BAB

Pemprograman Berorientasi objek (OOP)

Pertemuan kali ini akan membahas tentang Pemprograman Berorientasi objek atau disingkat dengan istilah OOP (Object Oriented Programming). Apakah kalian sudah pernah menggunakan OOP sebelumnya atau belum pernah menggunakan? Pada bab ini kita akan belajar atau mengenal lebih jauh apa itu OOP, bagaimana cara penggunaaanya dan kenapa harus menggunakan objek apakah sudah cukup dengan variable saja. Kita akan membahasnya satu persatu mulai bagaimana kita membuat objek, dasar – dasar penggunaan antarmuka, *exceptions, and magic methods.* Cara pendekatan Pemprograman Berorientasi objek lebih focus pada pendekatan secara konsepsual dari pada secara teknis, meskipun akan kita jumpai beberapa kata panjang yang digunakan dalam mendifinisikan dan demistifikasi saat penerapanya nantinya.

1. **Mengapa OOP penting dalam PHP ?**

Karena sangat memungkinkan ketika kita menuliskan situs web yang sangat kompleks bisa saja cukup menggunakan fungsi, anda mungkin akan bertanya kenapa harus menggunakan OOP mungkin bisa berfikiran akan membuat repot ketika menggunakan OOP dan kenapa dalam PHP harus menggunakan OOP. Pada mulanya PHP hanyalah berupa kumpulan script yang sederhana, seiring dengan perkembangan PHP maka ditambhakan fitur-fitur pemrograman berorientasi objek yang dimulai pada PHP 4 dan pada perkembangan terakhir di PHP 5 fitur-fitur pemrograman berorientasi objek semakin baik dan semakin cepat. Dengan penerapan pemrograman berorientasi objek pada PHP 5, script yang menerapkan konsep pemrograman berorientasi objek akan lebih cepat dan lebih efisien.

1. **Apa perbedaan diantara procedural dan OOP ?**

Pemrograman Prosedural adalah dilakukan dengan memberikan serangkaian perintah yang berurutan. Paradigma ini didasari oleh konsep mesin Von Newman (stored program concept) sekelompok tempat penyimpanan (memori), yang dibedakan menjadi memori instruksi dan memori data, masing-masing memori tersebut dapat diberi nama dan nilai, selanjutnya instruksi akan dieksekusi satu persatu secara sekuensial oleh sebuah proses tunggal.

Program dalam paradigma ini berdasarkan pada struktur informasi di dalam memori dan manipulasi dari informasi yang disimpan tersebut. Kata kunci yang sering digunakan dalam paradigma ini adalah:

Algoritma + Struktur Data = Program

Contoh bahasa pemrogaman yang menggunakan paradigma prosedural atau imperatif adalah: Bahasa-bahasa tingkat tinggi seperti Cobol, Basic, Pascal, Fortran, dan C/C++ mendukung kegiatan pemrograman prosedural, karena itu mereka dinamakan juga bahasa prosedural.

Kelebihan Prosedural :

1. Memiliki algoritma pemecahan masalah yanag sederhana, standar dan efektif.
2. Penulisan program memiliki struktur logika yang benar dan mudah dipahami
3. Program hanya terdiri dari 3(tiga) struktur dasar, yaitu struktur berurutan, struktur seleksi dan struktur perulangan.
4. Memiliki dokumentasi yang baik.
5. Menghindari penggunaan pernyataan GOTO, yang akan menjadikan program tidak terstruktur dengan baik.

Kekurangan Prosedural :

1. Program cukup sulit untuk proses perawatan.
2. Fungsi yang tersedia, susah untuk diubah tanpa harus mempengaruhi fungsi sistem secara keseluruhan.
3. Butuh usaha yang keras untuk menterjemahkan Business Models dalam programming models.
4. Mungkin dapat bekerja dengan baik pada saat terisolasi tapi tidak pada saat terintegrasi dengan sistem lain.

Pemrograman berbasis object (OOP) adalah sebuah paradigma pemrograman yang berorientasikan kepada object. Semua data dan fungsi dalam paradigma ini dibungkus dengan class-class atau object-object. Dalam pemrograman berbasis objek, kita diminta untuk memahami sebuah masalah dan memodelkan masalah tersebut menjadi sebuah class dan kemudian class akan diinstansiasi menjadi sebuah object pada saat runtime. Setiap class/object dalam pemrograman berbasis object dapat saling berinteraksi dan berkomunikasi satu sama lain untuk mendukung sebuah solusi dari suatu masalah

Kelebihan OOP dibandingkan procedural

1. Lebih terstruktur dan mudah dibaca.
2. Class-Class dapat digunakan kembali pada project yang lain (reuse).
3. Pemetaan masalah jadi lebih mudah sehingga mudah untuk membuat solusinya.
4. Pembatasan akses terhadap suatu fungsi dapat meningkatkan keamaan code.
5. Interaksi antara code lebih terasa.

