

PEMOGRAMAN FRAMEWORK



PHP | OOP

KONSEP OOP PADA PHP

(Semua Script Di Tes Dengan Menggunakan PHP 7.2)

TAMUS TAHIR

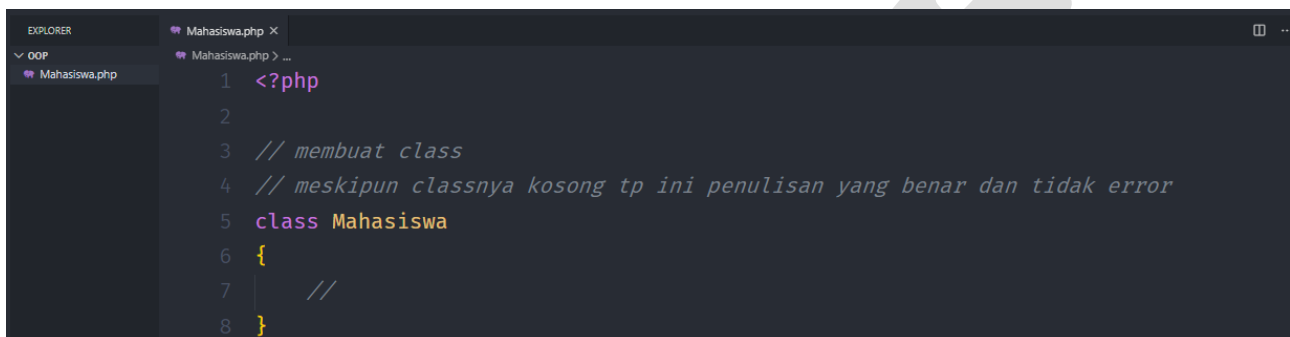
2020

OBJECT ORIENTED PROGRAMING

1. Jalankan xampp
2. Buat folder baru dan buka pada text editor
3. Jalankan pada browser

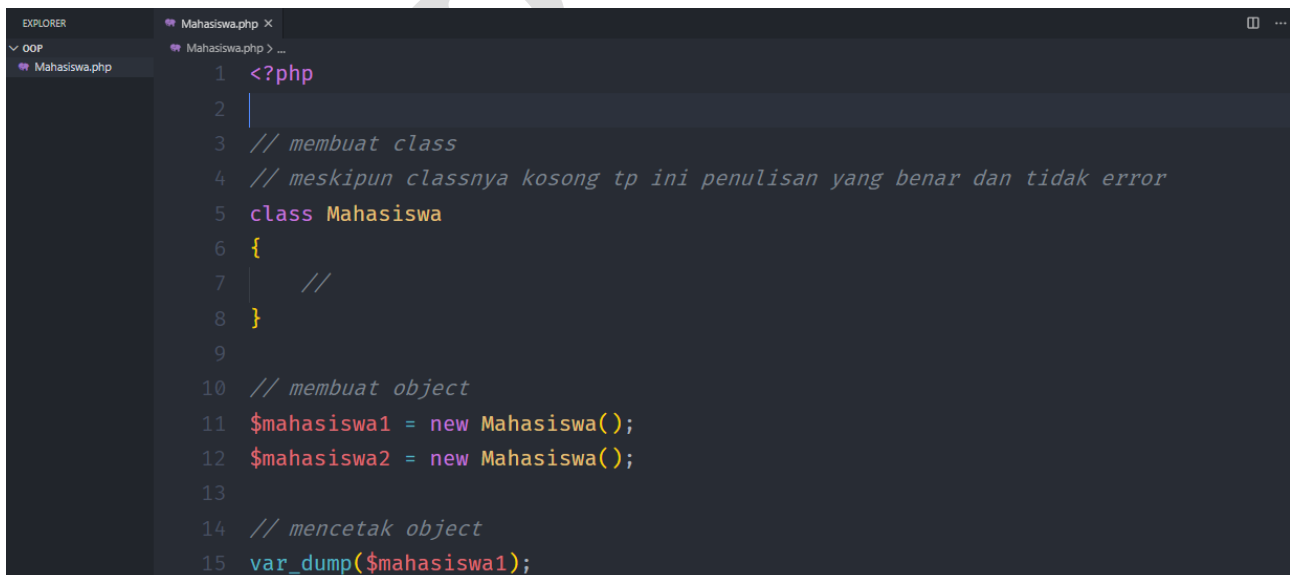
CLASS

1. Ditulis dengan keyword class kemudian nama classnya dibatasi dengan { }
2. Aturan nama class seperti variable
3. Nama class diawali dengan huruf besar
4. Nama file class sesuai dengan nama class dan diawali juga dengan huruf besar



```
EXPLORER
Mahasiswa.php
Mahasiswa.php > ...
1 <?php
2
3 // membuat class
4 // meskipun classnya kosong tp ini penulisan yang benar dan tidak error
5 class Mahasiswa
6 {
7     //
8 }
```

OBJECT



```
EXPLORER
Mahasiswa.php
Mahasiswa.php > ...
1 <?php
2
3 // membuat class
4 // meskipun classnya kosong tp ini penulisan yang benar dan tidak error
5 class Mahasiswa
6 {
7     //
8 }
9
10 // membuat object
11 $mahasiswa1 = new Mahasiswa();
12 $mahasiswa2 = new Mahasiswa();
13
14 // mencetak object
15 var_dump($mahasiswa1);
```

```
localhost/oop/Mahasiswa.php x +
object(Mahasiswa)#1 (0) { }
```

PROPERTY

1. Aturan penulisan sama seperti penulisan variable
2. Ditambah dengan visibility: public, private dan protected

```
EXPLORER
Mahasiswa.php x
