Seri Tutorial Praktis Pemrograman – Rosihan Ari Yuana

PANDUAN PRAKTIS OOP DI PHP

Disusun oleh:

Rosihan Ari Yuana, S.Si, M.Kom

http://blog.rosihanari.net

Daftar Isi Ebook

1. Pendahuluan

Bagi sebagian orang, khususnya para programmer pemula tentunya banyak menjumpai kesulitan ketika mencoba beralih dari gaya pemrograman prosedural ke OOP (Object Oriented Programming). Mengapa demikian? Ya... karena ketika mereka belajar programming pertama kali, doktrinasi gaya pemrograman prosedural banyak dilakukan kepada mereka. Tapi hal ini sangat beralasan karena ketika belajar pemrograman pertama kali, mereka harus dituntut memahami program yang mereka buat secara algoritmik. Dalam hal ini gaya pemrograman prosedural lah yang paling mudah dipahami secara algoritmik.

Di lain pihak, OOP sebenarnya pengembangan dari gaya pemrograman prosedural. Memang untuk OOP ini sangat disarankan bagi para programmer yang sudah memiliki level advanced. Mengapa OOP dikatakan pengembangan dari prosedural? Ya... karena, selain penguasaaan dari sisi algoritmik, programmer OOP ini harus mampu menyatakan problem ke dalam bentuk obyek-obyek.

Selanjutnya, jika Anda bertanya apakah sulit proses migrasi dari penguasaan gaya pemrograman prosedural ke gaya pemrograman OOP? Jawabnya adalah Ya... namun bagian yang sulit dari belajar OOP adalah di awal belajar saja atau di bagian dasar-dasar OOP nya saja. Setelah itu... dijamin pasti no problem. Selain itu kesulitan para programmer ketika mencoba beralih ke OOP adalah kurang dipahaminya referensi berupa buku-buku yang beredar di negeri ini. Hampir kebanyakan buku menjelaskan secara teoritis saja, tanpa adanya pembahasan detil tentang studi kasusnya, atau terkadang suatu buku hanya berisi terlalu banyak script contoh tapi miskin penjelasan sehingga membawa budaya copy paste tanpa dasar keilmuan dan pemahaman.

Nah.. oleh karena itu dalam ebook ini, saya akan mencoba menjelaskan OOP khususnya di PHP ini dengan gaya yang mudah dipahami Insya Allah. Dengan mengedepankan studi kasus diharapkan para pembaca ebook ini dapat mendalami konsep OOP ini dengan mudah.

Untuk bisa mempelajari ebook ini, saya sarankan Anda perdalam dahulu tentang PHP dasar meliputi function, variabel, looping, dan conditional statement. Jika belum, tolong pelajari dulu PHP dasar di http://blog.rosihanari.net/download-tutorial-php-dasar-gratis

2. Membuat Class

Dalam OOP, sebuah class merupakan blueprint dari suatu obyek. Mungkin Anda bertanya, apa bedanya class dengan sebuah function? Sebuah class bisa berisi variabel dan function. Variabel yang terletak di dalam class, dinamakan property dan function yang ada di dalam sebuah class dinamakan method.

Untuk membuat sebuah class, strukturnya adalah sbb:

```
<?php

class namakelas
{
	var namavariabel;
	.
	.
	.
}

?>
```

Sebagai contoh misalkan kita membuat class bernama kendaraan

```
<?php

class kendaraan
{
	var $jumlahRoda;
	var $warna;
	var $bahanBakar;
	var $harga;
	var $merek;

}

?>
```

Dalam contoh di atas, yang merupakan properti dari class kendaraan adalah: jumlahRoda, warna, bahanBakar dan harga.

Sebuah properties dari suatu class dapat Anda bayangkan sebagai sifat atau informasi yang melekat dari suatu obyek. Sebagai contoh misalkan kita pandang sebuah obyek 'mahasiswa', maka properties dari mahasiswa beberapa diantaranya adalah: nim, nama, alamat, nama orang tua, jurusan, fakultas dsb.

Latihan

1. Buatlah sebuah kelas bernama 'buku', kemudian deklarasikan beberapa properties dari buku tersebut, misalnya: judul buku, pengarang, penerbit, tahun tersebut dsb
2. Rancanglah sebuah kelas untuk menyatakan orang, kemudian tentukan sendiri properties nya dan selanjutnya tulis class tersebut ke dalam script PHP.

3. Membuat Function/Method dalam Class

Seperti yang saya sampaikan sebelumnya, bahwa dalam sebuah class bisa dibuat function. Sebuah function dalam suatu class dinamakan method, dan sebuah method jika kita bayangkan adalah segala hal yang terkait dengan pekerjaan atau proses yang dapat diberikan pada suatu obyek. Sebagai contoh method dalam kehidupan sehari-hari, adalah pada obyek seorang 'mahasiswa'. Sebuah method kita bisa berikan pada mahasiswa tersebut misalnya: 'tempuh kuliah'. Di dalam method 'tempuh kuliah' itu terdapat serangkaian proses mulai dari

- registrasi kuliah
- ikuti kuliah
- ikuti ujian
- Jika ujian tidak lulus, maka ulangi ikuti kuliah

Itu sebagai contoh gambaran method dalam kehidupan sehari-hari.

Berikut ini contoh sebuah function yang dibuat dalam sebuah class. Function dalam contoh berikut ini digunakan untuk menentukan apakah sebuah kendaraan harganya mahal atau tidak. Di sini kendaraan dikatakan mahal jika harganya di atas 50 juta, dan jika di 50 juta ke bawah dikatakan murah.

```
<?php

class kendaraan
{
	var $jumlahRoda;
	var $warna;
	var $bahanBakar;
	var $harga;
	var $merek;

	function statusHarga()
	{
		if ($this->harga > 50000000) $status = 'Mahal';
		else $status = 'Murah';
		return $status;
	}

}

?>
```

Perhatikan perintah:

$this->harga

Variabel $this merupakan built in variabel yang digunakan untuk mengakses properties atau method yang ada dalam class tersebut. Sehingga perintah $this->harga digunakan untuk mengakses atau membaca property dari $harga yang ada dalam class kendaraan.

Catatan:

Variabel $status dalam function statusharga() bukanlah termasuk property dari class kendaraan karena tidak didefinisikan dalam bentuk var $status;

Latihan

1. Dari kelas 'kendaraan' dalam contoh, tambahkan sebuah property 'tahun pembuatan'
2. Buatlah function dalam kelas 'kendaraan' dengan nama 'dapatSubsidi()' untuk menentukan apakah suatu kendaraan mendapat subsidi BBM atau tidak. Kendaraan yang mendapat subsidi adalah yang berbahan bakar 'Premium' dan tahun pembuatannya sebelum tahun 2005. Function ini mereturn 'Ya' jika mendapat subsidi, dan 'Tidak' jika tidak mendapat subsidi.
3. Buatlah function dalam kelas 'kendaraan' dengan nama 'hargaSecond()' untuk menentukan harga second dari kendaraan tersebut. Function ini mereturn harga second dari kendaraan dengan ketentuan:
    a. Jika tahun pembuatan di atas 2005, maka harga second nya turun 20% dari harga aslinya
    b. Jika tahun pembuatan 2000 s/d 2005, maka harga second nya turun 30% dari harga aslinya
    c. Jika tahun pembuatan di bawah 2000, maka harga second nya turun 40% dari harga aslinya.

