VERSI 2.0

JANUARI, 2020

PEMROGRAMAN WEB

PHP OOP – MODUL 5

TIM PENYUSUN: AMINUDIN, S.KOM., M.CS

-IHZA AHMAD ABROR AMRULLAH

PRESENTED BY: LAB. TEKNIK INFORMATIKA

UNIVERSITAS MUHAMMADIYAH MALANG

PEMROGRAMAN WEB

CAPAIAN PEMBELAJARAN MATA KULIAH

1. Mahasiswa mampu memahami konsep OOP pada PHP

SUB CAPAIAN PEMBELAJARAN MATA KULIAH

1. Mahasiswa mampu memahami konsep OOP pada PHP

KEBUTUHAN HARDWARE & SOFTWARE

Hardware dan Infrastruktur:

- Laptop/ PC
- Koneksi Internet

Software

- Text Editor (Atom/Sublime/Notepad++/atau lainnya)
- XAMPP (Web Server, MySql, PHP)

MATERI POKOK

1. PHP OOP

OOP merupakan metode pemrograman yang lebih berorientasi pada objek. Maksudnya dengan memecah alur program menjadi modul-modul sederhana yang disebut dengan objek. Setiap objek akan memiliki fungsi dan tugas tersendiri. Sehingga akan lebih sangat memudahkan kita di dalam membuat aplikasi. Sebenarnya OOP lebih di dukung pada pemrograman JAVA dan C++. tetapi di PHP sudah sangat di dukung pada versi PHP5.

a. Class

Class adalah ‘cetak biru’ atau ‘blueprint’ dari object. Class digunakan hanya untuk membuat kerangka dasar. Yang akan kita pakai nantinya adalah hasil cetakan dari class, yakni object. Di dalam pemrograman nantinya, contoh class seperti: koneksi_database dan profile_user.

Sebagai analogi, class disini ibarat mahasiswa. Mahasiswa memiliki ciri-ciri nama, NIM, Jenis kelamin, dan lainnya yang menyatakan benda itu adalah mahasiswa. Selain ciri-ciri, seorang mahasiswa juga melakukan tindakan, seperti : mahasiswa belajar atau mahasiswa berjalan.

Di dalam PHP, penulisan class diawali dengan keyword class, kemudian diikuti dengan nama dari class. Aturan penulisan nama class sama seperti aturan penulisan variabel dalam PHP, yakni diawali dengan huruf atau underscore untuk karakter pertama, kemudian boleh diikuti dengan huruf, underscore atau angka untuk karakter kedua dan selanjutnya. Isi dari class berada dalam tanda kurung kurawal.

Contoh penulisan Class pada PHP :

```php
<?php
    class mahasiswa {
        ...
    }
?>
```

b. Property / Atribut

Property (atau disebut juga dengan atribut) adalah data yang terdapat dalam sebuah class. Melanjutkan analogi tentang mahasiswa, property atau atribut dari mahasiswa bisa berupa nama, nim , jenis kelamin dan lainnya.

Jika anda sudah terbiasa dengan program PHP, property ini sebenarnya hanyalah variabel yang terletak di dalam class. Seluruh aturan dan tipe data yang biasa diinput ke dalam variabel, bisa juga diinput kedalam property. Aturan tata cara penamaan property sama dengan aturan penamaan variabel.

Contoh class dengan property :

```php
<?php
    class mahasiswa {
        var $nama;
        var $nim;
        var $jeniskelamin;
        ...
    }
?>
```

c. Method

Method adalah tindakan yang bisa dilakukan di dalam class. Jika menggunakan analogi class mahasiswa, maka contoh method : mahasiswa belajar, mahasiswa absen.

Method pada dasarnya adalah function yang berada di dalam class. Seluruh fungsi dan sifat function bisa diterapkan ke dalam method, seperti argumen/parameter, mengembalikan nilai (dengan keyword return), dan lain-lain.

