DFP Pranata Komputer Terampil
Pusdiklat Keuangan Umum
2012

Web Programming

by Hafid Mukhlasin

Outline

No	Materi
I	Introduction to the World of Web Standards
II	HTML Basic
III	CSS Basic
IV	Little Javascript Basic
V	PHP Basic
VI	**PHP OOP Basic**
VII	MySQL Basic
VIII	Implementation PHP & MYSQL

n

VI. PHP OOP Basic

Outline

A. Apa itu OOP
B. Perkembangan OOP di PHP
C. Istilah-istilah dalam OOP
D. Implementasi OOP Dasar di PHP

A. Apa itu OOP

Pemrograman berorientasi objek atau *object-oriented programming* (OOP) merupakan suatu pendekatan pemrograman yang menggunakan object dan class. Saat ini konsep OOP sudah semakin berkembang. Hampir setiap perguruan tinggi di dunia mengajarkan konsep OOP ini pada mahasiswanya. Pemrograman yang banyak dipakai dalam penerapan konsep OOP adalah Java dan C++.

OOP bukanlah sekedar cara penulisan sintaks program yang berbeda, namun lebih dari itu, OOP merupakan cara pandang dalam menganalisa sistem dan permasalahan pemrograman. Dalam OOP, setiap bagian dari program adalah *object*. Sebuah *object* mewakili suatu bagian program yang akan diselesaikan. Beberapa konsep OOP dasar, antara lain :

1. *Encapsulation* (*Class* dan *Object*)
2. *Inheritance* (Penurunan sifat), dan
3. *Polymorphisme*

B. Perkembangan OOP di PHP

PHP pada awalnya hanyalah kumpulan script sederhana. Dalam perkembangannya, selanjutnya ditambahkan berbagai fitur pemrograman berorientasi objek. Hal ini dimulai sejak PHP 3 & 4, namun masih sangat sederhana.

Karena perkembangan web application (ASP.NET, JSP) yang support full OOP dikembangkan PHP5 yang support full OOP untuk memenuhi kebutuhan Pengembangan aplikasi yang besar (Enterprise Application). Dengan lahirnya PHP 5, fitur-fitur pemrograman berorientasi objek semakin mantap dan semakin cepat. Dengan PHP 5, script yang menggunakan konsep *object-oriented* akan lebih cepat dan lebih efisien.

C. Istilah-istilah dalam OOP

Class, Object, Property, dan Method

Membuat aplikasi dengan menggunakan konsep OOP bisa diibaratkan dalam dunia nyata dengan membuat atau memproduksi mobil. Yang mana sebelum membuat mobil, diperlukan adanya desain atau sketsa tampilan mobil, inilah yang disebut dengan **Class**. Sketsa tersebut menggambarkan tentang warna, ukuran, jumlah roda dari mobil, inilah yang disebut dengan **Property**. Sketsa tersebut juga menjelaskan tentang apa yang nantinya bisa dilakukan oleh mobil seperti berjalan dengan roda, mengerem, membunyikan klakson dsb.., apa yang bisa dilakukan oleh mobil inilah yang disebut dengan **Method**. Kemudian, sketsa tersebut dikirimkan ke pabrik untuk dibuatkan bentuk fisiknya yaitu mobil, mobil – mobil hasil produksi inilah yang disebut dengan **Object**.

C. Istilah-istilah dalam OOP

Salah satu keuntungan program didefinisikan dengan konsep OOP adalah adanya pengkapsulan (encapsulation) program dalam class dan object, dimana programmer yang menggunakan class tidak perlu mengetahui isi dan jalannya class secara detail, hanya perlu tahu bagaimana cara menggunakannya.

Sama halnya dengan sebuah mobil misalnya. Seorang pemilik mobil tentunya tidak perlu mengetahui bagian-bagian mobil secara menyeluruh. Dia tidak perlu mengetahui bagaimana mesin mobil melakukan pembakaran dan bagaimana mesin mobil bisa menggerakkan roda, dsb. Dia hanya perlu tahu bagaimana cara menjalankan mobil, bagaimana menghentikan mobil, dan fungsi mobil lainnya.

D. Implementasi OOP Dasar di PHP

❑ Mendefinisikan Class

Bentuk umum mendefinisikan sebuah class adalah sbb :

```
class NamaClass
{
    Deklarasikan dan definisikan properties di sini
    Definisikan semua method di sini
}
```

Example #1 Simple Class definition

```
<?php
class MobilClass
{
        // property declaration
        public $jumlah_roda;

        // method declaration
        public function DisplayJumlahRoda() {
                echo $this->jumlah_roda;
        }
}
?>
```

❑ Menambahkan Konstruktor

Konstruktor merupakan sebuah *method* khusus yang akan secara otomatis dijalankan saat *object* terbentuk. Konstruktor tidak harus ada, namun dalam satu *class* hanya boleh ada satu konstruktor. Method konstruktor biasanya berisi pemberian nilai *default* dari masing-masing *properties* (variabel).