Bahasa yang mendukung konsep Pemrograman berbasis objek

Bila Anda menguasai OOP, maka Anda akan lebih mudah mempelajari bahasa pemrograman yang mendukung OOP. Adapun bahasa pemrograman yang mendukung OOP antara lain:

* PHP
* Java
* C++
* Python
* Net
* Ruby
* Go

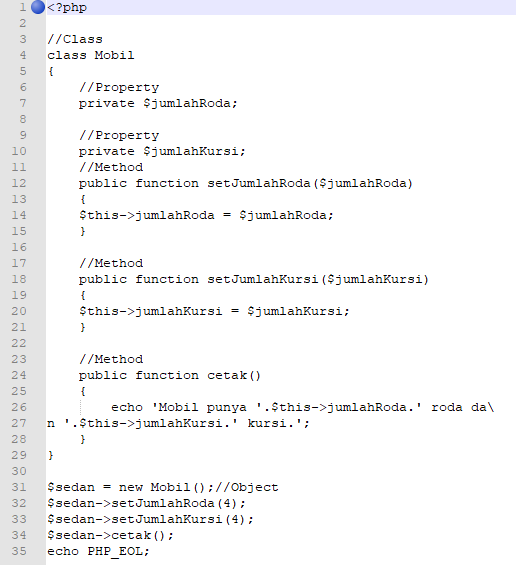
1. **Class dan Object**

3. 1. Pengertian Class

Secara gampang, class adalah sebuah model/cetakan sedangkan object adalah realisasinya. Dalam OOP, Class memiliki property dan method. Property adalah sesuatu yang dimiliki oleh class,sedangkan method adalah apa-apa saja yang bisa dilakukan oleh class. Bila diibaratkan dengan Mobil, maka property adalah roda, kursi, pintu, dan lain sebagainya. Sedangkan method adalah maju, mundur,berbelok, mengerem dan lain sebagainya.

* 1. Contoh Class

Setelah kita memahami pengertian dari class, tidak lengkap rasanya kalau tidak ada contoh penggunaannya. Contoh dibawah ini akan memberikan gambaran lebih dalam tentang class.



Pada contoh diatas, class Mobil adalah sebuah prototype/model sedangkan $sedan adalah realisasinya. Sementara $jumlahRoda dan $jumlahKursi adalah property. Sedangkan setJumlahRoda($jumlahRoda) sampai pada cetak() dinamakan method (akan dijelaskan secara khusus pada bab tersendiri).

**Catatan**: Class adalah cetakan, object adalah barang jadinya/realisasinya. Proses membuat object disebut instansiasi.

* 1. Latihan

1. Buatlah sebuah class bernama “buku”, kemudian deklarasikan properties dari buku tersebut, misalkan judul buku, penerbit, pengarang, tahun terbit dll.
2. Rancanglah sebuah class pohon, kemudian tuliskan sendiri properties yang dimiliki suatu mobil, selanjutnya nyatakan dalam program php.
   1. Bagaimana cara membuat Function/Method dalam Class

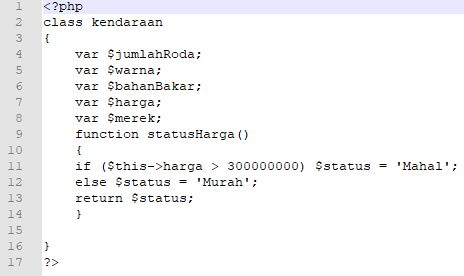
Bagaimana sebuah class bisa dibuat function? Suatu function dalam suatu class dinamakan method, dan sebuah method dapat kita membayangkannya adalah segala hal yang terkait dengan pekerjaan atau proses yang dapat diberikan pada suatu obyek. berikut contohnya method dalam kehidupan sehari-hari, adalah pada obyek seorang ‘siswa’. Sebuah method kita bisa berikan pada siswa tersebut misalnya: ‘daftar sekolah. Di dalam method ‘daftar sekolah’ itu terdapat serangkaian proses mulai dari

- pendaftaran sekolah

- mengikuti tes seleksi

- Jika tes tidak lulus, maka bisa mengikuti tes lagi

Itu tadi sebagai contoh gambaran method dalam kehidupan sehari-hari. Selanjutnya adalah 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 300 juta, dan jika di 300 juta ke bawah dikatakan murah.



Perhatikan perintah:

$this->harga

Variabel $this disini 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;

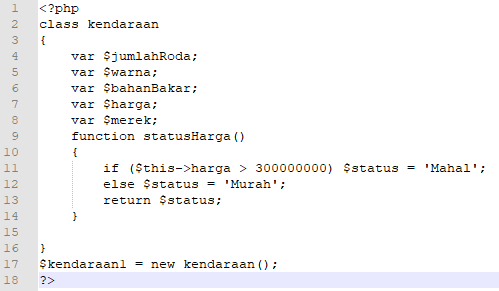
* 1. Latihan

1. Pada contoh kelas ‘kendaraan’, tambahkan sebuah property ‘tahun perakitan’
2. Buatlah function dalam kelas ‘kendaraan’ dengan nama ‘potongan pajak()’ untuk menentukan apakah suatu kendaraan mendapatkan potongan pajak atau tidak. Kendaraan yang mendapat potongan pajak adalah yang kapasitas mesin ‘dibawah 2000 cc’ dan tahun pembuatannya sebelum tahun 2010. Function ini untuk menentukan ‘Ya’ jika mendapat mendapat potongan pajak, dan ‘Tidak’ jika tidak mendapat potongan pajak.
3. Buatlah function dalam kelas ‘kendaraan’ dengan nama ‘hargabekas()’ untuk menentukan harga bekas dari kendaraan tersebut. Function ini menentukann harga bekas dari kendaraan dengan ketentuan:
4. Jika tahun pembuatan di atas 2019, maka harga second nya turun 15% dari harga aslinya
5. Jika tahun pembuatan 2015 s/d 2018, maka harga second nya turun 20% dari harga aslinya
6. Jika tahun pembuatan di bawah 2015, maka harga second nya turun 300% dari harga aslinya..
7. **Instantisasi Obyek**

