkan foto serta isi comment.

Untuk menampilkan foto, kita cukup memberikan link nya saja dalam tag `<img> </img>`. Sedangkan untuk menampilkan isi file text, kita harus membuka dan membaca file txt tersebut.



Untuk menampilkan isi direktori, kita terlebih dulu harus membuka direktori. Tambahkan kode berikut pada upload.php :

```
$myDir = "upload/$nama";  
$dir = opendir($myDir);
```

Direktori yang telah terbuka akan disimpan dalam variabel \$dir.

Selanjutnya, kita akan melakukan looping untuk membaca setiap entry yang ada dalam folder. Tambahkan kode berikut :

```
echo("Isi folder upload/$nama : <br>");  
while($tmp = readdir($dir)){  
    echo($tmp."<br>");  
}
```

Setelah selesai, tutup folder dengan kode berikut:

```
closedir($dir);
```

## E. Menghapus direktori

Untuk menghapus direktori, gunakan kode berikut :

```
rmdir($namaDirektori);
```

direktori yang akan dihapus **harus kosong**.

## F. Membaca file

Untuk menampilkan isi comment yang telah kita tulis ke dalam file, kita akan menempuh langkah-langkah yang sama dengan menulis file, yaitu:

Buka file – baca isinya – tutup file.

Ada sedikit modifikasi, kita akan memproses isi comment agar tidak mengandung tag-tag html. Tambahkan kode berikut pada upload.php :

```
$f = fopen($fileComment, "r");  
$isi = fread($f, filesize($fileComment));  
fclose($f);  
$output = nl2br(htmlspecialchars($isi));  
echo("Komentar anda : <br>");  
echo($output);
```

demikianlah isi file ditampilkan.

File lengkap upload.php adalah sebagai berikut :

```
<?
// -- Upload File -- //

// info tentang file user :
echo("Nama File User = $pic_name <br>");
echo("Nama File User di server= $pic <br>");
echo("Ukuran File User = $pic_size <br>");
echo("Jenis File User = $pic_type <br>");

// menulis file pic ke harddisk server
$oldmask = umask(0);
mkdir("upload/$nama", 0777);
umask($oldmask);
$fileGambar = "upload/$nama/$pic_name";
copy($pic, $fileGambar);

// menulis file comment ke file
$fileComment = "upload/$nama/$pic_name-comment.txt";
$fp = fopen($fileComment, "w");
fwrite($fp, $comment);
fclose($fp);

// -- Menampilkan isi folder -- //

$myDir = "upload/$nama";
$dir = opendir($myDir);

echo("<hr>Isi folder upload/$nama : <br>");
while($tmp = readdir($dir)){
    echo($tmp."<br>");
}

closedir($dir);

// -- menampilkan pic yang telah diupload -- //
echo("<hr>Pic $nama : <br>");
echo("<img src=\"\$fileGambar\" border=0 width=100
height=100>");

// -- menampilkan isi file text --//
echo("<hr>Komentar : <br>");
$f = fopen($fileComment, "r");
$isi = fread($f, filesize($fileComment));
fclose($f);
$output = htmlspecialchars($isi);
echo($output);
?>
```

## XII. Database Access

Menyimpan data dalam file biasa memiliki banyak keterbatasan. Semakin besar ukuran file, pencarian data menjadi lebih sulit. File biasa juga tidak memiliki kemampuan untuk mengolah data, misalnya menghitung total nilai, rata-rata, dan lain sebagainya.

Dan yang lebih penting, adanya keterbatasan untuk mengendalikan akses terhadap data. Kita tidak dapat menentukan siapa yang boleh dan siapa yang tidak boleh mengakses data. Siapa yang boleh membaca dan tidak boleh menulis, dan sebagainya.

Karena itu, sekarang kita akan menggunakan media penyimpanan data yang lebih mutakhir, yaitu database. Operasi data dalam database umumnya mengikuti pola yang sama, yaitu melalui rangkaian langkah sebagai berikut :

- ❑ Membuka sambungan dengan database server
- ❑ Memilih dan membuka database yang diinginkan
- ❑ Mengirim perintah untuk mengambil/mengubah/menghapus data.
- ❑ Mengakses hasil pengambilan data
- ❑ Mengakhiri sambungan

Kita akan membahas masing-masing langkah tersebut satu per satu.

Sebagai contoh, kita akan mengaktifkan fasilitas login pada aplikasi Contact Manager. User akan diminta untuk memasukkan username dan password. Bila data yang dimasukkan sama dengan yang ada dalam database, user akan diteruskan ke halaman welcome. Kalau salah, akan muncul pesan error.

