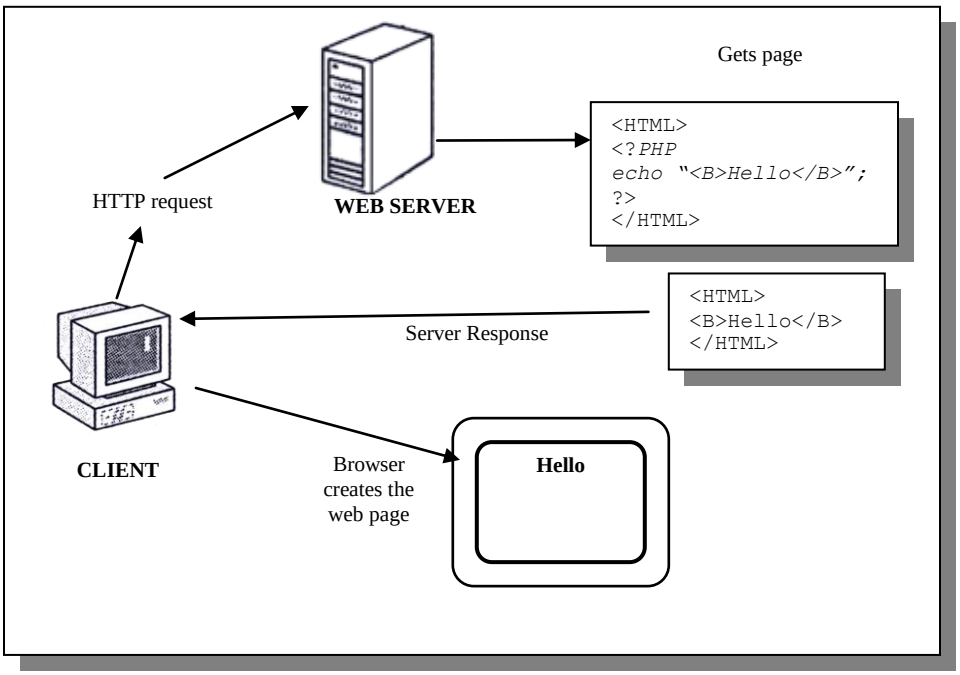
# 1. Pengenalan PHP

PHP (akronim dari PHP Hypertext Preprocessor) yang merupakan bahasa pemrogramman berbasis web yang memiliki kemampuan untuk memproses data dinamis.

PHP dikatakan sebagai sebuah server-side embedded script language artinya sintaks-sintaks dan perintah yang kita berikan akan sepenuhnya dijalankan oleh server tetapi disertakan pada halaman HTML biasa. Aplikasi-aplikasi yang dibangun oleh PHP pada umumnya akan memberikan hasil pada web browser, tetapi prosesnya secara keseluruhan dijalankan di server.

Pada prinsipnya server akan bekerja apabila ada permintaan dari client. Dalam hal ini client menggunakan kode-kode PHP untuk mengirimkan permintaan ke server (dapat dilihat pada gambar dibawah). Ketika menggunakan PHP sebagai server-side embedded script language maka server akan melakukan hal-hal sebagai berikut :

- Membaca permintaan dari client/browser
- Mencari halaman/page di server
- Melakukan instruksi yang diberikan oleh PHP untuk melakukan modifikasi pada halaman/page.
- Mengirim kembali halaman tersebut kepada client melalui internet atau intranet.




**Mengapa PHP*?***

- PHP dapat dijalankan pada platform yang berbeda-beda (Windows, Linux, Unix, etc.)
- PHP merupakan *web scripting open source*
- PHP mudah dipelajari

## 2. *Syntax* PHP

Kode PHP disimpan sebagai plain text dalam format ASCII, sehingga kode PHP dapat ditulis hampir di semua editor text seperti windows notepad, windows wordpad, dll. Kode PHP adalah kode yang disertakan di sebuah halaman HTML dan kode tersebut dijalankan oleh server sebelum dikirim ke browser.

Contoh file PHP (contoh.php):

```
<html>
  <?
Print ("Contoh text yang menggunakan kode PHP");
  ?>
</html>
```

Pada file .html, HTTP server hanya melewatkan content dari file menuju ke browser. Server tidak mencoba untuk mengerti atau memproses file, karena itu adalah tugas sebuah browser.