Untuk membuat konstruktor, cukup dengan mendefinisikan suatu fungsi dengan nama **__construct().** Perhatikan contoh sebagai berikut :

```
class MobilClass
{
        public $jumlah_roda;
        public function __construct() {
                $this->jumlah_roda = 4;
        }
        public function DisplayJumlahRoda() {
                echo $this->jumlah_roda;
        }
}
```

D. Implementasi OOP Dasar di PHP

Encapsulation (One Class Many Object)

- **Menciptakan Object Baru**

Untuk menciptakan **object** baru dari **class** yang sudah dibuat adalah sbb:

```
class MobilClass
{
        public $jumlah_roda;
        // constructor
        public function __construct() {
                $this->jumlah_roda = 4;
        }
        public function DisplayJumlahRoda() {
                echo $this->jumlah_roda;
        }
}
$mobil = new MobilClass(); // Create objek mobil
$mobil->DisplayJumlahRoda(); // Menjalankan method object mobil 4
echo $mobil->jumlah_roda; // Mengakses variabel public 4
$mobil->jumlah_roda=6; // set variabel ke 6
$mobil->DisplayJumlahRoda(); // 6
$mobil2 = new MobilClass(); // Create objek mobil baru lagi
$mobil2->DisplayJumlahRoda(); // Menjalankan method object mobil 4
```

D. Implementasi OOP Dasar di PHP

❑ Menambahkan Destruktor

Kebalikan dengan konstruktor, destruktor merupakan sebuah *method* yang akan secara otomatis dijalankan saat *object* dihapus atau dinullkan atau eksekusi skrip PHP telah selesai.

```
class MobilClass
{
        public $jumlah_roda;
        public function __destruct() {
                 echo "Object telah dihapus";
        }
        public function DisplayJumlahRoda() {
                echo $this->jumlah_roda;
        }
}

$mobil = new MobilClass(); // Create objek mobil
```

❑ Static Properties / Method

• Jika Method / Property dideklarasikan secara statik maka Method / Property tersebut dapat langsung diakses tanpa harus membuat instan class

```
class MobilClass
{
        public static $jumlah_roda=4;
        public static function DisplayJumlahRoda() {
                echo self::$jumlah_roda; // akses static value
        }
}

MobilClass::DisplayJumlahRoda(); // Akses static method 4
echo MobilClass::$jumlah_roda; // Akses static variabel 4

$mobil = new MobilClass(); // Create objek mobil
$mobil::DisplayJumlahRoda(); // 4
Echo $mobil::$jumlah_roda; // 4
```

❑ Constant Property

• Pada dasarnya constant property sama dengan static property, hanya saja cara pendeklarasiannya beda.

```
class MobilClass
{
        const JUMLAH_RODA=4;
        public function DisplayJumlahRoda() {
                echo self::JUMLAH_RODA; // akses constant
        }
}

echo MobilClass::JUMLAH_RODA; // Akses constant 4

$mobil = new MobilClass(); // Create objek mobil
$mobil->DisplayJumlahRoda(); // 4
```

❑ Cloning Object

• Kita bisa mengkloning object sehingga dua object akan memiliki property dan method yang sama seterusnya.

```
<?php
class MyClass {
        public $var = 1;
}

$obj1 = new MyClass();
$obj2 = $obj1;
$obj2->var = 2;
echo $obj1->var;
?>
```

D. Implementasi OOP Dasar di PHP

Inheritance & Polymorphism

- **Inheritance = Pewarisan**
  - Parent mewariskan sifat ke child
- **Polymorphism = Banyak Bentuk**
  - Method dengan nama yang sama tapi beda parameternya (Overloading)
  - Method dengan nama sama parameter sama tapi berada dalam kelas anak (Overriding)

```
<?php
class Hewan {
        public function suara(){
                echo "tergantung hewannya apa????";
        }
}

class Kucing extends Hewan {
        public function suara(){
                echo "meoong";
        }
}
```

D. Implementasi OOP Dasar di PHP

```
class Ayam extends Hewan {
        public function suara($gender=1){
                if($gender==1) echo "Kukuruyuk";
                else echo "Petok-petok";
        }
}
$hewan1 = new Hewan();
$hewan1->suara(); // tergantung hewannya apa????

$hewan2 = new Kucing();
$hewan2->suara(); // meoong

$hewan3 = new Ayam();
$hewan3->suara(0); // Petok-petok
?>
```

D. Implementasi OOP Dasar di PHP

Visibility of Properties and Methods

Visibility Property and Methods maksudnya adalah skope hak akses dari property and method. Di PHP digunakan 3 keyword yaitu private, protected, public.

- Private artinya method atau property hanya bisa diakses di dalam class itu sendiri
- Protected artinya method atau property bisa diakses di dalam class itu sendiri dan class-class anaknya.
- Public artinya method atau property bisa diakses dimanapun,

D. Implementasi OOP Dasar di PHP

```
<?php
class MyClass
{
    public $public = 'Public';
    protected $protected = 'Protected';
    private $private = 'Private';

    function printHello()
    {
        echo $this->public;
        echo $this->protected;
        echo $this->private;
    }
}

$obj = new MyClass();
echo $obj->public; // Works
echo $obj->protected; // Fatal Error
echo $obj->private; // Fatal Error
$obj->printHello(); // Shows Public, Protected and Private
?>
```

D. Implementasi OOP Dasar di PHP

```
class MyClass2 extends MyClass
{
    // We can redeclare the public and protected method, but not private
    protected $protected = 'Protected2';

    function printHello()
    {
        echo $this->public;
        echo $this->protected;
        echo $this->private;
    }
}

$obj2 = new MyClass2();
echo $obj2->public; // Works
echo $obj2->private; // Undefined
echo $obj2->protected; // Fatal Error
$obj2->printHello(); // Shows Public, Protected2, Undefined
```