Dasar-dasar Pemrograman PHP dan Database PostgreSQL

irsan riza

Dasar-dasar Pemrograman PHP dan Database PostgreSQL

irsan riza

Published: Januari 2013

Daftar Isi

PHP

03 Januari 2013
22:06

Apa kepanjangan PHP? Awalnya diketahui PHP adalah "Personal Home Page", namun semenjak dikenalkannya PHP versi 3, kepanjangannya diganti menjadi rekursif (pengulangan) yaitu "PHP: Hypertext Preprocessor". Baiklah sepertinya tidak penting hanya membahas singkatannya, sekarang mari kita mulai dengan materi yang lebih menantang. PHP merupakan bahasa pemrograman web yang bekerja di sisi server (server side scripting). Berbeda dengan html, css, dan javascript yang bekerja di sisi client, pendekatan pemrograman dengan cara ini memiliki keuntungan tersendiri yaitu syntax yang kita buat tidak akan terbuka pada sisi client yang berarti mengurangi celah keamanan pada website yang kita bangun. Selain PHP memang terdapat banyak sekali pilihan bahasa pemrograman web untuk server side scripting seperti asp, java server page (jsp), coldfusion, dan lain sebagainya. Bahkan dahulu kala sebelum menggunakan php, BPS sediri pernah mengalami era coldfusion, namun semuanya berubah semenjak negara api menyerang. Bukan, sebenarnya alasan memilih PHP untuk website yang akan kita bangun adalah karena PHP mmerupakan bahasa yang paling banyak digunakan di seluruh dunia dan mendukung pemrograman yang jauh lebih powerful dan kompleks. Struktur bahasanyapun lebih mudah untuk dipahami karena mengadopsi struktur pemrograman C, perl, dan java.

Saat ini PHP sudah mencapai versi 5 yang mendukung penuh gaya pemrograman terbaru yang sedang trend yaitu "object oriented programming" (OOP). Adapun turunan dari penggunaan gaya OOP ini bisa bermacam-macam, contohnya adalah konsep MVC (Model View Controller) yang diusung pada framework Yii. Lebih lanjut mengenai Yii ini akan dibahas pada kesempatan lain.

Pada pemrograman PHP kita membutuhkan PHP interpreter yang biasanya melekat pada web server. Program yang kita buat dalam bahasa PHP dapat kita sisipkan diantara syntax HTML dengan menggunanan tag <?php dan diakhiri dengan tag ?>. File ini nantinya harus disimpan dengan ekstensi *.php dan diletakkan pada folder yang bisa diakses oleh web server, asumsinya kita menggunakan web server apache pada platform windows, maka lokasi foldernya adalah "C:/xampp/htdocs/".

Cara Mudah Belajar PHP

06 Desember 2012
21:46

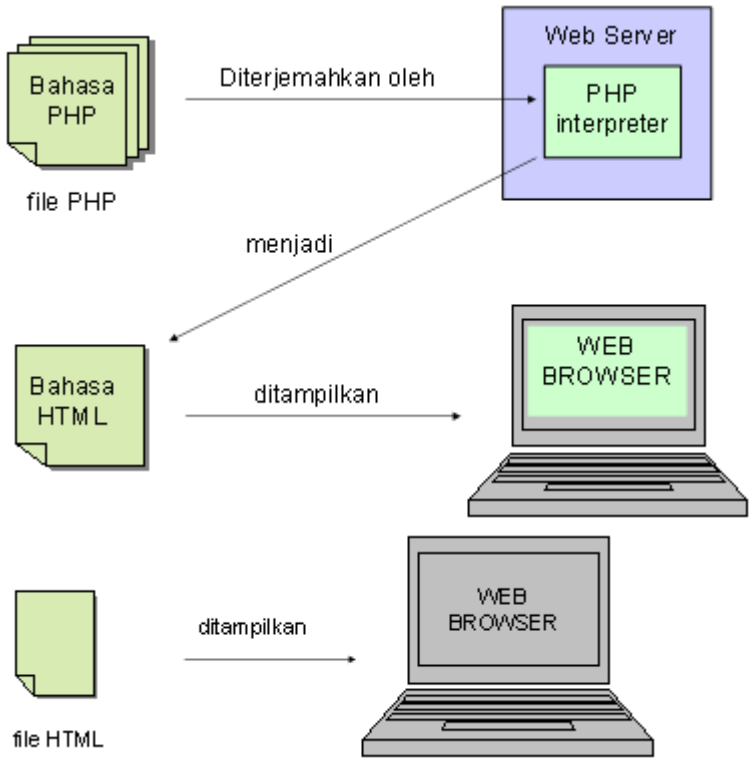
Apa itu PHP? Dan apa hubungannya dengan web site(s)?
Bagaimana PHP bekerja?
Bagaimana cara membuatnya?
Software apa yang harus diinstall untuk membuat dan menjalankan PHP?

Apa itu PHP?

06 Desember 2012
21:47

PHP = PHP Hypertext Processing = Personal Home Page
PHP → bahasa pemrograman yang dikhususkan untuk membuat halaman web
Bedanya dengan html?

Sedangkan PHP adalah file yang akan dieksekusi/dibaca terlebih dahulu oleh PHP interpreter pada web server baru kemudian ditampilkan oleh web browser
html adalah file yang dikenali dan akan dibuka oleh web browser (seperti mozilla, internet explorer atau opera)



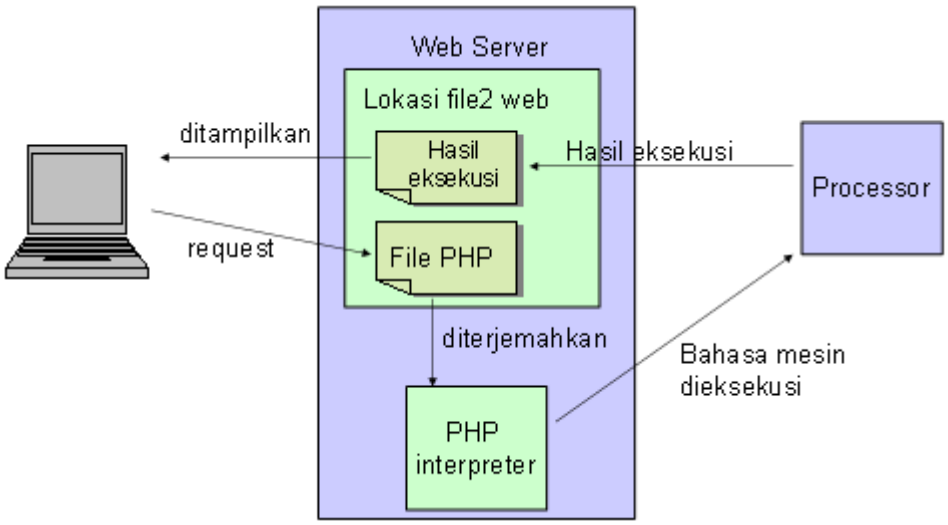
PHP dan Webserver

06 Desember 2012
21:53

PHP interpreter Adalah penterjemah bahasa PHP menjadi bahasa yang dimengerti oleh mesin komputer yang akan mengeksekusi perintah yang kita buat
Web Server Adalah mesin server atau komputer biasa yang diinstall dengan software khusus (misal: apache,IIS,server2go,vertrigo,dll) untuk menempatkan halaman web yang kita buat agar dapat diakses secara mudah dari komputer lain (yang terhubung dengan jaringan komputer server)

Cara Kerja PHP:

- User (seperti kita) yang menggunakan web broser untuk me-request (meminta untuk membuka) halaman web
- Web server mencari file yang diminta
- Bila file mengandung perintah PHP, PHP interpreter akan menterjemahkannya sebelum dieksekusi
- Hasil eksekusi akan dikirim ke web browser dan siap dibaca oleh user



Membuat PHP

06 Desember 2012
21:57

- Belajar bahasa PHP yang umum
- Sama seperi saat membuat program dengan bahasa lain (VB, java, C#, pascal, dll) tergantung text editor (program pembuat file text) yang kita gunakan

Tools yang diperlukan:

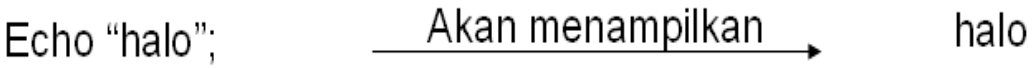
- Web server (misal: apache)
- PHP interpreter (catatan: untuk memudahkan, biasanya apache&PHP telah dibundle dalam 1 installer “apache2triad”)
- PHP editor (optional, tergantung kebutuhan. Misal: dreamweaver, PHP editor, notepad++, atau bahkan NOTEPAD…)

- File yang mengandung perintah PHP (yang kita buat) harus disimpan dalam ekstensi "*.php"
- diletakkan di folder yang bisa diakses webserver misalnya "C:\xampp\htdocs\"
- gunakan tag PHP ("<?php … ?>")di awal dan di akhir perintah PHP
- Jangan lupa menggunakan ";" pada setiap akhir statement

Istilah penting

06 Desember 2012
22:02

Echo digunakan untuk menampilkan string ke web browser:



- ==,>,<,>=,<=,!= → perbandingan dua nilai menghasilkan true/false
- & → and
- | → or
- = → memberi nilai variabel kiri dengan nilai pada variabel kanan
- { } →sebagai begin dan end untuk statement yang berkelompok (if … then, while)

Hello World!

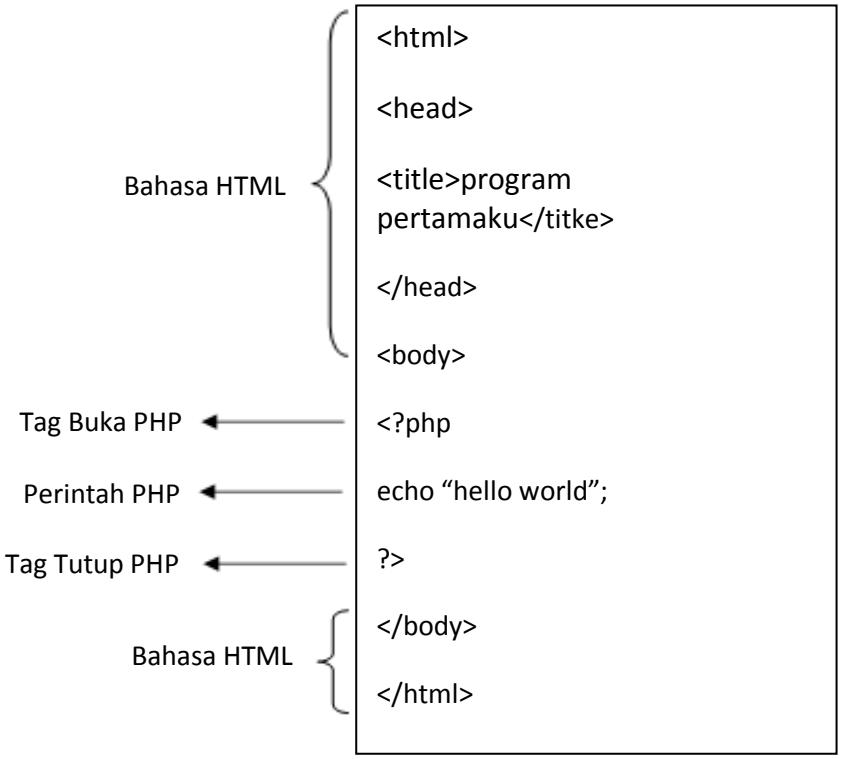
06 Desember 2012
22:05

Buka notepad atau text editor sederhana lainnya

Ketikkan kode sebagai berikut:

```
<html>

<head>

<title>program pertamaku</titke>

</head>

<body>

<?php

echo "hello world";

?>

</body>

</html>
```

Penjelasan Program:



Melihat Hasil Program

06 Desember 2012
22:12

- Simpan file berisikan kode PHP tadi ke “C:\xampp\htdocs\hello.php”
- Buka web browser (mozilla firefox)
- Ketikkan alamat http://localhost/hello.php
- Viola..! Anda berhasil membuat program “hello world”

Struktur PHP

03 Januari 2013
22:36

Seperti yang kita ketahui sebelumnya bahwa untuk menuliskan perintah php kita harus menggunakan awalan tag <?php dan diakhiri dengan tag ?>. File-file yang kita buat yang mengandung unsur perintah dalam bahasa PHP harus kita simpan dengan ekstensi "*.php" dan diletakkan di folder yang bisa diakses oleh web server. Diasumsikan bahwa kita menggunakan xampp sebagai web server lokal, maka lokasi penyimpanan file php secara default seharusnya berada di "C:/xampp/htdocs/". Kita bisa membuat sub folder dibawahnya untuk mempermudah manajemen penyimpanan file.
Dalam penulisan perintah php, setiap satu statement dalam php diakhiri dengan tanda titik koma ";". Untuk mengelompokkan beberapa statement misalnya untuk penggunaan if, for , ataupun fungsi

maka digunakan tanda bracket { dan }. Untuk mendefinisikan variabel kita menggunakan tanda $ pada awal nama variabel untuk membedakannya dengan nama fungsi, prosedur, atau objek. Contoh:

```
<?php
Echo "hai..";
$a = 1;
If ($a== 1) {
        Echo "nilai dari variabel a adalah = ", $a;
        Echo "<br>";
}
?>
```

Untuk keperluan dokumentasi program, PHP juga memiliki kode untuk menambahkan baris komentar yaitu tanda "//" utuk menandakan 1 baris komentar. Sedangkan untuk beberapa baris komentar digunakan tanda /* untuk memulai komentar, dan tanda */ untuk mengakhiri komentar.

```
<?php
// ini baris komentar
/* ini untuk beberapa baris komentar
Komentar baris kedua*/
?>
```

Variabel dan tipe data

03 Januari 2013
22:36

Pada PHP kita tidak perlu mendefinisikan variabel secara eksplisit, karena variabel tersebut akan didefinisikan secara langsung saat kita memberikan nilai pertama kali. Dan tipe data dari variabel itupun akan secara otomatis diberikan sesuai dengan nilai yang kita masukkan. Misalnya kita memasukkan nilai "1" (dengan tanda petik) maka tipe data variabel akan menjadi string. Dan bila kita memasukkan nilai 1, maka tipe datanya akan menjadi numerik.

