ASTRA polytechnic
member of ASTRA

Kampus Merdeka
INDONESIA JAYA

satu Indonesia
Semangat Astra Terpadu Untuk Indonesia

OBJECT ORIENTED PROGRAMMING (OOP)

Oleh :

Dwi Diana Wazaumi

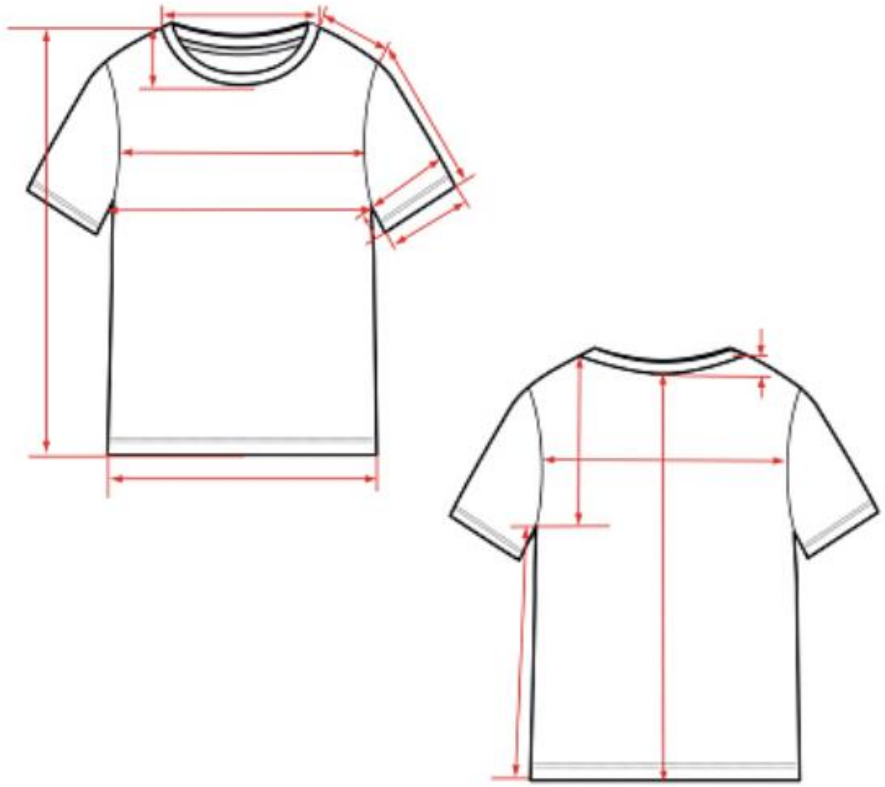
Eko Abdul Goffar

Overview

- Pengertian OOP
- Class
- Acces Control Class
- Contoh Konsep OOP PHP
- Interface

PENGERTIAN OOP

- OOP adalah salah satu paradigma dalam cara membuat program dengan memecah program menjadi modul-modul sederhana yang disebut dengan objek.
- Objek adalah entitas yang memiliki atribut (data) dan metode (fungsi)
- Selain OOP, juga terdapat paradigma lain dalam membuat program yaitu procedural programming.