4. Instantisasi Obyek

Seperti yang telah dijelaskan sebelumnya bahwa sebuah class merupakan blueprint dari obyek. Sebuah class belum menjadi obyek sebelum kita lakukan sebuah proses instantisasi obyek.

Untuk melakukan instantisasi obyek, perintahnya adalah sbb:

$handle = new namaclass();

Sebagai contoh, misalkan kita lakukan instantiasi pada class kendaraan

```
<?php

class kendaraan
{
      var $jumlahRoda;
      var $warna;
```

```
	var $bahanBakar;
	var $harga;
	var $merek;

	function statusHarga()
	{
		if ($this->harga > 50000000) $status = 'Mahal';
		else $status = 'Murah';
		return $status;
	}

}

$kendaraan1 = new kendaraan();

?>
```

Jika script di atas dijalankan, maka di browser tidak muncul apa-apa. Hal ini terjadi karena kita belum menyuruh PHP untuk melakukan sesuatu pada obyek $kendaraan1 tersebut.

Variabel $kendaraan1 dalam hal ini dinamakan 'handle' karena kita akan gunakan $kendaraan1 untuk mengontrol dan menggunakan obyek kendaraan.

Oya, kita juga bisa melakukan instantisasi obyek tanpa menggunakan kurung, perhatikan contoh berikut ini yang menunjukkan proses instantisasi beberapa obyek dari class kendaraan.

```
<?php

class kendaraan
{
	var $jumlahRoda;
	var $warna;
	var $bahanBakar;
	var $harga;
	var $merek;

	function statusHarga()
	{
		if ($this->harga > 50000000) $status = 'Mahal';
		else $status = 'Murah';
		return $status;
	}

}

$kendaraan1 = new kendaraan();
$kendaraan2 = new kendaraan;
$kendaraan3 = new kendaraan();

?>
```

5. Setting Properties

Setelah suatu obyek kita lakukan instantitasi, selanjutnya kita bisa mensetting properties dari obyek tersebut. Sebagai contoh, misalkan kita telah membuat obyek $kendaraan1, kemudian bagaimana kita menset properti harga dan merek dari obyek $kendaraan1 ini?

Kita dapat mensetting properties dari suatu obyek dengan perintah:

$namaobyek->properti = value;

Perhatikan contoh berikut ini:

```
<?php

class kendaraan
{
      var $jumlahRoda;
      var $warna;
      var $bahanBakar;
      var $harga;
      var $merek;

      function statusHarga()
      {
            if ($this->harga > 50000000) $status = 'Mahal';
            else $status = 'Murah';
            return $status;
      }

}

$kendaraan1 = new kendaraan();
$kendaraan1->merek = 'Yamaha MIO';
$kendaraan1->harga = 10000000;

?>
```

Perintah

$kendaraan1->merek = 'Yamaha MIO';

Digunakan untuk mensetting properti merek ‘Yamaha MIO’ dari obyek $kendaraan1.

Kita juga bisa menggunakan method untuk proses setting properti ini, dan ini adalah cara yang lebih direkomendasikan dalam OOP.

```
<?php
```

```
class kendaraan
{
	var $jumlahRoda;
	var $warna;
	var $bahanBakar;
	var $harga;
	var $merek;

	function statusHarga()
	{
		if ($this->harga > 50000000) $status = 'Mahal';
		else $status = 'Murah';
		return $status;
	}

	function setMerek($x)
	{
		$this->merek = $x;
	}

	function setHarga($x)
	{
		$this->harga = $x;
	}

}

$kendaraan1 = new kendaraan();
$kendaraan1->setMerek('Yamaha MIO');
$kendaraan1->setHarga(10000000);

?>
```

Latihan

Dari class 'kendaraan' di atas, buatlah obyek dengan beberapa properti sbb

Obyek	Merek	Jml Roda	Harga	Warna	Bhn Bakar
$kendaraan2	Toyota Yaris	4	160.000.000	Merah	Premium
$kendaraan3	Honda Scoopy	2	13.000.000	Putih	Premium
$kendaraan4	Isuzu Panther	4	170.000.000	Hitam	Solar

6. Menjalankan Method

Dalam bagian ini, akan dijelaskan cara menjalankan sebuah method dari suatu obyek. Ingat, bahwa menjalankan sebuah method dari suatu obyek pada intinya adalah memanggil function yang dalam class.

Sebenarnya, dalam contoh sebelumnya sudah diberikan contoh untuk menjalankan method yaitu salah satunya melalui perintah

```
$kendaraan1->setMerek('Yamaha MIO');
```

Perintah tersebut adalah menjalankan method setMerek() dari obyek $kendaraan1, dan dalam hal ini setMerek() adalah sebuah function dalam class kendaraan.

Contoh yang lain, misalkan kita akan menjalankan method statusHarga() yang digunakan untuk menampilkan status harganya apakah termasuk mahal atau murah.

```
 <?php

class kendaraan
{
      var $jumlahRoda;
      var $warna;
      var $bahanBakar;
      var $harga;
      var $merek;

      function statusHarga()
      {
            if ($this->harga > 50000000) $status = 'Mahal';
            else $status = 'Murah';
            return $status;
      }

      function setMerek($x)
      {
            $this->merek = $x;
      }

      function setHarga($x)
      {
            $this->harga = $x;
      }

}

$kendaraan1 = new kendaraan();
$kendaraan1->setMerek('Yamaha MIO');
$kendaraan1->setHarga(10000000);
echo $kendaraan1->statusHarga();

?>
```

Jika script di atas dijalankan, maka akan muncul ‘Murah’, karena harga nya kurang dari 50.000.000.

Perhatikan dari beberapa contoh pemanggilan method di atas, bahwa setiap kali pemanggilan method jangan lupa memberi tanda kurung (), seperti pada

$kendaraan1->setHarga(10000000);

Atau

$kendaraan1->statusHarga();

Karena kurung tersebut digunakan untuk meletakkan parameter bagi method tersebut.

Latihan

1. Perhatikan kembali soal latihan sebelumnya pada bab 5. Tampilkan status harga dari $kendaraan2, $kendaraan3 dan $kendaraan4.
2. Perhatikan kembali soal latihan pada bab 3 nomor 3. Tampilkan harga second dari $kendaraan2, $kendaraan3 dan $kendaraan4.