Berikut contoh class dengan method :

```php
<?php
    class mahasiswa {
        function mahasiswa_belajar() {
            ...
        }
        function mahasiswa_absen() {
            ...
        }
        ...
    }
?>
```

d. Object

Object atau Objek adalah hasil cetak dari class, atau hasil 'konkrit' dari class. Jika menggunakan analogi class mahasiswa, maka objek dari class mahasiswa bisa berupa: mahasiswa_ihza, mahasiswa_ahmad dan lain-lain. Objek dari class mahasiswa akan memiliki seluruh ciri-ciri mahasiswa, yaitu property dan method-nya.

Proses 'mencetak' objek dari class ini disebut dengan 'instansiasi'. Pada PHP, proses instansiasi dilakukan dengan menggunakan keyword 'new'. Hasil cetakan class akan disimpan dalam variabel untuk selanjutnya digunakan dalam proses program.

Berikut adalah contoh pembuatan object dari class mahasiswa:

```php
<?php
    class mahasiswa {
        ...
    }

    $mahasiswa_ihza = new mahasiswa();
    $mahasiswa_ahmad = new mahasiswa();
?>
```

Dari contoh di atas $mahasiswa_ihza dan $mahasiswa_ahmad adalah objek dari class mahasiswa. Kedua objek ini akan memiliki seluruh property dan method yang telah dirancang dari class mahasiswa.

e. Abstract Class dan Absctract Method

Abstract Class adalah sebuah class yang tidak bisa di-instansiasi (tidak bisa dibuat menjadi objek) dan berperan sebagai 'kerangka dasar' bagi class turunannya. Di dalam abstract class umumnya akan memiliki abstract method.

Abstract Method adalah sebuah 'method dasar' yang harus diimplementasikan ulang di dalam class anak (child class). Abstract method ditulis tanpa isi dari method, melainkan hanya 'signature'-nya saja. Signature dari sebuah method adalah bagian method yang terdiri dari nama method dan parameternya (jika ada).

Abstract class digunakan di dalam inheritance (pewarisan class) untuk 'memaksakan' implementasi method yang sama bagi seluruh class yang diturunkan dari abstract class. Abstract class digunakan untuk membuat struktur logika penurunan di dalam pemrograman objek.

- Membuat Abstract Class

  Membuat class Manusia sebagai abstract class, maka berikut adalah cara penulisannya di dalam PHP:

```php
<?php
    abstract class manusia {
        ...
     }

?>
```

Untuk membuat abstract class di dalam PHP, kita tinggal menambahkan keyword abstract sebelum nama class. Sebuah abstract class bisa memiliki property dan method biasa layaknya sebuah class 'normal', namun juga bisa memiliki abstract method.

- Membuat Abstract Method

  Jika sebuah method dinyatakan sebagai abstract method, maka kita tidak perlu membuat isi methodnya, tetapi hanya signature dari method tersebut. Jika sebuah method dinyatakan sebagai abstract method, isi dari method tersebut akan dibuat dalam class turunan. Abstract method harus berada di dalam abstract class. Signature terdiri dari nama method dan parameternya (jika ada) seperti contoh berikut:

```
abstract public function lihat_nama();
```

  Contoh penulisan abstract method di dalam abstract class :

```
<?php
    abstract class manusia {
        abstract public function lihat_nama();
     }
?>
```

- Abstract Class Bisa Memiliki Property dan Method biasa

  Jika sebuah class dinyatakan sebagai abstract class, class tersebut juga bisa memiliki property dan method 'normal'. Namun kita hanya bisa mengakses property dan method ini dari class turunan, karena abstract class tidak bisa diinstansiasi.

```
<?php
    abstract class mahasiswa {
        abstract public function lihat_nama();
        public function berjalan(){
            echo "Manusia berjalan";
        }
     }
?>
```