Untuk mendukung konsep pemrograman berorientasi objek PHP memiliki empat macam variabel yang dapat digunakan sesuai kebutuhan. Keempat jenis variabel itu adalah variabel lokal, global, static, dan parameter.

Variabel lokal adalah variabel yang didefinisikan di dalam sebuah fungsi dan hanya berlaku di dalam fungsi tersebut. Variabel ini biasanya akan dihapus setelah fungsi selesai dijalankan, dan nilainya tidak dapat diakses dari luar fungsi, begitu pula sebaliknya.

```
<?php
$a = 5; // global scope
function myTest()
{
echo $a; // local scope
}
myTest();
?>
```

Contoh diatas tidak akan menghasilkan output apa-apa, karena variabel $a didefinisikan diluar fungsi, dan di dalam fungsi variabel $a tersebut tidak pernah didefinisikan sehingga tidak dapat diproses lebih lanjut dengan perintah "echo".

Variabel global adalah variabel yang didefinisikan diluar suatu fungsi dan dapat digunakan di bagian manapun dari program kecuali didalam sebuah fungsi. Untuk dapat menggunakan variabel ini di dalam fungsi tertentu, maka variabel ini harus didefinisikan ulang sebagai variabel global di dalam fungsi tersebut. Dengan cara sebagai berikut:

```
<?php
$a = 5;
$b = 10;
function myTest()
{
global $a, $b;
$b = $a + $b;
}
myTest();
echo $b;
?>
```

Cara lain memanggil variabel global dalam lingkup fungsi adalah dengan menggunakan variabel $GLOBALS['nama_variabel'] sebagai berikut:

```
<?php
$a = 5;
$b = 10;
function myTest()
{
$GLOBALS['b'] = $GLOBALS['a'] + $GLOBALS['b'];
}
myTest();
echo $b;
?>
```

Kedua cara itu akan menghasikan output yang sama yaitu "15".

Variabel static adalah variabel yang ada di dalam suatu fungsi dan nilainya akan tetap tersimpan walaupun fungsi tersebut selesai dijalankan. Normalnya variabel di suatu fungsi akan langsung terhapus saat fungsi tersebut selesai dijalankan. Namun ada kalanya kita ingin variabel tersebut tetap tidak dihapus, agar bisa digunakan lagi kemudian. Jadi ketika fungsi tersebut dipanggil kembali, variabel static ini tetap akan menyimpan nilai seperti sebelumnya. Sebagai cacatan: variabel static ini tetap bersifat seperti variabel lokal, dimana nilainya hanya bisa diakses dari dalam fungsi, dan tidak berlaku untuk penggunaan di luar fungsi yang bersangkutan. Contoh penggunaannya:

```
static $nama_variabel;
```

Jenis variabel terakhir adalah parameter. Variabel parameter ini kadang disebut juga dengan nama argumen, yaitu sebuah variabel di dalam suatu fungsi yang nilainya dipoer dari luar fungsi (baik dari variabel global lain maupun penulisan nilai secara langsung). Contoh:

```
<html>
<body>
<?php
function add($x,$y)
{
$total=$x+$y;
return $total;
}
echo "1 + 16 = " . add(1,16);
?>
</body>
</html>
```

Output dari kode tersebut adalah:

```
1+16=17
```

Menampilkan Teks

03 Januari 2013
22:36

Kode PHP bisa digunakan untuk menampilkan tulisan pada web browser yaitu dengan menggunakan perintah echo. Cara lain menampilkan tulisan di browser adalah dengan menutup tag php ?> maka tulisan setelahnya akan dieksekusi layaknya kode HTML biasa.

Contoh:

```
<?php
If ($a==1) {
        Echo "variabel a bernilai 1";
} else {
        Echo "variabel a bukan 1";
}
?>
```

Atau

```
<?php
If ($a==1) {
?>
Variabel a bernilai 1
<?php
} else {
?>
Variabel a bukan 1
<?php
}
?>
```

Kedua contoh diatas akan menghasilkan output yang sama. Hanya saja pada cara kedua akan lebih efisien untuk digunakan bila kita perlu menuliskan output teks yang cukup banyak ke browser. Daripada mengirimkan nilai yang ingin ditampilkan dengan menggunakan echo, alangkah lebih baik bila menggunakan sistem buka tutup tag php untuk mengurangi beban server dan mempercepat prosesnya. Namun untuk output teks yang tidak terlalu banyak seperti contoh diatas sepertinya tidak perlu repot-repot untuk membuka menutup tag php karena selisih waktu yang didapat tidak terlalu signifikan.
Untuk menuliskan nilai variabel pada web browser, kita bisa langsung menuliskan nama variabelnya setelah perintah echo. Contohnya separti berikut:

```
<?php
$a = "saya";
$b = "ganteng";
Echo "kata orang " . $a . " " . $b;
?>
```

Pada contoh diatas untuk menuliskan beberapa variabel atau nilai dalam 1 perintah echo digunakan kode titik ".". Kode ini digunakan untuk melakukan penggabungan nilai teks (string). Jadi pada perintah diatas, penerjemah perintah php akan melakukan 2 kegiatan yaitu penggabungan nilai string terlebih dahulu, baru kemudian menampilkannya. Lain halnya bila kita mengganti kode titik "." menjadi koma "," maka penerjemah php hanya akan melakukan 1 kegiatan yaitu menampilkan semua variabel atau nilai yang mengikutinya. Penggunaan masing-masing cara ini tergantung kondisi dan kebutuhan. Perbedaan seperti ini sebenarnya tidak terlalu signifikan untuk server-server modern yang ada sekarang ini, namun ada baiknya kita lebih berhati-hati terhadap efisiensi kode yang kita buat agar website kita tetap stabil dalam kondisi apapun.

Array

03 Januari 2013
22:37

Sebuah variabel biasanya hanya bisa digunakan untuk satu nilai saja. Namun ada kalanya kita ingin mengelompokkan beberapa nilai yang berbeda dalam variabel yang sama untuk mempermudah melakukan proses secara sekaligus (batch processing). Misalnya kita memiliki beberapa variabel tunggal seperti:

```
$cars1="Saab";
$cars2="Volvo";
$cars3="BMW";
```

Mungkin bila kita hanya memiliki tiga veriabel seperti diatas, maka pemrosesannya seperti looping untuk menampilkan nilai masing-masing variabel tidak akan mengalami kendala berarti. Namun bayangkan bila kita memiliki lebih dari 3 nilai, misalnya 100 atau 500. tentunya akan menyulitkan bila harus menuliskan perintahnya sebanyak 500 kali. Untuk kasus seperti ini kita bisa menggunakan variabel array. Dalam sebuah array kita bisa menyimpan banyak nilai dalam satu variabel, dan untuk mengaksesnya kita bisa merujuk ke index dari array tersebut.
Ada tiga jenis array: numeric array, association array, dan multidimensional array. Numeric array adalah array yang masing-masing elemennya ditandai dengan nomor index (dimulai dari 0). Associative array menyimpan masing-masing elemennya dengan sebuah nama untuk menngantikan nomor index. Multidimensional array adalah array didalam array, digunakan untuk mendefinisikan banyak nilai dengan struktur yang lebih kompleks.

Contoh mendefinisikan numeric array:

```
$cars=array("Saab","Volvo","BMW","Toyota");
```

Atau

```
$cars[0]="Saab";
$cars[1]="Volvo";
$cars[2]="BMW";
$cars[3]="Toyota";
```

Contoh cara mengaksesnya:

```
<?php
$cars[0]="Saab";
$cars[1]="Volvo";
$cars[2]="BMW";
$cars[3]="Toyota";
echo $cars[0] . " and " . $cars[1] . " are Swedish cars.";
?>
```

Syntax diatas akan menghasilkan output sebagai berikut:

```
Saab and Volvo are Swedish cars.
```

Contoh mendefinisikan associative array:

```
$ages = array("Peter"=>32, "Quagmire"=>30, "Joe"=>34);
```

Atau

```
$ages['Peter'] = "32";
$ages['Quagmire'] = "30";
$ages['Joe'] = "34";
```

Contoh cara menggunakannya:

```
<?php
$ages['Peter'] = "32";
$ages['Quagmire'] = "30";
$ages['Joe'] = "34";
echo "Peter is " . $ages['Peter'] . " years old.";
?>
```

Kode diatas akan menghasilkan output sebagai berikut:

```
Peter is 32 years old.
```

Contoh mendefinisikan multidimensional array:

```
$families = array
 (
 "Griffin"=>array
 (
 "Peter",
 "Lois",
 "Megan"
 ),
 "Quagmire"=>array
 (
 "Glenn"
 ),
 "Brown"=>array
 (
 "Cleveland",
 "Loretta",
 "Junior"
 )
 );
```

Syntax diatas memiliki struktur yang lebih kompleks dari array biasa, adapun bila digambarkan dengan lebih jelas, struktur dari pendefinisian diatas adalah seperti ini:

```
Array
(
[Griffin] => Array
 (
 [0] => Peter
 [1] => Lois
 [2] => Megan
 )
[Quagmire] => Array
 (
 [0] => Glenn
 )
[Brown] => Array
 (
 [0] => Cleveland
 [1] => Loretta
 [2] => Junior
 )
)
```

Contoh cara mengakses nilai multidimensional array

```
echo "Is " . $families['Griffin'][2] . " a part of the Griffin family?";
```

Outputnya adalah seperti berikut:

```
Is Megan a part of the Griffin family?
```

Fungsi pada PHP

16 Januari 2013
9:45

Pada PHP kita juga bisa membuat fungsi, tidak jauh berbeda dengan fungsi yang pernah kita bahas pada materi sebelumnya mengenai javascript. Struktur utama penulisan fungsi php adalah:

```
<?php
Function nama_fungsi () {
Perintah_php;
}
?>
```

Dan untuk memanggil fungsi tersebut, cukup dengan menuliskan nama fungsinya secara langsung diikuti dengan tanda kurung "()". Contoh:

```
<?php
Function cetak_saya() {
        Echo "helow.. PHP";
}
cetak_saya();
?>
```

Seperti pada bahasan sebelumnya kita mengatahui bahwa variabel global dan lokal memiliki batasannya masing-masing dalam penggunaannya di dalam fungsi. Selain itu kita bisa menggunakan parameter untuk melakukan passing variabel ke dalam suatu fungsi. Cara mendefinisikan fungsi dengan suatu parameter adalah sebagai berikut:

```
<?php
Function nama_fungsi ($variabel_parameter) {
Perintah_php;
}
?>
```

