Pemrograman berbasis web 2

Graphical user interface

Description automatically generated Konsep OOP Dalam PHP

**DESKRIPSI MATAKULIAH**

**Pada mata kuliah ini mahasiswa belajar mendesain dan membuat web statis maupun dinamis dengan menerapkan Framework yang ada, seperti Bootstrap dan Codeigniter**

**CAPAIAN PEMBELAJARAN**

**Mahasiswa mampu menjelaskan dan membuat program sederhana menggunakan konsep OOP dalam PHP dengan baik dan benar.**

**DOSEN PENGAMPU**

**Wachid Darmawan, M.Kom**

**SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER**

**(STMIK) WIDYA PRATAMA**

konsep oop dalam php

**CAPAIAN PEMBELAJARAN**

**Mahasiswa mampu menjelaskan dan membuat program sederhana menggunakan konsep OOP dalam PHP dengan baik dan benar.**

**MATERI PEMBELAJARAN**

* **Kontrak Kuliah**
* **Pengertian Object Oriented Programming (OOP)**
* **Membuat Class**
* **Objek**
* **Method Constructor Dan Destructor**
* **Melindungi Method Dan Property Dengan Enkapsulasi (Encapsulation)**

**MATERI**

Programmer pemula ketika akan beralih dari gaya pemrogramman procedural ke dalam OOP masih merasa kesulitan Karena mereka sudah terbiasa bekerja dengan memahami program yang mereka buat secara algoritmik. Karena gaya pemrogramman prosedur yang paling mudah dipahami secara algoritmik. Dengan memahami konsep OOP diharapkan kedepan saat membuat program lebih mudah dengan cara mengikuti konsep dari OOP tersebut.

## **Kontrak Kuliah**

Matakuliah Pemrograman Berbasis Web II merupakan salah satu matakuliah yang ditawarkan pada program studi Teknik Informatika dengan bobot 4 SKS. Pada matakuliah ini mahasiswa mempelajari dan mampu menjelaskan: Konsep Object Oriented Programming pada PHP, Pengenalan framework, Konsep MVC, CRUID, File upload, Pagination dan searching, validation

Helper dan Library.

Kontrak Kuliah/Belajar merupakan tindak lanjut apa yang akan menjadi nilai akhir dari pembelajaran yang sudah kita laksanakan selama 1 semester kedepan. Kontrak Kuliah tidak bisa dibuat secara sepihak oleh Dosen atau salah satu Mahasiswa saja, akan tetapi dibuat dan disahkan secara musyawarah dan disepakati oleh semua mahasiswa. Kontrak Kuliah yang kami (Dosen) tawarkan adalah:

* **Presensi : 10%**
* **Tugas & Latihan : 40%**
* **UTS : 25%**
* **UAS : 25%**

## **Pengertian Object Oriented Programming (OOP)**

Pengertian Object Oriented Programming (OOP) adalah suatu metode pemrograman yang berorientasi kepada objek. Tujuan dari OOP diciptakan adalah untuk mempermudah pengembangan program dengan cara mengikuti model yang telah ada di kehidupan sehari-hari. Jadi setiap bagian dari suatu permasalahan adalah objek, nah objek itu sendiri merupakan gabungan dari beberapa objek yang lebih kecil lagi.

Untuk mempermudah ilustrasi dalam belajar konsep OOP, saya akan ambil contoh Pesawat, Pesawat adalah sebuah objek. Pesawat itu sendiri terbentuk dari beberapa objek yang lebih kecil lagi seperti mesin, roda, baling-baling, kursi, dll. Pesawat sebagai objek yang terbentuk dari objek-objek yang lebih kecil saling berhubungan, berinteraksi, berkomunikasi dan saling mengirim pesan kepada objek-objek yang lainnya. Begitu juga dengan program, sebuah objek yang besar dibentuk dari beberapa objek yang lebih kecil, objek-objek itu saling berkomunikasi, dan saling berkirim pesan kepada objek yang lain.