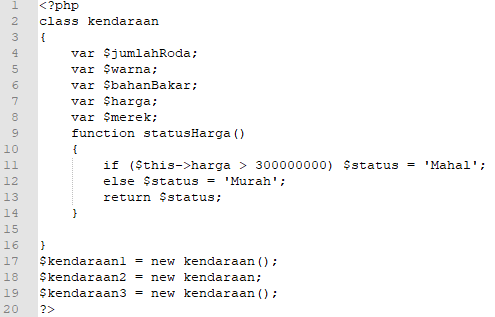
Jika kalian sudah memahami yang telah dijelaskan sebelumnya bahwa sebuah class merupakan blueprint dari obyek. Sebuah class belum menjadi obyek sebelum kita lakukan sebuah proses instantisasi obyek. Untuk melakukan instantisasi obyek, perintahnya adalah sbb:

$handle = new namaclass();

Sebagai contoh, misalkan kita lakukan instantiasi pada class kendaraan



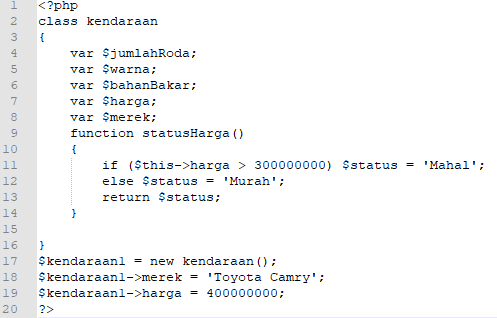
Jika script di atas dijalankan, maka di browser tidak muncul apa-apa. Hal ini terjadi karena kita belum menyuruh PHP untuk melakukan sesuatu pada obyek $kendaraan1 tersebut. Variabel $kendaraan1 dalam hal ini dinamakan ‘handle’ karena kita akan gunakan kendaraan1 untuk mengontrol dan menggunakan obyek kendaraan. kita juga bisa melakukan instantisasi obyek tanpa menggunakan kurung, perhatikan contoh berikut ini yang menunjukkan proses instantisasi beberapa obyek dari class kendaraan.



1. **Setting Properties**

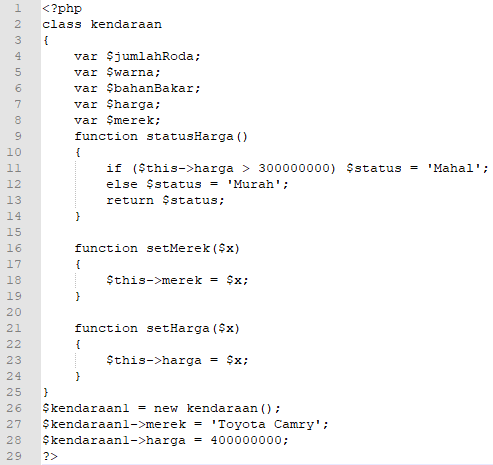
Setelah suatu obyek kita lakukan instantitasi, selanjutnya kita bisa mensetting properties dari obyek tersebut. Sebagai contoh, misalkan kita telah membuat obyek $kendaraan1, kemudian bagaimana kita menset properti harga dan merek dari obyek $kendaraan1 ini? Kita dapat mensetting properties dari suatu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

$namaobyek->properti = value; Perhatikan contoh berikut ini:



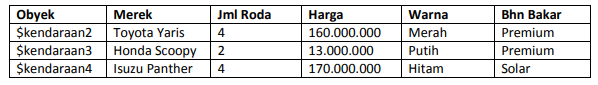
kendaraan1->merek = ‘Toyota camry';

Digunakan untuk mensetting properti merek ‘Toyota camry' dari obyek $kendaraan1. Kita juga bisa menggunakan method untuk proses setting properti ini, dan ini adalah cara yang lebih direkomendasikan dalam OOP.



* 1. Latihan

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



1. **Menjalankan Method**

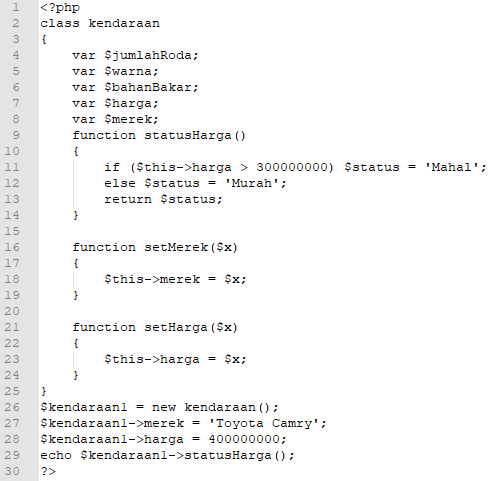
Dalam bagian ini, akan dijelaskan cara menjalankan sebuah method dari suatu obyek. Ingat, bahwa menjalankan sebuah method dari suatu obyek pada intinya adalah memanggil function yang dalam class.

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

$kendaraan1->setMerek(‘Toyota Camry’);

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.