Contoh penulisan fungsi dengan parameter:

```
<?php
Function cetak_nama ($nama) {
        Echo $nama;
}
 Echo "cara pertama: ";
cetak_nama("irsan keren");
 Echo "cara kedua: ";
$nama_irsan = "irsan ganteng";
cetak_nama($nama_irsan );
 Echo "cara ketiga: ";
$nama_irsan = "irsan ";
cetak_nama($nama_irsan . " waaaww");
?>
```

Ada kalanya fungsi yang kita definisikan menggunakan parameter, namun pada pemanggilan fungsinya ternyata tidak menyertakan variabel yang akan di-pass sebagai parameter. Untuk itu terkadang kita perlu mendefinisikan nilai default dari parameter yang kita definisikan. Caranya adalah sebagai berikut:

```
<?php
Function Cetak_nama($nama = "irsan") {
        Echo $nama;
}
$guweh = "riza";

Cetak_nama();
// perintah di atas ini akan menghasilkan "irsan"
 Cetak_nama($guweh);
// perintah di atas ini akan menghasilkan "riza"
?>
```

Pada contoh diatas kita bisa melihat bahwa cara kedua dan ketiga menggunakan variabel yang didefinisikan sebagai parameter di dalam fungsi. Namun nantinya nilai variabel ini tidak mengalami perubahan di luar fungsi. Artinya perubahan apapun yang kita lakukan terhadap variabel parameter di dalam fungsi tidak akan mengubah nilai variabel asalnya di luar fungsi. Kondisi passing variabel ini biasa disebut passing by value. Agar perubahan tersebut dapat dipertahankan di luar fungsi (biasa juga disebut passing by reference), maka kita perlu mengubah cara mendefinisikan parameternya yaitu dengan menambahkan karakter "&" diawal setiap variabel yang ingin di passing by reference. Contoh:

```
<?php
Function Hitung1($a, $b) {
        $b++;
        $a = $a + $b;
}

Function Hitung2(&$a, &$b) {
        $b++;
        $a = $a + $b;
}

$x = 1;
$y = 3;
Hitung1($x, $y);
// disini nilai variabel x = 1, dan y = 3 (tetap)

Hitung2($x, $y);
// disini nilai variabel x = 5, dan y = 4 (berubah)
// nilai variabel $a dan $b tidak bisa diakses dari luar fungsi karena merupakan variabel lokal
?>
```

Selain passing by reference, kita juga bisa mengembalikan nilai dari dalam suatu fungsi agar dapat diakses dari luar fungsi. Cara ini menggunakan perintah return, agar suatu nilai atau variabel dapat di-pass keluar fungsi (sebagai nilai dari fungsi) untuk digunakan diluar dari deklarasi fungsi. Contoh:

```
<?php
Function Jumlah ($a, $b) {
        Return $a + $b;
}
$x = 2;
$y = 3;
$z = Jumlah ($x, $y);
//disini nilai variabel $z = 5
?>
```

Tidak terbatas pada variabel tunggal, return suatu fungsi juga bisa merupakan variabel array.

Perintah Kondisional

03 Januari 2013
22:38

Dalam membuat program, ada kalanya kita ingin menjalankan perintah yang berbeda untuk beberapa kemungkinan kondisi yang berbeda. Dalam hal ini kita bisa menggunakan perintah if atau switch. Perintah if dapat dituliskan dengan cara sebegai berikut:

```
<?php
if (kondisi) perintah;

If (kondisi) {
        Perintah;
}
?>
```

If dapat dituliskan dengan kedua cara diatas, cara pertama bila hanya ada satu perintah yang akan dieksekusi saat kondisi bernilai true. Sedangkan cara kedua digunakan bila lebih dari satu perintah yang harus dieksekusi saat kondisi bernilai true.

Contoh:

```
<?php
If ($a < $b) {
        Echo "a lebih kecil dari b";
}
?>
```

Bila kondisi yang ingin kita definisikan lebih dari satu untuk beberapa baris perintah yang sama maka kita bisa menggunakan tambahan kondisi dengan perantara and atau or. Perlu diingat bahwa untuk menuliskan kondisi lebih dari satu, perlu dibuatkan tanda kurung () untuk masing-masing kondisi, dan keseluruhan kondisi. Beberapa jenis operator logical untuk kondisi adalah:

Example	**Name**	**Result**
$a and $b	And	**TRUE** if both $a and $b are **TRUE**.
$a or $b	Or	**TRUE** if either $a or $b is **TRUE**.
$a xor $b	Xor	**TRUE** if either $a or $b is **TRUE**, but not both.
! $a	Not	**TRUE** if $a is not **TRUE**.
$a && $b	And	**TRUE** if both $a and $b are **TRUE**.
$a || $b	Or	**TRUE** if either $a or $b is **TRUE**.

Perbedaan dari &&, ||, and, or adalah && dan || derajatnya lebih tinggi dibanding dengan and, or. Sehingga bila dituliskan berdampingan maka && dan || akan lebih dulu dieksekusi dibanding dengan and, or. && derajatnya lebih tinggi dari ||, begitu juga and lebih tinggi derajatnya dibanding or. Contoh:

a or b && c --> a or (b && c)
a or b and c --> a or (b and c)
a || b and c --> (a || b) and c

true or false and false = true
true or (false and false) = true
(true or false) and false = false
true || false and false = false
true || false && false = true

Contoh diatas bukan untuk menunjukkan rumitnya operator logical pada php, tapi hanya untuk sekedar mengingatkan bahwa kita harus konsisten dalam menggunakan operator, apakah ingin menggunakan lambang || dan &&, atau lebih cenderung untuk menuliskan AND dan OR.

```
<?php
If ((kondisi1) or (kondisi2)) {
        Perintah;
}
?>
```

Bila kita ingin mendapatkan lebih dari satu kali percobaan kondisi, dengan kata lain kita ingin mendapatkan hasil yang berbeda untuk beberapa kondisi yang berbeda, maka kita bisa menggunakan if () elseif () else. Cara penulisannya adalah sebagai berikut:

```
<?php
If (kondisi1) {
        Perintah1;
} elseif (kondisi2) {
        Perintah2;
} elseif (kondisi3) {
        perintah3
} else {
        perintah4
}
?>
```

Pada contoh diatas, bila kondisi satu bernilai true, maka perintah1 akan dieksekusi sedangkan kondisi dan perintah lain diabaikan. Bila kondisi satu false maka akan dicek kondisi2. Bila kondisi2 bernilai trus maka perintah2 akan dieksekusi, kondisi dan perintah lain diabaikan. Bila kondisi 1 dan 2 false maka akan dicek kondisi 3, dan bila kondisi 3 bernilai true maka perintah3 akan dieksekusi, perintah4 diabaikan. Bila semua kondisi dari 1 sampai 3 false, maka perintah4 akan dieksekusi. Selain if perintah lain yang berhubungan dengan perintah kondisional yaitu switch.

```
<?php
if ($i == "a") {
   echo "i = a";
} elseif ($i =="b") {
   echo "i = b";
} elseif ($i == "c") {
   echo "i = c";
}

switch ($i) {
   case "a":
      echo "i = a";
      break;
   case "b":
      echo "i = b";
      break;
   case "c":
      echo "i = c";
      break;
}
?>
```

Pada contoh di atas penggunaan if ataupun switch akan menghasilkan output yang sama. Perlu diperhatikan bahwa pada setiap case, selalu diakhiri dengan perintah "break;". Hal ini untuk menghentikan proses pengecekan terhadap case selanjutnya apabila kondisi case di atasnya sudah terpenuhi.

Looping/Pengulangan

22 Januari 2013
5:14

Looping seringkali kita perlukan dalam pemrograman dengan php, terutama bila kita berurusan dengan database atau array. Looping/pengulangan akan dilakukan dalam jumlah tertentu, ataupun selagi kondisi yang kita tentukan masih berlaku. Untuk looping ini kita bisa menggunakan perintah for atau while.

```
<?php
for (expr1; expr2; expr3) {
   statement
}
?>
```

Sama seperti bahasan pada javascript, expr1 hanya akan dieksekusi sekali pada saat looping dimulai, biasanya untuk mendefinisikan nilai awal. Expr2 akan dievaluate (biasanya berupa kondisi yang bernilai true atau false) setiap kali akan memulai satu putaran looping, apabila nilainya true maka looping akan dilanjutkan, bila bernilai false maka looping akan dihentikan. Expr3 akan dieksekusi setiap kali satu putaran looping berakhir, biasanya untuk memanipulasi nilai awal agar kondisi pada expr2 bisa berubah setelah beberapa kali putaran tertentu. Contoh penggunaan:

```
<?php
/* example 1 */
 for ($i = 1; $i <= 10; $i++) {
   echo $i;
}

/* example 2 */
 for ($i = 1; ; $i++) {
   if ($i > 10) {
      break;
   }
   echo $i;
}

/* example 3 */
 $i = 1;
for (; ; ) {
   if ($i > 10) {
      break;
   }
   echo $i;
   $i++;
}

/* example 4 */
 for ($i = 1, $j = 0; $i <= 10; $j += $i, print $i, $i++);
?>
```

Keempat contoh penggunaan for diatas sebenarnya akan menghasilkan output yang sama, hanya saja cara mendefinisikannya berbeda. Contoh pertama merupakan contoh penggunaan for yang ideal, dan paling sering ditemui karena penggunaan syntax yang efisien dan readable oleh manusia. Contoh kedua tidak menggunakan expr2 yaitu kondisi yang menentukan apakah looping akan berlanjut atau tidak, sehingga secara default looping akan terus dijalankan sampai kita mendefinisikan perintah break. Contoh ketiga benar-benar menghilangkan semua expr baik nilai awal, kondisi, maupun manipulasinya. Sehingga kita harus mendefiniskan sendiri secara manual nilai awal di luar looping, menentukan kondisi break di dalam looping dan juga memanipulasi secara manual nilai awal di setiap perulangan looping. Cara ini tidak salah, walaupun mungkin tidak terlalu lumrah, namun ada kalanya kita mengalami suatu kondisi yang membuat kita lebih nyaman untuk mengatur kondisi perulangan dari dalam looping itu sendiri. Contoh terakhir yang keempat merupakan contoh syntax yang paling efisien. Penggunaan variabel $j pada contoh ini hanya untu membuktikan bahwa kita bisa mengeksekusi beberapa perintah untuk setiap segmen expr (ekspresi). Namun cara ini mungkin kurang begitu populer karena kode yang tertulis disana tidak newbie friendly, sulit dibaca oleh manusia.
Khusus untuk membantu looping dalam sebuah array, ada fungsi khusus yang didefinisikan sebagai berikut:

```
foreach (array_expression as $value)
   statement

foreach (array_expression as $key => $value)
   statement
```

Array_expression merupakan ekspresi yang mendefinisikan sebuah array, ekspresi ini bisa berupa variabel dengan tipe array, atau bisa juga berupa pemberian nilai array secara langsung. Contoh pertama menjelaskan bahwa pada setiap nilai record array yang didefinisikan pada array_expression akan dikembalikan nilai setiap recordnya sebagai variabel $value. Sedangkan pada contoh kedua, untuk setiap record pada array akan didefinisikan dengan dua variabel yaitu $key sebagai index array dan variabel $value sebagai nilai array.

```
<?php
$arr = array(1, 2, 3, 4);
foreach ($arr as &$value) {
   $value = $value * 2;
} // $arr sekarang bernilai array(2, 4, 6, 8)
unset($value); // menghapus referensi pada variabel $value yang digunakan pada foreach
?>
```

Berikut contoh cara mengeprint nilai array dengan menggunakan while dan foreach:

```
<?php
$arr = array("one", "two", "three");
reset($arr);
while (list(, $value) = each($arr)) {
   echo "Value: $value<br />\n";
}
foreach ($arr as $value) {
   echo "Value: $value<br />\n";
}
?>
```

```
<?php
$arr = array("one", "two", "three");
reset($arr);
while (list($key, $value) = each($arr)) {
   echo "Key: $key; Value: $value<br />\n";
}
foreach ($arr as $key => $value) {
   echo "Key: $key; Value: $value<br />\n";
}
?>
```

kedua cara diatas baik yang menggunakan while maupun yang menggunakan foreach akan menghasilkan output yang sama. Berikut contoh beberapa cara untuk mengeprint setiap record array yang berbeda.

```
<?php
/* foreach example 1: value only */
$a = array(1, 2, 3, 17);

foreach ($a as $v) {
   echo "Current value of \$a: $v.\n";
}

/* foreach example 2: value (with its manual access notation printed for illustration) */
 $a = array(1, 2, 3, 17);

$i = 0; /* for illustrative purposes only */

foreach ($a as $v) {
   echo "\$a[$i] => $v.\n";
   $i++;
}

/* foreach example 3: key and value */
 $a = array(
   "one" => 1,
   "two" => 2,
   "three" => 3,
   "seventeen" => 17
);

foreach ($a as $k => $v) {
   echo "\$a[$k] => $v.\n";
}

/* foreach example 4: multi-dimensional arrays */
$a = array();
$a[0][0] = "a";
$a[0][1] = "b";
$a[1][0] = "y";
$a[1][1] = "z";
```

```
foreach ($a as $v1) {
   foreach ($v1 as $v2) {
      echo "$v2\n";
   }
}

/* foreach example 5: dynamic arrays */

foreach (array(1, 2, 3, 4, 5) as $v) {
   echo "$v\n";
}
?>
```