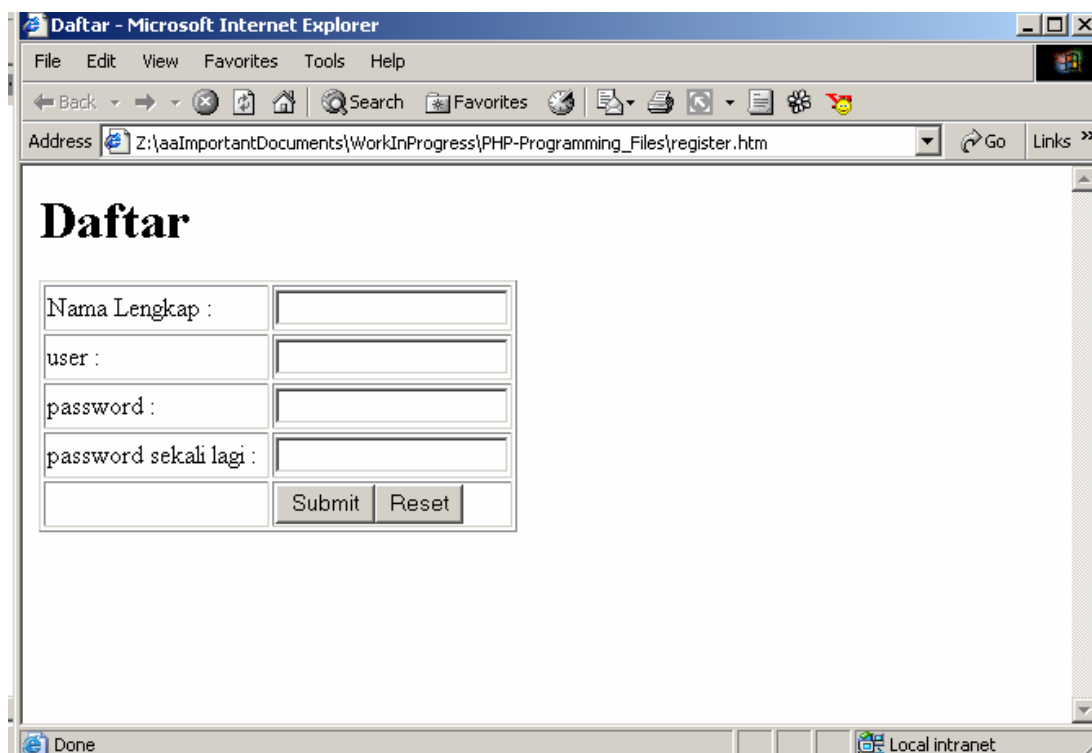
Sebagai tambahan, kita akan membuat halaman yang memungkinkan user untuk mengganti password.

User yang belum terdaftar dapat melakukan registrasi di halaman yang akan disediakan. Berikut screenshot halaman web yang dibutuhkan :

login.htm :



register.htm



## A. Sekilas SQL Query

SQL (Structured Query Language) adalah bahasa yang khusus digunakan untuk mengoperasikan database. Untuk memudahkan pelajaran, SQL query akan dikelompokkan menjadi tiga:

- ❑ Query untuk mengelola database
- ❑ Query untuk mengakses data dalam satu tabel
- ❑ Query yang melibatkan lebih dari satu tabel

### Query pengelolaan database

Yang termasuk ke dalam kelompok ini adalah query yang bertujuan untuk :

- ❑ Membuat database
- ❑ Menghapus database
- ❑ Membuat tabel
- ❑ Memodifikasi tabel
- ❑ Menghapus tabel
- ❑ Menambah user
- ❑ Mengatur permission
- ❑ Menghapus user

Membuat database dilakukan dengan perintah sebagai berikut :

```
CREATE DATABASE <nama database>
```

Contohnya : untuk aplikasi Content Management, kita akan membuat database ContentManager dengan query sebagai berikut :

```
CREATE DATABASE ContactManager
```

Menghapus database dilakukan dengan perintah :

```
DROP DATABASE <nama database>
```

Contoh :

```
DROP DATABASE ContactManager
```

Membuat tabel dilakukan dengan memberikan perintah sebagai berikut:

```
CREATE TABLE <nama tabel> (  
  <nama kolom> <tipe data>,  
  <nama kolom> <tipe data>,  
  ... ,  
  <nama kolom> <tipe data>  
);
```

contohnya:

```
CREATE TABLE userTable(
  UserId INT (3),
  UserName VARCHAR (50),
  Password VARCHAR (50),
  NamaLengkap VARCHAR (50)
);
```

Menghapus tabel dilakukan dengan menggunakan query

```
DROP TABLE <nama tabel>;
```

Contoh :

```
DROP TABLE userTable;
```

### Query satu tabel

Query satu tabel digunakan untuk mengelola data dalam satu tabel.

Beberapa hal yang dapat dilakukan pada satu tabel adalah :

Tujuan	Query
Memasukkan data	INSERT
Memodifikasi data	UPDATE
Mengambil data	SELECT
Menghapus data	DELETE
Menghitung banyaknya data	COUNT
Menghitung penjumlahan data	SUM
Menghitung nilai minimal	MIN
Menghitung nilai maksimal	MAX
Menghitung nilai rata-rata	AVG

Untuk memasukkan data ke dalam database, kita menggunakan perintah INSERT. Aturan penulisannya adalah :

```
INSERT INTO <nama tabel> (<nama kolom 1>, < nama kolom 1>, ..)
VALUES (<isi kolom 1>, <isi kolom 2>, ..);
```

