Mengenal Pernyataan Kontrol

A. Tujuan

a. Siswa mempelajari pernyataan if dalam pemrograman php
b. Siswa mempelajari pernyataan if-else dalam pemrograman php
c. Siswa mempelajari pernyataan if-else if dalam pemrograman php
d. Siswa mempelajari pernyataan switch-case dalam pemrograman php
e. Siswa mempelajari pernyataan while dalam pemrograman php
f. Siswa mempelajari pernyataan do-while dalam pemrograman php
g. Siswa mempelajari pernyataan for dalam pemrograman php
h. Siswa mempelajari pernyataan break dalam pemrograman php
i. Siswa mempelajari pernyataan continue dalam pemrograman php

B. Dasar Teori

Pernyataan if

Pernyataan if biasa dipakai untuk mengambil keputusan berdasarkan suatu kondisi. PHP memiliki tiga macam bentuk if :

- if saja
- if-else
- if-else if

a) Bentuk if saja

Bentuk pertama if berupa :

```
if (ekspresi)
      pernyataan
```

Pada bentuk ini, bagian *pernyataan* akan dijalankan hanya kalau bagian *ekspresi* bernilai besar.

Contoh penerapan if misalnya untuk menentukan diskon. Sebagai contoh, diberikan ketentuan bahwa bila pembeli berbelanja melebihi atau sama dengan 100.000, maka ia akan mendapat diskon.

diskon.php

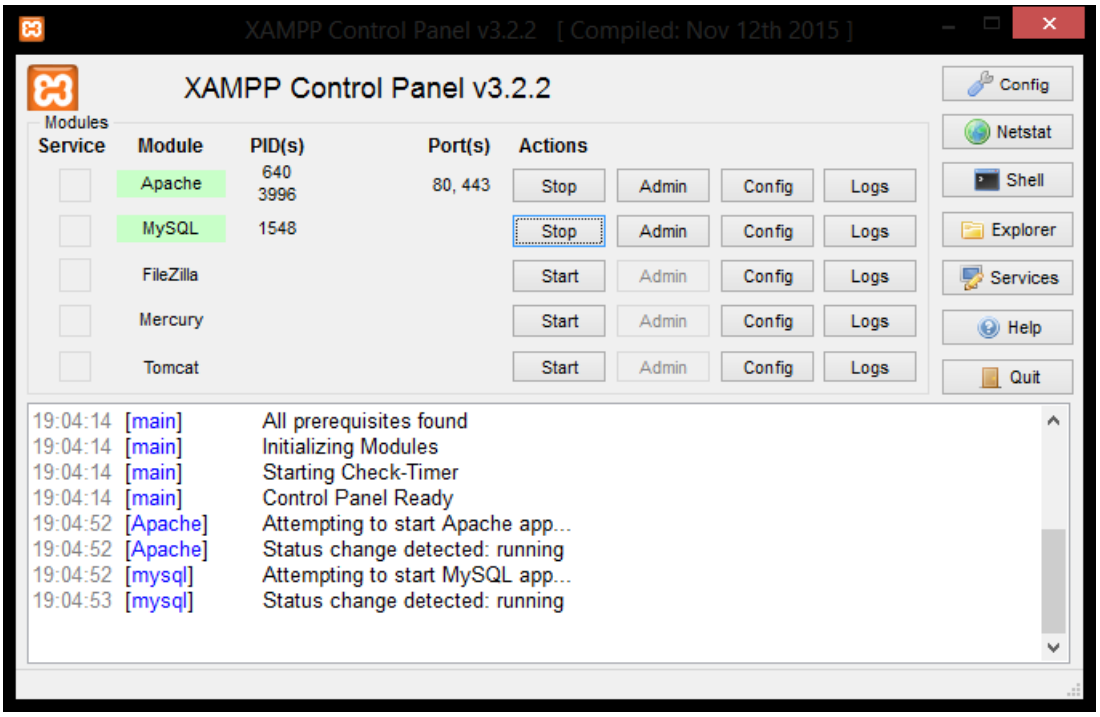
```
<html>
<head>
<title>Contoh Penentuan Diskon<title>
</head>
<body>

    <?php
    $total_beli = 200000;
    $keterangan = "Tak dapat diskon";

    if ($total_beli >= 100000)
        $keterangan = "Dapat diskon";

    print("$keterangan <br />");
    ?>
</body>
</html>
```