Pada file dengan ekstensi .php akan ditangani secara berbeda. Yang memiliki kode PHP akan diperiksa. Web server akan memulai bekerja apabila berada diluar lingkungan kode HTML. Oleh karena itu server akan melewati semua content yang berisi kode HTML, CSS, JavaScript, simple text di browser tanpa diinterpretasikan di server.

Blok *scripting* PHP selalu diawali dengan <?php dan diakhiri dengan ?>. Blok *scripting* PHP dapat ditempatkan dimana saja di dalam dokumen. Pada beberapa server yang mendukung, blok *scripting* PHP dapat diawali dengan <? dan diakhiri dengan ?>. Namun, untuk kompatibilitas maksimum, sebaiknya menggunakan bentuk yang standar (<?php ?>).

Setiap baris kode PHP harus diakhiri dengan semikolon (;). Semikolon ini merupakan *separator* yang digunakan untuk membedakan satu instruksi dengan instruksi lainnya.

PHP menggunakan // untuk membuat komentar baris tunggal atau /* dan */ untuk membuat suatu blok komentar.

## 3. Variabel PHP

Variabel digunakan untuk menyimpan suatu nilai, seperti text, angka atau array. Ketika sebuah variabel dibuat, variabel tersebut dapat dipakai berulang-ulang.

Pada PHP semua variabel harus dimulai dengan karakter '$'. Variabel PHP tidak perlu dideklarasikan dan ditetapkan jenis datanya sebelum kita menggunakan variabel tersebut. Hal itu berarti pula bahwa tipe data dari variabel dapat berubah sesuai dengan perubahan konteks yang dilakukan oleh user. Secara tipikal, variabel PHP cukup diinisialisasikan dengan memberikan nilai kepada variabel tersebut.

Contoh berikut akan mencetak "PHP" :

```
$text = "PHP";
print "$text";
```

Identifier dalam PHP adalah case-sensitive, sehingga $text dengan $Text merupakan variabel yang berbeda. Built-in function dan structure tidak case-sensitive, sehingga echo dengan ECHO akan mengerjakan perintah yang sama. Identifier dapat berupa sejumlah huruf, digit/angka, underscore, atau tanda dollar tetapi identifier tidak dapat dimulai dengan digit/angka.

**Aturan Penamaan Variabel**

- Nama variabel harus diawali dengan sebuah huruf atau garis bawah (underscore) "_"

- Nama variabel hanya boleh mengandung karakter alpha-numeric dan underscore (a-Z, 0-9, dan _ )
- Nama variabel tidak boleh mengandung spasi.

# 4. String di PHP

Variabel string digunakan untuk nilai yang mengandung karakter string. Pada contoh berikut, skript PHP memberikan string "Hello World" pada variabel string bernama $txt :

```
<?php
$txt="Hello World";
echo $txt;
?>
```

Keluaran kode tersebut adalah : Hello World

Hanya ada satu operator string di PHP. *Concatenation operator* (.). digunakan untuk menggabungkan dua string. Contoh :

```
<?php
$txt1="Hello World";
$txt2="1234";
echo $txt1 . " " . $txt2;
?>
```

Keluaran kode tersebut adalah : Hello World 1234

# 5. Operator di PHP

Operator digunakan untuk menentukan sebuah nilai dengan melakukan suatu prosedur, atau dengan suatu operasi dari beberapa nilai. Nilai yang digunakan dalam sebuah operasi disebut operand. Penjumlahan adalah sebuah contoh sederhana dari suatu operasi.

Sebagai contoh : 2 + 4

2 dan 4 adalah sebuah operand. Ekspresi ini akan menghasilkan 6.