Mahasiswa.php > ...
1 <?php
2
3 // membuat class
4 // meskipun classnya kosong tp ini penulisan yang benar dan tidak error
5 class Mahasiswa
6 {
7     // membuat property
8     public $nim, // public adalah visibility
9         $nama,
10        $gender,
11        $fakultas,
12        $prodi;
13 }
14
15 // membuat object
16 $mahasiswa1 = new Mahasiswa();
17 $mahasiswa2 = new Mahasiswa();
18
19 // mencetak object
20 var_dump($mahasiswa1);
```

```
localhost/oop/Mahasiswa.php x +
object(Mahasiswa)#1 (5) { ["nim"]=> NULL ["nama"]=> NULL ["gender"]=> NULL ["fakultas"]=> NULL ["prodi"]=> NULL }
```

MEMBERI NILAI DEFAULT PADA PROPERTY

```

5 class Mahasiswa
6 {
7     // membuat property
8     // memberikan nilai default
9     public $nim = 'NIM',
10         $nama = 'Nama',
11         $gender = 'Gender',
12         $fakultas = 'Fakultas',
13         $prodi = 'Prodi';
14 }

```

localhost/oop/Mahasiswa.php

object(Mahasiswa)#1 (5) { ["nim"]=> string(3) "NIM" ["nama"]=> string(4) "Nama" ["gender"]=> string(6) "Gender" ["fakultas"]=> string(8) "Fakultas" ["prodi"]=> string(5) "Prodi" }

MEMBERI NILAI PADA PROPERTY DARI OBJECT

```

15
16 // membuat object
17 $mahasiswa1 = new Mahasiswa();
18 $mahasiswa1->nim = '2020055003';
19 $mahasiswa1->nama = 'Monkey D Luffy';
20
21 $mahasiswa2 = new Mahasiswa();
22