- Class Turunan Harus Mengimplementasikan Abstract Method

  Jika sebuah class diturunkan dari abstract class, maka class tersebut harus membuat ulang seluruh abstract method yang terdapat dalam abstract class, dan juga harus sesuai dengan signature-nya.

```
<?php
    abstract class manusia {
        abstract public function lihat_nama();
        public function berjalan(){
            echo "Manusia berjalan";
        }
     }

     class mahasiswa extends manusia {
```

```
        public function lihat_nama(){
            ...
        }
    }

    $mhs = new mahasiswa();
?>
```

Dalam kode diatas, method lihat_nama() telah kita implementasikan di dalam class manusia. Fitur inilah yang menjadi fungsi dari abstract method, yakni 'memaksa' setiap class turunan untuk memiliki method lihat_nama().

f. Objek interface

Secara sederhana, Object Interface adalah sebuah 'kontrak' atau perjanjian implementasi method. Bagi class yang menggunakan object interface, class tersebut harus mengimplementasikan ulang seluruh method yang ada di dalam interface. Sama seperti abstract class, interface juga hanya berisi signature dari method, yakni hanya nama method dan parameternya saja (jika ada). Isi dari method akan dibuat ulang di dalam class yang menggunakan interface.

Jika kita menganggap abstract class sebagai 'kerangka' atau 'blue print' dari class-class lain, maka interface adalah implementasi method yang harus 'tersedia' dalam sebuah objek. Interface tidak bisa disebut sebagai 'kerangka' class.

Untuk membuat Interface di dalam PHP, penulisnya mirip seperti membuat class, tetapi menggunakan keyword interface, seperti contoh berikut:

```
<?php
    interface manusia {
        ...
    }
?>
```

Sesuai dengan tujuannya untuk membuat interface/antar muka bagi class, method di dalam perancangan interface harus memiliki hak akses public, atau tidak ditulis sama sekali (dimana PHP akan menganggapnya sebagai public).

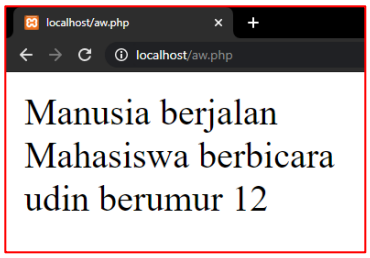
g. Inheritance (Pewarisan)

Inheritance atau Pewarisan/Penurunan adalah konsep pemrograman dimana sebuah class dapat 'menurunkan' property dan method yang dimilikinya kepada class lain. Konsep inheritance digunakan untuk memanfaatkan fitur 'code reuse' untuk menghindari duplikasi kode program.

Di dalam PHP, inheritance / penurunan dari sebuah class kepada class lain menggunakan kata kunci: 'extends', dengan penulisan dasar sebagai berikut:

```
<?php
    class manusia {
        var $nama;
        var $umur;
```

```php
        function berjalan(){
            echo "Manusia berjalan<br>";
        }
    }

    class mahasiswa extends manusia {
        function tampil_data(){
            echo $this->nama." berumur ".$this->umur;
        }
        function berbicara(){
            echo "Mahasiswa berbicara<br>";
        }
    }

    $mhs = new mahasiswa();

    $mhs->nama = "udin";
    $mhs->umur = 12;
    $mhs->berjalan();
    $mhs->berbicara();
    $mhs->tampil_data();
?>
```



Dalam contoh kode diatas, dibuat class manusia dengan beberapa property dan sebuah method. Property class komputer belum berisi nilai apa-apa. Dibawah class manusia, dibuat class mahasiswa extends class manusia. Disini class mahasiswa menurunkan class manusia. Di dalam class mahasiswa, kita bisa mengakses seluruh property dan method apapun dari class manusia selama memiliki hak akses public atau protected. Untuk membuktikan hal tersebut, buat objek $mhs_baru dari class mahasiswa. Perhatikan bahwa kita bisa mengakses property $nama, dan $umur yang semuanya adalah milik class manusia, bukan class mahasiswa. Method berjalan() juga sukses diakses dari objek $mhs. Inilah yang dimaksud dengan inheritance/penurunan class dalam OOP.

h. PHP Trait

Trait adalah suatu mekanisme dimana suatu class diizinkan untuk menggunakan kembali kode program (code reuse) yang berasal dari hirarki yang berbeda.