Contoh :

```
INSERT INTO userTable VALUES (
  1,
  'endy',
  'inipaswod',
  'Endy Muhardin'
);
```

Untuk mengubah data dalam database, digunakan perintah UPDATE, dengan aturan penulisan sebagai berikut :

```
UPDATE <nama tabel>
SET <nama kolom>=<isi kolom>
WHERE <kriteria>
```

Contoh :

```
UPDATE userTable
SET password='test'
WHERE UserName='endy'
```

Untuk menampilkan data dalam tabel, gunakan perintah SELECT. Aturan penulisannya adalah sebagai berikut:

```
SELECT <nama kolom>
FROM <nama tabel>
WHERE <kriteria>
```

Contoh :

```
SELECT *
FROM UserTable
WHERE UserId=1
```

Untuk menghapus data dari dalam tabel, gunakan perintah DELETE. Aturan penulisannya adalah :

```
DELETE FROM <nama tabel> WHERE <kriteria>
```

Contoh :

```
DELETE FROM UserTable
WHERE NamaLengkap='Endy Muhardin'
```

### Query banyak tabel

Query banyak tabel digunakan untuk menghubungkan data di lebih dari satu database. Pembahasan tentang join berada di luar materi PHP Programming dan ada di pembahasan tentang relational database.

## B. Membuat koneksi ke database

Sebelum mulai, kita terlebih dahulu akan membuat database. Kita membutuhkan tabel user untuk menyimpan data sebagai berikut :

Nama Field	Tipe Data
UserId	int(3)
UserName	varchar(50)
Password	varchar(50)
NamaLengkap	varchar(50)

Tabel dibuat dengan memberikan query sebagai berikut:

```
CREATE TABLE userTable(
  UserId INT (3) AUTO_INCREMENT PRIMARY KEY,
  UserName VARCHAR (50),
  Password VARCHAR (50),
  NamaLengkap VARCHAR (50)
);
```

tabel siap digunakan.

Kita akan mulai dengan halaman daftar, karena kita belum punya user.

File register.php akan menangani pendaftaran user baru. Tugasnya adalah memasukkan data yang diisi dalam form ke dalam database.

Selanjutnya, kita akan melakukan langkah-langkah seperti yang disebutkan di atas.

### **Membuka sambungan dengan database server**

Untuk membuka sambungan, kita membutuhkan alamat server serta username dan password untuk database.

Setelah itu, kita membuka sambungan dengan perintah sebagai berikut :

```
$dbServer = "localhost";  
$dbUser = "endy";  
$dbPass = "test";  
  
$dbConn = mysql_pconnect($dbServer, $dbUser, $dbPass);
```

sambungan dengan database disimpan dalam variabel \$dbConn untuk digunakan pada langkah-langkah selanjutnya.

### **Memilih dan membuka database yang diinginkan**

Selanjutnya, kita akan menggunakan database tutorial, yang sebelumnya telah dibuat dengan menggunakan PHPMYAdmin. Gunakan kode berikut untuk memilih dan membuka database phpTutorial.

```
$dbName = "phpTutorial";  
mysql_select_db($dbName);
```

sekarang database telah siap untuk digunakan. Kita dapat melangkah ke tahap selanjutnya, yaitu memasukkan data.

### **Mengirim perintah untuk mengambil/mengubah/menghapus data.**

Perintah untuk memasukkan data dibuat dengan menggunakan aturan SQL, dijelaskan lebih lengkap pada bagian selanjutnya. Query (perintah) untuk memasukkan data adalah :

```
INSERT INTO userTable VALUES (  
    1,  
    'endy',  
    'inipaswod',  
    'Endy Muhardin'  
);
```

perintah tersebut dieksekusi di dalam script PHP sebagai berikut :



```

$query = "
INSERT INTO userTable VALUES (
    1,
    'endy',
    'inipaswod',
    'Endy Muhardin'
);
";

$hasil = mysql_query($query);
if($hasil){
    echo(mysql_affected_rows()." orang ditambahkan ke dalam
sistem");
}

```

Script kita harus mengambil data dari HTML Form (register.htm) dan memasukkannya ke dalam database. Untuk itu, kita akan membuat file register.php yang berisi kode sebagai berikut :

```

<?
$dbServer = "localhost";
$dbUser = "endy";
$dbPass = "test";

$dbConn = mysql_pconnect($dbServer, $dbUser, $dbPass);

$dbName = "phpTutorial";
mysql_select_db($dbName);

$query = "
INSERT INTO userTable
(UserName, Password, NamaLengkap)
VALUES (
    '$user',
    '$pass',
    '$namaLengkap'
);
";

$hasil = mysql_query($query);
if($hasil){
    echo(mysql_affected_rows()." orang ditambahkan ke dalam
sistem");
}
?>

```

### C. Mengakses hasil query

Setelah register.htm dieksekusi dengan sukses, kita telah memiliki satu user dalam database. Dengan demikian, kita dapat menguji halaman login yang telah kita buat. Untuk itu, kita perlu mengakses data dalam database dan mencocokkannya dengan input yang diberikan user.

