PHP-COMPLETE

Balya Rochmadi

December 17, 2018

Contents

Chapter 1

Variabel dan Variabel Superglobal

Chapter 2

Class dalam PHP

2.1 Definisi Kelas dan Objek

Seperti halnya bahasa pemrograman berbasis objek yang lain, dalam PHP juga dikenal memiliki kelas. Kelas adalah definisi dari sebuah objek. Objek sendiri adalah representasi dari sebuah kelas. Dalam kehidupan nyata Objek dapat berupa benda yang memiliki karakteristik khusus. Misalkan meja makan, memiliki karakteristik meja dan digunakan untuk makan.

Dalam PHP kita dapat menulis kelas sebagai berikut dengan keyword *class*

```
1    class Meja{
2    }
3    class Konsumen{
4    }
5
```

Kelas didefinisikan dengan menggunakan kata kunci (*keyword*) *class* diikuti oleh nama kelas. Sedangkan untuk implementasinya diperlukan variabel yang di*assign* oleh *instantinasi* kelas. Berikut ini adalah kelas yang di *instanitasikan*. Hasil dari *instantinasi* tersebut adalah sebuah objek.

```
1    $meja1= new Meja();
2    $konsumen1= new Konsumen();
3
```

Asignasi variabel tersebut terhadap *instansi* dari objek sama

dengan *asignasi* nilai primitif PHP seperti *integer,float* atau *string*. *Instantinasi* dari sebuah kelas dapat berlangsung berkali-kali dan menghasilkan objek-objek yang berbeda.

```
1     $meja2 = new Meja();
2     $konsumen2 = new Konsumen();
3
```

Untuk dapat melihat hasil dari proses *instantinasi* dan *asignasi* dari kelas tersebut kita dapat menggunakan fungsi php *var_dump*()

```
1     var_dump($meja1);
2     var_dump($meja2);
3     var_dump($konsumen1);
4     var_dump($konsumen2);
5
```

Seperti yang kita ketahui bersama jika *var_dump*() menghasilkan *array* dari objek tersebut. Karena dalam objek yang kita buat sebelumnya tidak memiliki metode, sehingga hasil dari *var_dump* adalah sebagai berikut :

```
Output:

class Meja#1 (0) {
}

class Meja#2 (0) {
}

class Konsumen#3 (0) {
}

class Konsumen#4 (0) {
}
```

Dalam output tersebut terdapat empat objek yang ditandai oleh "#". Belum terdapat metode ataupun properti yang didefinisikan dalam kelas-kelas tersebut sehingga *array length* dari objek tersebut masih nol.

2.2 Metode dan Properti Kelas

Misalkan dalam *Meja* dan *Konsumen* tersebut memiliki karakteristik tertentu, maka kita dapat menambahkan metode dan properti dalam definisi kelasnya. Contoh dibawah ini adalah penambahan properti pada kelas Meja dan Konsumen dengan nilai *default*;

```
1    class Meja{
2      public $jumlahkaki =  4;
3      public $warna      =  "coklat";
4      public $bentuk     =  "lingkaran";
5      public $diameter   =  7;
6
7    }
8
9    class Konsumen{
10     public $nama    = "Eddy";
11     public $Alamat  = "Jalan Pasar Kembang";
12     public $JenisKelamin   ="Laki-Laki";
13
14   }
15
16
```

Untuk dapat menggunakan data yang terdapat pada kelas-kelas diatas, diperlukan *asignasi* dan *instantinasi* pada variabel.Pada contoh diatas terdapat *keyword* visibilitas yaitu *public*. *Keyword* ini digunakan agar properti kelas dapat diakses oleh objek/kelas lain diluar kelas. Berikut ini adalah contoh yang dapat diterapkan.

```
1    $meja1=new Meja();
2    echo "Jumlah kaki meja adalah: ".$meja1->jumlahkaki.PHP_EOL; //
   mengambil nilai jumlahkaki
3    $meja1->jumlahkaki=10; //mengganti properti jumlahkaki
4    echo "Jumlah kaki meja yang baru: ".$meja1->jumlahkaki.PHP_EOL;
5
```

Jika ingin mengetahui isi dari kelas tersebut kita dapat menggunakan *var_dump*().

```
1      $meja1=new Meja();
2      var_dump($meja1);
3      echo "Jumlah kaki meja adalah: ".$meja1->jumlahkaki.PHP_EOL; //
   mengambil nilai jumlahkaki
4      $meja1->jumlahkaki=10; //mengganti properti jumlahkaki
```

```
5       var_dump($meja1);
6       echo "Jumlah kaki meja yang baru: ".$meja1->jumlahkaki.PHP_EOL;
7
```

Output dari variabel $meja1 adalah sebagai berikut:

```
:
class Meja#1 (4) {
public $jumlahkaki =>
int(4)
public $warna =>
string(6) "coklat"
public $bentuk =>
string(9) "lingkaran"
public $diameter =>
int(7)
}
Jumlah kaki meja adalah: 4


class Meja#1 (4) {
public $jumlahkaki =>
int(10)
public $warna =>
string(6) "coklat"
public $bentuk =>
string(9) "lingkaran"
public $diameter =>
int(7)
}
Jumlah kaki meja yang baru: 10
```

Perlu diingat bahwa objek-objek yang telah diinstantinasi masing-masing menempati ruang berbeda di virtual memori sehingga perubahan pada objek satu sama lain saling *mutually exclusive* /independen. Berikut ini adalah contoh dua objek dengan kelas yang sama

```
1     $konsumen1=new Konsumen();
2     $konsumen2=new Konsumen();
3
4     $konsumen1->nama="Robert Downey Jr";
5     $konsumen1->Alamat="NYC";
6     var_dump($konsumen1);
7
8     $konsumen2->nama="Peter Parker";
9     $konsumen2->Alamat="Queens";
10    var_dump($konsumen2);
```

```
11
12
13
```

Output dari kode tersebut adalah sebagai berikut:

```
class Konsumen#2 (3) {
public $nama =>
string(16) "Robert Downey Jr"
public $Alamat =>
string(3) "NYC"
public $JenisKelamin =>
string(9) "Laki-Laki"
}

class Konsumen#3 (3) {
public $nama =>
string(12) "Peter Parker"
public $Alamat =>
string(6) "Queens"
public $JenisKelamin =>
string(9) "Laki-Laki"
}
```

Dapat dilihat bahwa $konsumen1 dan $konsumen2 adalah 2 objek yang berbeda dan $konsumen1 menempat objek Konsumen#2 dan sama sekali tidak mempengaruhi $konsumen2 yang memnempati objek Konsumen#2. Objek juga akan berbeda jika berada pada skrip php yang berbeda walaupun memiliki nama yang sama, hal ini dikarenakan php melakukan *freeing*/pembebasan memori setelah eksekusi skrip. Maka dari itu untuk mempertahankan nilai baik yang berupa objek ataupun properti agar tetap dapat digunakan oleh skrip php yang lain pemanfaatan variabel *superglobals* seperti $_COOKIE, ataupun $_SESSION diperlukan.

Properti saja tidaklah cukup, karakteristik bawaan dari objek yang lain adalah metode objek. Metode adalah fungsi yang menjadi representasi dari aksi sebuah kelas. Berikut ini adalah karakterisitik dari metode:

1. merupakan fungsi;

2. dapat menerima nilai ataupun *void*
3. dapat menghasilkan return nilai yang lain
4. Untuk PHP 7.0 fungsi dapat memiliki