**Arithmetic Operator (Operator Aritmatika)**

PHP menggunakan operator arithmetic dasar :

| Operator | Aksi | Contoh | Penejelasan |
|---|---|---|---|
| + | Penjumlahan | 5 + 9 | Menghasilkan nilai : 14 |
| - | Pengurangan | 9 - 4 | Menghasilkan nilai : 5 |
| * | Perkalian | 5 * 6 | Menghasilkan nilai : 30 |
| / | Pembagian | 20 / 4 | Menghasilkan nilai : 5 |
| % | Modulus | 9 % 4 | Menghasilkan nilai : 1 |

PHP mengabaikan karakter spasi dalam sebuah operasi. Ekspresi $x = 5 + 9 dengan dengan $x=5+9 akan menghasilkan nilai yang sama. Penggunaan spasi disesuaikan dengan keinginan dari para user.

### Unary Operator

Tanda minus ( - ) digunakan dengan sebuah nilai numerik tunggal untuk menegatifkan sebuah bilangan (untuk membuat negatif bilangan postif atau membuat postif bilangan negatif).

Contoh :

```
$x = 2;
$y = -$x;              // $y = -2

$i = -234;
$j = -$i;    // $j = 234
```

### Variable Assignment Operator

Tanda sama dengan ( = ) digunakan untuk mengatur atau menetapkan nilai suatu variabel. Oleh karena itu tanda tersebut dikenal sebagai operator penugasan.

Contoh :

```
$x = 1;
$y = x + 1;
$luas = $panjang * $lebar;
$statements = "Yes";
```

Variabel disebelah kiri tanda ( = ) akan diberikan nilai dari ekspresi disebelah kanan tanda ( = ).

#### Comparison Operator (Operator Perbandingan)

Operator perbandingan digunakan untuk menguji suatu kondisi. Ekspresi yang menggunakan operator perbandingan akan selalu menghasilkan nilai boolean, yaitu antara true (benar) atau false (salah).

Contoh :

```
$i = 4;

if ($i < 6) print "akan dilakukan pencetakan";
// ekspresi '$i < 6' adalah benar

if ($i > 6) print "tidak akan tercetak";
// ekspresi '$i > 6' adalah salah
```

Beberapa operator perbandingan yang lain adalah sebagai berikut :

| Operator | Arti | Contoh | Menghasilkan benar (true) ketika : |
|---|---|---|---|
| == | Sama dengan | $i == $j | $i dan $j mempunyai nilai yang sama |
| < | Kurang dari | $i < $j | $i kurang dari $j |

| | | | |
|---|---|---|---|
| > | Lebih dari | $i > $j | $i lebih dari $j |
| <= | Kurang dari atau sama dengan | $i <= $j | $i kurang dari atau sama dengan $j |
| >= | Lebih dari atau sama dengan | $i >= $j | $i lebih dari atau sama dengan $j |
| != | Tidak sama dengan | $i != $j | $i tidak sama dengan $j |
| <> | Tidak sama dengan | $i <> $j | $i tidak sama dengan $j |
| === | Identik | $a === $b | Benar jika $a sama dengan $b, dan keduanya memiliki type data yang sama.(hanya dalam PHP4) |

Ingat !

Tanda ( == ) merupakan operator perbandingan untuk menguji suatu variabel sedangkan tanda ( = ) adalah operator penugasan untuk memberikan nilai kepada suatu variabel. Perhatikan dua contoh dibawah ini.

Contoh salah !

```
$i = 3;
if($i = 5) print "lima";
// akan mencetak lima. Pernyataan $i=3 akan diabaikan

$i = 3;
if(5 = $i) print "lima";
/* parse error terdapat kesalahan. PHP berusaha memberikan nilai
$i kepada bilangan 7
*/
```

Contoh Benar !

```
$i = 3;
if($i == 5) print "lima";
// $i == 5 menghasilkan nilai false (salah) sehingga pernyataan
// print  "lima" tidak dijalankan.

$i = 3;
if(5 == $i) print "lima";
// 5 == $i menghasilkan nilai false (salah) sehingga pernyataan
// print  "lima" tidak dijalankan.

$a = "7";
$b = 7.00;
```

```
print ($a == $b);                // mencetak 1 (true)
print (($a == $b) and (gettype($a) == gettype($b))); // mencetak 0
```