Langkah-langkah untuk mengakses data dalam database adalah :

- ❑ Membuat sambungan database
- ❑ Memilih database
- ❑ Membuat query
- ❑ Menjalankan query
- ❑ Mengambil hasilnya
- ❑ Memproses hasil

Query yang akan digunakan adalah :

```
SELECT * FROM UserTable WHERE userName='input dari user'
```

Query ini akan dimasukkan ke dalam script menjadi :

```
$query = SELECT * FROM UserTable WHERE userName='$login'
```

Dan dieksekusi dengan kode :

```
$hasil = mysql_query($query);
```

Bila userName yang diinputkan user tidak ada dalam database, query akan mengembalikan 0 (nol) baris hasil. Bila ada, query akan menghasilkan array yang berisi data pada masing-masing kolom dalam database.

Untuk mengetahui jumlah hasil yang didapat dari query, digunakan kode sebagai berikut:

```
$jumlahHasil = mysql_num_rows($hasil);
```

Apabila hasilnya tidak nol (berarti username telah terdaftar dalam sistem), kita akan mengambil data password. Untuk itu, kita gunakan perintah:

```
$data = mysql_fetch_array($hasil);
```

data password ada dalam array, diakses dengan cara :

```
$passDb = $data["Password"];
```

Dengan demikian, keseluruhan script login.php akan menjadi :

```
<?
// variabel yang diperlukan untuk akses database
$user = "endy";
$pass = "test";
$db = "ContentManager";
$server = "localhost";

// membuat koneksi
$koneksi = mysql_connect($server, $user, $pass);

// memeriksa koneksi
if(!$koneksi){
    echo("Koneksi ke database gagal");
    exit;
}

// membuka database
mysql_select_db($db);

// membuat query
$query = "SELECT * FROM userTable WHERE
userName='".$login.'"";

// mengeksekusi query
$hasil = mysql_query($query);

// mengakses hasil
$jumlahHasil = mysql_num_rows($hasil);
if($jumlahHasil < 1){
    echo("User $login tidak ada dalam database <br>");
}else{
    echo("User $login ada dalam database<br>");
    $data = mysql_fetch_array($hasil);
    $passwordAsli = $data["password"];

    if($password == $passwordAsli){
        echo("Password untuk user $login benar<br>");
    }else{
        echo("Password untuk user $login salah<br>");
    }
}
?>
```

## XIII. String Handling

String handling adalah kumpulan function PHP yang berguna untuk memanipulasi string. Ada banyak kegunaan yang bisa didapat dengan menggunakan fungsi-fungsi ini, misalnya:

- Pencarian kata dalam website
- Pemeriksaan user input
- Memformat file untuk keperluan khusus (misalnya email).
- Dan sebagainya

### A. Sekilas Regular Expression

Untuk dapat menggunakan fungsi-fungsi string handling dengan baik, kita perlu menguasai teknik pembuatan pola kalimat. Contoh pola kalimat misalnya, alamat email yang valid (misalnya endy@ngoprek.org) selalu memiliki pola sebagai berikut:

*satu atau lebih huruf/angka, diikuti tanda @ kemudian diikuti dengan satu atau lebih huruf/angka, dipisahkan oleh tanda titik, lalu diakhiri dengan satu atau lebih huruf/angka.*

Pola tersebut, dapat dinyatakan dengan seperangkat kode sebagai berikut :

Pola	kode
Harus di awal kata	^
Huruf	A-Za-z
Angka	0-9
Huruf atau angka	[A-Za-z0-9]
Semua jenis karakter	.
Berjumlah satu atau lebih	+ atau {1,}
Berjumlah nol atau lebih	* atau {0,}
Berjumlah tiga sampai sepuluh	{3,10}
Diikuti dengan @	@
Tanda titik	¥.
Harus berada di akhir kalimat	\$

Pola alamat email di atas dapat dinyatakan dengan satu baris kode sebagai berikut.

```
^.\+@\.\+\.\+\$
```

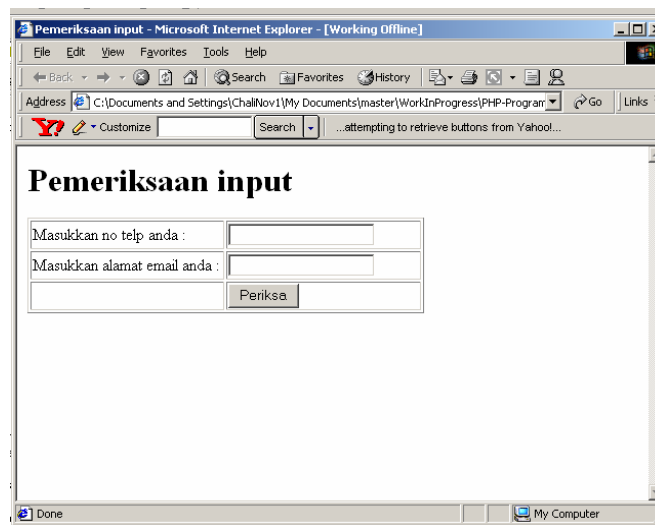
Atau kita dapat membatasi alamat email yang digunakan oleh user agar hanya menerima domain com, net, atau edu dengan menggunakan pola berikut :

```
^.+@.+.\.((com)|(edu)|(net))$
```

## B. Penggunaan Regular Expression

Untuk mendemonstrasikan kemampuan string handling, kita akan menggunakan input form yang akan memvalidasi alamat email dan nomer telepon yang dimasukkan user.