Dari contoh diatas kita dapat melihat perbedaan cara mengeprint masing-masing record untuk bebera[a jenis array yang berbeda. Untuk multidimensionala array kita juga dapat menberikan beberapa perlakuan saat mengakses setiap recordnya sebagai berikut:

```
<?php
$array = [
   [1, 2],
   [3, 4],
];

foreach ($array as list($a, $b)) {
   // $a contains the first element of the nested array,
   // and $b contains the second element.
   echo "A: $a; B: $b\n";
}
?>
```

Contoh diatas akan menghasilkan output sebegai berikut:

```
A: 1; B: 2
A: 3; B: 4
```

Contoh lain penggunaan list pada looping pemanggilan record array:

```
<?php
$array = [
   [1, 2],
   [3, 4],
];

foreach ($array as list($a)) {
   // Note that there is no $b here.
   echo "$a\n";
}
?>
```

Contoh diatas akan menghasilkan output sebagai berikut:

```
1
3
```

Lalu apa yang terjadi bila kita salah mendefinisikan list yang jumlah nya lebih dari elemen array itu senndiri? Berikut contohnya:

```
<?php
$array = [
    [1, 2],
    [3, 4],
];

foreach ($array as list($a, $b, $c)) {
    echo "A: $a; B: $b; C: $c\n";
}
?>
```

Kode diatas akan menghasilkan output sebagai berikut:

```
Notice: Undefined offset: 2 in example.php on line 7
A: 1; B: 2; C:

Notice: Undefined offset: 2 in example.php on line 7
A: 3; B: 4; C:
```

Setelah kita memahami penggunaan for dan foreach, kita akan mencoba membahas masih mengenai looping, namun kali ini dengan menggunakan perintah while. Struktur dasar penulisan syntax while adalah sebagai berikut:

```
while (expr)
    statement
```

Maksud dari penggunaan while seperti pada struktur diatas adalah untuk mengulangi baris perintah yang diwakili dengan "statement" terus menerus selagi kondisi pada "expr" bernilai true. Saat kondisi "expr" berubah menjadi false, maka looping akan berhenti. Ada kalanya kondisi "expr" akan bernilai false pada saat pertama kali dijalankan yang mengakibatkan baris perintah "statement" tidak akan pernah dijalankan satu kalipun.

Beberapa cara menuliskan perintah while yang memiliki beberapa baris perintah "statement":

```
while (expr):
    statement
    ...
endwhile;
 While (expr) {
   statement
   ...
}
```

Berikut beberapa contoh penggunaan while:

```
<?php
/* example 1 */

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

/* example 2 */

$i = 1;
while ($i <= 10):
   echo $i;
   $i++;
endwhile;
?>
```

Kedua contoh diatas akan menghasilkan output yang sama. Dimana pada contoh pertama variabel $i akan dicetak kemudian nilainya langsung ditambahkan 1 pada baris perintah yang sama. dan pada contoh kedua nilai variabel $i dicetak terlebih dahulu, baru kemudian pada baris selanjutnya ada perintah untuk menambahkan nilai variabel $i.
Ada cara lain untuk menuliskan perintah while, yaitu dengan do-while. Namun cara ini bekerja dengan cara yang berbeda dari while yang kita bahas diatas.

```
<?php
$i = 0;
do {
   echo $i;
} while ($i > 0);
?>
```

Pada contoh diatas kita lihat kondisi while dituliskan di bagian bawah dari looping. Hal ini berarti statement yang berada di dalam looping pasti akan dieksekusi dulu minimal satu kali, baru setelah itu kondisi while diperiksa, apabila bernilai true maka looping akan dilanjutkan, dan bila bernilai false maka looping akan berhenti. Dejavu dengan javascript yang pernah kita bahas pada materi sebelumnya? Ya, do-while pada php ini memiliki konsep yang sama seperti yang ada pada javascript, pada dasarnya apapun bahasa pemrogramannya masing-masing akan memiliki "pola syntax" yang sama, hanya cara penulisan ejaaannya yang berbeda.

Konsep Object Oriented Programming

22 Januari 2013
22:37

Konsep dari pemrograman berorientasi objek (object oriented programming - OOP) yaitu membuat struktur dari program yang merepresentasikan objek seperti pada dunia nyata, secara sederhana ini adalah konsep dimana objek tersebut mempunyai sifat tertentu (atribut), dan memiliki kemampuan

untuk bereaksi terhadap suatu tindakan (function). Objek ini juga bisa mempunyai sifat khusus yang tidak bisa dilihat dari luar dan hanya berlaku untuk intern objek itu sendiri. Pada OOP juga dikenal penurunan sifat pada objek lainnya yang sejenis, dan juga beberapa fungsi yang sama namun bisa menghasilkan output yang berbeda untuk objek yang berbeda.

Contoh sederhana pada objek di dunia nyata misalnya untuk objek "sepeda". Sepeda memiliki atribut seperti warna cat, ukuran roda depan dan belakang, tinggi tempat duduk, dan lain-lain. Sepeda juga memiliki function tertertu misalnya "dikayuh" atau "parkir". Sepeda bisa memiliki sub kelompok seperti sepeda gunung, sepeda lipat, sepeda keranjang, dan lainnya. Masing-masing sepeda memiliki fungsi "dikayuh" namun untuk masing-masing sepeda, terutama yang keompoknya berbeda, akan memiliki sifat yang berbeda pula walaupun nama fungsinya sama-sama "dikayuh", yaitu misalnya pada sepeda gunung daya kayuhnya bisa lebih besar dari sepeda keranjang walaupun tenaga yang kita keluarkan sama.

Dari sedikit penjelasan diatas, kita dapat memahami beberapa konsep utama dalam OOP yaitu: encapsulation, inheritance, polymorphism dan message.

Encapsulation (kapsul) adalah menyembunyikan detail dari objek yang tidak perlu diketahui oleh selain objek itu sendiri, jadi objek hanya dikenal dari sebagian sifatnya, bukan keseluruhannya. Ada atribut atau fungsi yang hanya berlaku dan bisa diakses dari dalam objek itu sediri, dan tidak bisa diakses dari luar objek. Hal ini untuk meminimalisir kompleksitas dari suatu objek. Dari luar objek akan terlihat sederhana, dan penggunaannyapun menjadi sederhana, tanpa melihat apa yang terjadi didalamnya. Untuk analogi dengan konsep sepeda tadi bisa diumpamakan seperti tekanan udara pada roda yang tidak terlihat dari luar namun dari dalam objek itu sendiri memiliki peranan penting untuk menunjang berat sepeda dan pengendaranya.

Inheritance (penurunan sifat) yaitu kondisi dimana sifat-sifat dari suatu objek dapat juga dimiliki oleh objek lain yang secara hirarki merupakan turunan dari objek tersebut. Hal ini memungkinkan beberapa kelas objek yang berbeda memiliki atribut dan fungsi yang sama. Namun penurunan sifat ini tidak bersifat mengikat, karena kelas objek turunannya tetap bisa memiliki sifat sendiri yang berbeda. Analogi pada cerita sepeda seperti sebelumnya adalah seperti dari superclass sepeda, kita bisa mempunyai beberapa kelas turunan seperti sepeda gunung, sepeda keranjang, sepeda lipat, dan lainnya.

Polymorphism (kesamaan bentuk) yang dalam hal ini menggambarkan kesamaan dalam hal fungsi yang digunakan terutama untuk objek yang berbeda kelas dalam satu turunan yang sama. Misalnya sebuah objek dari suatu kelas memiliki fungsi dengan nama dan parameter yang sama, namun cara memproses parameter tadi bisa saja berbeda untuk objek yang berada pada kelas yang berbeda. Sesuai analogi sepeda tadi, kondisi ini bisa diumpamakan seperti sepeda gunung dan sepeda lipat memiliki fungsi yang sama yaitu "dikayuh". Namun ternyata proses "menayuh" dari kedua jenis sepeda ini penanganannya bisa berbeda. Pada sepeda gunung kita bisa menyetel gigi agar lebih enteng dikendarai, sedangkan pada sepeda lipat, tidak ada sub proses memindahkan gigi, dan tenaga yang dikeluarkan untuk "mengayuhnya" diperkirakan sama.

Message (interaksi antar objek) masing-masing objek yang terbentuk dari class tertentu bisa saling berkomunikasi (bertukar nilai variabel atau saling memanggil fungsi). Disini berarti suatu objek tidak terlepas hanya bisa mengendalikan fungsi yang berlaku pada objek itu saja. Bisa juga suatu objek akan memanggil fungsi lain ynang tidak pernah dibuat sebelumnya. Untuk contoh analogi sepeda-nya bisa kita bayangkan seperti ada objek sepeda dengan fungsi "dikayuh" untuk setiap jenis kelas objek. Fungsi ini bisa diakses dari luar objek yaitu misalnya objek "manusia" akan bisa mengakses fungsi "dikayuh" yang ada pada objek sepeda.

Ada dua hal penting yang harus kita ingat dalam pemahaman awal dari pembuatan objek dalam php terutama dalam yii framework yaitu class dan objek itu sendiri. Class merupakan kerangka dari suatu object yang berisi definisi-definisi dari variabel dan fungsi. Sedangkan objek adalah sebuah variabel yang dibentuk dari suatu class. Class dapat merupakan implementasi dari tabel yang terdapat di dalam database. Dalam hal ini class akan berisi fungsi-fungsi yang berkaitan secara langsung dengan operasi database tersebut. Namun ada kalanya juga class mendefinisikan kerangka objek yang tidak berkaitan dengan database. Contoh penulisan class dalam yii adalah sebagai berikut:

```
class Menu extends CActiveRecord
{
	/**
	 * Returns the static model of the specified AR class.
	 * @param string $className active record class name.
	 * @return Menu the static model class
	 */
	public static function model($className=__CLASS__)
	{
		return parent::model($className);
	}

	/**
	 * @return string the associated database table name
	 */
	public function tableName()
	{
		return 'menu';
	}

	/**
	 * @return array validation rules for model attributes.
	 */
	public function rules()
	{
		// NOTE: you should only define rules for those attributes that
		// will receive user inputs.
		return array(
			array('flag', 'numerical', 'integerOnly'=>true),
			array('menu_idn, menu_eng', 'length', 'max'=>30),
			array('aksi, alt_idn, alt_eng', 'length', 'max'=>60),
			array('id_parent, id_menu_kategori', 'safe'),
			// The following rule is used by search().
			// Please remove those attributes that should not be searched.
			array('id, menu_idn, menu_eng, id_parent, aksi, flag, id_menu_kategori, alt_idn,
			alt_eng', 'safe', 'on'=>'search'),
		);
	}

	/**
	 * @return array relational rules.
	 */
```

```
public function relations()
{
	// NOTE: you may need to adjust the relation name and the related
	// class name for the relations automatically generated below.
	return array(
		'idMenuKategori' => array(self::BELONGS_TO, 'MenuKategori',
		'id_menu_kategori'),
	);
}

/**
 * @return array customized attribute labels (name=>label)
 */
public function attributeLabels()
{
	return array(
		'id' => 'ID',
		'menu_idn' => 'Menu Idn',
		'menu_eng' => 'Menu Eng',
		'id_parent' => 'Id Parent',
		'aksi' => 'Aksi',
		'flag' => 'Flag',
		'id_menu_kategori' => 'Id Menu Kategori',
		'alt_idn' => 'Alt Idn',
		'alt_eng' => 'Alt Eng',
	);
}

/**
 * Retrieves a list of models based on the current search/filter conditions.
 * @return CActiveDataProvider the data provider that can return the models based on the
search/filter conditions.
 */
public function search()
{
	// Warning: Please modify the following code to remove attributes that
	// should not be searched.

	$criteria=new CDbCriteria;

	$criteria->compare('id',$this->id);
	$criteria->compare('menu_idn',$this->menu_idn,true);
	$criteria->compare('menu_eng',$this->menu_eng,true);
	$criteria->compare('id_parent',$this->id_parent,true);
	$criteria->compare('aksi',$this->aksi,true);
	$criteria->compare('flag',$this->flag);
	$criteria->compare('id_menu_kategori',$this->id_menu_kategori,true);
	$criteria->compare('alt_idn',$this->alt_idn,true);
	$criteria->compare('alt_eng',$this->alt_eng,true);

	return new CActiveDataProvider($this, array(
```

```
                'criteria'=>$criteria,
        ));
    }

        /**
     * fungsi tambahan
     */
    private function RetrieveMenu($kategori='1', $lang='idn')
    {
        $sql = "select id, menu_$lang as menu, aksi, alt_$lang as alt, id_parent, have_child from
        menu where id_menu_kategori=$kategori and flag=1 order by id asc";
        $hasil=Yii::app()->db->createCommand($sql)->queryall();
        return $hasil;
    }


    private function susunMenu($arr, $parent) {
        foreach  ($arr as $row) {
            if ($row["id_parent"]==$parent) {
                if ($row["have_child"]==TRUE) {
                    $child = Menu::model()->susunMenu($arr,
                    $row["id"]);
                    $menu[] = array('label'=>$row['menu'], 'url'=>$row['aksi'],
                    'items'=>$child);
                } else {
                    $menu[] = array('label'=>$row['menu'], 'url'=>$row['aksi']);
                }
            }

        }
        return $menu;
    }

    public function MenuHorizontal($lang="idn")
    {

        $menuH = Menu::model()->RetrieveMenu("1", $lang);
        $menu = array('items'=>Menu::model()->susunMenu($menuH, "0"));
        return $menu;
    }
}
```