### Logical Operator (Operator Logika)

Operator logika digunakan untuk mengkombinasikan kondisi, sehingga beberapa kondisi dapat dievaluasi atau diperiksa dalam sebuah ekspresi. Sebagai contoh logika AND akan bernilai true jika semua kondisi benar. Tabel berikut ini menunjukkan semua anggota dari operator logika :

| Operator | Contoh | Bernilai benar jika : |
|---|---|---|
| AND / and | $i && $j atau $i AND $j | $i dan $j bernilai bernilai benar |
| OR / or | $i \|\| $j atau $i OR $j | Salah satu atau kedua variabel bernilai benar |
| XOR | $i XOR $j | Salah satu variabel bernilai benar, tetapi tidak keduanya benar |
| NOT | !$i | $i tidak bernilai benar |

Contoh :

```
$i = 1;
$j = 2;
$k = 3;

if($i==1 && $j==2 && $k==3) print "akan tercetak";
// akan mengeksekusi pernyataan print

if($i==1 OR $k==3) print "akan tercetak";
// akan mengeksekusi pernyataan print

if($i==1 XOR $j==2) print "akan tercetak";
// tidak mengeksekusi pernyataan print karena kedua variabel //bernilai benar

if !($i==1 && $k==3) print "akan tercetak";
// tidak akan mengeksekusi pernyataan print

if (($i==1 && $k==3) XOR ($i==1 || $j=2) XOR ($i==1)) print "akan tercetak";
// akan mengeksekusi pernyataan print
```

### String Concatenation Operator

Tanda titik ( . ) sebagai operator concatenate (penggabung) digunakan untuk menggabungkan dua atau lebih nilai string menjadi sebuah string tunggal.

Contoh :

```
$subjek = "saya";
```

```
$predikat = "sedang belajar";
$objek = "PHP";
$kalimat = $subjek." ".$predikat." ".$objek;

print $kalimat;                 // akan mencetak saya sedang belajar PHP
print "$kalimatversi 3";        // akan akan mencetak 3
print "$kalimat versi 3";       // akan mencetak saya sedang belajar PHP versi 3
print "${kalimat} versi 3";// akan mencetak saya sedang belajar PHP versi 3

$bilangan = "<B>1</B> ";
$bilangan .= "<I>2</I> ";
$bilangan .= "<U>3</U>";

print $bilangan;      // akan mencetak 1 2 3
```

### Variable Assignment Shortcut

Pada PHP dimungkinkan untuk melakukan penggunaan jalan pintas untuk operator pada pernyataan penugasan dimana operand pertama adalah sebuah variabel dan hasilnya disimpan pada variabel yang sama.

| Contoh | Ekivalen dengan |
|---|---|
| $x += $y | $x = $x + $y |
| $x -= $y | $x = $x - $y |
| $x *= $y | $x = $x * $y |
| $x /= $y | $x = $x / $y |
| $x %= $y | $x = $x % $y |
| $x &= $y | $x = $x & $y |
| $x \|= $y | $x = $x \| $y |
| $x ^= $y | $x = $x ^ $y |
| $x .= $y | $x = $x . $y |
| $x >>= 2 | $x = $x >> 2 |
| $x <<= 2 | $x = $x << 2 |
| $x++ | $x = $x + 1 |
| $x-- | $x = $x - 1 |

Contoh :

```
$x = 10;    // $x bernilai 10
$x++;       // $x bernilai 11
```

```
$x = 10;    // $x bernilai 10
++$x;       // $x bernilai 11
```

tetapi,

```
$x = 10;    // $x bernilai 10
$y = $x++;          // $x bernilai 11 tetapi $y bernilai 10
$x = 10;    // $x bernilai 10
$y = ++$x;          // $x dan $y bernilai 11
            // penugasan terjadi setelah penambahan
```

## 6. *Control Structures* di PHP

Skrip PHP terdiri dari rangkaian pernyataan. Sebuah pernyataan dapat berupa *assignment*, pemanggilan fungsi, sebuah *loop*, pernyataan kondisional atau bahkan pernyataan kosong. Pernyataan biasanya diakhiri dengan semikolon. Sebagai tambahan, pernyataan-pernyataan dapat dikelompokkan menjadi suatu kelompok pernyataan menggunakan kurung kurawal ( {} ). Sebuah kelompok pernyataan merupakan sebuah pernyataan juga.