- Simpanlah file tersebut dengan nama diskon.php pada folder belajar (modul sebelumnya)
- Bukalah browser kalian dan ketikkan alamat http://localhost/nama_folder/nama_file.php. Dari kasus di atas setelah kita membuka browser pada field alamat maka ketikkan **http://localhost/belajar/diskon.php**.
- Jangan lupa, XAMPP nya diaktifkan pada Apache dan MySQL



- Ubahlah skrip di atas agar menampilkan tampilan keterangannya yaitu “Dapat Diskon” atau “Tak Dapat Diskon”.

Pada kode di atas, penentuan keterangan dilakukan dengan memberikan nilai “Tidak ada diskon” ke variabel keterangan. Selanjutnya, pernyataan if di atas akan membuat variabel keterangan diisi dengan “Dapat Diskon” hanya kalau ekspresi $total_beli > 100000 bernilai benar. Tentu saja pada contoh ini, ekspresi tersebut akan bernilai benar mengingat variabel total_beli diisi dengan 200000. Dengan demikian bila kalian memanggil skrip tersebut pada browser maka akan tampil “Dapat diskon”. Namun, setelah kalian mencobanya, ubahlah isi total_beli menjadi 50000 (jangan lupa untuk melakukan penyimpanan) dan kemudian tes lagi pada browser. Kalian akan melihat tampilan “Tak dapat diskon”. Hal ini memberikan gambaran bahwa dengan menggunakan if kalian bisa mengatur sutau tindakan berdasarkan suatu kondisi.

Sekiranya bagian pernyataan pada if berupa sejumlah pernyataan, kalian bisa meletakkan pernyataan-pernyataan tersebut dalam { }. Contoh berikut merupakan pengembangan skrip sebelumnya, yang memungkinkan pemakai memasukkan besar pembelian dan komputer akan menghitung jumlah yang harus dibayar.

b) Bentuk if-else

diskon2.php

```
<html>
<head>
<title>Contoh Penentuan Diskon</title>
</head>
<body>
    <form>
    Besar Pembelian:
    <input type=text name=total_beli><br /><br />
    <input type=submit value="Tentukan Diskon">
    </form>

<?php
IF (ISSET($total_beli))
{
    $total_beli = intval($total_beli);
    $diskon = 0;
    if ($total_beli >= 100000)
        $diskon = intval(0.1 * $total_beli);
    printf("Diskon      = %d <br />", $diskon);
    printf("Pembayaran  = %d <br />", $total_beli - $diskon);
}
?>
</body>
</html>
```

Skrip di atas ubahlah menjadi pernyataan if-else jika total_beli < 100000 maka tidak akan mendapatkan diskon atau diskon = 0. Dan ubahlah juga skrip di atas agar muncul tampilan Diskon dan Pembayaran ketika kalian mengisikan Besar Pembelian dan menekan tombol Tentukan Diskon !

c) Pernyataan if-else if

Pernyataan if-else if sangat bermanfaat untuk melakukan pengambilan keputusan yang melibatkan banyak alternatif. Sebagai contoh, pada skrip berikut, if-else if digunakan untuk menentukan nama hari sekarang (diambil dari tanggal sistem).

hariini.php

```
<html>
<head>
<title>Latihan Menentukan Nama Hari</title>
</head>

    Hari ini:
    <?php
        $nama_hari = date("l");
        if ($nama_hari == "Sunday")
            print("Minggu");
        elseif ($nama_hari == "Monday")
            print("Senin");
        elseif ($nama_hari == "Tuesday")
            print("Selasa");
        elseif ($nama_hari == "Wednesday")
            print("Rabu");
        elseif ($nama_hari == "Thursday")
            print("Kamis");
        elseif ($nama_hari == "Friday")
            print("Jumat");
        else
            print("Sabtu");
    ?>
</body>
</html>
```

d) Pernyataan switch

Bila kalian menjumpai persoalan yang melibatkan banyak alternatif semacam penentuan hari pada skrip sebelumnya, kalian juga bisa menggunakan pernyataan switch. Secara umum, bentuk pernyataan ini adalah sebagai berikut:

```
switch (ekspresi)
{
   case ekspresi_case_1 :
       pernyataan_1;
       break;
   case ekspresi_case_2 :
       pernyataan_2;
       break;
   case ekspresi_case_3 :
```

```
    ......
    default :
        pernyataan_n;
}
```