7. Mengakses Properties

Sekarang akan dijelaskan bagaimana cara mengakses properties dari suatu obyek. Sebelumnya, pernah saya katakan bahwa properties dari suatu obyek itu merupakan value dari variabel yang ada dalam class.

Bagaimana cara mengakses properties dari suatu obyek? Perhatikan contoh berikut ini

```
<?php

class kendaraan
{
	var $jumlahRoda;
	var $warna;
	var $bahanBakar;
	var $harga;
	var $merek;

	function statusHarga()
	{
		if ($this->harga > 50000000) $status = 'Mahal';
		else $status = 'Murah';
		return $status;
	}

	function setMerek($x)
	{
		$this->merek = $x;
	}
```

```
	function setHarga($x)
	{
		$this->harga = $x;
	}

}

$kendaraan1 = new kendaraan();
$kendaraan1->setMerek('Yamaha MIO');
$kendaraan1->setHarga(10000000);
echo 'Harga dari '.$kendaraan1->merek.' adalah Rp. '.$kendaraan1->harga;

?>
```

Perhatikan pada bagian perintah

$kendaraan1->harga

dan

$kendaraan1->merek

Kedua perintah di atas adalah digunakan untuk mengakses value dari property obyek $kendaraan1, yaitu 'merek' dan 'harga'. Jika script di atas dijalankan, maka akan diperoleh output

"Harga dari Yamaha MIO adalah Rp. 10000000"

Selain cara di atas, dapat pula menggunakan method dalam membaca properties dari suatu obyek, dan cara inilah yang paling disarankan dalam OOP. Perhatikan contoh berikut ini

```
<?php

class kendaraan
{
	var $jumlahRoda;
	var $warna;
	var $bahanBakar;
	var $harga;
	var $merek;

	function statusHarga()
	{
		if ($this->harga > 50000000) $status = 'Mahal';
		else $status = 'Murah';
		return $status;
	}

	function setMerek($x)
	{
```

```
            $this->merek = $x;
      }

      function setHarga($x)
      {
            $this->harga = $x;
      }

      function bacaMerek()
      {
            return $this->merek;
      }

      function bacaHarga()
      {
            return $this->harga;
      }
}

$kendaraan1 = new kendaraan();
$kendaraan1->setMerek('Yamaha MIO');
$kendaraan1->setHarga(10000000);
echo 'Harga dari '.$kendaraan1->bacaMerek().' adalah Rp. '.$kendaraan1-
>bacaHarga();

?>
```

Dalam contoh di atas, untuk mengakses properti merek dibuat function sbb

```
function bacaMerek()
{
      return $this->merek;
}
```

Sedangkan function untuk mengakses properti harga kendaraan adalah

```
function bacaHarga()
{
      return $this->harga;
}
```

Selanjutnya untuk mengakses properti nama merek kendaraan, cukup dipanggil saja method bacaMerek() sbb:

$kendaraan1->bacaMerek()

Demikian pula untuk mengakses properti harga kendaraan melalui method bacaHarga();

$kendaraan1->bacaHarga()

Latihan

Perhatikan kembali soal latihan pada bab 5, berdasarkan obyek yang telah dibuat, tampilkan properti setiap obyek sedemikian hingga tampilan script apabila dijalankan di browser sebagai berikut:

- Kendaraan **Toyota Yaris**, memiliki **4** roda, berbahan bakar **Premium** dan harganya Rp **160000000**.
- Kendaraan **Honda Scoopy**, memiliki **2** roda, berbahan bakar **Premium** dan harganya Rp **13000000**.
- Kendaraan **Isuzu Panther**, memiliki **4** roda, berbahan bakar Solar dan harganya Rp **170000000**.

8. Modularitas Class

Pada contoh-contoh script di atas, class dan juga proses instantisasi dijadikan satu dalam sebuah script. Hal ini dirasa kurang efektif apabila class tersebut juga digunakan dalam script yang lain nantinya. Sehingga untuk alasan kemudahan penggunaan, biasanya sebuah class atau kumpulan class diletakkan dalam sebuah script tersendiri, yang selanjutnya tinggal di include kan dalam sebuah script apabila class tersebut akan digunakan. Dengan demikian kita tidak perlu menulis kembali isi class secara penuh dalam setiap scriptnya.

Sebagai contoh, perhatikan kembali contoh script pada bab 7 yang berbentuk sbb:

```
<?php

class kendaraan
{
	var $jumlahRoda;
	var $warna;
	var $bahanBakar;
	var $harga;
	var $merek;

	function statusHarga()
	{
		if ($this->harga > 50000000) $status = 'Mahal';
		else $status = 'Murah';
		return $status;
	}

	function setMerek($x)
	{
		$this->merek = $x;
	}

	function setHarga($x)
```

```
	{
		$this->harga = $x;
	}

	function bacaMerek()
	{
		return $this->merek;
	}

	function bacaHarga()
	{
		return $this->harga;
	}
}

$kendaraan1 = new kendaraan();
$kendaraan1->setMerek('Yamaha MIO');
$kendaraan1->setHarga(10000000);
echo 'Harga dari '.$kendaraan1->bacaMerek().' adalah Rp. '.$kendaraan1-
>bacaHarga();

?>
```

Kita dapat memisahkan class 'kendaraan' ini dalam file tersendiri misalkan diberinama 'class-kendaraan.php' yang isinya

class-kendaraan.php

```
<?php

class kendaraan
{
	var $jumlahRoda;
	var $warna;
	var $bahanBakar;
	var $harga;
	var $merek;

	function statusHarga()
	{
		if ($this->harga > 50000000) $status = 'Mahal';
		else $status = 'Murah';
		return $status;
	}

	function setMerek($x)
	{
		$this->merek = $x;
	}

	function setHarga($x)
	{
```

```
            $this->harga = $x;
      }

      function bacaMerek()
      {
            return $this->merek;
      }

      function bacaHarga()
      {
            return $this->harga;
      }
}
?>
```

Selanjutnya kita include kan file class-kendaraan.php ini ke dalam script lain apabila kita memerlukannya,

contoh.php

```
<?php
include 'class-kendaraan.php';

$kendaraan1 = new kendaraan();
$kendaraan1->setMerek('Yamaha MIO');
$kendaraan1->setHarga(10000000);
echo 'Harga dari '.$kendaraan1->bacaMerek().' adalah Rp. '.$kendaraan1-
>bacaHarga();
?>
```

9. Constructor

Perhatikan kembali proses instantisasi yang ada di bab 4 dan setting properties di bab 5. Jika kita perhatikan, maka proses instantisasi dan setting properties ini dilakukan secara terpisah. Tentu saja proses ini agak terlalu bertele-tele. Ternyata kita bisa langsung melakukan instantisasi obyek sekaligus melakukan setting propertiesnya. Proses ini dapat dilakukan dengan menggunakan ‘constructor’.