Seperti yang kita lihat pada contoh diatas class "Menu" diawali dengan syntax

```
class Menu extends CActiveRecord
```

Yang berarti kelas "Menu" merupakan turunan dari kelas "CActiveRecord". Sedangkan "CactiveRecord" sendiri merupakan kelas yang telah didefinisikan oleh Yii. Dengan perintah tersebut berarti kelas "Menu" akan memiliki semua sifat (fingsi dan variabel) yang dimiliki oleh kelas "CActiveRecord". Peurunan sifat seperti ini yang kita namakan inheritance dalam OOP.

Class memiliki banyak fungsi yang didefinisikan di dalamnya. Kita bahkan juga bisa mengubah syntax fungsi yang sudah ada dari superclass (parent) agar dapat melakukan proses yang lebih detil dari sebelumnya. Pada contoh diatas memang belum ada yang diganti karena untuk saat ini dirasa belum perlu. Namun cara seperti ini yang dinamakan polymorphism dalam OOP.Dari beberapa fungsi-fungsi ini ada yang bersifat public dan private seperti pada deklarasinya. Fungsi yang bersifat public artinya fungsi tersebut dapat diakses dari luar objek. Fungsi seperti ini dalam OOP yang kita namakan message. Untuk mendeklarasikan objek dari kelas, dan syntax memanggil salah satu fungsinya adalah sebagai berikut:

```
$model=new Menu;
$listMenu = $menu->MenuHorizontal();
```

Berikut cara memanggil fungsi yang terdapat di dalam objek tanpa mendefinisikan objeknya terlebih dahulu

```
$listMenu = Menu::model()->MenuHorizontal();
```

Selain fungsi yang bersifat public yang dapat diakses dari luar, ada juga fungsi yang bersifat private yang berarti fungsi ini tidak akan bisa diakses dari luar suatu objek. Fungsi ini hanya digunakan dari dalam objek itu sendiri untuk menunjang fungsi lain dan sekaligus membuat kode progam yang kita buat lebih reusable dan menyembunyikan detil fungsi-fungsi yang tidak diperlukan dari luar objek untuk membuat objek tersebut terlihat sederhana. sifat seperti ini yang kita namakan encapsulation. Pada contoh class diatas kita melihat fungsi

```
private function RetrieveMenu($kategori='1', $lang='idn')
```

Fungsi ini tidak dapat diakses dari luar objek, karena didefinisikan sebagai private. Hal ini juga dibuat untuk mencegah user berinteraksi langsung dengan database. Dalam pemrograman dengan framework Yii sebisa mungkin kita mendefinisikan semua fungsi yang berkaitan dengan database di dalam objek model. Yang kemudian user hanya akan melihat hasil requestnya dalam bentuk view yang secara fisik sudah tidak merepresentasikan objek lagi. Kenapa harus demikian? Hal ini kita lakukan untuk menurunkan tingkat kompleksitas yang harus dihadapi programmer terutama saat kita mengembangkan aplikasi yang nantinya akan dikelola oleh banyak orang dan atau secara bergantian. Ini juga berguna untuk memisahkan proses bisnis dengan tampilan, saat kita melakukan perubahan pada tampilan, maka kita tidak akan perlu mengubah sedikitpun kode proses bisnis yang lain. Begitu pula sebaliknya bila kita melakukan perubahan proses bisnis untuk perbaikan performance, maka kita tidak perlu mengubah kode pada tampilannya.

Database PostgreSQL

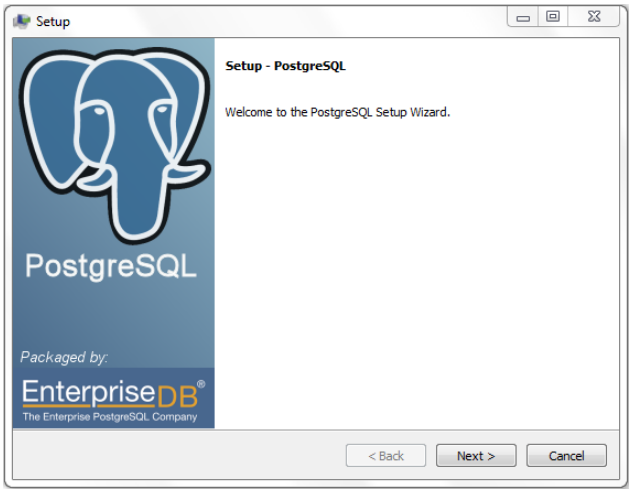
22 Januari 2013
21:15

Untuk menunjang website yang kita buat agar menjadi lebih dinamis, kita memerlukan database untuk menyimpan data yang akan ditampilkan. Ada beberapa alternatif database yang bisa kita gunakan, pada pembuatan website sebelumnya kita menggunakan database server MySQL karena lebih sederhana dan tentunya gratis. Sedangkan kali ini kita juga akan menggunakan database yang bersifat gratis, namun lebih powerful dalam hal ketahanan menyimpan data dalam jumlah besar dan juga mendukung fungsi proses database yang lebih kompleks. Yap, kita akan mencoba menggunakan database server PostgreSQL.
Secara default bila kita menggunakan web server XAMPP, database PostgreSQL belum termasuk dalam bundel aplikasinya. Jadi kita harus menginstallnya secara terpisah. Namun bagi yang belum pernah mencoba menginstalll PostgreSQL tidak usah khawatir karena installer database ini sudah termasuk installer next generation (klik next sampai finish). Namun tidak seperti mySQL yang sudah mempunyai password default , pada PostgreSQL kita harus mendefinisikan password sendiri saat instalasi. jangan sampai lupa untuk mengisikan default password untuk membuka databasenya. Password bebas bisa diisi apa saja, asal JANGAN SAMPAI LUPA. Karena bila password ini sampai lupa akan repot untuk merestorenya kembali.
Untuk pembelajaran ini kita akan mencoba untuk menggunakan database PostgreSQL di komputer lokal bersama dengan web servernya. Mulai dari instalasinya membuat tabel sampai pada membuat backup database. Namun untuk implementasinya di server BPS kita akan mewelatkan bagian membuat database dan tabel, karena kita hanya perlu merestore database kita yang sudah kita buat sebelumnya di server lokal.

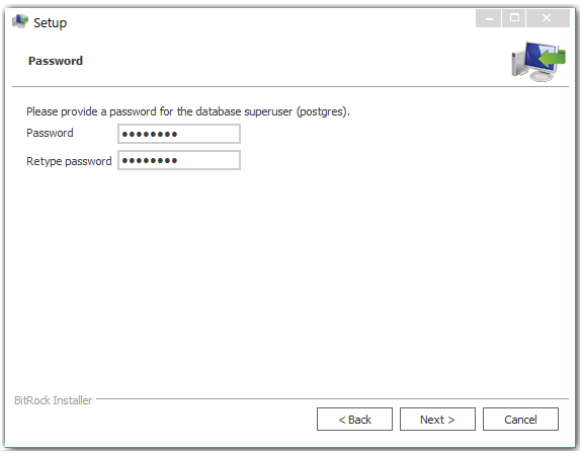
Instalasi PostgreSQL

22 Januari 2013
5:25

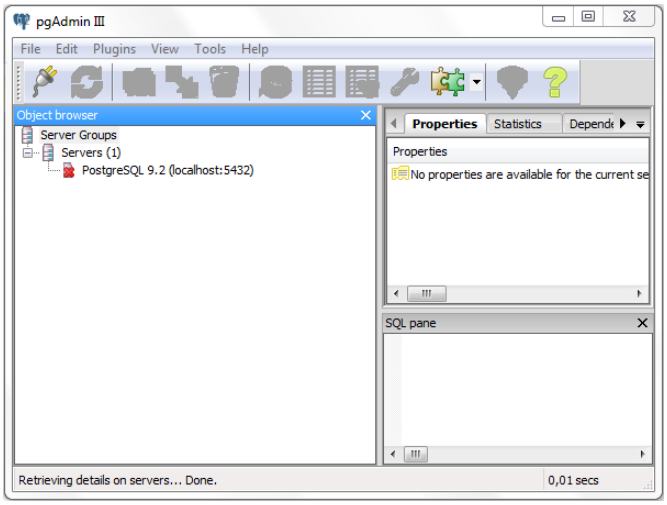
Seperti yang disebutkan sebelumnya instalasi database PostgreSQL ini termasuk dalam installer next generation. Cukup dengan klik ganda pada file installernya misalnya "postgresql-9.2.1-1-windows.exe". Selanjutnya adalah klik next, next, sampai installer meminta untuk memasukkan password. Perlu diingat bahwa password yang dimasukkan ini (bebas apa saja) akan digunakan untuk user "postgres" yang levelnya adalah admin. Setelah itu ikuti next terus sampai pada layar finish, user akan diminta untuk menjalankan stackbuilder. Unchek saja karena kita tidak perlu menjalankan stackbuilder berfungsi untuk mencari update program pendukung,



Isikan password, bebas bisa diisi kombinasi huruf, angka dan underscore jangan sampai lupa passwordnya.



Setelah selesai instalasi, service database postgreSQL secera otomatis langsung dijalankan. posgreSQL memiliki banyak interface untuk memanipulasi databasenya bahkan juga bisa dengan menggunakan web browser seperti halnya mySQL dengan phpMyAdmin, postgreSQL juga memiliki phpPgAdmin. Namun untuk memudahkan kita dalam mengoperasikan database dengan lebih optimal, kita akan menggunakan interface PgAdmin3 yang secara otomatis sudah terinstall bersama dengan database itu sendiri.

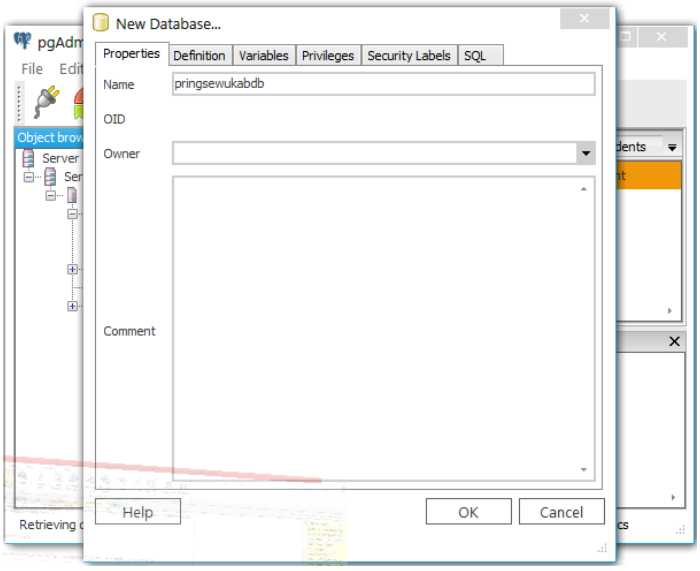


Saat pertama menjalankan PgAdmin3, klik pada tree icon "PostgreSQL 9.2 (localhost:5432)", dan postgre akan meminta password untuk postgres. Masukkan password yang sama seperti pada waktu instalasi. Dengan ini kita sudah masuk ke environment postgre dengan level admin dan siap untuk melakukan proses database lebih lanjut.