Blueprint Desain Baju

CLASS

Baju

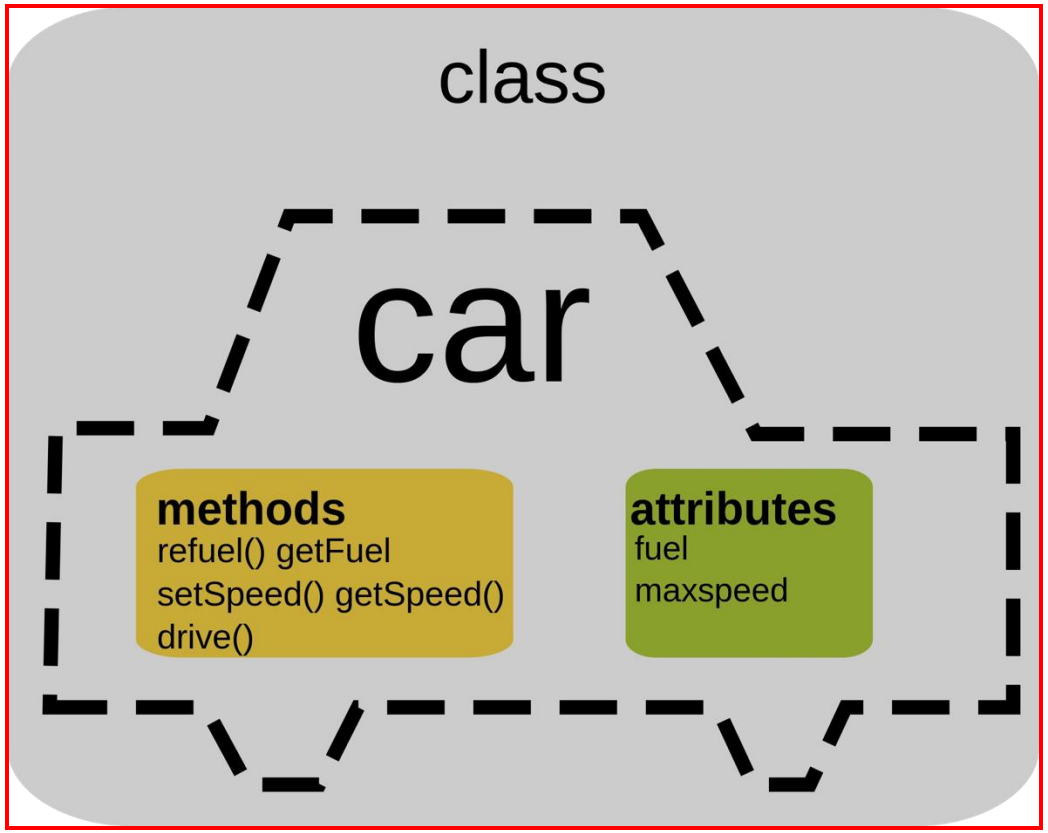
OBJECT

Baju berbagai warna

OBJECT 1, OBJECT 2, DST.

CLASS

- Class adalah blueprint dari object.
- Class dibuat sebagai kerangka dasar yang akan dipakai sebagai hasil dari cetakan class yaitu object.
- Dalam PHP penulisan class diawali dengan *keyword* **class**.
- Misalnya, kita bisa memiliki class ”Baju" dengan syntax sbb:

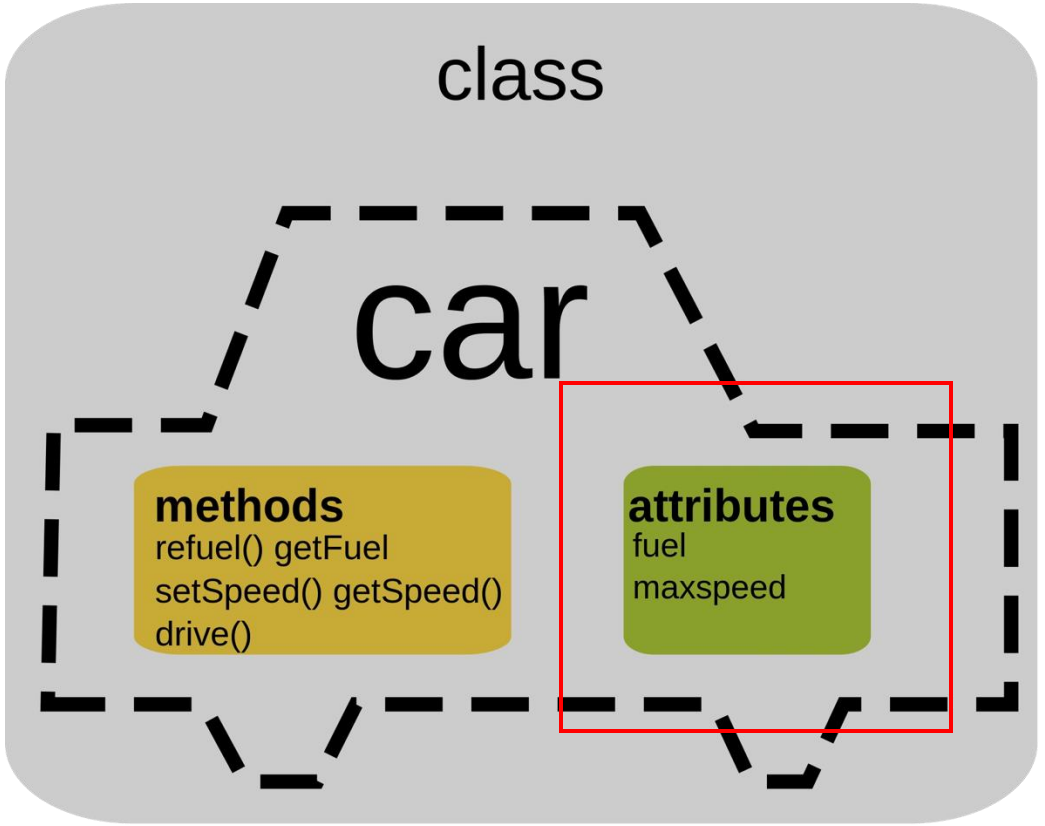



```php
<?php
  class Baju
  {
    // isi dari class
  }
?>
```