Pada pernyataan switch, ekspresi_case_1 akan diperiksa terlebih dahulu. Bila nilainya cocok dengan nilai ekspresi maka pernyataan_1 akan dijalankan dan kemudian eksekusi dilanjutkan ke pernyataan yang terletak sesudah tanda penutup switch (}). Kalau tidak cocok, pembandingan nilai ekspresi dan ekspresi_case_2 akan dilakukan. Jika hasil pada pembandingan yang kedua ini benar maka pernyataan_2 akan dijalankan, dan kemudian eksekusi dilanjutkan ke pernyataan yang terletak sesudah tanpa penutup tanda penutup switch (}). Sekiranya tak ada bagian ekspresi_case yang cocok maka pernyataan_n yang mengikuti kata default akan dijalankan.

Skrip berikut merupakan modifikasi dari skrip hariini.php. Pernyataan if-else if diganti dengan switch.

hariini2.php

```php
<html>
<head>
<title>Latihan Menentukan Hari</title>
</head>

    Hari ini:
    <?php
        $nama_hari = date("l");
        switch ($nama_hari)
        {
            case "Sunday" :
                print("Minggu");
                break;
            case "Monday" :
                print("Senin");
                break;
            case "Tuesday" :
                print("Selasa");
                break;
            case "Wednesday" :
                print("Rabu");
                break;
            case "Thursday" :
                print("Kamis");
                break;
            case "Friday" :
                print("Jumat");
                break;
            default :
                print("Sabtu");
    }
    ?>
</body>
</html>
```

Pada pernyataan switch, pernyataan break memegang peran yang sangat penting. Pernyataan break-lah yang membuat eksekusi dilanjutkan ke pernyataan yang terletak sesudah tanda penutup switch (}).

Untuk melihat efek break, cobalah untuk menuliskan skrip berikut dan kemudian memanggilnya melalui browser.

harikerja.php

```
<html>
<head>
<title>Latihan Menentukan Hari</title>
</head>

    Hari ini:
    <?php
        $nama_hari = date("l");
        switch ($nama_hari)
        {
            case "Monday" :
            case "Tuesday" :
            case "Wednesday" :
            case "Thursday" :
            case "Friday" :
                print("Hari kerja");
                break;
            case "Saturday" :
            case "Sunday" :
                print("Hari libur");
    }
    ?>
</body>
</html>
```

Pada contoh di atas, apabila isi variabel nama_hari berupa “Monday”, “Tuesday”, “Wednesday”, “Thursday”, atau “Friday”, maka pernyataan yang dijalankan adalah:

```
print(“Hari kerja”);
break;
```

Pernyataan break akan mengakhiri switch. Adapun jika isi variabel nama_hari berupa “Saturday” atau “Sunday”, maka pernyataan yang akan dijalankan adalah:

```
print(“Hari libur”);
```

e) Pernyataan while

Pernyataan while merupakan salah satu pernyataan yang berguna untuk melakukan suatu pengulangan. Sebagai contoh, kalian bisa menampilkan bilangan 1 sampai dengan 25 cukup dengan menggunakan kode yang pendek.

Bentuk pernyataan ini:

```
while (ekspresi)
{
   pernyataan_pernyataan
}
```

Pernyataan while akan memeriksa nilai ekspresi terlebih dahulu. Jika bernilai benar maka pernyataan-pernyataan yang terdapat dalam { } akan dijalankan dan kemudian ekspresi dievaluasi lagi. Proses ini diulang terus-menerus sampai ekspresi bernilai salah.