Untuk membuat constructor, kita cukup membuat sebuah function dalam class dengan bentuk

```
function __construct (parameter)
{
      .
      .
      .
}
```

Keterangan: Tanda __ merupakan tanda underscore (_) yang ditulis double.

Berikut ini contoh constructor untuk obyek kendaraan, dimana sekaligus mensetting properti 'merek' dan 'harga' kendaraan.

class-kendaraan.php

```
 <?php

class kendaraan
{
	var $jumlahRoda;
	var $warna;
	var $bahanBakar;
	var $harga;
	var $merek;

	function statusHarga()
	{
		if ($this->harga > 50000000) $status = 'Mahal';
		else $status = 'Murah';
		return $status;
	}

	function setMerek($x)
	{
		$this->merek = $x;
	}

	function setHarga($x)
	{
		$this->harga = $x;
	}

	function bacaMerek()
	{
		return $this->merek;
	}

	function bacaHarga()
	{
		return $this->harga;
	}

	function __construct($x, $y)
	{
		$this->merek = $x;
		$this->harga = $y;
	}
}
?>
```

Perhatikan

```
function __construct($x, $y)
{
      $this->merek = $x;
      $this->harga = $y;

}
```

Function tersebut kita buat 2 parameter, dimana $x menyatakan merek kendaraan, dan $y adalah harganya. Selanjutnya, perintah

$this->merek = $x;

Digunakan untuk setting property merek kendaraan berdasarkan nilai $x. Demikian juga perintah

$this->harga = $y;

Untuk setting property harga kendaraan berdasarkan nilai $y.

Selanjutnya, bagaimana cara melakukan instantisasi sekaligus setting propertiesnya? Perhatikan script berikut ini.

contoh.php

```
<?php
include 'class-kendaraan.php';

$kendaraan1 = new kendaraan('Yamaha MIO', 10000000);

echo 'Harga dari '.$kendaraan1->bacaMerek().' adalah Rp. '.$kendaraan1-
>bacaHarga();
?>
```

10. Encapsulation

Di dalam dasar-dasar OOP, ada istilah encapsulation. Istilah ini terkait dengan aksesibilitas properties dalam suatu class. Dengan encapsulation ini, kita bisa mengatur sebuah properti apakah hanya bisa diakses dalam class tersebut saja, atau tidak.

Aksesibilitas properties dalam encapsulation ini ada tiga sifat:

- Public : properti dapat diakses darimanapun
- Private : properti hanya dapat diakses dari dalam class saja

- Protected : properti hanya dapat diakses dari dalam class atau class turunan (inheritanced class)

Untuk membedakan ketiganya, perhatikan contoh berikut ini

class-kendaraan.php

```
 <?php

class kendaraan
{
	protected $jumlahRoda;
	public $warna;
	public $bahanBakar;
	public $harga;
	private $merek;

	function statusHarga()
	{
		if ($this->harga > 50000000) $status = 'Mahal';
		else $status = 'Murah';
		return $status;
	}

	function setMerek($x)
	{
		$this->merek = $x;
	}

	function setHarga($x)
	{
		$this->harga = $x;
	}

	function bacaMerek()
	{
		return $this->merek;
	}

	function bacaHarga()
	{
		return $this->harga;
	}

	function __construct($x, $y)
	{
		$this->merek = $x;
		$this->harga = $y;
	}
}
?>
```

Perhatikan class di atas. Untuk properti 'warna', 'bahan bakar' dan 'harga' dibuat sebagai public properties. Sedangkan untuk properti 'jumlahRoda' dan 'merek', masing-masing sebagai protected dan private properties. Selanjutnya, perhatikan script contoh berikut ini

contoh.php

```
<?php
include 'class-kendaraan.php';

$kendaraan1 = new kendaraan('Yamaha MIO', 10000000);
echo 'Nama merek : '.$kendaraan1->merek;

?>
```

Dalam script di atas, setelah proses instantisasi dan setting properti untuk obyek $kendaraan1, akan dilakukan pengaksesan ke properti merek secara langsung (tanpa method), dengan memberikan perintah

$kendaraan1->merek

Apa yang terjadi jika script di atas dijalankan? Ternyata akan muncul error

Fatal error: Cannot access private property kendaraan::$merek

Hal ini terjadi karena properti merek bersifat private, sehingga properti ini tidak bisa diakses dari luar class.

Bagaimana dengan akses ke properti harga secara langsung? Perhatikan script berikut ini

contoh.php

```
<?php
include 'class-kendaraan.php';

$kendaraan1 = new kendaraan('Yamaha MIO', 10000000);
echo 'Harga : '.$kendaraan1->harga;

?>
```

Ternyata jika script di atas dijalankan,bisa memunculkan harga dari Yamaha Mio.

Nah... yang menjadi pertanyaan, apakah bisa kita mengakses sebuah properti yang sifatnya private dalam class dari luar? Jawabnya adalah bisa, namun tidak dilakukan secara langsung dengan mengakses properti nya namun menggunakan method. Sebagai contoh, misalkan kita ingin mengakses properti merek yang sifatnya private, maka kita bisa menggunakan method bacaMerek().

contoh.php

```
<?php
include 'class-kendaraan.php';

$kendaraan1 = new kendaraan('Yamaha MIO', 10000000);
echo 'Harga : '.$kendaraan1->bacaMerek();

?>
```

Oya, bagaimana dengan deklarasi properties menggunakan ‘var’ seperti pada contoh-contoh di awal, misalnya:

```
class kendaraan
{
      var $jumlahRoda;
      var $warna;
      var $bahanBakar;
      var $harga;
      var $merek;
      .
      .
      .
}
```

Penggunaan ‘var’ di depan nama properties, secara otomatis akan bersifat sebagai public.

Berikutnya, muncul pertanyaan apakah yang bisa dibuat encapsulation dg sifat private, protected dan public ini hanya untuk properties saja? Jawabnya adalah TIDAK, sebuah function atau method pun bisa diterapkan hal ini. Sebagai contoh misalkan kita buat method statusHarga() sebagai private method.

class-kendaraan.php

```
<?php

class kendaraan
{
      protected $jumlahRoda;
      public $warna;
      public $bahanBakar;
      public $harga;
      private $merek;

      private function statusHarga()
      {
            if ($this->harga > 50000000) $status = 'Mahal';
            else $status = 'Murah';
            return $status;
      }

      function setMerek($x)
      {
```

```
            $this->merek = $x;
      }

      function setHarga($x)
      {
            $this->harga = $x;
      }

      function bacaMerek()
      {
            return $this->merek;
      }

      function bacaHarga()
      {
            return $this->harga;
      }

      function __construct($x, $y)
      {
            $this->merek = $x;
            $this->harga = $y;
      }
}
?>
```

Kemudian kita cek, apakah efek jika sebuah method dibuat private dengan memanggil method statusHarga() di dalam script.

contoh.php

```
<?php
include 'class-kendaraan.php';

$kendaraan1 = new kendaraan('Yamaha MIO', 10000000);
echo 'Status harga : '.$kendaraan1->statusHarga();

?>
```

Jika script di atas dijalankan, maka akan muncul pesan error sbb:

Fatal error: Call to private method kendaraan::statusHarga() from context ''

Yang menginformasikan bahwa method statusHarga() bersifat private sehingga tidak bisa diakses dari luar class.