PROPERTY

- Property atau attribut merupakan data yang terdapat dalam class.
- Property adalah variabel yang berada di dalam class.
- Contoh:

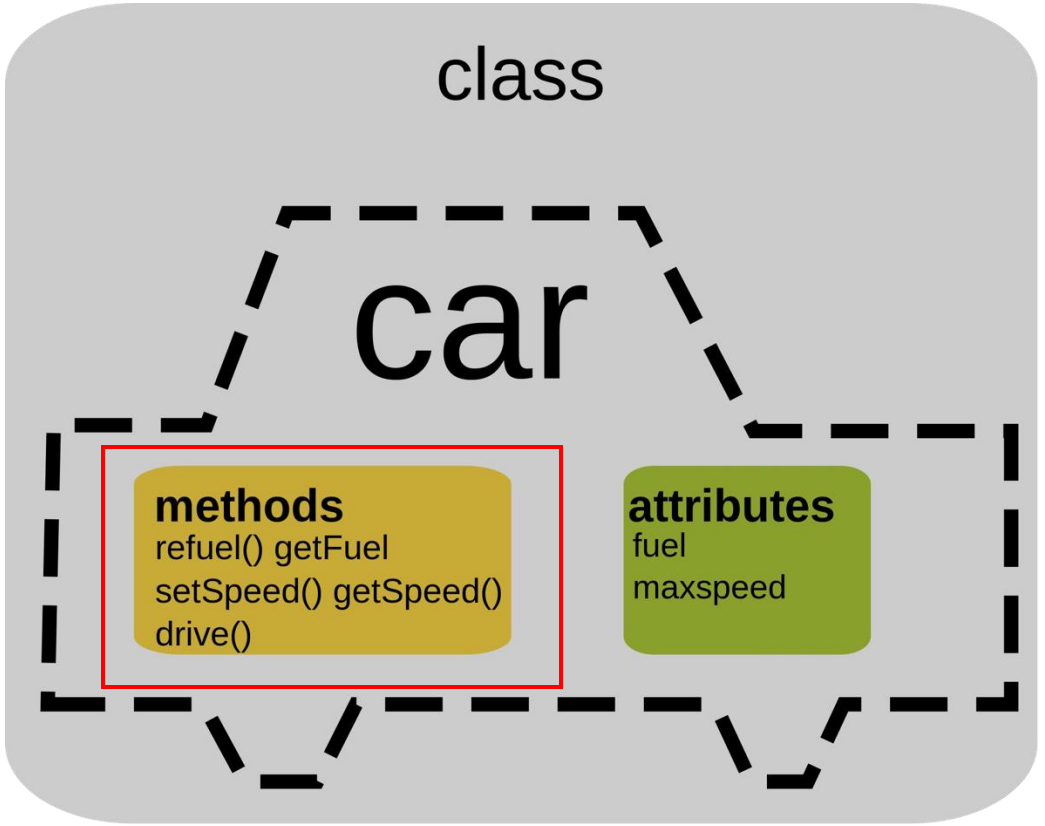
```php
<?php
class Baju {
    // Atribut (sifat) dari class Baju
    public $warna;
    public $merek;
    public $ukuran;
}
?>
```



METHOD

- Method adalah fungsi yang bisa dilakukan di dalam class.
- Method adalah fungsi 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.
- Contoh:

```php
<?php
class Baju {
    // Atribut (sifat) dari class Baju
    public $warna;
    public $merek;
    public $ukuran;

    // Metode untuk menampilkan informasi baju
    public function infoBaju() {
      // isi method
    }

    // Metode untuk mengganti warna baju
    public function gantiWarna($warnaBaru) {
      // isi method
    }
}

?>
```