```

localhost/oop/Mahasiswa.php

localhost/oop/Mahasiswa.php

Monkey D Luffy

METHOD

1. Method adalah function di dalam class
2. Method sebaiknya mengembalikan nilai
3. \$this untuk mengambil isi dari property yang ada di dalam class

```
EXPLORER
Mahasiswa.php x
Mahasiswa
Mahasiswa.php
1 <?php
2
3 // membuat class
4 // meskipun classnya kosong tp ini penulisan yang benar dan tidak error
5 class Mahasiswa
6 {
7     // membuat property
8     // memberikan nilai default
9     public $nim = 'NIM',
10         $nama = 'Nama',
11         $gender = 'Gender',
12         $fakultas = 'Fakultas',
13         $prodi = 'Prodi';
14
15     // membuat method
16     public function getNamaMahasiswa()
17     {
18         return $this->nama;
19     }
20 }
```

4. Cetak method

```
40
41 // mencetak method
42 echo $mahasiswa1->getNamaMahasiswa();
43
```

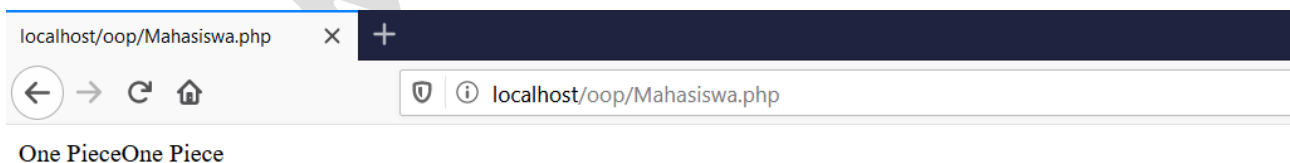


CONSTRUCTOR

1. Merupakan sebuah method yang khusus ada di dalam sebuah class
2. Method yang otomatis dijalankan ketika sebuah kelas kita buat objeknya

```
EXPLORER
Mahasiswa.php x
Mahasiswa.php > ...
1 <?php
2 // membuat class
3 // meskipun classnya kosong tp ini penulisan yang benar dan tidak error
4 class Mahasiswa
5 {
6     // membuat property
7     // memberikan nilai default
8     public $nim = 'NIM',
9         $nama = 'Nama',
10        $gender = 'Gender',
11        $fakultas = 'Fakultas',
12        $prodi = 'Prodi';
13
14    // membuat constructor
15    public function __construct()
16    {
17        echo 'One Piece';
18    }
19
20    // membuat method
21    public function getNamaMahasiswa()
22    {
23        return $this->nama;
24    }
25 }
26
27 // membuat object
28 $mahasiswa1 = new Mahasiswa();
29
```

3. Disini one piece tercetak dua kali karena kita mempunyai 2 object



4. Setiap dipanggil construct ini akan menerima parameter untuk mengisi property yang ada di dalam class
5. Nilai yang dikirimkan oleh object akan mengisi parameter construct kemudian akan mengisi property dari class

```
Mahasiswa.php x
Mahasiswa.php > Mahasiswa
1  <?php
2
3  // membuat class
4  // meskipun classnya kosong tp ini penulisan yang benar dan tidak error
5  class Mahasiswa
6  {
7      // membuat property
8      // memberikan nilai default
9      public $nim = 'NIM',
10         $nama = 'Nama',
11         $gender = 'Gender',
12         $fakultas = 'Fakultas',
13         $prodi = 'Prodi';
14
15     // membuat constructor
16     public function __construct($nim, $nama, $gender, $fakultas, $prodi)
17     {
18         // $this->nim akan menerima nilai dari paramater $nim pada construct
19         $this->nim = $nim;
20         $this->nama = $nama;
21         $this->gender = $gender;
22         $this->fakultas = $fakultas;
23         $this->prodi = $prodi;
24     }
25
```

6. Sekarang ketika membuat object kita harus mengirimkan parameter

```
33 // membuat object
34 $mahasiswa1 = new Mahasiswa(2020055003, 'Luffy', 'Laki-Laki', 'Teknik', 'Informatika');
35 // $mahasiswa1->nim = '2020055003';
36 // $mahasiswa1->nama = 'Monkey D Luffy';
37 $mahasiswa2 = new Mahasiswa(2020055025, 'Nami', 'Perempuan', 'Teknik', 'Elektro');
38
39 // mencetak object
40 var_dump($mahasiswa1);
41
```

```
localhost/oop/Mahasiswa.php x +
localhost/oop/Mahasiswa.php
object(Mahasiswa)#1 (5) { ["nim"]=> int(2020055003) ["nama"]=> string(5) "Luffy" ["gender"]=> string(9) "Laki-Laki" ["fakultas"]=> string(6) "Teknik" ["prodi"]=> string(11) "Informatika" }
```