11. Pewarisan (Inheritance)

Perhatikan kembali class 'kendaraan', selanjutnya bagaimana jika kita ingin membuat obyek baru akan tetapi obyek ini nanti berupa 'kereta api' ? Khusus kereta api ini nanti, ada properti yang digunakan untuk menyatakan jumlah gerbong. Sedangkan properti yang lain seperti merek, jumlah roda, harga dan bahan bakar sama seperti dalam class kendaraan. Oleh karena itu untuk obyek kereta api ini kita perlu membuat class baru yang merupakan pengembangan dari class kendaraan.

Dalam OOP, kita tidak perlu lagi membuat class baru ini, tapi cukup kita membuat class baru yang merupakan turunan atau warisan dari class sebelumnya. Class turunan ini, akan memiliki properti dan method yang sama seperti class pewarisnya, namun terdapat properti atau method tambahan khusus untuk class ini. Istilah pewarisan class ini dalam OOP dinamakan inheritance.

Bagaimana cara membuat class turunan ini?

```
class namaclassbaru extends namaclasslama
{
      .
      .
      .
}
```

Sebagai contoh perhatikan script berikut ini

class-kendaraan;

```
<?php

class kendaraan
{
      protected $jumlahRoda;
      public $warna;
      public $bahanBakar;
      public $harga;
      private $merek;

      private function statusHarga()
      {
            if ($this->harga > 50000000) $status = 'Mahal';
            else $status = 'Murah';
            return $status;
      }

      function setMerek($x)
      {
            $this->merek = $x;
```

```
	}

	function setHarga($x)
	{
		$this->harga = $x;
	}

	function bacaMerek()
	{
		return $this->merek;
	}

	function bacaHarga()
	{
		return $this->harga;
	}

	function __construct($x, $y)
	{
		$this->merek = $x;
		$this->harga = $y;
	}
}

class keretaApi extends kendaraan
{
	public $jumGerbong;

	function setGerbong($x)
	{
		$this->jumGerbong = $x;
	}

	function bacaGerbong()
	{
		return $this->jumGerbong;
	}

}
?>
```

Perhatikan class ‘keretaApi’ yang merupakan turunan dari class ‘kendaraan’ dalam script di atas. Dalam class tersebut, dibuat properti bernama ‘jumGerbong’ (jumlah gerbong). Selain itu, khusus untuk class ‘keretaApi’ ini dibuat juga method untuk mensetting properti jumGerbong ini dengan nama setGerbong(), serta method bacaGerbong() untuk mengakses properti jumlah gerbong.

Selanjutnya perhatikan script yang di dalamnya ada proses instantisasi obyek kereta api ini, setting properties serta memanggil method.

```
<?php
include 'class-kendaraan.php';
```

```
$kereta1 = new keretaApi('KA Lokomotif', 15000000);
$kereta1->setGerbong(20);
echo 'Jumlah gerbong dari '.$kereta1->bacaMerek().
     ' yang seharga '.$kereta1->bacaHarga().
     ' adalah '.$kereta1->bacaGerbong();

?>
```

Jika script tersebut diperhatikan, maka terdapat constructor pada class keretaApi dimana dapat dilakukan instantisasi sekaligus setting properties untuk nama merek dan harganya. Mengapa kok bisa? Ya... karena class keretaApi adalah turunan dari class kendaraan, dimana di dalam class kendaraan terdapat constructor, sehingga untuk class keretaApi inipun dapat dilakukan hal yang sama.

Selanjutnya diberikan perintah

```
$kereta1->setGerbong(20);
```

Perintah tersebut akan mensetting properties jumlah gerbong pada obyek $kereta1.

Selain itu, perintah untuk memanggil method bacaMerek() dan bacaHarga() pun juga dapat dilakukan karena class keretaApi merupakan turunan dari class kendaraan.

Adapun output di browser apabila script tersebut dijalankan adalah sbb:

“Jumlah gerbong dari KA Lokomotif yang seharga 15000000 adalah 20”

Latihan

1. Dalam script ‘class-kendaraan.php’, buatlah class baru bernama ‘pesawat’ yang merupakan turunan dari class kendaraan
2. Dalam class ‘pesawat’ yang telah dibuat, definisikan sebuah properti ‘tinggiMaks’ dengan sifat private untuk menyatakan ketinggian maksimum pesawat dan ‘kecepatanMaks’ dengan sifat private untuk menyatakan kecepatan maksimum pesawat
3. Dalam class ‘pesawat’, buatlah sebuah method bernama setTinggiMaks() untuk mensetting properti ‘tinggiMaks’ dan setKecepatanMaks() untuk setting properti kecepatan maksimum pesawat.
4. Dalam class ‘pesawat’, buatlah method bernama bacaTinggiMaks() untuk mengakses properti ‘tinggiMaks’.
5. Dalam class ‘pesawat’, buatlah method bernama biayaOperasional() untuk menentukan biaya operasional pesawat, dimana untuk menghitung biaya ini tergantung dari harga pesawat yaitu dirumuskan:

- Jika tinggi maksimum pesawat lebih dari 5000 feet dan kecepatan maks lebih dari 800 km/jam, maka biaya operasional = 30% dari harga pesawat
- Jika tinggi maksimum pesawat 3000-5000 feet dan kecepatan maks 500 – 800 km/jam, maka biaya operasional = 20% dari harga pesawat
- Jika tinggi maksimum pesawat kurang dari 3000 feet dan kecepatan maks kurang dari 500 km/jam, maka biaya operasional = 10% dari harga pesawat
- Selain itu, biaya operasionalnya = 5% dari harga pesawat

6. Berdasarkan ketentuan pada nomor 1 s/d 5, tentukan biaya operasional dari pesawat-pesawat ini

Merek Pesawat	Harga (juta)	Tinggi Maks (feet)	Kecept Maks (km/jam)
Boeing 737	2.000	7500	650
Boeing 747	3.500	5800	750
Cassa	750	3500	500

Contoh tampilan output yang diharapkan adalah sebagai berikut

Biaya operasional pesawat 'Boeing 737' dengan harga Rp 2.000.000.000 yang memiliki tinggi maksimum 7500 feet dan kecepatan maksimum 650 km/jam adalah Rp. XXXXXXX

12. Studi Kasus 01 - Operasi Bilangan dengan OOP

Pada studi kasus yang pertama ini, kita akan mencoba membuat script operasi bilangan yaitu menjumlahkan dan mengalikan dua bilangan dalam perspektif OOP.