Perhatikan contoh di bawah ini, buat tiga buah class:

```php
<?php
    class ApaKabar
    {
        public function apaKabar() {
            return "Apa kabar?";
        }
    }

    class SelamatPagi
    {
        public function selamatPagi() {
            return "Selamat pagi?";
        }
    }

    class Pesan
    {
    //
    }
?>
```

Jika kita menghendaki class Pesan bisa menggunakan method apaKabar() maka kita tinggal melakukan extends ke class ApaKabar. Bagaimana jika kita ingin class Pesan dapat menggunakan method apaKabar() dan selamatPagi()? class Pesan meng-extends class ApaKabar dan class SelamatPagi? tentu ini tidak diperbolehkan oleh PHP, PHP hanya memperbolehkan satu parent class . Untuk itulah adanya Trait.

1. Membuat Trait

   Sama seperti pembuatan class, hanya saja class diganti dengan trait.

```php
trait ApaKabar
    {
        public function apaKabar() {
            echo "Apa kabar?";
        }
    }
```

2. Menggunakan trait

   Cara memakai trait yaitu dengan menggunakan keyword use dalam sebuah class dan diiringi nama trait.

```php
    trait ApaKabar
    {
        public function apaKabar() {
            echo "Apa kabar?";
        }
    }
```

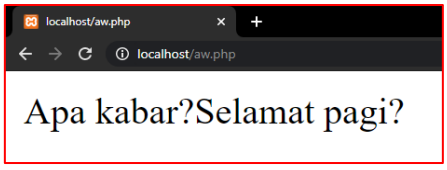
```
    class Pesan
    {
        use ApaKabar;
    }
```

3. Multiple Trait

   Satu class boleh memakai lebih dari satu trait, cara memanggilnya dipisahkan oleh koma.

```
<?php
    trait ApaKabar
    {
        public function apaKabar() {
            echo "Apa kabar?";
        }
    }

    trait SelamatPagi
    {
        public function selamatPagi() {
            echo "Selamat pagi?";
        }
    }

    class Pesan
    {
        use ApaKabar,SelamatPagi;
    }

    $psn = new Pesan();

    $psn->apaKabar();
    $psn->selamatPagi();
?>
```

Hasil :

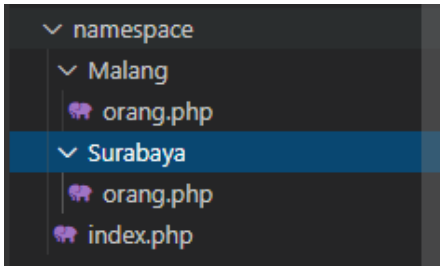


i. PHP Namespace

Dengan namespace, bisa mengorganisis kode dengan lebih rapi dan terstruktur, sehingga lebih mudah di-maintenance. Cara kerja namespace mirip dengan cara kerja folder dan file. Dalam satu folder tidak bisa membuat 2 buah file dengan nama yang sama. Tetapi hal tersebut bisa dilakukan jika foldernya berbeda.

- Membuat namespace

  Untuk mempelajari bagaimana membuat namespace, mari kita praktekkan dengan contoh sederhana. Pada folder htdocs buat folder baru yang berisi 2 folder .Buat beberapa file php dengan struktur seperti berikut:



Dalam folder namespace berisi folder malang, Surabaya dan index.php. Pada folder malang dan Surabaya berisi orang.php.

File Malang/orang.php :

```php
<?php
    class orang {
        public function bicara() {
            echo "HALO";
        }
    }
?>
```

File Surabaya/orang.php:

```php
<?php
    class orang {
        public function bicara() {
            echo "HAI";
        }
    }
?>
```

File index.php :

```php
<?php

    include 'Malang/orang.php';
    include 'Surabaya/orang.php';
```

```php
    //toton, orang Malang
    $tono = new orang;
    $tono->bicara();

    echo '<br>';

    //yusuf, orang Malang
    $yusuf = new orang;
    $yusuf->bicara();
?>
```

Jika menjalanakan file index.php maka akan mendapati error karena tidak boleh membuat class dengan nama yang sama.