Jika script di atas dijalankan, maka akan muncul ‘Mahal’, karena harga nya lebih dari 300.000.000. Perhatikan dari beberapa contoh pemanggilan method di atas, bahwa setiap kali pemanggilan method jangan lupa memberi tanda kurung (), seperti pada $kendaraan1->setHarga(300000000);

Atau

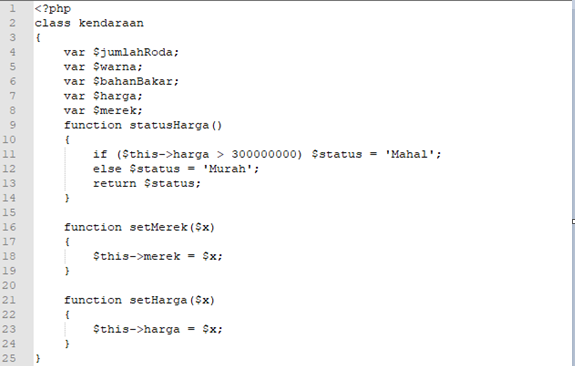
$kendaraan1->statusHarga();

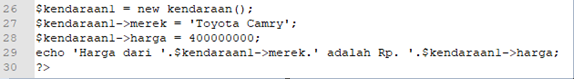
Karena kurung tersebut digunakan untuk meletakkan parameter bagi method tersebut.

Latihan

1. Perhatikan kembali soal latihan sebelumnya . Tampilkan status harga dari $kendaraan2, $kendaraan3 dan $kendaraan4.
2. Perhatikan kembali soal latihan sebelumnya. Tampilkan harga bekas dari $kendaraan2, $kendaraan3 dan $kendaraan4.
3. **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





Perhatikan pada bagian perintah

$kendaraan1->harga

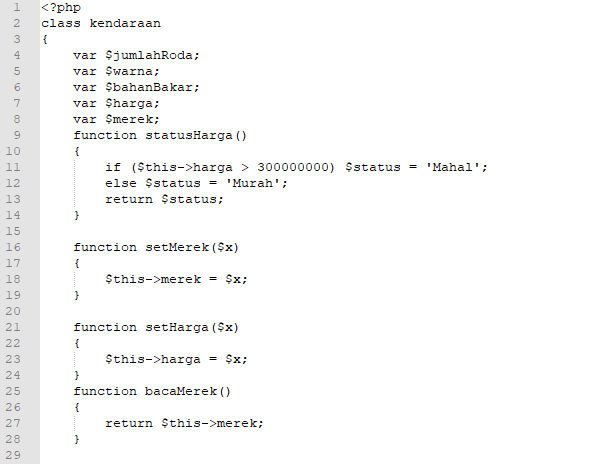
dan

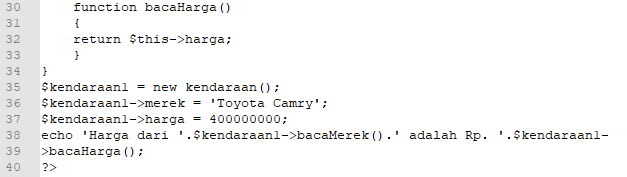
$kendaraan1->merek

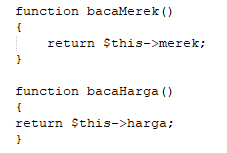
Kedua perintah di atas adalah digunakan untuk mengakses value dari property obyek $kendaraan1, yaitu ‘merek’ dan ‘harga’. Jika script di atas dijalankan, maka akan diperoleh output

**“Harga dari Toyota Camry adalah Rp. 400000000”**

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.





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

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()

**7.1. Latihan**

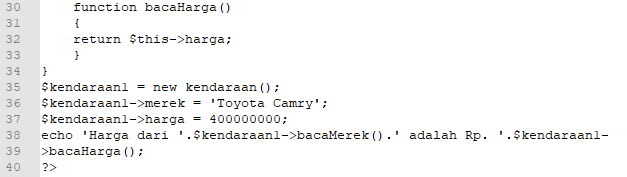
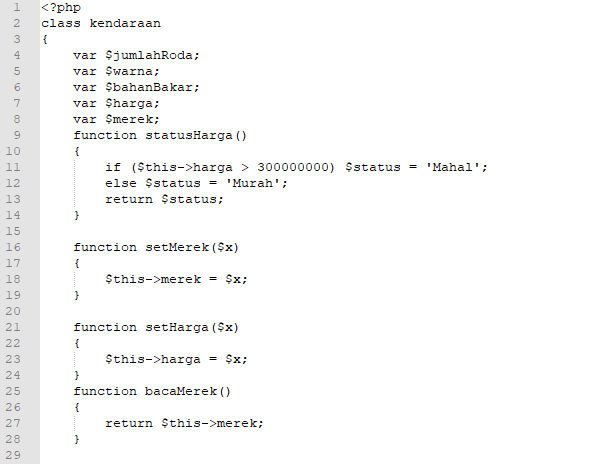
Perhatikan kembali soal latihan sebelumnya, berdasarkan obyek yang telah dibuat, tampilkan properti setiap obyek sedemikian hingga tampilan script apabila dijalankan di browser sebagai berikut:

* 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.