CekMail.htm berbentuk seperti gambar di bawah:



input yang dimasukkan user akan diperiksa oleh script CekMail.php yang berisi kode sebagai berikut :

```
<?
// validasi alamat email
$pol>Email = "^.+@.+.\.+$";
if(!ereg($pol>Email, $email)){
    echo("Masukkan alamat email yang valid, misal :
    endy@ngoprek.org");
}else{
    echo("Alamat email valid");
}

// validasi no telp
$pol>Telp = "^\\+[0-9]{2}-[0-9]+$";
if(!ereg($pol>Telp, $telp)){
    echo("Masukkan nomer telepon yang valid, misal : +62-
    315054307");
}else{
    echo("No telp valid");
}
?>
```

Function yang digunakan untuk memeriksa email pada script di atas adalah `eregi`. Menerima input berupa pola yang diinginkan dan string yang akan diperiksa. Function ini akan menghasilkan nilai `true` bila polanya sesuai dan `false` bila pola yang dicari tidak ada pada string input.

## XIV. Session

### A. No Session

Pada waktu kita browsing menjelajahi website, misalnya e-mail, kita akan diminta memasukkan username dan password. Prosedur ini hanya dilakukan sekali pada awal proses. Selanjutnya sistem akan 'mengingat' identitas kita, dan menampilkan informasi yang sesuai. Tanpa adanya session, sistem akan salah dan menampilkan data rahasia ke orang yang salah.

Untuk mengetahui kegunaan session, kita akan membuat aplikasi tanpa session. Kita membutuhkan satu halaman web dan tiga script php : login.htm, loginNoSession.php, welcomeNoSession.php dan displayNoSession.php

#### login.htm

```
<html>
<head>
<title>Login</title>
</head>
<body>
<h1>Login </h1>
<form method="POST" action="loginNoSession.php">
  <table border="1" width="44%">
    <tr>
      <td width="27%">user : </td>
      <td width="73%"><input type="text" name="login"
size="20"></td>
    </tr>
    <tr>
      <td width="27%">password : </td>
      <td width="73%"><input type="text" name="pass"
size="20"></td>
    </tr>
    <tr>
      <td width="27%">&nbsp;</td>
      <td width="73%"><input type="submit" value="Submit"
name="B1"><input type="reset" value="Reset" name="B2"></td>
    </tr>
  </table>
</form>
<p><a href="register.htm">Daftar</a></p>
</body>
</html>
```

### loginNoSession.php

```
<?
// -- cek user dan password -- //

if($login == "endy" && $pass == "test"){
    header("Location: welcomeNoSession.php?user=endy");
}elseif($login == "oky" && $pass == "coba"){
    header("Location: welcomeNoSession.php?user=oky");
}else{
    echo("User dan password salah");
}
?>
```

### welcomeNoSession.php

```
<?
echo("Welcome $user <br>");
echo("<a href=\"displayNoSession.php?user=$user\">");
echo("klik di sini untuk mengakses daftar kenalan
anda</a>");
?>
```

### displayNoSession.php

```
<?
$dbUser = "endy";
$dbPass = "test";
$db = "latihan";
$server = "localhost";

// membuat koneksi
$koneksi = mysql_connect($server, $dbUser, $dbPass);

// memeriksa koneksi
if(!$koneksi){
    echo("Koneksi ke database gagal");
    exit;
}

// membuka database
mysql_select_db($db);

// membuat query dan mengakses hasil
$query = "SELECT * FROM UserData WHERE user='$user'";
$hasil = mysql_query($query);

// menampilkan hasil
echo("<table border=1>");
echo("<tr>");
echo("<td>NamaTeman</td>");
echo("<td>Keterangan</td>");
echo("</tr>");
while($row = mysql_fetch_array($hasil)){
    echo("<tr>");
    echo("<td>".$row["namaTeman"]."</td>");
    echo("<td>".$row["keterangan"]."</td>");
    echo("</tr>");
}
echo("</table>");
?>
```



kode ini membutuhkan data contact yang diambil dari tabel UserData dengan definisi sebagai berikut:

```
CREATE TABLE userData (
  user VARCHAR(50),
  namaTeman VARCHAR(50),
  keterangan VARCHAR(50)
);
```