OBJECT

- Object adalah hasil cetak dari class.
- Banyaj object bisa dibuat menggunakan 1 class.
- Contoh:

```php
<?php
class Baju {
    // Atribut (sifat) dari class Baju
    public $warna;
    public $merek;
    public $ukuran;

    // Metode untuk menampilkan informasi baju
    public function infoBaju() {
      // isi method
    }

    // Metode untuk mengganti warna baju
    public function gantiWarna($warnaBaru) {
      // isi method
    }
}
// Membuat objek dari class Baju
$bajuAlfian = new Baju();
$bajuTyo = new Baju();
?>
```

ACCESS CONTROL CLASS

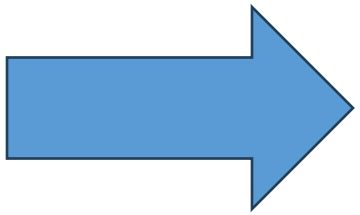
- Enkapsulasi adalah sebuah metoda untuk mengatur struktur class dengan cara menyembunyikan alur kerja dari class tersebut.
- Struktur class yang dimaksud adalah property dan method. Dengan enkapsulasi, kita bisa membuat pembatasan akses kepada property dan method, sehingga hanya property dan method tertentu saja yang bisa diakses dari luar class. Enkapsulasi juga dikenal dengan istilah 'information hiding'.

HAK AKSES PUBLIC, PROTECTED DAN PRIVATE

- Public yaitu seluruh kode program di luar class bisa mengaksesnya, termasuk class turunan.
- Protected yaitu tidak bisa diakses dari luar class, namun bisa diakses oleh class itu sendiri atau turunan class tersebut.
- Private yaitu hanya bisa diakses oleh class itu sendiri.

CONTOH HAK AKSES PUBLIC

```php
<?php
// buat class Baju
class Baju {

    // buat public property
    public $pemilik;

    // buat public method
    public function pakaiBajuOK() {
        return "Pakai Baju AstraTech";
    }
}

// buat objek dari class Baju (instansiasi)
$bajuAlfian = new Baju();

// set property
$bajuAlfian->pemilik = "Dean ";

// tampilkan property
echo $bajuAlfian->pemilik;

// tampilkan method
echo $bajuAlfian->pakaiBajuOK();
?>
```

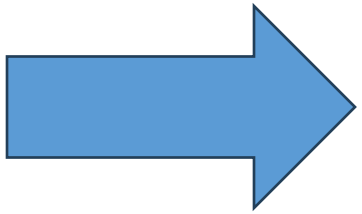


Output:

Dean Pakai Baju AstraTech

CONTOH HAK AKSES PROTECTED

```php
<?php
// Buat class Seragam
class Seragam {

    // Buat protected property
    protected $pemilik;

    // Buat protected method
    protected function status() {
        return "Seragam sedang dipinjam.";
    }
}

// Buat objek dari class Seragam (instansiasi)
$seragam_anto = new Seragam();

// Set protected property akan menghasilkan error
$seragam_anto->pemilik = "Anjariska";
// Fatal error: Cannot access protected property Seragam::$pemilik

// Tampilkan protected property akan menghasilkan error
echo $seragam_anto->pemilik;
// Fatal error: Cannot access protected property Seragam::$pemilik

// Jalankan protected method akan menghasilkan error
echo $seragam_anto->status();
// Fatal error: Call to protected method Seragam::Status() from context
?>
```



Output:

```
PHP Fatal error:  Uncaught Error: Cannot access protected property
Stack trace:
#0 {main}
```

CONTOH HAK AKSES PRIVATE

```php
<?php

// buat class komputer
class komputer {

    // property dengan hak akses protected
    private $jenis_processor = "Intel Core i7-4790 3.6Ghz";

    public function tampilkan_processor() {
      return $this->jenis_processor;
    }
}

// buat class laptop
class laptop extends komputer{

    public function tampilkan_processor() {
      return $this->jenis_processor;
    }
}

// buat objek dari class laptop (instansiasi)
$komputer_baru = new komputer();
$laptop_baru = new laptop();

// jalankan method dari class komputer
echo $komputer_baru->tampilkan_processor(); // "Intel Core i7-4790 3.6Ghz"

// jalankan method dari class laptop (error)
echo $laptop_baru->tampilkan_processor();
// Notice: Undefined property: laptop::$jenis_processor
?>
```