Contoh berikut menunjukkan penggunaan while untuk menampilkan bilangan 1 hingga 25.

bilangan.php

```
<html>
<head>
<title>Menampilkan Bilangan 1-25</title>
</head>
<body>
    <?php
        $bilangan = 1;
        while ($bilangan <= 25)
        {
            print("$bilangan <br />");
            $bilangan++ ;
        }
    ?>
</body>
</html>
```

Pada contoh skrip di atas, isi variabel bilangan berperan dalam melakukan pengulangan perintah yang berada dalam { }. Pada keadaan seperti ini harus dipastikan bahwa ada pernyataan yang mengubah nilai bilangan sehingga suatu ketika kondisi dalam while (yaitu pada contoh ini, $bilangan <= 25) bernilai salah. Jika tidak maka akan terjadi pengulangan selamanya.

f) Pernyataan do-while

Pernyataan do-while mempunyai kegunaan yang serupa dengan pernyataan while. Bentuk pernyataan ini:

```
do
{
   pernyataan_pernyataan
}  while (ekspresi);
```

Pengulangan akan berakhir jika ekspresi (yang diuji sesudah pernyataan-pernyataan dijalankan) bernilai salah.

Berdasarkan diagram alir di atas terlihat bahwa paling tidak pernyataan yang berada dalam { } akan dieksekusi sekali.

Skrip berikut memperlihatkan penggunaan do-while untuk menampilkan bilangan 1 sampai dengan 25.

dowhile.php

```php
<html>
<head>
<title>Menampilkan Bilangan 1-25</title>
</head>
<body>

    <?php
        $bilangan = 1;
        do
        {
            print("$bilangan <br />");

            $bilangan++;
        }   while ($bilangan < 26);
    ?>
</body>
</html>
```

g) Pernyataan for

Pernyataan for juga merupakan pernyataan yang biasa digunakan untuk menangani pengulangan proses. Pernyataan ini mempunyai bentuk sebagai berikut:

```
for (ekspr1; ekspr2; ekspr3)
{
   pernyataan_pernyataan;

   ekspr3;
}
```

Jadi:

- ekspr1 adalah ekspresi untuk memberi nilai awal terhadap variabel yang akan digunakan untuk melakukan pencacahan pengulangan
- ekspr2 berlaku sebagai kondisi untuk menentukan pengulangan terhadap pernyataan yang ada di dalam { } akan dilakukan atau tidak
- ekspr3 digunakan untuk mengatur nilai variabel yang digunakan dalam ekspr1.

Contoh skrip yang menggunakan pernyataan for dapat dilihat di bawah ini:

for.php

```
<html>
<head>
<title>Menampilkan Bilangan 1-25</title>
</head>
<body>
    <?php
        for ($bilangan = 1; $bilangan <=25; $bilangan++)
            print("$bilangan <br />");
    ?>
</body>
</html>
```

Tampak bahwa dalam kasus menampilkan bilangan 1 sampai dengan 25, pernyataan for lebih sederhana

h) Pernyataan break

Berikut dengan pengulangan proses, PHP menyediakan pernyataan break. Kegunaannya adalah untuk keluar dari suatu kalang (proses yang berulang).

Untuk melihat efek pernyataan break pada suatu kalang, kalian bisa mencoba program berikut ini:

break.php

```
<html>
<head>
<title>Contoh untuk Memperlihatkan Efek Break</title>
<body>
    <?php
        for ($i = 1; $i <= 25; $i++)
        {
            print("$i <br />");

            if ($i == 10)
            break;
        }
        print("Selesai <br />");
    ?>
</body>
</html>
```

Tampak bahwa bilangan 11 hingga 25 tidak ditampilkan, disebabkan setelah bilangan 10 ditampilkan, break mengakhiri pernyataan for. Selanjutnya, eksekusi dilanjutkan ke pernyataan setelah for, yaitu:

```
print("Selesai <br />");
```

Kalian juga bisa memberikan angka di belakang kata break. Misalnya:

```
break 2;
```