dan sampel data sebagai berikut:

```
INSERT INTO userData VALUES (
  'endy',
  'Imanudin',
  'Teman ceting'
);

INSERT INTO userData VALUES (
  'oky',
  'Iswahyudi',
  'Pak Lurah kos adhyaksa'
);

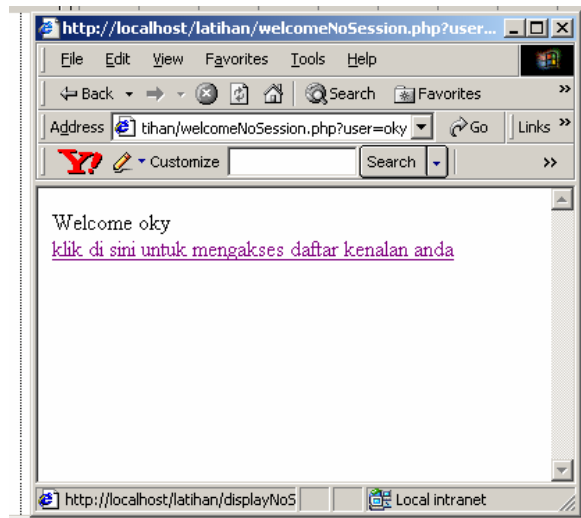
INSERT INTO userData VALUES (
  'oky',
  'Anton Raharja',
  'Rekan kerja di kantor'
);

INSERT INTO userData VALUES (
  'endy',
  'Widya Andhini',
  'Gebetn lama tapi masih mesra'
);
```

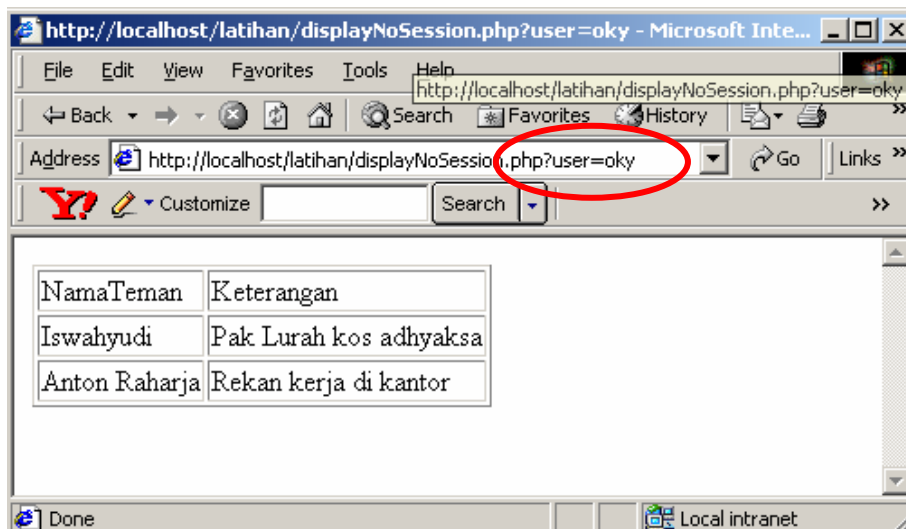
**Jalannya aplikasi :**

login dengan user = oky dan password = coba





klik link untuk mendisplay output, tampilan akan muncul sebagai berikut:

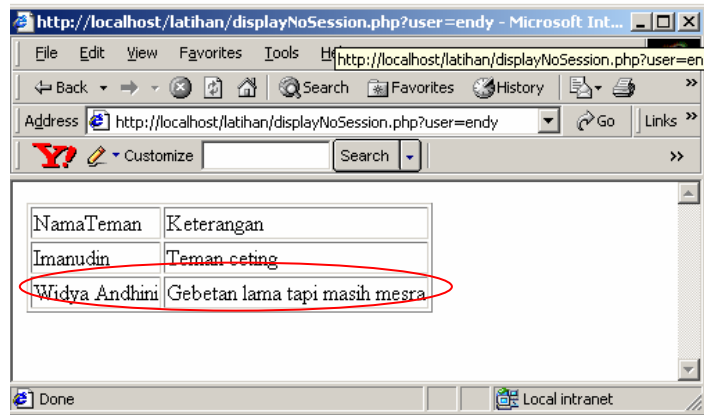


sekarang, kita akan "memalsu" link yang ada di address bar untuk menampilkan data user endy.

Buka browser baru dan masukkan link berikut di address bar:

`http://localhost/latihan/displayNoSession.php?user=endy`

Kemudian tekan enter.



Ooops... dengan mudah data rahasia akan muncul.

Cara mencegah hal ini terjadi adalah dengan meminta user untuk memasukkan login dan password setiap kali ia berpindah halaman. Tetapi tentu saja cara ini menjengkelkan bagi user. Selain itu, mempassing variabel login dan password di setiap halaman sangat tidak aman dan rentan penyadapan.

Oleh karena itulah kita menggunakan session.

## B. Konsep session

Session bekerja seperti tempat penitipan barang di supermarket. Kita datang, menitipkan barang bawaan, dan petugas akan memberikan tanda pengenal.

Pada saat kita akan mengambil barang, kita memberikan tanda pada petugas, dan petugas akan dapat mengambil barang yang ditiptkan tanpa tertukar dengan barang orang lain.

Permasalahan yang terjadi adalah, bagaimana "tanda pengenal" tersebut akan disimpan pada setiap halaman yang diakses user, sehingga dapat diakses oleh sistem setiap kali user mengklik link.

Ada tiga alternatif yang dapat digunakan untuk menyimpan data session :

- o Cookie
- o Hidden Form
- o URL Embedding

### C. Implementasi : Cookie

Cookie adalah informasi yang disimpan pada browser user. Script menjalankan function setcookie yang akan menulis data di harddisk user.