MEMBERI NILAI DEFAULT PADA CONSTRURTOR

```

5 class Mahasiswa
6 {
7     // membuat property
8     // memberikan nilai default
9     public $nim,
10         $nama,
11         $gender,
12         $fakultas,
13         $prodi;
14
15     // membuat constructor
16     public function __construct($nim = 'NIM', $nama = 'Nama', $gender = 'Gender', $fakultas
= 'Fakultas', $prodi = 'Prodi')
17     {
18         // $this->nim akan menerima nilai dari paramater $nim pada construct
19         $this->nim = $nim;
20         $this->nama = $nama;
21         $this->gender = $gender;
22         $this->fakultas = $fakultas;
23         $this->prodi = $prodi;
24     }

```

1. Karena nilai defaultnya sudah ada kita bisa kosongkan parameter ketika membuat object

```

36 // $mahasiswa1->nama = 'Monkey D Luffy';
37 $mahasiswa2 = new Mahasiswa(2020055025, 'Nami', 'Perempuan', 'Teknik', 'Elektro');
38 $mahasiswa3 = new Mahasiswa();
39
40 // mencetak object
41 var_dump($mahasiswa3);

```

localhost/oop/Mahasiswa.php x +

localhost/oop/Mahasiswa.php

object(Mahasiswa)#3 (5) { ["nim"]=> string(3) "NIM" ["nama"]=> string(4) "Nama" ["gender"]=> string(6) "Gender" ["fakultas"]=> string(8) "Fakultas" ["prodi"]=> string(5) "Prodi" }

INHERITANCE / PEWARISAN

1. Kita dapat mengambil property dan method yang ada pada class sebelumnya pada class yang baru yang akan kita buat.
2. Buat file baru Alumni.css

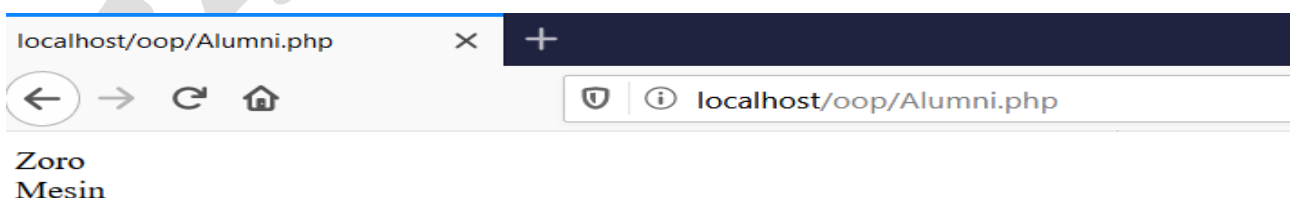

```
EXPLORER
Mahasiswa.php
Alumni.php x
Alumni.php
Mahasiswa.php

1 <?php
2
3 require_once 'mahasiswa.php';
4
5 // extends adalah mewarisi semua fungsi dan semua property dalam class parrent
6 // Mahasiswa adalah class parrent
7 class Alumni extends Mahasiswa
8 {
9     //
10 }
11
12 $alumni1 = new Alumni(20200555003, 'Zoro', 'Laki-Laki', 'Teknik', 'Mesin');
13 // disini kita gunakan method yang ada pada class mahasiswa
14 echo $alumni1->getNamaMahasiswa();
```