Misalkan untuk implementasi kasus ini, kita akan buat class bernama 'operasiBilangan'. Di mana nanti akan memiliki dua properties, yaitu bilangan 1 dan bilangan 2. Kedua bilangan itu nanti akan dioperasikan melalui method-method, yaitu method untuk menjumlahkan dan mengalikan.

Untuk memudahkan implementasi, ada baiknya kita buat constructor supaya proses instantisasi dan setting properties bilangan 1 dan bilangan 2 nya bisa dilakukan dalam satu perintah saja.

Dari desain skenario tersebut, kita bisa buat scriptnya sbb:

kasus-01.php

```
<?php

class operasiBilangan
```

```
{
	// properties dari class
	private $bilangan1;
	private $bilangan2;

	// constructor
	function __construct($x, $y)
	{
		$this->bilangan1 = $x;
		$this->bilangan2 = $y;
	}

	// method untuk membaca properti bilangan1
	function bacaBilangan1()
	{
		return $this->bilangan1;
	}

	// method untuk membaca properti bilangan2
	function bacaBilangan2()
	{
		return $this->bilangan2;
	}

	// method untuk menjumlahkan bilangan1 dan bilangan2
	function jumlahkan()
	{
		$hasil = $this->bilangan1 + $this->bilangan2;
		return $hasil;
	}

	// method untuk mengalikan bilangan1 dan bilangan2
	function kalikan()
	{
		$hasil = $this->bilangan1 * $this->bilangan2;
		return $hasil;
	}
}

// instantisasi dan setting properties
$operasi1 = new operasiBilangan(4, 5);

// menampilkan hasil penjumlahan
echo '<p>Hasil penjumlahan '.$operasi1->bacaBilangan1().' dan '.$operasi1-
>bacaBilangan2().' adalah '.$operasi1->jumlahkan().'</p>';

// menampilkan hasil perkalian
echo '<p>Hasil perkalian '.$operasi1->bacaBilangan1().' dan '.$operasi1-
>bacaBilangan2().' adalah '.$operasi1->kalikan().'</p>';

?>
```

Script di atas digunakan untuk menjumlahkan dan mengalikan bilangan 4 dan 5.

Latihan

1. Tambahkan method baru bernama 'kurangkan' pada class operasiBilangan untuk melakukan proses pengurangan bilangan 1 terhadap bilangan 2.
2. Tambahkan method baru bernama 'modulo' pada class operasiBilangan untuk menghitung hasil modulo bilangan 1 terhadap bilangan 2.
3. Tambahkan method baru bernama 'pangkat' pada class operasiBilangan untuk menghitung hasil bilangan 1 dipangkatkan bilangan 2.

13. Studi Kasus 02 – Koneksi ke Database MySQL dengan OOP

Untuk studi kasus kali ini, kita akan mencoba mengimplementasikan OOP ini pada script PHP yang terkait dengan management data dalam database, khususnya MySQL. Dengan script OOP nantinya kita akan melakukan insert data ke MySQL, hapus data, edit data dan menampilkan data. Pembahasan hal ini nanti akan dibagi ke beberapa bab, mulai Studi Kasus 02 s/d Studi Kasus 06.

Sebagai contoh kasus, nantinya kita buat database untuk keperluan katalog buku. Berikut ini struktur tabel nya:

```
CREATE TABLE `buku` (
  `id` int(11) AUTO_INCREMENT,
  `judul` text,
  `pengarang` varchar(200),
  `penerbit` varchar(200),
  `tahunTerbit` varchar(4),
  PRIMARY KEY (`id`)
)
```

Sehingga untuk Studi Kasus 02 s/d Studi Kasus 06 kita akan menggunakan tabel 'buku' di atas.

Pada studi kasus 02 ini, kita akan coba membuat script PHP untuk koneksi ke MySQL dengan menggunakan OOP.

Class yang nanti akan kita buat, misalnya diberinama 'database', dan berikut ini adalah tabel properties dan method yang akan kita buat

Nama	Tipe	Sifat	Keterangan
dbHost	Properti	Private	Nama host MySQL
dbUser	Properti	Private	Username MySQL
dbPass	Properti	Private	Password MySQL

dbName	Properti	Private	Nama database MySQL
connectMySQL()	Method	Public	Melakukan koneksi ke database MySQL

Keterangan:
Untuk menentukan sifat properties dan method, apakah private, public atau protected sebenarnya tidak ada ketentuan pasti. Semuanya terserah kepada Anda dalam menentukannya. Intinya adalah jika Anda ingin properties atau method bisa dipanggil dari luar class, maka buat sebagai Public. Tapi kalau tidak, cukup dibuat private saja.

Berikut ini adalah script class 'database' nya

kasus02-class.php

```
<?php

class database
{
	// properties
	private $dbHost;
	private $dbUser;
	private $dbPass;
	private $dbName;

	// constructor
	function __construct($a, $b, $c, $d)
	{
		$this->dbHost = $a;
		$this->dbUser = $b;
		$this->dbPass = $c;
		$this->dbName = $d;
	}

	// method koneksi mysql
	function connectMySQL()
	{
		mysql_connect($this->dbHost, $this->dbUser, $this->dbPass);
		mysql_select_db($this->dbName);
	}
}

?>
```

Perhatikan class di atas, di dalam class kita buat constructor. Pembuatan constructor ini tidak wajib, dan ini suka-suka si pembuat script :-) Pembuatan constructor ini bertujuan untuk memudahkan dalam proses instantisasi dan setting propertiesnya saja.

Berikutnya, kita bisa gunakan script class di atas untuk melakukan koneksi ke MySQL.

kasus02.php

```
<?php
include 'kasus02-class.php';

// parameter koneksi mysql
$host = 'localhost';
$user = 'root';
$pass = '';
$mydb   = 'test';

// instantitasi dan setting properties obyek database
$db = new database($host, $user, $pass, $mydb);

// koneksi ke mysql via method
$db->connectMySQL();

?>
```

14. Studi Kasus 03 – Insert Data ke Database MySQL dengan OOP

Studi kasus berikutnya adalah bagaimana proses insert data ke MySQL dengan script OOP. Jika sebelumnya kita sudah membuat class 'database' dan method untuk koneksi ke mysqlnya, maka untuk insert data ini kita bisa tambahkan method tersendiri ke dalam class 'database' tersebut.

Di dalam class 'database' kita buat function 'addBuku'.

```
function addBuku($judul, $pengarang, $penerbit, $thnTerbit)
{
      $query = "INSERT INTO buku (judul, pengarang, penerbit, tahunTerbit)
                VALUES ('$judul', '$pengarang', '$penerbit', '$thnTerbit')";
      $hasil = mysql_query($query);
      if ($hasil) echo "Data buku sudah disimpan ke DB";
      else echo "Data buku gagal disimpan ke DB";
}
```

Dalam function addBuku() tersebut, terdapat 4 parameter yaitu judul, pengarang, penerbit, dan tahun terbit.