Cookie ditulis pada browser user dengan menggunakan perintah :

```
setcookie("sessionId", "1234");
```

Setelah itu, variabel \$sessionId akan tersedia setiap kali browser user mengakses sistem.

Cookie akan hilang dari browser user setelah melewati masa kadaluarsa yang ditentukan atau dihapus melalui perintah:

```
setcookie("sessionId");
```

Pendekatan ini banyak mengundang kontroversi. Karena cookie dapat menghilangkan privacy user dan berpotensi mengandung virus. Oleh karena itu, user dapat menonaktifkan cookie melalui setting di browser.

Dengan demikian, pendekatan ini kurang handal untuk diterapkan pada aplikasi kita.

Untuk melihat dengan lebih jelas implementasi cookie, kita akan memodifikasi aplikasi di atas sebagai berikut :

login.htm

```
<form method="POST" action="loginSessionCookie.php">
```

loginSessionCookie.php

```
<?
// -- cek user dan password -- //
if($login == "endy" && $pass == "test"){
    setcookie("user", $login);
    header("Location: welcomeSessionCookie.php");
}elseif($login == "oky" && $pass == "coba"){
    setcookie("user", $login);
    header("Location: welcomeSessionCookie.php");
}else{
    echo("User dan password salah");
}
?>
```

welcomeSessionCookie.php

```
<?
if(!$user){
    echo("Illegal Access");
    exit;
}
echo("Welcome $user <br>");
echo("<a href=\"displaySessionCookie.php\">");
echo("klik di sini untuk mengakses daftar kenalan
anda</a>");
?>
```

tambahkan baris berikut pada bagian atas displayNoSession.php dan save as displaySessionCookie.php

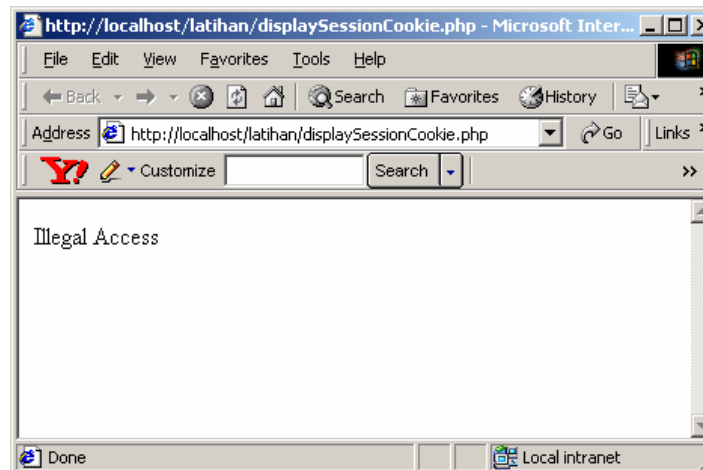
```
if(!$user){
    echo("Illegal Access");
    exit;
}
```

Untuk mengujinya, buka browser baru dan masukkan link berikut di address bar:

```
http://localhost/latihan/displaySessionCookie.php
```

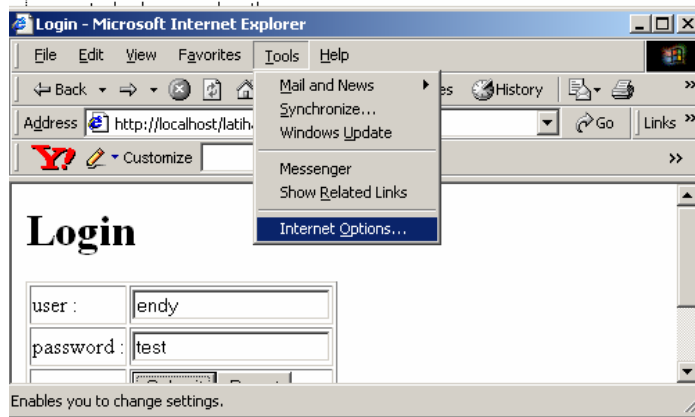
Kemudian tekan enter.

Anda akan mendapatkan:

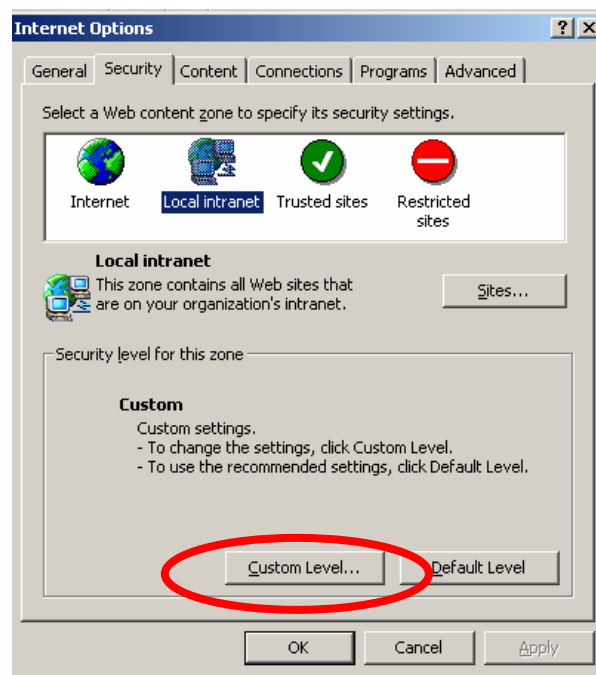


Sekarang, disable cookie dengan cara :