## **Membuat Class**

Class adalah sebuah konsep OOP yang digunakan untuk mengencapsulasi /membungkus data dan abstraksi prosedural yang diperlukan dalam menggambarkan isi dan tingkah laku berbagai entitas. Kelas juga merupakan deskripsi tergeneralisir (misal templet, pola, cetak biru) yang menggambarkan kumpulan objek yang sama, untuk membuat class dalam pemograman PHP bisa digunakan dengan nenambahkan keyword class lalu di ikuti oleh nama class yang akan dibuat seperti contoh berikut ini :

<?php

    class person {

    }

?>

Jika disimpulkan class bisa juga dapat diartikan sebagai berikut:

1. Class adalah penampung sekumpulan elemen data (variable) dan kode program (function) yang digunakan oleh pengolahan datanya. Sifat ini disebut sebagai enkapsulasi (Encapsulation) .
2. Class disusun berdasarkan karakteristik sebuah objek benda. Sifat ini disebut sebagai Abstraksi (Abstraction)
3. Class dapat disusun secara hierarki sehingga suatu class dapat mewariskan beberapa atau semua karakteristiknya ke class lain (class anak/child class). Sifat ini disebut sebagai Pewarisan (Inheritance)

Aturan pembuatan class, diantaranya:

1. Definisi suatu class dimulai dengan keyword Class, diikuti dengan nama classnya.
2. Isi suatu class diapit dengan kurung kurawal ({ }).
3. Umumnya isi class diawali dengan menuliskan properti-propertinya dan diikuti dengan method-methodnya.
4. **Property Pada Sebuah Class**

Berdasarkan sifat abstraksi, suatu class menyimpan karakteristik suatu benda dalam suatu variable. Variabel ini sering disebut sebagai property dari Class, sedangkan Method biasa dalam bentuk fungsi. Pada Class person kita akan menambahkan sebuah properti baru dengan nama name, pada konsep OOP dalam bahasa pemograman PHP untuk menyatakan sebuah properti bisa dengan menggunakan keyword var lalu di ikuti dengan nama propertinya seperti dibawah ini :

<?php

    class person {

        var $name;

    }

?>

Contoh Property:

1. Class Orang mempunyai properti seperti: Nama, Warna Rambut, Tinggi, Berat, Tahun Lahir dan lain-lain.
2. Class KoneksiDB mempunyai properti seperti: Alamat Server, User Name, Password dan Nama Database
3. **Method Dalam Konsep OOP PHP**

Method adalah suatu operasi berupa fungsi-fungsi yang dapat dikerjakan oleh suatu object. Method didefinisikan pada class akan tetapi dipanggil melalui object. untuk membuat method bisa dengan dimulai dengan keyword function lalu di ikuti oleh nama function nya. berikut ini adalah contoh method Setter dan Getter untuk memberikan value kepada properti dan mengakses properti tersebut :

<?php

    class person {

        var $name;

        function set\_name($new\_name) {

            $this->name = $new\_name;

        }

        function get\_name() {

            return $this->name;

        }

    }

?>

Contoh Method:

1. Class Orang mempunyai method seperti: Makan, Minum, Tidur, Ucap Salam, dan lain-lain
2. Class KoneksiDB mempunyai method seperti: Koneksi, EksekusiQuery, AmbilData dan lain-lain

## **Objek**

Objek merupakan instance/insansiasi/membuat objek dari suatu Class. Sebuah class tidak dapat digunakan tanpa diinstansiasikan dulu (kecuali untuk Static Class). Objek sendiri sebenarnya sebuah variabel.

1. **Instance Object / Membuat Object**

Object atau Objek adalah hasil cetak dari class, atau hasil ‘konkrit’ dari class. Jika menggunakan analogi class person di atas maka object dari person bisa berupa anton, sari dan lain nya. sekarang kita akan membuat sebuah file baru dengan nama index.php yang akan kita gunakan sebagai file utama. setelah itu kita juga akan memanggil file class\_lib.php tadi agar bisa menggunakan class yang sudah kita buat sebelumnya :

<?php include("class\_lib.php"); ?>

<?php

    $wachid = new person();

    $ahmad = new person;

    $wachid->set\_name("Wachid Darmawan");

    $ahmad->set\_name("Muhammad Darmawan");

    echo "Wachid's full name: " . $wachid->get\_name(). "<br>";

    echo "Ahmad's full name: " . $ahmad->get\_name();

?>

Hasilnya sebagai berikut:

Graphical user interface, text, application, chat or text message

Description automatically generated

Untuk mengakses sebuah properti sebenarnya tidak selalu harus melalui method, bisa saja anda memanggil properti itu langsung dari luar class dengan syarat bahwa properti itu bersifat public (akan kita bahas lebih lengkap pada pembahasan visibility setelah ini) seperti dibawah ini :

<?php include("class\_lib.php"); ?>

<?php

    $wachid = new person();

    $ahmad = new person;

    $wachid->set\_name("Wachid Darmawan");

    $ahmad->set\_name("Muhammad Darmawan");

    // directly accessing properties in a class is a no-no.

    echo "Wachid's full name: " . $wachid->name. "<br>";

    echo "Ahmad's full name: " . $ahmad->name;

?>

Hasilnya sebagai berikut:

Graphical user interface, text, application, chat or text message

Description automatically generated

Contoh kasus lain terkait OOP dalam PHP, berikut source code program sederhana mengguanakn OOP dengan PHP yang mana cara mensetting nilai propertisnya langsung pada objectnya, sebagai berikut:

<?php

//membuat class kendaraan

class Kendaraan

{

    //membuat propertisenya

    var $merk;

    var $harga;

    var $jmlroda;

    //membuat function kondisi mobil

    function KondisiHarga()

    {

        if ($this->harga > 10000000) {

            return "Mahal";

        } else {

            return "Murah";

        }

    }

}

//membuat object agar program bisa ditampilkan

$objMobil = new Kendaraan();  //object dari kelas kendaraan

//mensetting nilai propertise di dalam object

$objMobil->merk = ("Avanza");

$objMobil->harga = (15000000);

$objMobil->jmlroda = (4);

//perintah untuk menampilkan program yang telah disetting

echo "Merek Mobil adalah" . $objMobil->merk . "<br>";

echo "Harga Mobil = " . $objMobil->harga . "<br>";

echo "Jumlah Rodanya = " . $objMobil->jmlroda . "<br>";

//menampilkan function kondisi yang telah dibuat tadi

echo "Status Harga =" . $objMobil->KondisiHarga();

?>

Hasilnya sebagai berikut:

Graphical user interface, text, application, chat or text message

Description automatically generated

## **Method Constructor Dan Destructor**

Constructor adalah fungsi khusus / fungsi unik yang akan di panggil secara otomatis pada saat object dibuat. Secara umum, pada saat pembuatan object, class harus meng-inisialisasikan variablenya untuk menghindari error yang tidak diinginkan sedangkan Destructor adalah method khusus yang dijalankan secara otomatis pada saat sebuah objek dihapus. Di dalam PHP, seluruh objek secara otomatis dihapus ketika halaman PHP dimana objek itu berada selesai diproses. berikut ini adalah contoh nya :

<?php

    class person {

        var $name;

        function \_\_construct($persons\_name) {

            echo "<p>initialize class</p>";

        }

        function set\_name($new\_name) {

             $this->name = $new\_name;

        }

         function get\_name() {

             return $this->name;

         }

                function \_\_destructor(){

                         echo "<p>end class</p>";

                }

    }

?>

Contoh kasus terkait Method Constructor dan Destructor, sebagai berikut:

<?php

//membuat class kendaraan

class kendaraan

{

    //membuat properti

    private $merk;

    private $harga;

    //membuat method konstruktor, dimana mengambil parameter merk

    dan harga dari class kendaraan

    function \_\_construct($merk, $harga)

    {

        $this->merk = $merk;

        $this->harga = $harga;

    }

    //fungsi getter

    //membuat method bacamerk

    function BacaMerk()

    {

        return $this->merk;

    }

    //membuat method bacaharga

    function BacaHarga()

    {

        return $this->harga;

    }

    function \_\_destruct()

    {

        echo "Merk dan Harga telah dihapus";

    }

}

//membuat objec mobil dari kelas kendaraan, yang berisi nilai dari  parameter konstructor

$mobil = new kendaraan("Avanza", 100000000);

//perintah untuk menampilkan pada browser dimana mengambil dari  method BacaMerk dan BacaHarga

echo "Merk Kendaraan =" . $mobil->BacaMerk() . "<br>";

echo "Harga Mobil = " . $mobil->BacaHarga() . "<br>";

?>

Hasilnya sebagai berikut:

Graphical user interface, text, application, chat or text message

Description automatically generated

## **Melindungi Method Dan Property Dengan Enkapsulasi (Encapsulation)**

Dengan enkapsulasi, kita bisa membuat pembatasan akses kepada property dan method, sehingga hanya property dan method tertentu saja yang bisa diakses dari luar class. Enkapsulasi juga dikenal dengan istilah ‘information hiding’. ada 3 jenis enkapsulasi yang bisa anda manfaatkan yaitu Public, Private Protected.