1. **Modularitas Class**

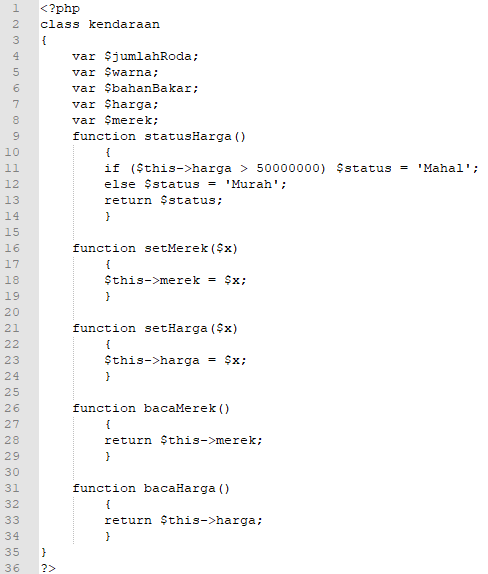
Pada contoh-contoh script di atas, class dan juga proses instantisasi dijadikan satu dalam sebuah script. Hal ini dirasa kurang efektif apabila class tersebut juga digunakan dalam script yang lain nantinya. Sehingga untuk alasan kemudahan penggunaan, biasanya sebuah class atau kumpulan class diletakkan 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 berbentuk sbb:



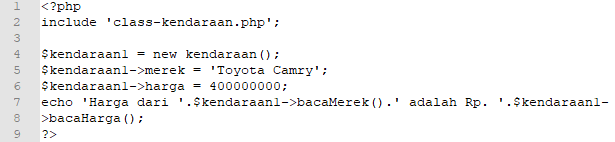
Kita dapat memisahkan class ‘kendaraan’ ini dalam file tersendiri misalkan diberinama ‘classkendaraan.php’ yang isinya

**class-kendaraan.php**



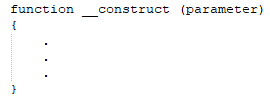
Selanjutnya kita include kan file class-kendaraan.php ini ke dalam script lain apabila kita

memerlukannya,

**contoh.php** 

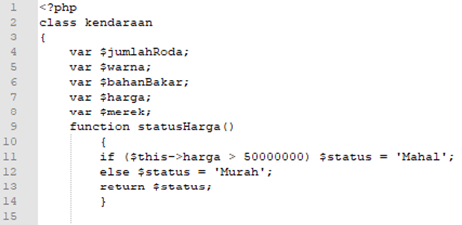
1. **Constructor**

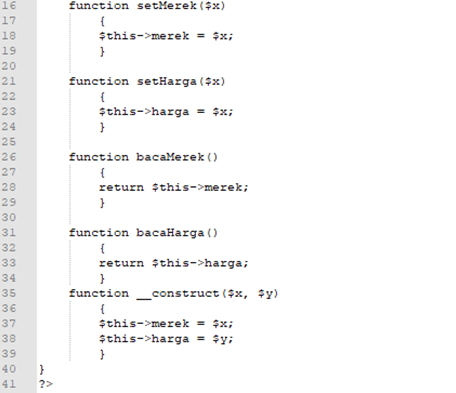
Perhatikan kembali proses instantisasi dan setting properties yang kita bahas sebelumnya. Jika kita perhatikan, maka proses instantisasi dan setting properties ini dilakukan secara terpisah. Tentu saja proses ini agak terlalu bertele-tele. Ternyata kita bisa langsung melakukan instantisasi obyek sekaligus melakukan setting propertiesnya. Proses ini dapat dilakukan dengan menggunakan ‘constructor’. Untuk membuat constructor, kita cukup membuat sebuah function dalam class dengan bentuk



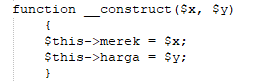
Keterangan: Tanda \_\_ merupakan tanda underscore (\_) yang ditulis double.

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



****

Perhatikan



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

$this->merek = $x;

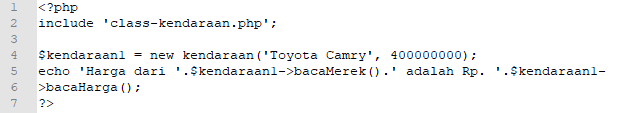
Digunakan untuk setting property merek kendaraan berdasarkan nilai $x. Demikian juga perintah

$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**



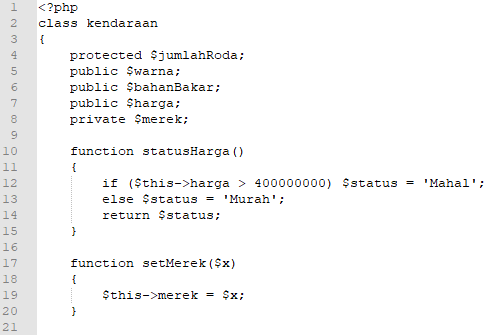
1. **Encapsulation**

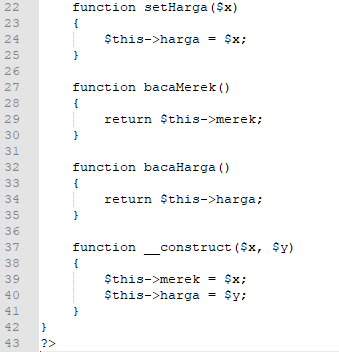
Di dalam dasar-dasar OOP, ada istilah encapsulation. Istilah ini terkait dengan aksesibilitas properties dalam suatu class. Dengan encapsulation ini, kita bisa mengatur sebuah properti apakah hanya bisa diakses dalam class tersebut saja, atau tidak.