Hal seperti ini berguna sekiranya kalian meletakkan break di dalam pernyataan switch, tetapi kalian maksudkan bukan untuk keluar dari switch, melainkan untuk keluar dari kalang. Untuk lebih jelasnya, kalian bisa mencoba skrip berikut ini:

break2.php

```
<html>
<head>
<title>Efek Pernyataan Break 2</title>
<body>
    <?php
        for ($i = 1; $i <= 25; $i++)
        {
            switch($i)
            {
                case 5:
                    print("5 - break 1 <br />");
                    break 1;
                case 7:
                    print("7 - break 2 <br />");
                    break 2;
                default:
                    print("$i <br />");
                    break;
            }
        }
        print("Selesai <br />");
    ?>
</body>
</html>
```

Pada kode di atas, break (tanpa angka) ataupun break 1 mempunyai makna yang sama yaitu keluar dari switch. Namun, break 2 berarti keluar dari switch dan sekaligus keluar dari for.

Bila break 2 diganti dengan break 1 atau break saja, angka 8, 9, hingga 25 akan ditampilkan.

i) Pernyataan continue

Pernyataan continue digunakan untuk menuju ke iterasi (putaran) berikutnya pada pernyataan-pernyataan yang terkait dengan pengulangan. Pada pernyataan for, ekspresi ketiga yang terletak di dalam tanda kurung akan dijalankan terlebih dahulu dan kemudian baru menguji ekspresi kedua yang terletak dalam tanda kurung. Pada selain while dan do-while, eksekusi akan dilanjutkan ke pengujian ekspresi yang terletak di dalam tanda kurung.

Kalian bisa mencoba skrip berikut ini untuk memahami efek continue:

continue.php

```
<html>
<head>
<title>Contoh untuk Memperlihatkan Efek Break</title>
<body>
    <?php
        for ($i = 1; $i <= 25; $i++)
        {
            if ($i >= 10 and $i <= 15)
            continue;

            print("$i <br />");

            $i++;
        }
    ?>
</body>
</html>
```

Tampak bahwa angka 10 hingga 15 tidak ditampilkan disebabkan oleh perintah continue, berdasarkan instruksi:

if ($i >= 10 and $i <=15)
    continue;

Seperti halnya break, kalian juga bisa meletakkan sebuah angka di belakang kata continue. Untuk melihat efeknya, kalian bisa mencoba skrip berikut:

continue2.php

```
<html>
<head>
<title>Contoh untuk Memperlihatkan Efek Break</title>
<body>
    <?php
        for ($i = 1; $i <= 5; $i++)
        {
            print("Proses for. Iterasi ke-$i <br />");

            for ($j = 1; $j <=5; $j++)
            {
                if ($j == 3)
                    continue 1;
                print($j);
            }
            print("<br />");
        }
    ?>
</body>
</html>
```

j) Pernyataan exit

Pernyataan exit berguna untuk mengakhiri pengeksekusian. Dengan dijalankannya perintah ini, segala kode baik PHP maupun HTML tidak dikirim ke browser. Untuk melihat efek exit, kalian bisa mencoba skrip berikut:

exit.php

```
<html>
<head>
<title>Contoh untuk Memperlihatkan Efek Exit</title>
<body>
    <?php
        for ($i = 1; $i <= 25; $i++)
        {
            print("$i <br />");

            if ($i == 10)
            exit;
        }
        print("Selesai <br />");
    ?>
</body>
</html>
```

Terlihat bahwa hasil terakhir berupa angka 10. Pernyataan:

```
print(“Selesai <br />”);
```

tidak dijalankan sama sekali. Sebenarnya, tidak hanya itu. Kode HTML yang terletak sesudah ?> juga tidak dijalankan.

k) Sintaks Alternatif

PHP juga menawarkan sintaks alternatif terhadap pernyataan-pernyataan kontrol seperti if, while, dan for. Sebagai pengganti kurung buka ({), kalian bisa menggunakan tanda:, sedangkan kurung tutup (}) dapat diganti dengan:

- endif; (untuk if)
- endfor; (untuk for)
- endwhile; (untuk while)
- endforeach; (untuk foreach)
- endswitch; (untuk switch)

C. Latihan

Pada beberapa kode program di atas, silahkan kalian praktikkan pada Notepad++ atau Sublime. Tujuannya adalah agar kalian tetap belajar meskipun kalian sedang melakukan kegiatan Prakerin. Selamat Belajar !!!