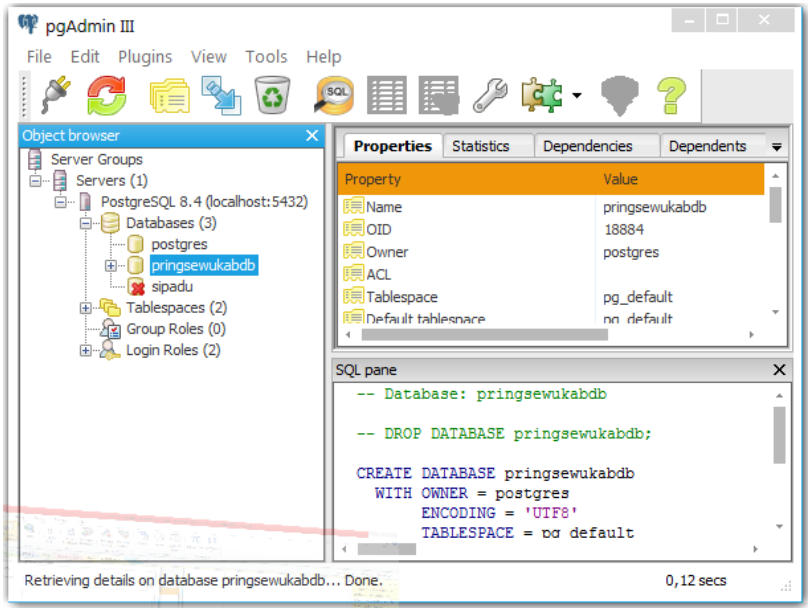
Membuat database baru

22 Januari 2013
5:38

Untuk membuat database kita akan menggunakan fasilitas yang ada pada aplikasi PgAdmin3. setelah kita login ke aplikasi ini, kita akan melihat sidebar di sebelah kiri layar. Klik kanan pada baris "databases" lalu pilih "new database".
Berikutnya akan muncul window untuk mendefinisikan database baru. Kita hanya perlu mengisikan nama databasenya, sisanya kita biarkan dalam keadaan default.



Kita asumsikan nama database diisi dengan "pringsewukabdb". Klik ok untuk membuat database ini. Sekarang kita telah memiliki database dengan nama "pringsewukabdb", namun isinya masih kosong, belum ada tabel ataupun data di dalamnya.



Tipe Data

23 Januari 2013
8:26

Pada postgre ada nayak macam tipe data. Namun yang akan kita bahas adalah tipe data yang mungkin akan sering kita gunakan. Pertama adalah jenis numeric yang terdiri dari beberapa tipe data yaitu:

Name	**Storage Size**	**Description**	**Range**
smallint	2 bytes	small-range integer	-32768 to +32767
integer	4 bytes	typical choice for integer	-2147483648 to +2147483647
bigint	8 bytes	large-range integer	-9223372036854775808 to 9223372036854775807
decimal	variable	user-specified precision, exact	up to 131072 digits before the decimal point; up to 16383 digits after the decimal point
numeric	variable	user-specified precision, exact	up to 131072 digits before the decimal point; up to 16383 digits after the decimal point
real	4 bytes	variable-precision, inexact	6 decimal digits precision

double precision	8 bytes	variable-precision, inexact	15 decimal digits precision
smallserial	2 bytes	small autoincrementing integer	1 to 32767
serial	4 bytes	autoincrementing integer	1 to 2147483647
bigserial	8 bytes	large autoincrementing integer	1 to 9223372036854775807

Pasted from <http://www.postgresql.org/docs/9.2/interactive/datatype-numeric.html>

Lalu yang berikutnya adalah tipe data dengan jenis karakter (satuan huruf):

Name	Description
character varying(n), varchar(n)	variable-length with limit
character(n), char(n)	fixed-length, blank padded
text	variable unlimited length

Pasted from <http://www.postgresql.org/docs/9.2/interactive/datatype-character.html>

Yang berikutnya adalah time data yang berkaitan dengan waktu:

Name	Storage Size	Description	Low Value	High Value	Resolution
timestamp [(p)] [without time zone]	8 bytes	both date and time (no time zone)	4713 BC	294276 AD	1 microsecond / 14 digits
timestamp [(p)] with time zone	8 bytes	both date and time, with time zone	4713 BC	294276 AD	1 microsecond / 14 digits
date	4 bytes	date (no time of day)	4713 BC	5874897 AD	1 day
time [(p)] [without time zone]	8 bytes	time of day (no date)	00:00:00	24:00:00	1 microsecond / 14 digits
time [(p)] with time zone	12 bytes	times of day only, with time zone	00:00:00+1459	24:00:00-1459	1 microsecond / 14 digits
interval [fields] [(p)]	12 bytes	time interval	-178000000 years	178000000 years	1 microsecond / 14 digits

Pasted from <http://www.postgresql.org/docs/9.2/interactive/datatype-datetime.html>

Contoh dari timestamp ini adalah:

TIMESTAMP tanpa timezone	'2004-10-19 10:23:54'
TIMESTAMP dengan timezone	'2004-10-19 10:23:54+02'

Untuk outputnya, jenis waktu ini memiliki beberapa kemungkinan output standar/default:

SQL	traditional style	12/17/1997 07:37:16.00 PST
Postgres	original style	Wed Dec 17 07:37:16 1997 PST

Tipe data yang terakhir yang akan kita bahas adalah boolean:

Name	**Storage Size**	**Description**
boolean	1 byte	state of true or false

Pasted from <http://www.postgresql.org/docs/9.2/interactive/datatype-boolean.html>

Tipe data ini hanya memiliki nilai true atau false. Namun untuk mendefinisikan true dan false bisa dengan beberapa cara

TRUE	FALSE
't'	'f'
'true'	'false'
'y'	'n'
'yes'	'no'
'on'	'off'
'1'	'0'

Normalisasi Database

23 Januari 2013
8:27

Normalisasi database adalah proses untuk membuat database kita menjadi "normal" dan terbebas dari segala bentuk anomali. Anomali sendiri adalah kondisi dimana tabel pada database bersifat tidak efisien dan cenderung menghabiskan sumber daya secara berlebihan. Normalisasi database kita lakukan setelah kita membuat rancangan database dan sebelum kita mengimplementasikannya ke aplikasi database.
Misalnya kita memiliki tabel dengan nama "artikel" yang memiliki field-field dan tipe data sebagai berikut:

Tabel_Artikel

field	Tipe data
id	Serial (big int)
Judul_artikel	Var char (60)
Isi_artikel	Text
Kategori_artikel	Var char (60)

Misalnya pada tabel diatas kita akan mengisikan "judul_artikel" dan "isi_artikel" seperti biasa, dengan field "id" sebagai identitas yang di generate secara otomatis, dan field "kategori_artikel" akan diisikan kategori seperti "berita resmi statistik", atau "berita kegiatan", atau "tentang BPS", dan berbagai macam kategori lain yang kita buat. Cara seperti ini bisa saja kita lakukan dan tidak akan menimbulkan error apapaun pada aplikasi yang kita buat. Namun bila kita melihatnya dalam skala yang lebih besar, misalnya kita telah memiliki 100 artikel saja dan 80 dari artikel itu adalah dari kategori "berita resmi statistik", maka kita akan memiliki 80 record yang kolom kategorinya berisi nilai "berita resmi statistik". Ini berarti kita telah mengulangi isian yang sama dengan tipe teks sebanyak 80 kali. Karena tipe teks kapasitas penyimpanannya jauh lebih besar daripada tipe numeric ini artinya kita telah memboroskan sumberdaya yang ada pada server untuk menyimpan data dan untuk menarik data dalam jumlah yang lebih besar.
Untuk menghentikan pemborosan sumber daya ini sebenarnya kita masih bisa melakukan pengelompokan pada kolom kategori daripada menuliskan kategorinya berulang-ulang (walaupun hal ini bisa dilakukan oleh php). Untuk itu akan lebih baik jika kita memiliki dua buah tabel yaitu

Tabel_Artikel

field	Tipe data
id	Serial (big int)
Judul_artikel	Var char (60)
Isi_artikel	Text
Id_kategori_artikel	integer

Tabel_Kategori

field	Tipe data
id	Serial (big int)
kategori	Var char (60)

Dengan menggunakan 2 tabel diatas, maka kapasitas penyimpanan akan menjadi lebih optimal terutama untuk data dalam jumlah besar.

Dalam dunia per-database-an, normalisasi bertujuan untuk meminimalisasi pengulangan data (redundancy), dan untuk memudahkan idetifikasi entitas. Normalisasi sebenarnya ada beberapa tingkatan yaitu:

1nf:	• setiap atribut hanya memiliki nilai tunggal pada setiap record • Tidak boleh ada grup atribut yang berulang
2nf:	• Memenuhi kaidah 1nf • setiap atribut yang bukan primary key tergantung secara fungsional terhadap semua primary key dan bukan hanya sebagian primary key (ketergantungan penuh pada primary key)
3nf:	• Memenuhi 2nf • setiap atribut yang bukan primary key tidak tergantung secara fungsional terhadap atribut bukan primary key yang lain dalam relasi tsb (tidak bergantung pada non primary key)
Bcnf(Boyce Codd normal form):	• jika dan hanya jika setiap determinan yang ada pada relasi tersebut adalah candidate key • Untuk normalisasi ke bentuk BCNF, maka tabel 3NF didekomposisi menjadi beberapa tabel yang masing-masing memenuhi BCNF

Namun kita tidak perlu merisaukan segala bentuk normalisasi ini. Asalkan kita sudah mengoptimalkan bentuk database kita dan menghindari redundancy, maka secara otomatis kita telah membantu menormalkan database yang kita buat.

Relasi Antar Tabel

22 Januari 2013
21:30

Relasi antar tabel adalah hubungan antara primary key suatu tabel dengan foreign key pada tabel lain. Bagi yang terbiasa menggunakan database Microsoft Access, ini sama saja dengan relationship diagram. Namun cara mendefiniskannya saja yang berbeda. Pada postgre kita harus mendefinisikan satu persatu field mana yang berhubungan dengan field pada tabel lain. Nantinya relasi yang terbentuk akan bersifat one to many, dimana one pada primary key suatu tabel dan many pada foreign key di tabel yang lain.

Seperti pada contoh normalisasi di bahasan sebelumnya, relasi antar tabel ini menjembatani antar tabel yang terkait. Misalnya:

Tabel_Artikel

field	Tipe data
id	Serial (big int)
Judul_artikel	Var char (60)
Isi_artikel	Text
Id_kategori_artikel	integer

Tabel_Kategori

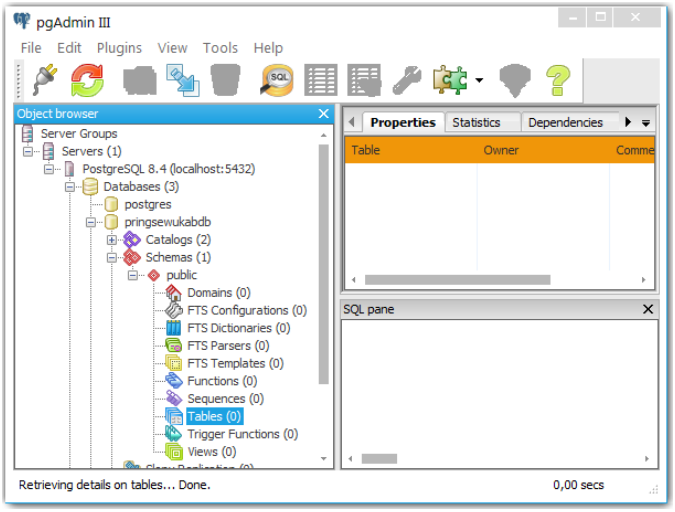
field	Tipe data
id	Serial (big int)
kategori	Var char (60)

Diatas kita melihat pada tabel "Tabel_Artikel" ada field yang bernama "id_kategori"artikel". Field ini berkaitan dengan tabel "Tabel_Kategori" pada field "id". Pada tabel "Tabel_Artikel" field "id_kategori_artikel" merupakan foreign key yang merujuk pada tabel "Tabel_Kategori" khususnya pada field "id" yang merupakan primary key. Ini kan menciptakan hubungan one to many, dimana 1 baris pada tabel "Tabel_Kategori" akan menjelaskan beberapa baris data pada tabel "Tabel_Artikel". Dengan mendefinisikan relasi antar tabel ini berarti kita telah mendefinisikan ketergantungan suatu tabel terhadap tabel yang lain. Dimana pada contoh diatas tabel "Tabel_Artikel" tidak akan pernah bisa ditambahkan record baru apabila kategorinya belum ditambahkan pada "Tabel_Kategori". Sehingga tidak akan terjadi miss link dimana ada artikel yang tidak mempunyai kategori.

