[MODUL 8]

PHP Introduction

Modul Ini Disusun Untuk Membantu Proses Pembelajaran Bagi Mahasiswa

DOSEN : CEPI RAHMAT HIDAYAT, S.KOM

STMIK TASIKMALAYA

SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER

1. Pengenalan

1.1. Sejarah

- Pada awalnya PHP merupakan kependekan dari *Personal Home Page* (Situs Personal)
- PHP Pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995.
- PHP awalnya merupakan sekumpulan skrip yang digunakan untuk mengolah formulir dari web.
- Pada tahun 1997, sebuah perusahaan bernama Zend menulis ulang interpreter PHP menjadi lebih bersih, lebih baik dan lebih cepat.
- Pada tahun 1998 Perusahaan Zend merilis versi terbaru yaitu PHP 3.0 dan singkatan PHP diubah menjadi akronim berulang ***PHP : Hhypertext Preprocessing***
- Pada pertengahan tahun 1999, zend merilis interpreter PHP baru dan rilis tersebut dikenal dengan PHP 4.0
- Pada Juni 2004, Zend merilis PHP 5.0. Dalam verisi ini interpreter PHP mengalami perubahan besar , versi ini sudah mendukung model pemrograman berorientasi objek.

1.2. Kelebihan PHP

- PHP Berbasis Server Side Scripting
- Command Line Scripting pada PHP
- PHP Dapat Membuat Aplikasi Desktop
- Digunakan Untuk Berbagai Macam Platform OS
- Mendukung Berbagai Macam Web Server
- Object Oriented Programing atau Procedural
- Output file PHP pada XHTML, HTML, & XML
- Mendukung Banyak RDBMS (Database)
- Menudkung Banyak Komunikasi
- Pengolahan Teks yang sangat baik
- Open Source
- Gratis Lisensi GPL

2. Membuat Program PHP

- Untuk membuat program dengan PHP kita diharuskan menginstal web server terlebih dahulu. Program yang dibuat nantinya akan memiliki ekstensi **.php** dan disimpan di folder root suatu web server untuk dijalankan.
- Tools yang bisa digunakan untuk mempermudah dalam membangun web server diantaranya ada Xampp, Appserv dan lain-lain.
- Proses pembuatan program PHP tidak terlalu berbeda dengan HTML. Agar lebih jelasnya coba perhatikan contoh kode program HTML dan PHP dibawah ini :

helloHTML.html

```
<html>
<head>
<title> Contoh Program HTML </title>
</head>
<body>
<h4> Contoh 1 Program HTML </h4>
Hello, ii program HTML
</body>
</html>
```

helloPHP.php

```
<html>
<head>
<title> Contoh Program PHP </title>
</head>
<body>
<h4> Contoh 1 Program PHP </h4>
<?php
Echo ” Hello, ini program PHP Pertamaku” ;
?>
</body>
</html>
```

- Jika anda perhatikan pada program helloPHP diatas terdapat program HTML, karena php disebut scripting inilah sehingga mudah disisipkan pada bahasa markup sekalipun
- Untuk melihat hasilnya file PHP tersebut haruslah disimpan di sebuah folder root web server, contoh : jika menggunakan Tools Appserv simpan di folder **“www”** . Jika menggunakan Xampp simpan di folder **“Htdocs”**. Untuk memanggilnya caranya dengan mengetikan **“ localhost/helloPHP.php ”** pada address bar browser.

3. Komponen Dasar PHP

3.1. Tag Pembuka dan Tag Penutup PHP

Sama dengan HTML PHP juga memiliki tag pembuka (**<?php**) dan tag penutup (**?>**) sebagai awal dan akhir program. Perhatikan contoh berikut :

```
<?php /* tag pembuka PHP */
/* Isi program PHP */
/* tag penutup PHP */ ?>
```

