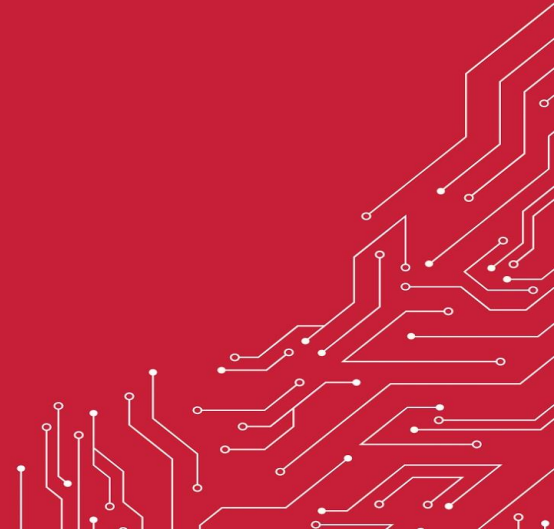




OOP PHP

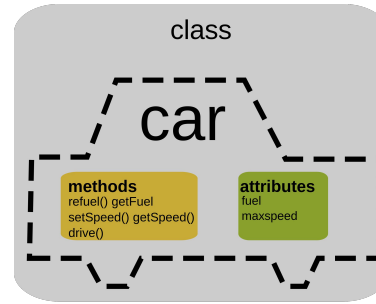


Apa OOP?

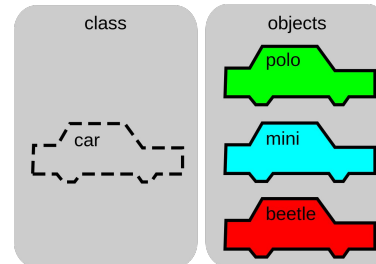
OOP (Object Oriented Programming) atau pemrograman berbasis objek merupakan paradigma pemrograman yang berorientasikan objek. Semua data dan fungsi di dalam paradigma ini dibungkus dalam kelas-kelas atau objek-objek.

Di dalam OOP, kita dapat memetakan persoalan-persoalan dalam program dengan memecah masalah ke dalam *class-class* yang lebih kecil dan simpel agar solusi yang dibuat lebih spesifik.

Setiap *class* dalam OOP mempunyai method atau fungsi serta property atau atribut. *method* adalah kemampuan dari class untuk melakukan sesuatu, sedangkan *property* adalah segala sesuatu yang dimiliki oleh class.



Class dan Object. class adalah cetakan atau blueprint dari objek. Di dalam class terdapat property dan method. contohnya di bawah ini, terdapat class car yang merupakan cetakan dari objek-objek mobil. Pada gambar sebelumnya *class car* bisa memiliki method yaitu *refuel()*, *getSpeed()*, *setSpeed()*, *drive()* dan memiliki property *fuel*, *maxspeed*.




Contoh Class

```
<?php
class Mobil
{
    public $roda = 4;
    public function jalan()
    {
        echo "Mobil berjalan";
    }
}
?>
```

Pada contoh di atas, terdapat class Mobil yang di dalamnya terdapat method `jalan()` dan property `$roda`.

Membuat Object (Intansiasi). pada penjelasan sebelumnya bahwa class merupakan *blueprint* atau cetakan dari objek. Untuk membuat objek dari cetakan tersebut kita harus melakukan instansiasi atau pembuatan objek. caranya adalah seperti berikut:

```
$mini = new Mobil();
```



Pada contoh disamping dibuat sebuah object baru dengan nama `$mini` yang merupakan hasil instansiasi dari class Mobil. Karena `$mini` dihasilkan dengan cetakan mobil maka dia memiliki property dan method yang sama dengan class Mobil. Cara memanggil property dan method yaitu dengan tanda panah `->`.

```
$<?php
$mini = new Mobil();
$mini->jalan(); // menampilkan echo 'Mobil
berjalan'
echo $mini->roda; // 4

?>
```

\$this pada Class

Di dalam class kita akan sering menulis **\$this** yang berarti merujuk kepada class itu sendiri. scope atau ruang lingkup **\$this** adalah segala sesuatu yang ada di dalam tanda kurung kurawal { } setelah penamaan class NamaClass.Contohnya seperti berikut

```
class Mobil {  
    protected $roda = 4;  
    public function jumlah_roda() {  
        echo $this->roda;  
    }  
}  
  
$mini = new Mobil;  
$mini->jumlah_roda(); // 4
```

pada contoh di atas, di dalam function **jumlah_roda()** dipanggil **\$this->roda** yang merujuk kepada property **\$roda** pada **class Mobil**

Visibilitas pada OOP

Dalam PHP, visibilitas dibagi menjadi 4 yaitu private, protected, public dan default. visibilitas digunakan untuk mengatur hak akses terhadap property dan method pada class. Hal ini dimaksudkan agar menjamin keamanan informasi yang terdapat pada property maupun method.

1.Private. property atau method dengan visibilitas private maka property atau method tersebut hanya dapat diakses dari lingkup class dimana property atau method tersebut didefinisikan. contohnya:

```
<?php  
class Mobil  
{  
    private $roda = 4;  
    private function jalan()  
    {  
        echo 'Mobil berjalan';  
    }  
}  
$savanza = new Mobil();  
echo $savanza->jalan();  
echo PHP_EOL;  
echo $savanza->roda;  
echo PHP_EOL;  
?  
>
```

2.Protected. method atau property yang diberikan visibilitas protected maka method atau property tersebut dapat diakses dari lingkup class dimana property atau method tersebut didefinisikan dan pada class turunan (inheritance) dari class tersebut.

```
<?php

class Mobil
{
    protected $roda = 4;
}

class MobilSport extends Mobil
{
    protected $maxSpeed;
}

$ferrari = new MobilSport;
echo $ferrari->roda ; // 4

?>
```

pada contoh di atas, class MobilSport merupakan inheritance atau turunan dari class Mobil. property \$roda yang dimiliki oleh class Mobil diturunkan ke class MobilSport dan tetap bisa dipakai di class MobilSport jika menggunakan visibilitas protected.

3.Public Jika property atau method diberikan visibilitas public maka method atau property tersebut dapat diakses baik dari lingkup class maupun object yang sudah diinstansiasi.

```
<?php
class Mobil
{
    private $roda = 4;
    public function jumlahRoda()
    {
        echo $this->roda;
    }
}

$kijang = new Mobil();
$kijang->jumlahRoda(); // menampilkan 4
```

Constructor

Constructor pada class yaitu method atau function yang akan dipanggil pertama kali ketika class tersebut diinstansiasi menjadi object. untuk membuat constructor kita buat method dengan nama __construct().



Contoh constructor sebagai berikut :

```
<?php

class Mobil {
    protected $roda= 4;
    public $merk;
    public function __construct($merk)
    {
        $this->merk= $merk;
    }
}

?>

$Xeniya = new Mobil("Xeniya");

echo $Xeniya->merk; // Xeniya
```

pada contoh di atas, property \$merk pada class Mobil hanya didefinisikan tanpa diberikan nilai. Lalu pada method construct dilakukan assign nilai merk diisi dengan parameter \$merk pada method construct tersebut.

Ketika object \$Xeniya dibuat, cara instansiasinya adalah dengan mengetik `new Mobil("Xeniya")` . parameter "Xeniya" pada instansiasi tersebut adalah parameter yang akan terbaca pada metode construct sehingga object \$Xeniya memiliki property \$merk yang bernilai "Xeniya".



THANK YOU