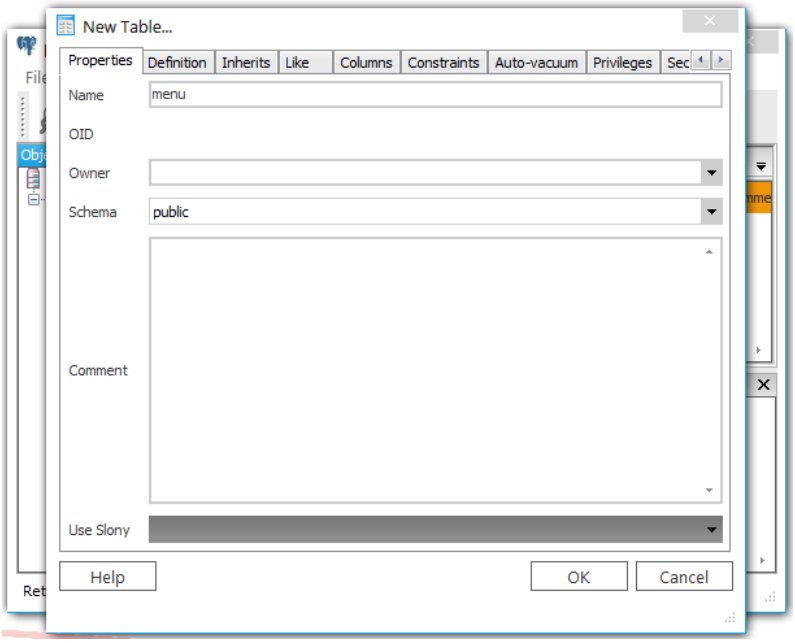
Membuat Tabel

22 Januari 2013
5:39

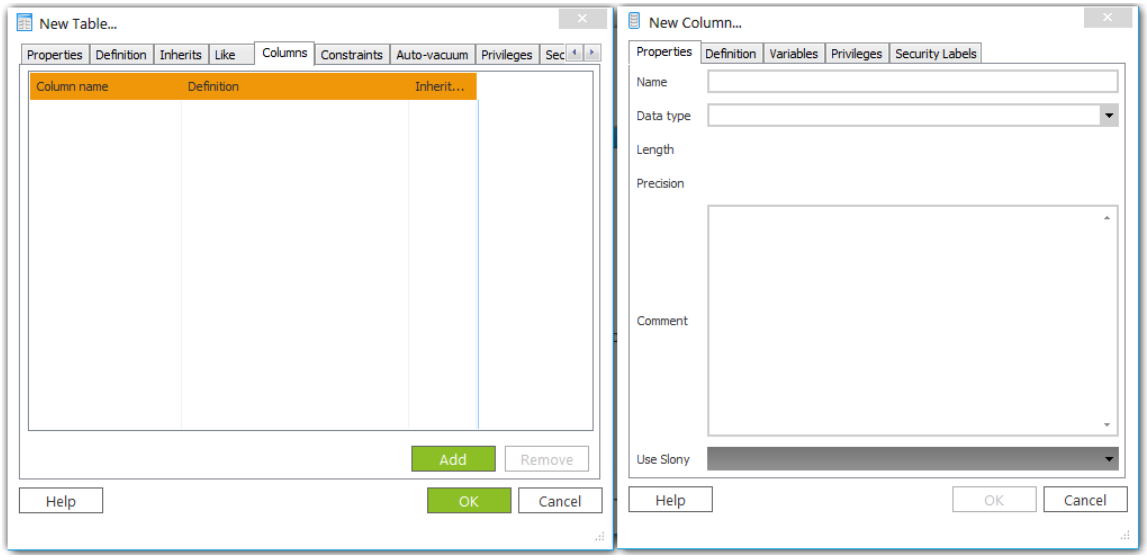
Setelah kita memiliki database, kita akan mencoba mengisi database ini dengan tabel yang dibutuhkan dalam website kita. Pada sidebar kiri layar, expand pada nama database > schemas > public. Dibawah item tersebut akan ada satu item dengan nama "tables".



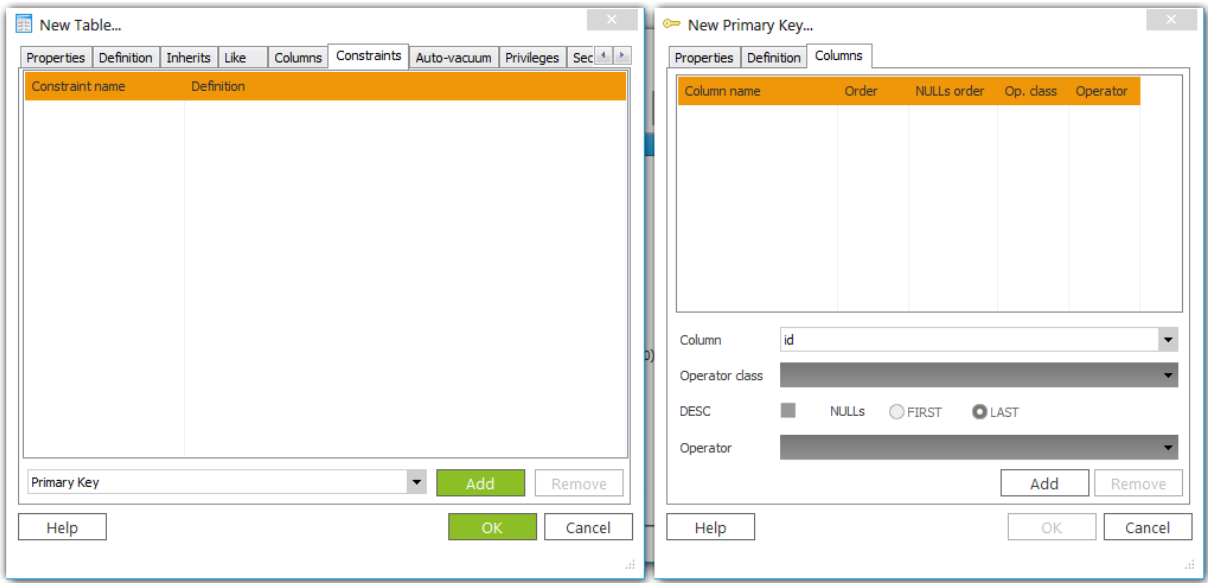
Klik kanan pada item "tables" tadi, lalu pilih "new table". Kemudian akan muncul window baru untuk memasukkan rincian tabel. Adapun yang harus kita isi adalah pada tab general -> nama tabel.



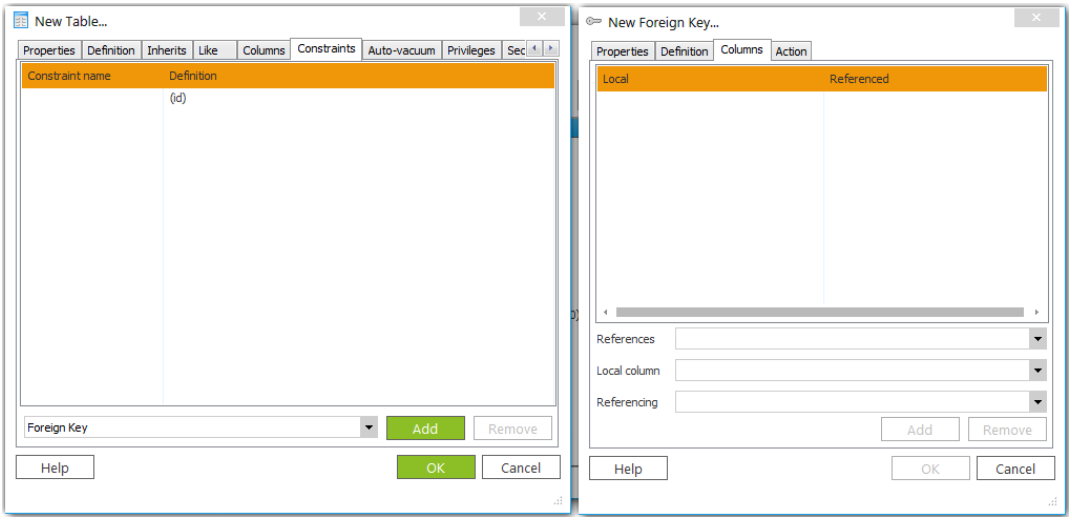
Lalu pada tab column -> klik tombol add, maka akan muncul window baru untuk mengisikan detail kolom-kolom pada tabel tersebut.



Isikan nama kolom, dan tipe datanya untuk masing-masing kolom, lalu klik OK pada window pengisian column. Namun jangan klik OK pada window Table dulu, karena kita masih harus mengulangi proses penambahan kolom untuk kolom-kolom selanjutnya pada tabel tersebut. Setelah selesai mengisikan kolom-kolom (field) yang diperlukan, selanjutnya dalah mendefinisikan primary key dan (atau) foreign key. Buka tab constraints lalu di sebelah kiri tombol "add" pilih "primary key" lalu baru klik "add". Berikutnya akan muncul window untuk memilih primary key.



Buka tab "columns" pada window primary key, lalu pada rincian "column" pilih nama kolom yang akan dijadikan identitas tabel, lalu klik Add. Tambahkan lagi kolom lainnya bila ingin memiliki primary key lebih dari satu field, bila tidak langsung klik tombol OK pada window "new primary key" ini. Setelah itu kita bisa mendefinisikan relasi antar tabel, dengan cara memilih "foreign key" di sebelah kanan tombol "add" pada window new table sebelumya, lalu klik add. Nanti akan muncul window "new foreign key".

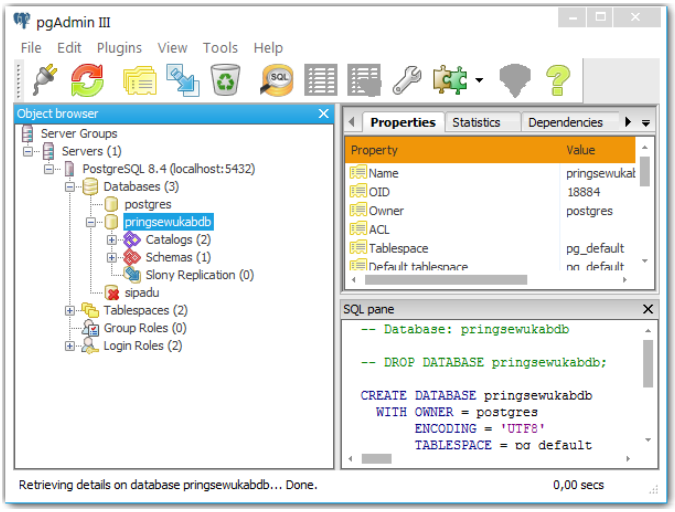


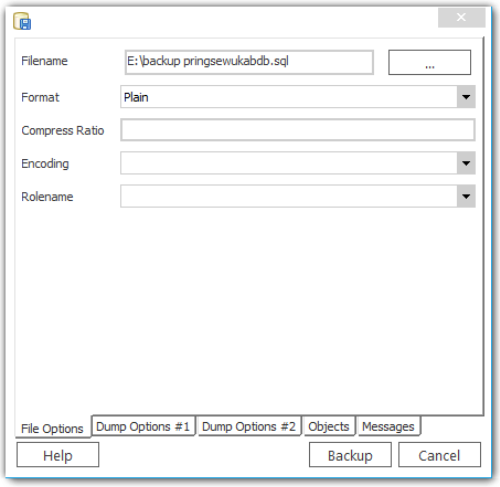
Kita baru bisa mendefinisikan foreign key apabila tabel rujukan dari relasi ini juga telah dibentuk. Isikan rincian "references" dengan nama tabel rujukan. Pilih field dari tabel lokal di rincian "local column" yang akan merujuk ke field dari tabel lain. Lalu pada rincian "referencing" pilih nama field dari tabel lain yang akan dijadikan rujukan oleh field lokal. Klik "add", lalu ulangi bila masih ada foreign key lainnya. Bila sudah selesai, klik ok pada window "foreign key". Sekarang setelah kita selesai mendefinisikan tabel, klik OK untuk membentuk tabel ini. Ulangi mendefinisikan tabel lain hingga seluruh tabel yang diperlukan terbentuk.