Adapun cara penggunaan method addBuku(), perhatikan script berikut ini

Kasus03.php

```
<?php
include 'kasus03-class.php';
```

```
// parameter koneksi mysql
$host = 'localhost';
$user = 'root';
$pass = '';
$mydb = 'test';

// instantitasi dan setting properties obyek database
$db = new database($host, $user, $pass, $mydb);

// koneksi ke MySQL via method
$db->connectMySQL();

// insert data buku via method
$db->addBuku('Pemrograman OOP di PHP', 'Rosihan Ari Yuana', 'Penerbit
Sendiri', '2012');
?>
```

15. Studi Kasus 04 – Menampilkan Data dari MySQL dengan OOP

Setelah proses insert data, selanjutnya kita tampilkan semua data yang sudah diinsert ke database menggunakan gaya OOP.

Pertama, kita tambahkan method untuk menampilkan data ini, misalkan diberi nama tampilBuku()

```
function tampilBuku()
{
      echo "<table border='1'>";
      echo "<tr><th>No</th><th>Judul
Buku</th><th>Pengarang</th><th>Penerbit</th><th>Tahun
Terbit</th><th>Action</th></tr>";

      // query untuk menampilkan semua data buku
      $query = "SELECT * FROM buku ORDER BY id";
      $hasil = mysql_query($query);
      $i = 1;
      while ($data = mysql_fetch_array($hasil))
      {
            echo
"<tr><td>".$i."</td><td>".$data['judul']."</td><td>".$data['pengarang']."</td
><td>".$data['penerbit']."</td><td>".$data['tahunTerbit']."</td><td><a
href='".$_SERVER['PHP_SELF']."?op=edit&id=".$data['id']."'>Edit</a> | <a
href='".$_SERVER['PHP_SELF']."?op=del&id=".$data['id']."'>Hapus</a></td></tr>
";
            $i++;
      }

      echo "</table>";
}
```

Dalam method tampilBuku() di atas, data kita sajikan dalam bentuk tabel. Selain tampilan data, dalam tabel tersebut juga kita berikan link untuk edit dan hapus data pada setiap baris datanya.

Kemudian, perhatikan script yang menggunakan method tampilBuku() tersebut.

Kasus04.php

```
<?php
include 'kasus04-class.php';

// parameter koneksi mysql
$host = 'localhost';
$user = 'root';
$pass = '';
$mydb = 'test';

// instantitasi dan setting properties obyek database
$db = new database($host, $user, $pass, $mydb);

// koneksi ke mysql via method
$db->connectMySQL();

// tampilkan data buku via method
$db->tampilBuku();
?>
```

Cukup mudah bukan dengan OOP? Setiap kali apabila kita ingin menampilkan data buku, cukup memanggil method tampilBuku() saja.

Dalam script Kasus04.php di atas, untuk fitur edit data dan hapus datanya belum bisa digunakan karena belum dibuat method atau functionnya. Pembahasan tentang hal ini akan dijelaskan di studi kasus berikutnya.

16. Studi Kasus 05 – Hapus Data dari MySQL dengan OOP

Berikutnya kita tinjau proses hapus data. Untuk keperluan ini, kita buat method dengan nama hapusBuku() pada class 'database' nya.

```
function hapusBuku($id)
{
      $query = "DELETE FROM buku WHERE id = '$id'";
      mysql_query($query);
      echo "Data buku ID ".$id." sudah dihapus";
```

```
}
```

Perhatikan function di atas, dalam function tersebut terdapat sebuah parameter $id. Parameter ini digunakan untuk menyatakan ID buku mana yang akan dihapus.

Selanjutnya perhatikan contoh script untuk proses hapus datanya.

Kasus05.php

```
<?php
include 'kasus05-class.php';

// parameter koneksi mysql
$host = 'localhost';
$user = 'root';
$pass = '';
$mydb = 'test';

// instantitasi dan setting properties obyek database
$db = new database($host, $user, $pass, $mydb);

$db->connectMySQL();

// proses hapus data
if (isset($_GET['op']))
{
      if ($_GET['op'] == 'del')
      {
            // baca ID dari parameter ID buku yang akan dihapus
            $id = $_GET['id'];
            // proses hapus data buku berdasarkan ID via method
            $db->hapusBuku($id);
      }
}

// tampilkan semua data buku
$db->tampilBuku();
?>
```

Jika kita perhatikan script di atas, maka letak proses hapus data dilakukan sebelum (di atas) method untuk menampilkan data buku. Hal ini bertujuan supaya data yang tampil pada method tampilBuku() merupakan list data yang sudah terbaru setelah proses penghapusan. Jika letak proses hapus data dilakukan setelah atau di bawah tampilBuku() maka Anda harus merefresh script kasus05.php ini setelah proses penghapusan supaya data yang sudah dihapus tidak muncul di tampilBuku().

17. Studi Kasus 06 – Edit Data dari MySQL dengan OOP

Untuk proses edit data ini, nantinya kita akan membuat 2 method, yaitu method untuk membaca data buku yang akan di edit berdasarkan ID tertentu. Data buku yang dibaca ini selanjutnya akan ditampilkan di form edit. Kemudian, method berikutnya kita gunakan untuk proses update datanya.

Sekarang perhatikan method bacaDataBuku() untuk baca data buku berdasarkan ID nya berikut ini

```
function bacaDataBuku($type, $id)
{
      $query = "SELECT * FROM buku WHERE id = '$id'";
      $hasil = mysql_query($query);
      $data  = mysql_fetch_array($hasil);
      if ($type == 'judul') return $data['judul'];
      else if ($type == 'pengarang') return $data['pengarang'];
      else if ($type == 'penerbit') return $data['penerbit'];
      else if ($type == 'thnTerbit') return $data['tahunTerbit'];
}
```

Function bacaDataBuku() di atas terdapat dua parameter yaitu $type dan $id. Parameter $type digunakan untuk menentukan tipe data apa yang menjadi return value nya.

Jika nilai $type nya adalah 'judul', maka method bacaDataBuku() ini akan menghasilkan return value judul buku dari ID buku tertentu. Jika $type nya 'pengarang' maka return value nya adalah nama pengarangnya, demikian juga untuk $type nya 'penerbit' maupun 'thnTerbit'.

Manfaat dari bentuk method seperti ini kita cukup membuat sebuah method saja untuk membaca semua keterangan data buku berdasarkan ID bukunya.