Aksesibilitas properties dalam encapsulation ini ada tiga sifat:

1. Public : properti dapat diakses darimanapun
2. Private : properti hanya dapat diakses dari dalam class saja
3. Protected : properti hanya dapat diakses dari dalam class atau class turunan (inheritance class)

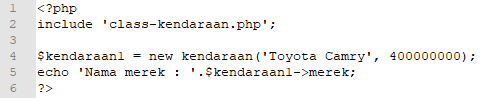
Untuk membedakan ketiganya, perhatikan contoh berikut ini

**class-kendaraan.php**



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**



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

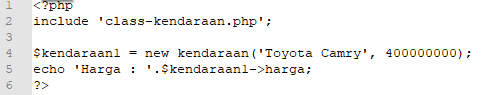
$kendaraan1->merek

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

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

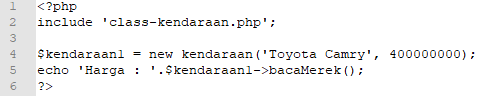
Hal ini terjadi karena properti merek bersifat private, sehingga properti ini tidak bisa diakses dari luar class. Bagaimana dengan akses ke properti harga secara langsung? Perhatikan script berikut ini

contoh.php



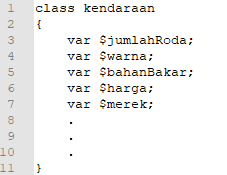
Ternyata jika script di atas dijalankan,bisa memunculkan harga dari Toyota camry, yang menjadi pertanyaan, apakah bisa kita mengakses sebuah properti yang sifatnya private dalam class dari luar? Jawabnya adalah bisa, namun tidak dilakukan secara langsung dengan mengakses properti nya namun menggunakan method. Sebagai contoh, misalkan kita ingin mengakses property merek yang sifatnya private, maka kita bisa menggunakan method bacaMerek().

**contoh.php**

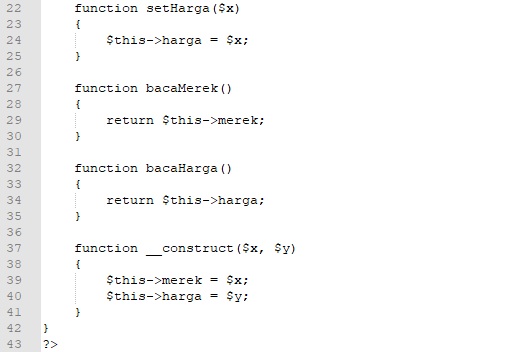
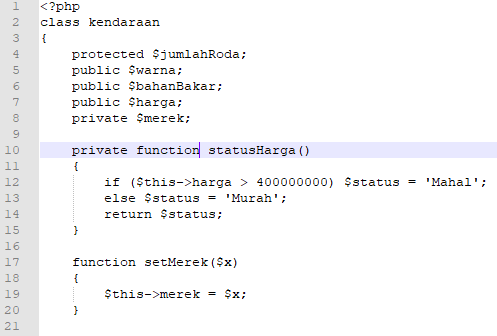


Terus bagaimana dengan deklarasi properties menggunakan ‘var’ seperti pada contoh-contoh di awal,

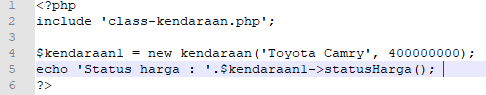
misalnya:



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** 

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

contoh.php 

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

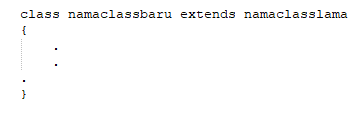
**Fatal error**: Call to private method kendaraan::statusHarga() from context ''

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

1. **Pewarisan (Inheritance)**

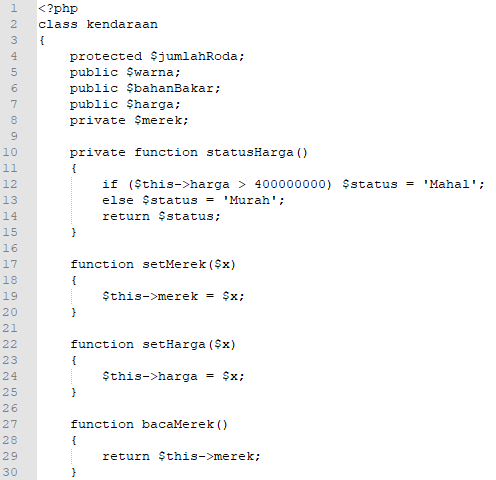
Perhatikan kembali class ‘kendaraan’, selanjutnya bagaimana jika kita ingin membuat obyek baru akan tetapi obyek ini nanti berupa ‘kereta api’ ? Khusus kereta api ini nanti, ada properti yang digunakan untuk menyatakan jumlah gerbong. Sedangkan properti yang lain seperti merek, jumlah roda, harga dan 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.

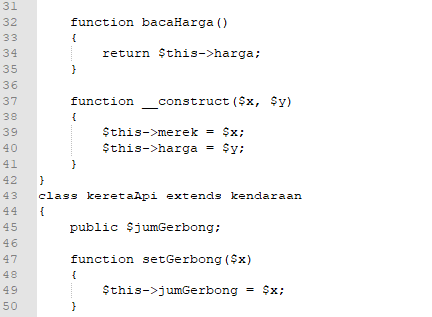
Dalam OOP, kita tidak perlu lagi membuat class baru ini, tapi cukup kita membuat class baru yang merupakan turunan atau warisan dari class sebelumnya. Class turunan ini, akan memiliki properti dan method yang sama seperti class pewarisnya, namun terdapat properti atau method tambahan khusus untuk class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

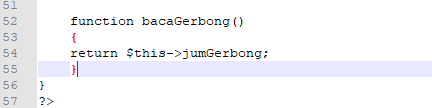


Sebagai contoh perhatikan script berikut ini

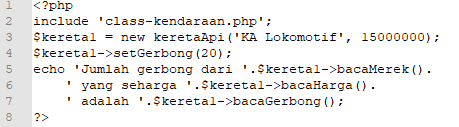
class-kendaraan;







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.