Membuat Backup Database

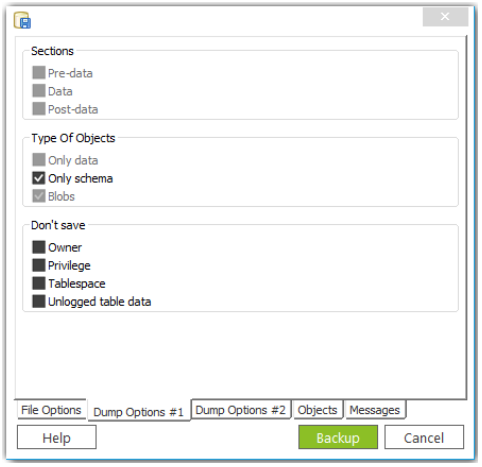
23 Januari 2013
5:40

Backup database sangat kita perlukan untuk membuat cadangan kondisi database seandainya sewaktu-waktu website kita mengalami gangguan, jadi kita tidak akan kehilangan data yang kita simpan di database. Selain itu backup database ini juga kita perlukan untuk "memindahkan" database dari komputer lokal kita ke remote server yaitu server yang ada di BPS RI. Caranya adalah masih dengan menggunakan aplikasi PgAdmin3, setelah login dan memilih database kerja, klik kanan pada nama database yang akan di buatkan backup-nya, lalu pilih "backup".
Agar memudahkan kita dalam penyimpanan data, kita akan memisahkan backup skema tabel dan backup datanya. Untuk itu kita akan melakukan backup sebanyak dua kali untuk keperluan tersebut.



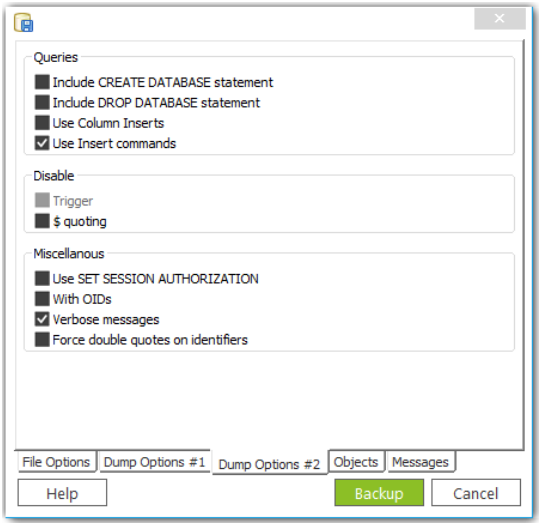


Nanti dari window kecil yang muncul, pada tab "file option" (bagian bawah layar) kita bisa mendefinisikan rincian yang ingin kita backup. Masukkan format "plain" lalu tentukan nama dan lokasi file penyimpanan backup. Lalu buka tab "Dump option #1"



Pada tab ini pilih rincian "only schema", beri tanda (check). Lalu klik backup untuk membuat backup skema tabel. Setelah proses selesai, klik "done". Perlu dikatahui bahwa backup skema ini adalah hanya membuat backup struktur tabelnya saja mulai dari nama tabel, field-field tabel beserta tipe

data masing-masingnya, primary key dan foreign key juga akan disipan informasinya. Untuk membuat backup isi datanya sendiri dapat kita lakukan dengan cara yang hampir sama yaitu klik nama database, lalu pilih backup. Kemudian masih sama dengan sebelumnya yaitu memilih format "plain" lalu menentukan nama file backup. Setelah itu yang agak sedikit berbeda adalah saat kita membuka tab "Dump Option #1", kita akan menandai (check) pada pilihan "only data". Lalu buka tab "Dump Option #2".

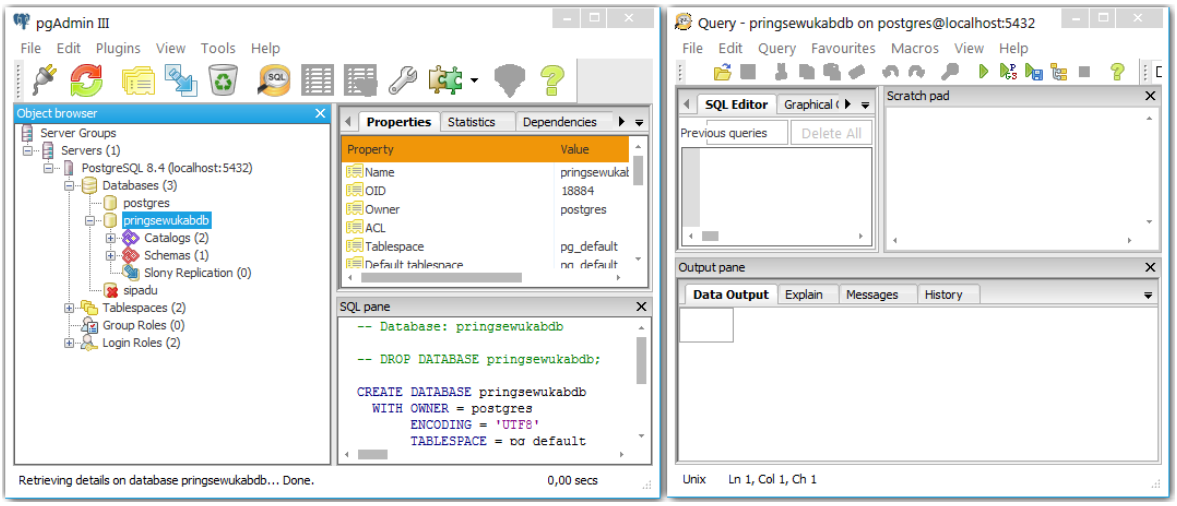


Tandai pilihan "use insert commands", lalu klik "backup". Setelah proses backup selesai, klik "Done". Dengan melakukan kedua jenis backup ini berarti kita sudah menyimpan keseluruhan database kita ke dalam format plain teks yang nantinya bisa kita implementasikan ke server BPS dengan lebih mudah. Dan perlu diketahui juga apabila kita sudah mempunyai backup skema dan selanjutnya kita tidak melakukan perubahan struktur database (seperti menambahkan tabel baru, ataupun field baru dari suatu tabel, hanya menambahkan record baru saja) maka kita tidak perlu melakukan backup skema berkali-kali. Kita hanya perlu melakukan backup rutin untuk data-nya saja untuk keperluan keamanan.

Merestore Database

23 Januari 2013
5:38

Kita melakukan restore database hanya pada daat kita memindahkan tabel database lokal ke remote. Atau pada saat kita sudah memiliki contoh skema database yang sudah dibuat terlebih dahulu pada komputer lain dan ingin kita implementasikan pada kompter yang kita gunakan. Kita akan mencoba merestore dengan menggunakan aplikasi PgAdmin3, untuk restore di server BPS tergantung dari aplikasi yang di support oleh database server di sana. Setelah kita membuka aplikasi dan masuk sebagai salah satu usernya, selanjutnya kita klik pada nama database yang ingin kita restore. Bila database-nya belum ada, maka harus kita buat dulu. Setelah itu kita klik pada icon SQL dengan gambar kaca pembesar.



Setelah itu akan muncul window query, klik icon open (bergambar folder) lalu pilih file backup yang skema terlebih dahulu (bila database masih kosong). Lalu klik icon run (bergambar segitiga berbentuk tanda panah ke arah kanan berwarna hijau). Setelah skema berhasil terbentuk, kita kan mengulangi cara yang sama untuk me-restore data-nya.

Koneksi database melalui PHP

23 Januari 2013
13:25

Pada Yii framework, koneksi database melalui PHP dilakukan melalui perintah

```
'db'=>array(
        'connectionString' => 'mysql:host=localhost;dbname=testdrive',
        'emulatePrepare' => true,
        'username' => 'root',
        'password' => '',
        'charset' => 'utf8',
),
```

Perintah ini ada pada file main.php pada folder website: "/pringsewu/protected/config"
Baris perintah ini kita ganti menjadi:

```
'db'=>array(
        'connectionString' => 'pgsql:host=localhost;port=5432;dbname=pringsewukabdb',
        'emulatePrepare' => true,
        'username' => 'postgres',
        'password' => 'password',
        'charset' => 'utf8',
),
```

Perintah kueri pada database dapat kita lihat contohnya misalnya pada class Menu berikut ini:

```
public function RetrieveMenu($kategori='1', $lang='idn')
        {
                $sql = "select id, menu_$lang as menu, aksi, alt_$lang as alt, id_parent, have_child from
                menu where id_menu_kategori=$kategori and flag=1 order by id asc";
```

```
        $hasil=Yii::app()->db->createCommand($sql)->queryall();
        return $hasil;
    }
```

Dengan syntax diatas, terutama bari perintah:

$hasil=Yii::app()->db->createCommand($sql)->queryall();

kita akan mendapatkan hasil berupa array yang berisi record-record yang direquest berdasarkan SQL yang didefinisikan sebelumnya. Bahasa perintah SQL sendiri yang paling banyak kita gunakan adalah "select" untuk memanggil record database sesuai kriteria tertentu. SQl select mempunyai sruktur sebegai berikut:

SELECT nama_field1, nama_field2 from nama_tabel WHERE nama_field_3 = 'nilai_yang diinginkan' ORDER BY nama_field4 ASC

"Order by" sampai "ASC" bisa dihilangkan bila kita tidak ingin mengurutkan record hasil berdasarkan field tertentu. "ASC" sendiri berarti ascending yaitu urutan dari kecil ke besar. Untuk mengurutkan dari besar ke kecil gunakan "DESC" untuk menggantikan "ASC". Bila kita ingin mengembalikan semua field dari tabel yang diinginkan, kita bisa mennganti nama_field di sebelah perintah SELECT dengan tanda *. Contoh:

SELECT * FROM nama_tabel

Perintah diatas akan mengembalikan semua field dari nama_tabel tanpa persyaratan apapaun (semua record akan dikembalikan) karena kita tidak mendefinisikan klausa WHERE.

Daftar Pustaka

Pemrograman Berbasis Web Menggunakan PHP 5, Didik Dwi Prasetyo, 2004

Http://www.w3schools.com/

http://www.postgresql.org/docs/manuals/

http://php.net/manual/en/

http://en.wikipedia.org/wiki/PHP/

Dasar-dasar Pemrograman PHP dan Database PostgreSQL

PHP merupakan bahasa pemrograman yang dikhususkan untuk pengembangan website. Diantara beberapa pilihan, PHP merupakan bahasa pemrograman yang paling populer dengan pengguna mencapai sekitar 75% (berdasarkan ‘‘wikipedia” tahun 2007) dari seluruh website di seluruh dunia yang diketahui menggunakan pemrograman dari sisi server. Selain didukung oleh banyak komunitas programer yang handal, bahasa PHP sendiri dikenal cukup sederhana dan mudah diimplementasikan. Mungkin tidak perlu dijelaskan lagi kalau program PHP ini bisa kita dapatkan dengan gratis. Selain itu paket programnya juga banyak ditemukan dibundel bersama dengan aplikasi lain seperti XAMPP, PHP2Triad, Apache2Triad, WAMP, dan beberapa paket aplikasi lainnya yang juga bersifat gratis.

Buku ini akan mengulas dasar-dasar pemrograman PHP dengan cara yang simpel namun cukup memadai untuk membagun aplikasi dengan yang pemrogramannya berorientasi objek. Dan diharapkan para pembaca dapat memahami konsep OOP dan penggunaan database server untuk menunjang pembuatan website yang dinamis.

Adapun materi yang dicakup dalam buku ini adalah sebagai berikut:

- Struktur PHP
- Variabel dan tipe data PHP
- Perintah berdasarkan kondisi (control structure)
- Looping
- Object Oriented Programming
- Pengenalah Database PostgreSQL
- Tipe data Postgre
- Membuat database dan tabel
- Backup dan restore database