1. **Jenis Enkapsulasi Public**

Jika anda menggunakan jenis enkapsulasi public pada properti atau method dalam sebuah class, itu artinya method dan propertinya bisa di akses secara bebas baik dari dalam class, dari luar class bahkan dari class turunan nya sekalipun. berikut ini adalah contoh nya :

<?php

    class person {

        public $name;

        function set\_name($new\_name) {

            $this->name = $new\_name;

        }

        function get\_name() {

            return $this->name;

        }

    }

?>

<?php

$person1 = new Person();

// properti bisa di akses secara langsung

echo "Hai ".$person1->name='Wachid Dw';

echo "<hr>";

// method bisa di akses secara langsung

echo $person1->get\_name();

?>

Seperti yang anda lihat pada script di atas bahwa kita bisa mengakses method atau properti secara langsung jika menggunakan keyword public, seperti pada gambar ini:

Graphical user interface, text, application, email

Description automatically generated

1. **Jenis Enkapsulasi Private**

Lain halnya jika anda menggunakan jenis enkapsulasi private, itu artinya method dan propertinya hanya bisa di akses dari dalam class itu sendiri dan dari class turunan nya, berikut ini adalah contoh nya:

<?php

    class person {

        private $name;

        function set\_name($new\_name) {

            $this->name = $new\_name;

        }

        function get\_name() {

            return $this->name;

        }

    }

?>

<?php

$person1 = new Person();

// properti bisa di akses secara langsung

echo "Hai ".$person1->name='Wachid Dw';

echo "<hr>";

// method tidak bisa di akses secara langsung

echo $person1->get\_name();

?>

Jika anda mengganti jenis enkapsulasinya menjadi private maka akan muncul error seperti ini jika anda melakukan akses properti dari luar class.

Graphical user interface, text, application, email

Description automatically generated

1. **Jenis Enkapsulasi Protected**

Lalu yang terakhir adalah jenis enkapsulasi protected, jenis enkapsulasi ini biasanya digunakan untuk melindungsi informasi yang bersifat rahasia, dimana jika anda menggunakan protected pada method atau properti maka anda hanya bisa mengakses properti tersebut hanya dari dalam class tersebut, berikut contoh script nya :

<?php

    class person {

        protected $name;

        function set\_name($new\_name) {

            $this->name = $new\_name;

        }

        function get\_name() {

            return $this->name;

        }

    }

?>

<?php

$person1 = new Person();

// set value dari properti name

$person1->set\_name('Muhammad Darmawan');

// akses value dari properti name

echo $person1->get\_name();

// properti tidak bisa di akses secara langsung, karena muncul  error

echo "Hai ".$person1->name='Wachid Dw';

echo "<hr>";

?>

Seperti yang sudah di jelaskan di atas bahwa perbedaan utama antara private dan protected hanyalah jika protected tidak bisa diakses kecuali dari dalam class itu sendiri. jika anda mengakses nya dari luar class maka akan muncul error seperti ini :

Graphical user interface, text, application, email

Description automatically generated

**RANGKUMAN**

* Materi lain terkait OOP dalam PHP bisa baca di [w3schools.com/php/php\_oop\_what\_is.asp](https://www.w3schools.com/php/php_oop_what_is.asp).
* Dari materi yang ada, silakan jenengan buat rangkuma/ringkasannya dengan cara **di tulis tangan** pada buku jenengan masing-masing (minimal 1 lembar).
* Source code yang ada silakan di uji coba sebagai bahan belajar dan pemahaman di rumah

**TES FORMATIF**

1. Dari contoh kasus yang ada, silakan buat objek dari kelas kendaraan dengan ketentuan sbb:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Objek | Merek | Jumlahroda | Harga | Warna | bahanbakar |
| $kendaraan2 | Panther | 4 | 200.000.000 | Biru Metalik | Solar |
| $kendaraan3 | Virza | 2 | 23.000.000 | Hitam Merah | Pertamax |
| $kendaraan4 | Inova | 4 | 450.000.000 | Putih | Pertalite |

Tampilkan hasil instansiasi dari soal adalah sebagai berikut:

1. Kendaraan **Panther** memiliki roda sebanyak **4** dengan harga **200000000** dan berbahan bakar **Solar**
2. Kendaraan **Virza** memiliki roda sebanyak **2** dengan harga **23000000** dan berbahan bakar **Pertamax**
3. Kendaraan **Inova** memiliki roda sebanyak **4** dengan harga **4500000000** dan berbahan bakar **Pertalite**
4. Simpan semua source code (materi dan latihan) yang ada dalam **FOLDER** dengan nama folder: **NIM Lengkap** (tanpa titik), contoh: **202400789**, untuk nama file bisa menyesuaiakan sendiri-sendiri atau menyesuaikan dengan nama class.
5. Dari folder yang sudah dibuat dijadikan ZIP (**202400789.zip**) kemudian kumpulkan ke classroom.

**DAFTAR PUSTAKA**

Akbar, Nuris. 2017. *Belajar Konsep OOP Dalam Pemrograman PHP*. Bandung: Belajar PHP.

Dhika, Bernan. 2019. Tutorial PHP OOP. Jakarta: Bernandhika.

Heryandi, Andri. *Modul Aplikasi Teknologi Online*. Bandung: IF-UNIKOM