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:

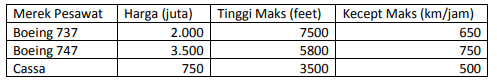
“Jumlah gerbong dari KA Lokomotif yang seharga 15000000 adalah 20”

11.1. 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 property ‘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

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



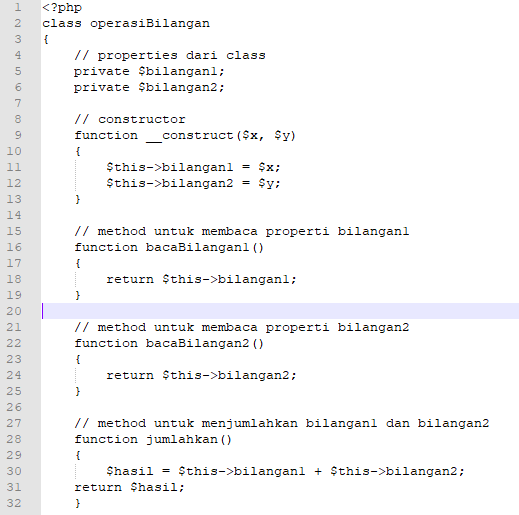
Contoh tampilan output yang diharapkan adalah sebagai berikut

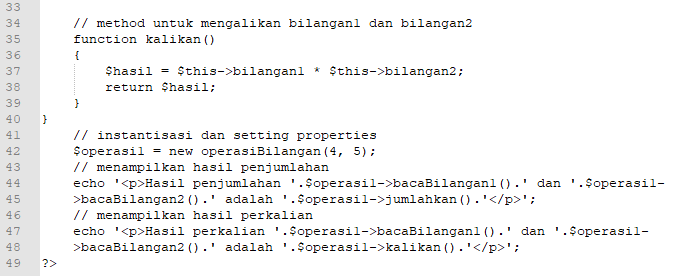
Biaya operasional pesawat ‘Boeing 737’ dengan harga Rp 2.000.000.000 yang memiliki tinggi maksimum 7500 feet dan kecepatan maksimum 650 km/jam adalah Rp. XXXXXXX

**X11. Studi Kasus**

**12.1. 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 



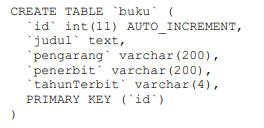
Script di atas digunakan untuk menjumlahkan dan mengalikan bilangan 4 dan 5

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.

**12.2. 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:



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.

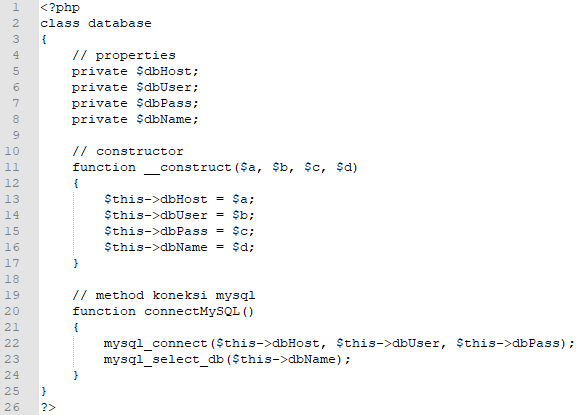
Class yang nanti akan kita buat, misalnya diberinama ‘database’, dan berikut ini adalah tabel properties dan method yang akan kita buat

|  |  |  |  |
| --- | --- | --- | --- |
| 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 koneneksi 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

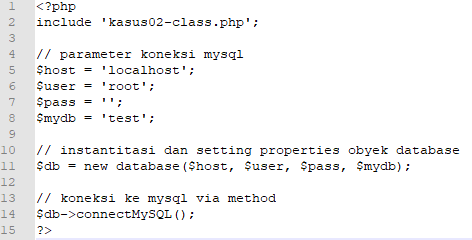
kasus02-class.php



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.

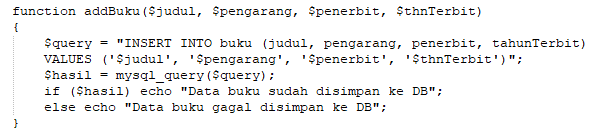
kasus02.php



**12.3. 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.

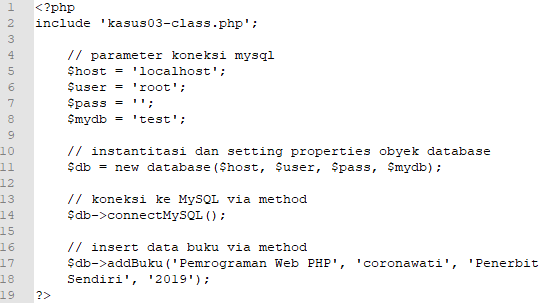
Di dalam class ‘database’ kita buat function ‘addBuku’.