3.2. Komentar dalam PHP

PHP mendukung penulisan sebuah komentar seperti halnya bahasa pemrograman yang lainnya. Perhatikan contoh beikut :

```
<?php
Echo " ini adalah contoh" ; // contoh gaya komentar satu baris

/* Ini adalah contoh komentar
 lebih dari satu baris */

echo "Contoh lagi"; #contoh gaya komentar satu baris pada shell
 ?>
```

3.3. Variabel dalam PHP

- Pembuatan variabel di PHP diawali dengan tanda dollar ($) di depan nama variabel. Nama variabel adalah case-sensitive, artinya huruf besar dan huruf kecil adalah berbeda.
- Karakter yang bisa digunakan untuk mendefinisikan variabel PHP diantaranya dimulai dengan huruf atau underscore kemudian diikuti oleh, nomor,huruf,underscore.

```
<html>
<body>
 <?php
 $kampus = "STMIK TASIKMALAYA";
echo "Kampusku : $kampus" ;
?>
 </body>
 </html>
```

3.4. Konstanta dalam PHP

Didalam PHP suatu konstanta dapat kita definisikan dengan menggunakan fungsi define()

```
<html>
<body>
 <?php
define("kampus", "STMIK TASIKMALAYA");
echo "Kampusku :". kampus;
?>
 </body>
 </html>
```

3.5. Method Post dan Get

Ada dua metode ketika ingin memparsing nilai inputan dari suatu file ke file lainnya yaitu dengan method post atau get.

- **Method GET** merupakan suatu metode pengiriman data menggunakan query string yang ditampilkan pada addrees bar/ URL browser

  Method get yang akan ditampilkan pada addree bar browser akan ditampilkan seperti ini :

  localhost/validate.php?username=admin&password=demo

  Contoh penulisan program untuk mengambil nilai dari query string tersebut adalah seperti dibawah ini :

```
<?php
echo $_GET[username]; //meampilkan nilai dari variable get username
echo $_GET[password]; //menampilkan nilai variable get password
?>
```

- **Method POST** merupakan suatu metode pengiriman data yang digunakan untuk memparsing hasil inputan form tetapi tidak menampilkan query string pada address bar layaknya method GET. Method POST akan sangat berguna penggunaanya untuk halaman-halaman yang membutuhkan keamanan lebih, karena data inputan kita tidaka akan tampil pada address bar browser

  Contoh penulisan program untuk mengambil nilai dari query string tersebut adalah seperti dibawah ini :

```
<?php
echo $_POST[username]; //meampilkan nilai dari variable post username
echo $_POST[password]; //menampilkan nilai variable post password
?>
```

3.6. Operator Aritmatika

3.6.1. Aritmatika Dasar

Operator	Arti	Contoh	Hasil
+	Penjumlaham	2 + 4	6
-	Pengurangan	6-2	4
*	Perkalian	5*3	15
/	Pembagian	20/2	10
%	Modulus/Sisa hasil bagi	43%10	3
++	Tambah 1	$X=1; $X++;	X=2
--	Kurang 1	$X=1; $X++;	X=0

Contoh :

```
<?php
$x = 10;
$y = 4;
 $a = $x + $y;
echo $a."<br>"; //hasilnya 14
$b = $x - $y;
echo $b."<br>"; //hasilnya 6
$c = $x * $y;
echo $c."<br>"; //hasilnya 40
$d = $x / $y;
echo $d."<br>"; //hasilnya 2.5
$e = $x % $y;
 echo $e."<br>"; //hasilnya 2
?>
```

3.6.2. Compound Aritmatika

Operator	Arti	Contoh $var=4	Hasil
+=	Penjumlahan	$var +=2;	6
-=	Pengurangan	$var -=2;	2
*=	Perkalian	$var *=2;	8
/=	Pembagian	$var /=2;	2
%=	Modulus/Sisa hasil bagi	$var %=2;	0
.=	Concatenante	$var .=2;	42

3.7. Operator Perbandingan

Operator perbandingan digunakan untuk membandingkan dua kondisi atau nilai. Biasanya digunakan pada percabangan (*if*) maupun *loop*.

Misalnya x=3 dan y=8.