- IF

  Syntax : if (expr) statement
  Contoh : - if ($a > $b) `print "a is bigger than b";`
  - Jika statemen lebih dari satu maka :

```
if ($a > $b) {
  print "a is bigger than b";
  $b = $a;
}
```

  - Else

```
if ($a > $b) {
  print "a is bigger than b";
} else {
  print "a is NOT bigger than b";
}
```

  - Elseif

```
if ($a > $b) {
  print "a is bigger than b";
} elseif ($a == $b) {
  print "a is equal to b";
} else {
  print "a is smaller than b";
}
```

  - Switch

Pernyataan switch mirip dengan rangkaian pernyataan IF dengan ekspresi yang sama. Pernyataan switch digunakan untuk membandingkan variabel yang sama (atau ekspresi) dengan banyak nilai yang berbeda, dan menjalankan kode-kode yang berbeda tergantung pada nilai mana variabel tersebut sama.

Sangat penting untuk memahami bagaimana pernyataan switch dieksekusi agar terhindar dari kesalahan. Pernyataan switch dieksekusi per pernyataan. Di awal, tidak ada kode yang dieksekusi. Ketika pernyataan *case* sesuai dengan ekspresi pada switch, PHP mulai mengeksekusi pernyataan-pernyataan tersebut. PHP terus mengeksekusi pernyataan-pernyataan tersebut hingga akhir blok switch, atau pada saat pertama kali bertemu pernyataan break. Jika tidak ada pernyataan break, PHP akan mengeksekusi pernyataan-pernyataan pada case berikutnya. Contoh:

```
switch ($i) {
    case 0:
        print "i equals 0";
    case 1:
        print "i equals 1";
    case 2:
        print "i equals 2";
}
```

Pada pernyataan switch, kondisi (ekspresi) hanya diperiksa sekali dan hasilnya dibandingkan dengan setiap pernyataan case.

- While

Perulangan *while* merupakan perulangan yang paling sederhana di PHP. Bentuk dasar pernyataan while adalah :

```
while (expr) statement
```

Pada while, PHP mengeksekusi pernyataan-pernyataan bersarang (*nested statement(s)*) berulang-ulang, selama ekspresi yang dievaluasi bernilai benar (*TRUE*). Nilai ekspresi tersebut diperiksa setiap saat di awal perulangan. Jika hasil evaluasi ekspresi adalah salah (*FALSE*) sejak awal, pernyataan-pernyataan bersarang tersebut tidak akan dijalankan meskipun sekali.

Contoh :

```
$i = 1;
while ($i <= 10) {
    print $i++;  /* the printed value would be
                    $i before the increment
                    (post-increment) */
}
```

- Do ... while

Perulangan do..while loops hamper sama dengan perulangan while, kecuali kebenaran ekspresi dicek di akhir iterasi. Perbedaan mendasar dari perulangan while adalah iterasi pertama pada do…while pasti akan dijalankan.

Contoh :

```
$i = 0;
do {
  print $i;
} while ($i>0);
```

- For

`Syntax` : for (expr1; expr2; expr3) statement

Ekspresi pertama (expr1) dievaluasi (dieksekusi) sekali di awal perulangan. Di awal setiap iterasi, expr2 dievaluasi. Jika benar, perulangan dilanjutkan dan pernyataan-pernyataan bersarang dieksekusi. Jika salah, perulangan dihentikan. Di akhir setiap iterasi, expr3 dievaluasi (dieksekusi).

Contoh :

```
for ($i = 1; $i <= 10; $i++) {
    print $i;
}
```

# 7. Array di PHP

**Array sederhana**

Dalam PHP, sebuah variabel dapat dinyatakan sebagai sebuah tempat untuk sebuah nilai tunggal. Sedangkan Array adalah sebuah tempat untuk sekumpulan nilai. Sebuah array terdiri dari sejumlah element, yang masing-masing memiliki sebuah nilai - data yang tersimpan pada elemen array tersebut - dan sebuah key atau index, dimana elemen tersebut dapat dirujuk. Normalnya, sebuah index berupa integer. Secara default, array adalah basis nol, artinya elemen pertama dari array memiliki index nol. akan tetapi index dapat juga berupa string.