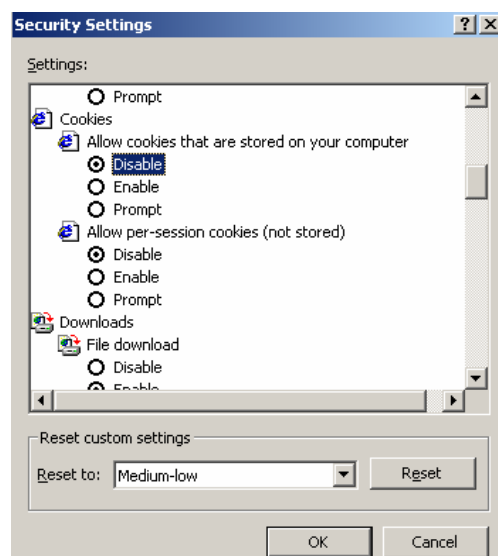
Klik Tools – Internet Options



Klik tab Security dan klik tombol Custom Level



klik disable cookie



coba lagi mengakses halaman contact. Anda akan mendapat `Illegal Access`.

#### D. Implementasi : Hidden Form

Cara kedua untuk menyimpan data session adalah dengan menyimpan data tersebut pada hidden input yang ada dalam form. Misalnya dengan menggunakan kode :

```
<input type="hidden" name="user" value="$user">
```

cara ini juga tidak dapat dilakukan di segala kondisi, karena ada saat-saat di mana kita tidak dapat menggunakan form.

#### E. Implementasi : URL Embedding

Cara terakhir yang pasti berhasil adalah dengan "menempelkan" variabel session di link/URL. Misalnya link menuju `welcome.php` dimodifikasi menjadi `welcome.php?sessionId=1234`.

Satu-satunya kerugian cara ini adalah, link URL menjadi tidak bisa di'bookmark'. Selain itu, link menjadi tidak indah dipandang mata. Tetapi secara teknis, metode inilah yang paling efektif dan reliable.

#### F. Implementasi : Session API

Session API PHP menggabungkan teknik cookie dan teknik URL Embedding. Apabila user mengaktifkan cookie, PHP akan menyimpan data session pada cookie. Tetapi apabila cookie di-disable user, PHP akan menyimpan data session pada URL. Mekanisme ini terjadi secara otomatis apabila PHP dicompile dengan option `--enable-trans-sid`.

Cara penggunaan PHP Session API dapat dilihat pada `loginSessionApi.php`, `welcomeSessionApi.php`, dan `displaySessionApi.php`.

Anda dapat memodifikasi setting Internet Explorer untuk melihat bahwa metode ini dapat dijalankan walaupun cookie di-disable. Kita juga tidak dapat mengetik langsung URL

```
http://localhost/latihan/displaySessionApi.php
```

untuk membuka halaman pribadi.

loginSessionApi.php:

```
<?
// -- cek user dan password -- //
if($login == "endy" && $pass == "test"){
    session_start();
    $user = $login;
    session_register("user");
    header("Location: welcomeSessionApi.php");
}elseif($login == "oky" && $pass == "coba"){
    session_start();
    $user = $login;
    session_register("user");
    header("Location: welcomeSessionApi.php");
}else{
    echo("User dan password salah");
}
?>
```

welcomeSessionApi.php

```
<?
session_start();
if(!session_is_registered("user")){
    echo("Illegal Access");
    exit;
}
echo("Welcome $user <br>");
echo("<a href=\"displaySessionApi.php\">");
echo("klik di sini untuk mengakses daftar kenalan
anda</a>");
?>
```



## displaySessionApi.php

```
<?
session_start();
if(!session_is_registered("user")){
    echo("Illegal Access");
    exit;
}
$dbUser = "endy";
$dbPass = "test";
$db = "latihan";
$server = "localhost";

// membuat koneksi
$koneksi = mysql_connect($server, $dbUser, $dbPass);

// memeriksa koneksi
if(!$koneksi){
    echo("Koneksi ke database gagal");
    exit;
}

// membuka database
mysql_select_db($db);

// membuat query dan mengakses hasil
$query = "SELECT * FROM UserData WHERE user='$user'";
$hasil = mysql_query($query);
echo($sessionId);
// menampilkan hasil
echo("<table border=1>");
echo("<tr>");
echo("<td>NamaTeman</td>");
echo("<td>Keterangan</td>");
echo("</tr>");
while($row = mysql_fetch_array($hasil)){
    echo("<tr>");
    echo("<td>".$row["namaTeman"]."</td>");
    echo("<td>".$row["keterangan"]."</td>");
    echo("</tr>");
}
echo("</table>");

?>
```