Operator	Arti	Contoh	Hasil
==	Sama Dengan	$x==y	False
!=	Tidak sama dengan	$x!=y	True
>	Lebih besar	$x>y	False
<	Lebih Kecil	$x<y	True
>=	Lebih besar atau sama dengan	$x>=y	False
<=	Lebih kecil atau sama dengan	$x<=y	True

4. Kondisi Percabangan

Baris kode percabangan digunakan untuk melakukan aksi yang berbeda untuk kondisi yang berbeda pula. Di dalam PHP ada beberapa tipe percabangan :

4.1. If Statement

Syntak berikut ini digunakan untuk melakukan serangkaian aksi tertentu hanya jika satu kondisi tepenuhi,

Syntak

```
If (Kondisi) {

Statement yang ingin dijalankan jika benar

}
```

Contoh

```
<?php
$nilai = 90;
if($nilai == 90){
 echo "Nilai anda baik";
}
?>
```

4.2. If-Else Statement

Syntak berikut ini digunakan untuk melakukan serangkaian aksi tertentu hanya jika satu kondisi tepenuhi dan serangkaian kode lain yang jika kondisinya tidak terpenuhi (else),

Syntak

```
if(kondisi){
    kode yang dijalankan jika benar
}else{
    kode yang dijalankan jika tidak ada yang memenuhi
}
```

Contoh

```
<?php
 $nilai = 50;
if($nilai == 90){
 echo "Nilai anda baik";
}else{
 echo "Nilai anda bukan 90";
}
?>
```

Syntak

```
if(kondisi 1){
  Statement yang dijalankan jika kondisi 1 benar
}else if(kondisi 2){
 Statement yang dijalankan jika kondisi 2 benar
}else if(kondisi 3){
 Statement yang dijalankan jika kondisi 3 benar
}else{
 Statement jika salah satu kondisi di atas tidak ada yang benar
}
```

Contoh

```
<?php
$nilai = 80;
if($nilai >= 85){
    echo "A";
}else if($nilai >= 70 && $nilai < 85){
    echo "B";
}else if($nilai >= 60 && $nilai < 70){
    echo "C";
}else{
    echo "D";
}
?>
```

4.4. Switch Statement

Statemen ini juga terdiri dari banyak kondisi dengan aksi yang bersesuaian. Jumlah kondisinya biasanya 2 atau bahkan lebih. Kelebihan dari penggunaan Case adalah eksekusi pemeriksaan kondisi tidak harus satu per satu secara berurutan seperti pada if – else if –else statement. Hal ini membuat performance akan sedikit lebih baik.

Syntak

```
switch(ekspresi){
    case kondisi1 :
        kode yang dijalankan jika kondisi1 benar;
        break;
    case kondisi2 :
        kode yang dijalankan jika kondisi2 benar;
        break;
    case kondisi3 :
        kode yang dijalankan jika kondisi3 benar;
        break;
}
```

Contoh

```
<?php
$bulan = "1";
switch ($bulan) {
    case "1":
        echo "Januari
        break;
    case "2":
        echo "Februari";
        break;
    case "3":
        echo "Maret";
        break;
```

4.5. For Loop

Pengulangan jenis ini ditandai dengan batas yang lebih jelas antara sampai mana pengulangan itu akan dijalankan..

Syntak

```
for(awal; kondisi; penambahan){

    kode untuk dijalankan

}
```

Contoh

```
<?php
for ($i = 1; $i <= 10; $i++) {
    echo $i." STMIK TASIKMALAYA<br>";
}
?>
```

Syntak

```
while(kondisi){

    kode untuk dijalankan;

}
```

Contoh

```
<?php
$i=1;
while($i<=5)
{ echo "Nomor : " . $i . "<br />";
    $i++; }
?>
```

4.7. Foreach

Pengulangan jenis ini digunakan untuk array yang mempunyai nilai .

Syntak

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

Contoh

```
<?php
$jabatan = array("Direktur", "Manager", "Karyawan");
foreach($jabatan as $key => $value) {
    echo "$key $value<br />\n";
}
?>
```