Bentuk sederhana array terdiri dari serangkaian elemen yang bertanda dimulai dari nol dan bertambah secara sekuensial. Sebagai contoh sebuah array bernama $branch, setiap elemen berisi nama kota cabang sebuah perusahaan.

| $branch[0] | $branch[1] | $branch[2] | $branch[3] | $branch[4] |
|---|---|---|---|---|
| "Semarang" | "Surabaya" | "Medan" | "Bandung" | "Yogyakarta" |

Array dalam PHP dapat berisi elemen dari sejumlah tipe data yang berbeda. Artinya array dalam PHP tidak harus memiliki tipe data yang sama. Setiap elemen dapat berupa tipe data apa saja.

Ada tiga jenis array di PHP:

- Numeric array – Array dengan dengan kunci ID numerik
- Associative array – Array dimana setiap kunci ID berasosiasi dengan sebuah nilai
- Multidimensional array - Array yang menyimpan satu atau lebih array

**Inisialisasi array**

Ada banyak cara untuk melakukan inisialisasi sebuah array. Cara pertama yang sederhana adalah cukup dengan memberikan nilai kepada variabel array.

```
$branch[] = "Semarang";
$branch[] = "Surabaya";
$branch[] = "Medan";
```

Jika tanda kurung siku pada variabel array tidak diberikan nilai index, maka secara default maka element sebenarnya bernilai index 0,1,2,.... contoh dibawah akan menghasilkan array yang sama dengan contoh diatas.

```
$branch[0] = "Semarang";
$branch[1] = "Surabaya";
$branch[2] = "Medan";
```

Dalam prakteknya, pemberian index dilakukan secara sekuensial atau berurutan. Tetapi dilain hal dapat dilakukan peng-indekkan secara acak sesuai keinginan user.

```
$branch[20] = "Semarang";
$branch[22] = "Surabaya";
$branch[23] = "Medan";
print $branch[23] // print Medan
```

Array tersebut memiliki tiga buah elemen juga tetapi indeksnya merupakan bilangan acak yaitu 20, 22, 23.

Jika menginginkan jumlah dari elemen array yang terdapat pada sebuah variabel array, dapat digunakan fungsi count(). Fungsi tersebut mengembalikan nilai fungsi berupa integer yang menyatakan jumlah elemen array.

```
$branch[20]   = "Semarang";
$branch[23]   = "Surabaya";
$branch[] = "Medan";              // memiliki indeks 24
                                  // bilangan indeks kedua setelah
                                  // bilangan indeks terbesar
print count ($branch)             // print 3
print $branch[]                   // print nothing
print $branch[24]                 // print Medan
```

Cara lain untuk menginisialisasi array adalah dengan konstruksi array array(). Nilai dikirimkan kedalam array yang akan diberikan.

```
$branch = array("Semarang", "Surabaya", "Medan");
print $branch[2];  // print Medan
```

Jika user ingin mengesampingkan indeks secara default, operator (=>) dapat digunakan untuk memberikan indeks spesifik untuk elemen array. Pada contoh sebelumnya $branch memiliki tiga elemen dengan indeks 0, 1, dan 2. jika user menginginkan array dengan basis satu (indeks dimulai dari 1, 2, 3, ...), maka dapat dituliskan dengan menggunakan operator (=>).

```
$branch = array(1 => "Semarang", "Surabaya", "Medan");
$city = array("Solo", 7 => "Gresik", "Brastagi");
print $branch[3];  // print Medan
print $city[8];      // print Brastagi
```

# 8. Form di PHP

Contoh Form :

```
<html>
<body>
  <form action="welcome.php" method="post">
        Name: <input type="text" name="name" />
        Age: <input type="text" name="age" />
        <input type="submit" />
  </form>
</body>
</html>
```

Contoh halaman HTML di atas mengandung dua field input dan sebuah tombol submit. Ketika user mengisi form dan mengklik tombol submit, data form akan dikirim ke file.

File "welcome.php" :

```
<html>
<body>
  Welcome <?php echo $_POST["name"]; ?>.<br />
  You are <?php echo $_POST["age"]; ?> years old.
</body>
</html>
```

Keluaran dari skrip tersebut adalah :

Welcome John.

You are 28 years old.

Untuk mengambil data form di PHP digunakan variabel $_GET atau $_POST.