3. Membuat method baru

```
EXPLORER
Mahasiswa.php
Alumni.php x
Alumni.php
Mahasiswa.php

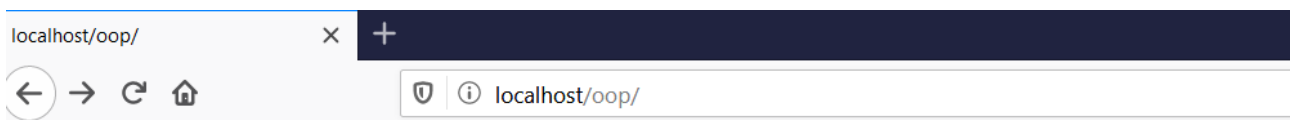
1 <?php
2
3 require_once 'mahasiswa.php';
4
5 // extends adalah mewarisi semua fungsi dan semua property dalam class parrent
6 // Mahasiswa adalah class parrent
7 class Alumni extends Mahasiswa
8 {
9     // membuat method baru yang hanya berfungsi pada class alumni
10    // sedangkan class mahasiswa tidak bisa menggunakan method ini
11    public function getProdiMahasiswa()
12    {
13        return $this->prodi;
14    }
15 }
16
17 $alumni1 = new Alumni(20200555003, 'Zoro', 'Laki-Laki', 'Teknik', 'Mesin');
18 // disini kita gunakan method yang ada pada class mahasiswa
19 echo $alumni1->getNamaMahasiswa();
20 echo '<br>';
21 echo $alumni1->getProdiMahasiswa();
```



AUTOLOAD

1. Ketika kita membuat file baru yang membutuhkan method yang ada pada sebuah class kita harus memanggil file tersebut dengan require atau include

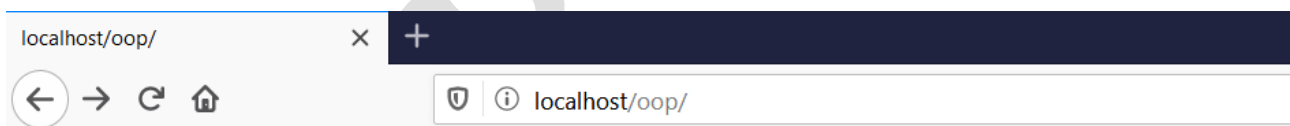
```
1 <?php
2
3 require_once 'mahasiswa.php';
4 require_once 'alumni.php';
5
6
7 $alumni1 = new Alumni(20200555029, 'Sanji', 'Laki-Laki', 'Teknik', 'Elektro');
8 echo $alumni1->getProdiMahasiswa();
```



Elektro

2. Hal ini akan menjadi rumit jika filenya sudah banyak, disinilah kita membutuhkan autoload

```
1 <?php
2
3 spl_autoload_register(function ($class_name) {
4     include $class_name . '.php';
5 });
6
7
8 $alumni1 = new Alumni(20200555029, 'Sanji', 'Laki-Laki', 'Teknik', 'Elektro');
9 echo $alumni1->getProdiMahasiswa();
10
```



Elektro

VISIBILITY

1. Visibility adalah digunakan mengatur hak akses terhadap property dan method dari sebuah class, jadi dengan menggunakan visibility ini anda dapat mengatur, property dan method

2. Visibility public adalah visibilitas tertinggi yang ada pada OOP, jika sebuah property atau method menggunakan visibility public, maka property atau method tersebut dapat diakses melalui class itu sendiri, ataupun melalui object.
3. Visibility berikutnya adalah protected, jika property atau method menggunakan hak akses protected maka, property atau method tersebut hanya bisa diakses melalui class itu sendiri dan class turunannya.
4. Visibility Private adalah hak akses yang paling rendah, jika property atau method yang menggunakan hak akses private, maka property atau method tersebut hanya dapat diakses di lingkup class dimana property atau method tersebut didefinisikan.
5. Untuk saat ini kita masih menggunakan visibility public, nantinya kita akan mencoba visibility ini pada codeigniter

DASAR DARI OOP MASIH SANGAT BANYAK, SILAHKAN CARI INFORMASI TAMBAHAN. SELANJUTNYA KITA AKAN MASUK KE PENGGUNAAN CODEIGNITER.