> **Fatal error**: Cannot declare class orang, because the name is already in use in **C:\xampp\htdocs\namespace\Surabaya\orang.php** on line **2**

Untuk mengatasi error di atas, mari tambahkan namespace ke masing-masing class.

File Malang/orang.php

```php
<?php
    namespace Malang;
    class orang {
        public function bicara() {
            echo "HALO";
        }
    }
?>
```

File Surabaya/orang.php

```php
<?php
    namespace Surabaya;
    class orang {
        public function bicara() {
            echo "HAI";
        }
    }
?>
```

File index.php :

```php
<?php

    include 'Malang/orang.php';
    include 'Surabaya/orang.php';

    //toton, orang Malang
    $tono = new \Malang\orang;
    $tono->bicara();

    echo '<br>';

    //yusuf, orang Malang
    $yusuf = new \Surabaya\orang;
    $yusuf->bicara();
?>
```

Untuk mengakses semua resource dalam namespace tersebut Anda harus menulisnya dengan full path.

Hasilnya :

j. Magic Methods

Dalam pemrograman class dengan PHP kita akan menjumpai method yang diawali dengan double underscore. Method ini digunakan untuk memberikan response pada suatu event atau kejadian-kejadian tertentu.

Sesuai dengan dokumentasi PHP pada http://php.net/manual/en/language.oop5.magic.php kita mengenal method-method magic seperti berikut ini:

1. __construct()
   Method ini akan diakses pada saat pembuatan object.

2. __destruct()
   Method ini akan diakses saat object dihapus.

3. __call()
   Method ini akan diakses saat object menjalankan method yang invisible atau tidak didefinisikan

4. __callStatic()
   Method ini akan diakses jika kita menjalankan suatu static method yang invisible atau tidak didefinisikan

5. __get()
   Method ini akan diakses jika program ingin mengambil nilai dari property yang invisible.

6. __set()
   Method ini akan diakses jika program ingin memberikan nilai dari suatu property yang inivisible.

7. __isset()
   Method ini akan diakses jika program menjalankan isset() atau empty() pada property yang tidak bisa diakses.

8. __unset()
   Method ini akan diakses jika program menjalankan unset() pada property yang tidak bisa diakses.

9. __sleep()
   Berhubungan dengan serialisasi object.

10. __wakeup()
    Berhubungan dengan unserialisasi object.

11. __toString()
    Method ini akan diakses jika kita memperlakukan class sebagai string.

12. __invoke()
    Method ini akan diakses jika program memperlakukan object sebagai function.

13. __set_state()
    Method yang dipakai saat mengeksport class dengan function var_export(), yaitu function yang mengeksport properti dalam method kedalam array ('property' => value, ...).
14. __clone()
    Method ini berhubungan dengan cloning object. Saat selesai proses cloning object, method ini akan dijalankan.

15. __debugInfo()
    Method ini akan diakses jika program menjalankan var_dump() pada suatu object.

LEMBAR KERJA

TUGAS 1

Dengan studi kasus bebas, usahakan berbeda dengan teman anda.

1. Buatlah sebuah parent atau super-class.
2. Buatlah minimal 3 atribut atau property.
3. Buatlah minimal 2 method atau fungsi abstract dan 1 fungsi biasa untuk menampilkan nilai – nilai dari atribut di atas.
4. Buatlah minimal 2 buah child atau sub-class yang meng-extends super-class yang sudah dibuat.
5. Masing – masing sub-class memiliki constructor.
6. Isi dari constructor adalah menginisialisasi atau memberikan nilai pada atribut - atribut yang di buat pada point 2.

RUBRIK PENILAIAN

1. Mengerjakan dan dapat menjelaskan tugas 1 (Point Max 100)