CONTOH KONSEP OOP PHP

```php
<?php
// buat class laptop
class laptop {

   // buat property untuk class laptop
   public $pemilik="Andi";
   public $merk;

   // buat method untuk class laptop
   public function hidupkan_laptop() {
     return "Hidupkan Laptop $pemilik";
   }
}

// buat objek dari class laptop (instansiasi)
$laptop_baru = new laptop();

echo $laptop_baru->hidupkan_laptop(); // "Hidupkan Laptop"
?>
```

Contoh disamping akan menampilkan error:
Notice: Undefined variable: pemilik in D:\xampp\htdocs\PRG5_20201\Materi\M5\M5.php on line 11 Hidupkan Laptop

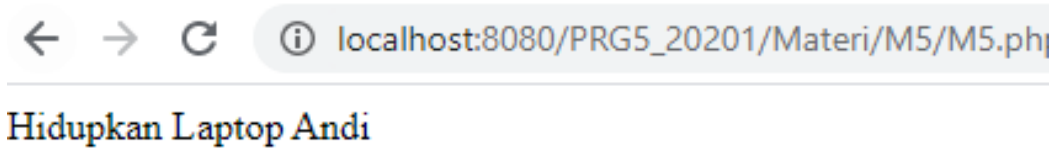
Kenapa hal tersebut bisa terjadi ?

FUNGSI VARIABEL $THIS

- Variabel **$this** adalah sebuah variabel khusus dalam OOP PHP yang digunakan sebagai penunjuk kepada objek, ketika kita mengaksesnya dari dalam class. Dalam manual PHP, $this disebut dengan istilah: **pseudo-variable**.

PERBAIKAN CONTOH KONSEP OOP

```php
<?php
// buat class laptop
class laptop {

    // buat property untuk class laptop
    public $pemilik="Andi";
    public $merk;

    // buat method untuk class laptop
    public function hidupkan_laptop() {
      return "Hidupkan Laptop $this->pemilik";
    }
}

// buat objek dari class laptop (instansiasi)
$laptop_baru = new laptop();

echo $laptop_baru->hidupkan_laptop(); // "Hidupkan Laptop"
?>
```



INTERFACE

- Interface adalah sebuah kontrak atau perjanjian implementasi method.
- class yang menggunakan object interface, class tersebut harus mengimplementasikan ulang seluruh method yang ada di dalam interface.
- Contoh:

```php
<?php
interface mouse{
    public function klik_kanan();
    public function klik_kiri();
}

class laptop implements mouse{
    public function klik_kanan(){
      return "Klik Kanan...";
    }
    public function klik_kiri(){
      return "Klik Kiri...";
    }
}

$laptop_baru = new laptop();
echo $laptop_baru->klik_kanan();
// Klik Kanan...
?>
```

INTERFACE

- Method Interface Harus di set Sebagai Public.
- 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***).
- Jika kita mengubah hak akses method di dalam *interface* menjadi ***private*** atau ***protected***, PHP akan mengeluarkan *error*

```php
<?php
interface mouse{
    public function klik_kanan();
    protected function klik_kiri();
}

class laptop implements mouse{
    public function klik_kanan(){
      return "Klik Kanan...";
    }
    public function klik_kiri(){
      return "Klik Kiri...";
    }
}

$laptop_baru = new laptop();
// Fatal error: Access type for interface
// method mouse::klik_kiri() must be omitted
?>
```

INTERFACE

- Interface bisa di Turunkan (Inherit)
- Prosesnya sama dengan penurunan *class*, yakni dengan menggunakan kata kunci ***extends.***

```php
<?php
interface mouse{
   public function klik_kanan();
   public function klik_kiri();
}

interface mouse_gaming extends mouse{
   public function ubah_dpi();
}

class laptop implements mouse_gaming{
   public function klik_kanan(){
     return "Klik Kanan...";
   }

   public function klik_kiri(){
     return "Klik Kiri...";
   }

   public function ubah_dpi(){
     return "Ubah settingan DPI mouse";
   }
}

$laptop_baru = new laptop();
echo $laptop_baru->ubah_dpi();
// Ubah settingan DPI mouse
?>
```

REFERENSI

- https://www.duniailkom.com/tutorial-belajar-oop-php-pemrograman-berbasis-objek-php/
- https://www.duniailkom.com/tutorial-belajar-oop-php-pengertian-class-object-property-dan-method/
- https://www.duniailkom.com/tutorial-belajar-oop-php-pengertian-enkapsulasi-objek-public-protected-dan-private/
- https://www.duniailkom.com/tutorial-belajar-oop-php-pengertian-object-interface-dalam-pemrograman-berbasis-objek/

TERIMA KASIH