Adapun cara pemanggilan method bacaDataBuku() ini adalah sebagai berikut:

$db->bacaDataBuku('judul', $id);

Untuk membaca Judul Buku dari ID buku $id,

$db->bacaDataBuku('pengarang', $id);

Untuk membaca nama pengarang dari ID buku $id

dan seterusnya.. Anda bisa lihat detailnya di script kasus06.php

Selanjutnya kita buat method updateDataBuku() untuk proses update datanya

```
function updateDataBuku($id, $judul, $pengarang, $penerbit, $thnTerbit)
{
      $query = "UPDATE buku SET
```

```
                judul = '$judul', pengarang = '$pengarang',
                penerbit = '$penerbit', tahunTerbit = '$thnTerbit'
                WHERE id = '$id'";
        mysql_query($query);
        echo "Data buku sudah diupdate";
}
```

Pada function updateDataBuku() tersebut terdapat 5 parameter di mana $id menunjukkan ID dari buku yang akan diedit, $pengarang menyatakan nama pengarang buku yang diupdate, $penerbit untuk nama penerbit, dan $thnTerbit menunjukkan tahun terbit buku.

Cara pemanggilan method updateDataBuku() bisa Anda lihat di script kasus06.php

Sampai bab ini, diharapkan Anda sudah bisa menguasai teknik pemrosesan data MySQl dengan gaya pemrograman OOP. Jika diperhatikan, dalam setiap script untuk masing-masing studi kasus selalu dibuat proses instantisasi dan setting properties sbb:

```
// parameter koneksi mysql
$host = 'localhost';
$user = 'root';
$pass = '';
$mydb   = 'test';

// instantitasi dan setting properties obyek database
$db = new database($host, $user, $pass, $mydb);
```

Hal ini dilakukan karena masing-masing script letaknya terpisah, sehingga untuk melakukan instantisasi ini dilakukan pada setiap script. Namun, apabila proses insert data, tampil data, edit data dan hapus data diletakkan dalam script yang sama, maka proses instantisasi cukup dilakukan sekali saja.

Latihan

1. Tambahkan field baru bernama ‘jenis’ dalam tabel ‘buku’.
   Keterangan:
   Field ‘jenis’ ini menunjukkan jenis bukunya, yaitu: text book, majalah, atau tutorial
2. Modifikasi script untuk insert data, dan edit data supaya script bisa digunakan untuk struktur data di tabel ‘buku’ yang baru (setelah penambahan field ‘jenis’).
3. Buatlah method baru dengan nama: cariBuku($keyword) yang akan digunakan untuk proses pencarian buku berdasarkan $keyword judul buku.
4. Buatlah form pencarian buku berdasarkan judul buku menggunakan method cariBuku($keyword) tersebut.

18. Studi Kasus 07 – Membuat Script Login dengan OOP

Pada studi kasus ini kita akan membuat script login dimana scriptnya kita buat dengan gaya pemrograman OOP.

Sebelum kita membuat detil script PHP nya, terlebih dahulu kita rancang class-class beserta properties dan methodnya. Berikut ini gambaran class yang akan dibuat. Perancangan class ini perlu dilakukan supaya sejak awal kita sudah membuat semacam perencanaan, karena perancangan class ini merupakan suatu roadmap dari aplikasi yang akan kita buat. Di dalam perjalanannya nanti, class dapat saja berubah properties maupun methodnya seiring dengan kebutuhannya.

Class: Database

Class ini nanti digunakan untuk mengatur segala tentang koneksi database ke MySQL.

Nama Properties	Deskripsi
dbHost	Nama host
dbUser	Username MySQL
dbPass	Password MySQL
dbName	Nama Database

Nama Method	Deskripsi
connectMySQL()	Melakukan koneksi ke MySQL

Class: login

Class ini digunakan untuk hal-hal terkait dengan proses login seorang user

Nama Properties	Deskripsi
Username	Nama user login
Password	Password user login

Nama Method	Deskripsi
prosesLogin()	Melakukan proses login
bacaNamaUser()	Membaca nama user yang sedang login
bacaStatusLogin()	Membaca status seorang user apakah dia sudah login atau belum
redirect($url)	Melakukan redirecting ke halaman $url setelah login sukses
sapaUser()	Menampilkan string sapaan bagi user yang telah login

prosesLogout()	Melakukan proses logout
validasiLogin()	Menampilkan 'Anda belum login' ketika seseorang mengakses halaman tertentu ketika belum login

Class: menu

Class ini digunakan untuk hal yang terkait dengan menu si user setelah login sukses

Nama Properties	Deskripsi
listMenu	Daftar menu

Nama Method	Deskripsi
tampilMenu()	Menampilkan daftar menu

NB: Rancangan class yang terdiri dari properties dan method di atas bisa dikembangkan sendiri sesuai kebutuhan Anda.

Selanjutnya untuk studi kasus ini, struktur tabel yang digunakan adalah sbb:

```
CREATE TABLE `users` (
  `username` varchar(50),
  `password` varchar(100),
  `nama` varchar(100),
  PRIMARY KEY (`username`)
)
```

Secara detail untuk method dalam setiap class, bisa dilihat di script 'kasus07-class.php'

Penggunaan class dari script 'kasus07-class.php' nya ada dalam script 'kasus07.php'

Latihan

1. Tambahkan sebuah field baru bernama 'level' di dalam tabel 'users'. Field ini menunjukkan level si user apakah 'administrator' atau 'operator'
2. Buatlah method di dalam class 'login' dengan nama getLevel($username) yang digunakan untuk membaca level si user berdasarkan $username nya
3. Dengan menggunakan method getLevel() ini, modifikasilah method tampilMenu() dalam class 'menu' supaya menampilkan daftar menu sesuai levelnya. Jika levelnya 'administrator' maka akan tampil semua menu, namun jika 'operator' hanya akan tampil menu halaman 1 dan 2 saja.

Tentang Penulis

Penulis merupakan staff pengajar (dosen) di program studi Pendidikan Matematika pada Fakultas Keguruan dan Ilmu Pendidikan (FKIP) di Universitas Sebelas Maret Surakarta (UNS). Sehari-hari penulis mengajar matakuliah pemrograman komputer, computer aided learning, dan web based learning. Selain mengajar, penulis juga diberikan amanah untuk mengelola ICT Center di FKIP UNS (http://ficos.fkip.uns.ac.id)

Di samping menjadi staff pengajar, penulis juga merupakan seorang developer software khususnya yang berbasis SMS Gateway. Salah satu proyek besar yang telah dibuatnya adalah GampSMS. (http://gampsms.rosihanari.net)

Di sela-sela kesibukannya, penulis berusaha menyempatkan untuk menshare berbagai tutorial tentang programming di blognya (http://blog.rosihanari.net) dan menulis beberapa buku tentang komputer serta matematika.

Penulis berharap, semoga karya-karya yang dihasilkan bisa memberikan manfaat positif bagi ummat dan membawa keberkahan bagi semuanya. Amin....

Baarokallahu fiikum...