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

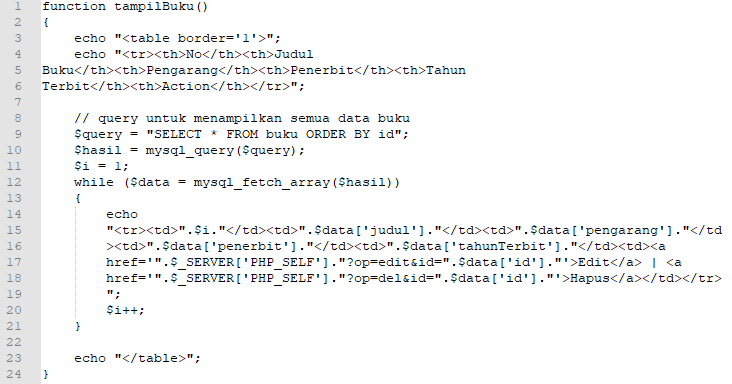
Adapun cara penggunaan method addBuku(), perhatikan script berikut ini

Kasus03.php



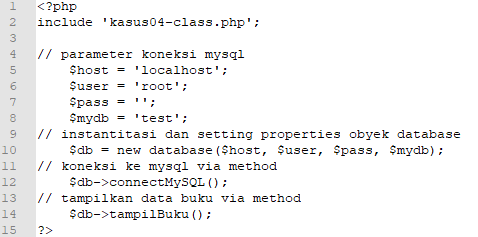
12.4. Studi Kasus 04 – Menampilkan Data dari MySQL dengan OOP

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()



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.

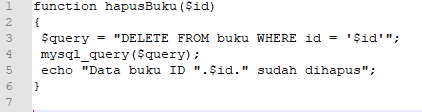
Kasus04.php



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.

**12.5. Studi Kasus 05 – Hapus Data dari MySQL dengan OOP**

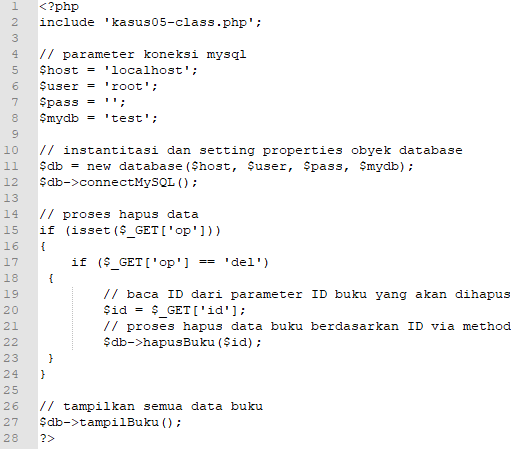
Berikutnya kita tinjau proses hapus data. Untuk keperluan ini, kita buat method dengan nama hapusBuku() pada class ‘database’ nya.



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

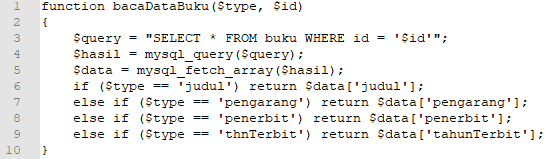


Jika kita perhatikan script di atas, maka letak proses hapus data dilakukan sebelum (di atas) method untuk menampilkan data buku. Hal ini bertujuan supaya data yang tampil pada method tampilBuku() merupakan list data yang sudah terbaru setelah proses penghapusan. Jika letak proses hapus data dilakukan 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().

**12.6. 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() 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);

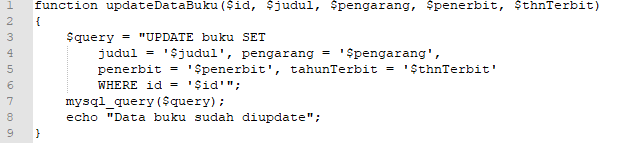
Untuk membaca Judul Buku dari ID buku $id,

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

Untuk membaca nama pengarang dari ID buku $id

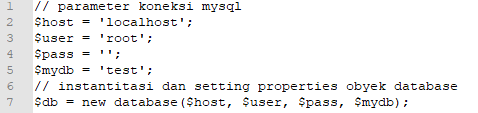
dan seterusnya.. Anda bisa lihat detailnya di script kasus06.php

Selanjutnya kita buat method updateDataBuku() untuk proses update datanya.



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:



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

1. Modifikasi script untuk insert data, dan edit data supaya script bisa digunakan untuk struktur data di tabel ‘buku’ yang baru (setelah penambahan field ‘jenis’).
2. Buatlah method baru dengan nama: cariBuku($keyword) yang akan digunakan untuk proses pencarian buku berdasarkan $keyword judul buku.
3. Buatlah form pencarian buku berdasarkan judul buku menggunakan method cariBuku($keyword) tersebut.

12.7. 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 MySQL |
| dbUser | Username MySQL |
| dbPass | Password MySQL |
| dbName | Nama database MySQL |
| connectMySQL() | Melakukan koneneksi ke database MySQL |
| Nama Method | Deskripsi |
| connectMySQL() | Melakukan koneneksi ke database 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 paasword |

|  |  |
| --- | --- |
| 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 |

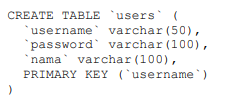
Class: menu

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: 

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.